DEVELOPMENT OF AN IMPROVED MODEL FOR BIG DATA ANALYTICS USING DYNAMIC MULTI-SWARM OPTIMIZATION AND UNSUPERVISED LEARNING ALGORITHMS

BY

OLEJI CHUKWUEMEKA PHILIPS
(B.Sc., PGD (UPH), MSc (FUTO))

REG. NO 20164024878

A THESIS SUBMITTED TO THE POSTGRADUATE SCHOOL FEDERAL UNIVERSITY OF TECHNOLOGY OWERRI

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF DOCTOR OF PHILOSOPHY, Ph.D IN COMPUTER SCIENCE

JULY, 2021

CERTIFICATION

This is to certify that this work "Development of an Improved Model for Big Data Analytics Using Dynamic Multi-Swarm Optimization, and Unsupervised Learning Algorithms" was carried out by Oleji, Chukwuemeka Philips (20164024878) in partial fulfilment for the award of the Degree of Ph.D. in Computer Science in the Department of Computer Science of the Federal University of Technology, Owerri.

.....
Dr. (Mrs) E. E. Nwokorie
Supervisor

23/06/2021
Date

.....
Engr. Prof. (Mrs) G. A. Chukwudebe
Co-Supervisor

23/06/2021
Date

.....
Dr. I. I. Ayogu
Co-Supervisor

23/06/2021
Date

.....
Dr. (Mrs) J. N. Odii
HOD, Computer Science

23/06/2021
Date

.....
Prof. (Mrs) U. F. EZE
Dean, SCIT

23/06/2021
Date

.....
Prof. B. O. Esonu
Dean, Postgraduate School

.....
Date

.....
Prof Adebayo O. Adetunmbi
External Supervisor

14/12/2022
Date

DEDICATION

This work is dedicated to Almighty God and my late beloved father Engr. Oleji, Imo Victor.

ACKNOWLEDGEMENTS

My profound gratitude to the Almighty God for His kindness, strength and being an additional advantage to my academic success and breakthrough;

My greatest indebtness goes to my Supervisors Engr. Prof. (Mrs.) G. Chukwudebe, Prof. E.O. Nwachukwu, Dr. (Mrs.) C.E. Nwokorie and Dr. I. I. Ayogu, who spent their precious time and energy in reading through this work. I really appreciate their guidance, advice and suggestion made during the write up. And I say, God Bless You.

I will not fail to acknowledge all my Lecturers: Prof. Erumaka, Prof. Aloy C. Onyeka, Prof. Prince Asagba, Dr. (Mrs.) Juliet N. Odii, Dr. Celestine Njoku, Mr. Joseph I. Eke, Dr. Stanley A. Okolie, Mr. Obilor A. Njoku, Dr. Jacinta C. Odirichukwu, Dr. (Mrs.) Uchenna C. Onemauche, Dr. (Mrs.) Mercy E. Benson-Emenike, Mr. Stanley O. Diala, Dr. Chinwe G. Onukwugha, Dr. Chukwema D. Anyiam, Dr. (Mrs.) Chidimma I. Okpalla, Mr. Godson E. Ahamba, Mrs. FranciscaO. Nwokoma, Mr. Donatus O. Njoku, Mr. Kelechi A. Douglas, Mr. Peter K. Joseph, Mr. Friday J. Oforma, Mrs. Ezi Achama Eke, Mr. Chukwudi N. Akujobi, Mr.s. Idu E. Iheukwumere, Mr. Ikechukwu Onyeanu, and Mrs, Stella Egwu-ahaotu in Department of Computer Science for their Educational and Moral support throughout my programme.

I wish to express my heartfelt gratitude to my late Father Engr. Oleji Victor; and my Elder brother, Obinna David Oleji, for their encouragements and financial support to make this project a reality. My gratitude goes to my lovely sister and brothers, Joy Oleji, Ugochukwu Oleji, Okechukwu Oleji and also to my aunty Dada Ozioma Imo for their support in one way or the other in achieving this goal.

I have not forgotten my Internal Reviewer Engr. Dr. Kennedy C. Okafor, thanks for your professional experience and contributions to improve this work.

TABLE OF CONTENTS

Chapter III: METHODOLOGY

Chapter IV: RESULTS AND DISCUSSION

Chapter V: CONCLUSION AND RECOMMENDATIONS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ACO	Ant Colony Optimization
AI	Artificial Intelligence
BA	Bees Algorithm
BART	Regression Tree,
BDAF	Big Data Architecture Framework
BP	Back Propagation
CART	Classification and Regression Trees
CBDMASP	Cloud-Based Big Data Mining and Analyzing Services Platform
CCPSO	Cooperative coevolving particle swarm Optimization
CF	Cyber Forensics
CO	Combinatorial Optimization
CoS	Classify or Send for Classification
CPU	Central Processing Unit
CRF	Chi-square Relative Frequency
CSO	Competitive Swarm Optimizer
CSV	Comma Separated Values
CUDA	A parallel computing platform and programming model developed by NVIDIA
CVs	Cluster Variation
DB	Database
DL	Deep Learning
DLCF	Cyber Forensics
DMPSO	Dynamic Multi-swarm Particle Swarm Optimization
DMSO	Dynamic Multi-Swarm Optimization
DMSO	Dynamic Multi-swarm Optimizer

DOT	Data Transformation
ED	Euclidean Distance
ELN	National Liberation Army
FA	Firefly Algorithm
FCM	Fuzzy C-means
FFNNs	Feed-Forward Neural Network
GA	Genetic Algorithms
GLADE	Generalized Linear Aggregates Distributed Engine
GPU	Graphical Processing Units
GTD	Global Terrorist Database
HAC	Hierarchical Agglomerative clustering
HACE	Heterogenous Autonomous Complex Evolving (Big data theory)
HDFS	Hadoop Distributed File System
HTML	Hyper Text Markup Language
IoS	Internet of Services
IoT	Internet of Things
ISIS	Islamic State of Iraq and Syria
JAR	Java Archive
KDD	Knowledge Discovery in Database
KMNS	Spherical K-Means
L^p	Function spaces
LR	Logistic Regression
LS	Linear Sum
MBP	Multiple Back-Propagation
MI	Mutual Information

MIMD	Multiple Instruction, Multiple Data
MISD	Multiple Instruction, Single Data
ML	Machine Learning Algorithm
MLib	Machine Learning Library
SOM	Self-organizing
MPB	Multiple back-propagation
MRAM	Map Reduce Agent Mobility
MSF	Multiple Species Flocking
NN	Neural Network
NVIDIA	Nvidia Corporation is an American multinational technology company incorporated in Delaware and based in Santa Clara, California. designs graphics processing units for the gaming and professional markets.
OOAD	Object Oriented Analysis and Design methodology
OSKMNS	On-line Spherical K-Means
PCA	Principal Component Analysis
PSO	Particle swarm optimization algorithm
RANKPRO	Random Search with K-prototypes Algorithm
RDBMS	Relational Data Base Management System
RDDs	Resilient Distributed Datasets
RDW	Reference Distance Weighted
RF	Random Forest
SI	Swarm Intelligence
SIMD	Single Instruction Multiple Data
SISD	Single Instruction, Single Data
SKMNS	Spherical K-Means

SODSS	Service-Oriented Decision Support System
SOM	Self-Organizing Map
SQL	Structural Query Language
SS	Square Sum
SVM	Support Vector Machines,
TMCM	Two-step Method for Clustering Mixed
TXT	Text File Extension
UCI	University of California Ivory
UFT	Unsupervised Feature Transformation
UML	Unified Modeling language
USIBD	Under Sampled Imbalance Big Data
WEKA	Waikato Environment for Knowledge Analysis
PB	Petabytes
EB	Exabytes
ZB	Zettabyte

ABSTRACT

An improved model for big data analytics was developed in this work using dynamic multi-swarm optimization and unsupervised machine learning algorithms. The problems of premature convergence of traditional data mining models due to the influence of heterogeneous data types and the voluminous nature of big data were solved with the developed Dynamic-K-reference Clustering Algorithm. Java programming language was used for implementation and Python Jupyter Notebook, Apache Spark frameworks were utilized for the virtualization of the clustered output results. The developed model was used to analyze a big dataset of Boko Haram insurgency attacks in Nigeria. The big dataset of Boko Haram terrorist attacks was scraped from the social media. The attributes of the dataset including the area of attacks, period of attacks, death tolls, and attack strategies were used for the analysis for the period of 2008 to May 2019. The output clustered results of the area of attack produced 64% at Borno, Abuja 1.3%, Adamawa 1.3%, Gombe 3.8%, Kano, 2.5%, Kastina 2.5%, Maiduguri 20% and Yobe 5% respectively. The output clustered results of death tolls at different years produced 4.1% on 2011, 15.6% on 2012, 3.4% on 2013, 6.0% on 2014, 42.6% on 2015, 0.0% on 2016, 2.8% on 2017, 6.0% on 2018 and 19.5% on 2019 respectively. The results show constant attacks of Boko Haram insurgency in the study area, which had led to millions of people currently displaced and killed. The Dynamic-K-reference clustering algorithm is resourceful enough to provide clustering accuracy of 0.9820 and clustering sum of square error of 0.0018 from the analysis of the Boko Haram attacks dataset. In other to validate Dynamic-K-references clustering algorithm its performance was compared with the existing algorithms on six datasets from the machine learning repository: Hepatitis, Australian Credit Approval, German Credit Data, Starlog Heart, Soybean and Yeast. The analysis of four datasets with Dynamic-K-references clustering algorithm when compared with PSO-based K-prototype algorithm produced performance improvement of 22%, 17%, 34%, and 12%, respectively. Similar analysis of Soybean and Yeast datasets with the existing MixK-meansXFon algorithm and the Dynamic-K-reference clustering algorithm produced performance improvement of 13.8% and 13.7% respectively. From the analysis the Dynamic-K-reference algorithm was found to be robust and very efficient at expelling outliers from its dissimilar clusters/classifications. Future work should develop big data analytic services with the improved Dynamic-K-reference clustering algorithm and other improved models of its kind using a service-oriented architectural methodology for real time analysis and prediction.

Key Words: *Swarm Intelligence, Analytics, Dynamic Multi-swarm, Big data, Unsupervised Learning, Dynamic-K-reference clustering.*

CHAPTER ONE

INTRODUCTION

1.1 **Background Information**

Big data analytics can be defined as the process of collecting, organizing, and analyzing big data to discover and visualize patterns, correlations, knowledge, and intelligence as well as other information within the big data for supporting decision making (Minelli, Chamber & Dhiraji, 2013; Loshin, 2013). The main components of big data analytics include big data descriptive analytics, big data predictive analytics and big data prescriptive analytics (Minelli *et al*., 2013). "Big data descriptive analytics, also called business report addresses the problems such as what happened, and when, as well as what is happening" (Sun, & Yearwood, 2014). "Big data predictive analytics (Sun, Sun, & Strang, 2016) focuses on forecasting trends by addressing the problems such as what will happen, what's going to happen, what is likely to happen and why it will happen" (Delena & Demirkanb, 2013). Big data prescriptive (Sun & Yearwood, 2014) addresses the problems such as what we should do, why we should do it and what should happen with the best outcome under uncertainty" (Minelli *et al*., 2013). Big data consists of complex or very large data that the traditional data mining applications cannot efficiently analyze. The fast growths of the application of new technology development in this modern time have brought about very large unstructured and structured information from various areas of its applications and processes. These very large unstructured and structured data have the following characteristics such as variety, large volume, velocity, veracity and complexity. These characteristics are used to describe the concept and challenges of big data analytic.

Veracity (accuracy) of analytic results produce correlation of big data analytics for decision support system (Marcu, Danubianu, & Pentiuc, 2019). Also, inaccuracy in the analytic of big data results in waste of resources and jeopardizes the future prediction of the organization. The problem of inaccurate results caused by some influencing factors like outliers, voluminous, and variety or heterogeneous data types, affect the computational performance of various traditional machine learning analytic mechanisms. Outliers in large datasets such as in big data deviate the computational

mechanisms of most supervised and unsupervised algorithms (like k-mean, etc.) causing it to converge prematurely when creating clusters. Volumes of the large datasets pose a great challenge to computational procedures of unsupervised machine learning algorithm's starting point of data training iteration. The voluminous nature of large dataset influences the traditional machine learning algorithms (Betser & Belanger, 2015). It causes them to exploit the instance and attributes of the elements in the large dataset from the wrong direction. Many algorithms suffer from the challenges of dimensionality, which implies that their performance deteriorates quickly as the dimension of the search space increases (Onuodu & Nwachukwu, 2014).

Any algorithm that lacks the computational strength to manipulate large volumes of dataset and the complexity of outliers in its contents produces inaccurate results. These issues cause partitioning clustering algorithms as well as most meta-heuristic algorithms to converge prematurely at the local minimum.

Most artificial intelligence algorithms lack efficient mechanisms (such as MapReduce techniques) to handle multi-model problems resulting from the voluminous nature of big data analysis. Also (Palak & Vikas, 2017) commented that the size of big datasets posed challenges on traditional algorithms, their work applied Genetic algorithm that is self-sufficiency to handle the problem of big data in global space. Another challenge faced by traditional machine learning algorithms is the analytics of heterogeneous data types (Borne, 2014). These problems limit the application of some unsupervised machine learning algorithms such as K-means, K-mode and K-medoid algorithms on the analysis of mixed datasets. The extensions of these algorithms such as K-prototype clustering algorithm produces poor accuracy of clusters because of the limitations inherited from K-means and K-mode algorithm (Oleji, Nwokorie, Onuodu & Obinna, 2016; Onuodu *et al*., 2014).

Unsupervised Learning algorithms are a subset of machine learning that devices solutions from concurrent testing datasets with an unknown domain. Traditional Unsupervised Machine Learning algorithms are weak and inefficient to handle Big Data analytics. The Unsupervised Machine learning

algorithms, such as the K-prototype Clustering consists of (Lakshmi, Karthikeyani Visalakshi, Shanthi & Parvathavarthini, 2018) the integration of k-means and k-mode clustering algorithm (Garbade, 2018; Oleji *et al*., 2016). K-means algorithm is well known for clustering large numerical datasets. The use of k-means is often limited to numerical attributes. K-mode algorithm is the extension of K-means to create clusters of categorical datasets, based on simple mismatches between objects in the given region (Garbade, 2018). These partitional clustering algorithms are not efficient to handle Big data unique features such as massive, high dimensional, heterogeneous, complex, unstructured, incomplete, noisy, and erroneous, which affects their accuracy and time constrain processes.

Swarm Intelligent (SI) systems respond well to the rapidly changing environment of the dynamics of big data manipulations, making use of their inherent auto-configuration and self-organization capability. The adaptability character of SI allows them to systematically adapt their behavior to the external environment dynamically on run-time processes, with substantial flexibility (Scott, 2018). Five remarkable principles defined the SI paradigm used to explain the capabilities of DMSO to handle big data analytic challenges. It includes Proximity, quality, diversity, adaptability, and stability. First is the Principle of Proximity: the swarm has the potentials to carry out simple space and time computation interactively; the Second principle of quality: The Swarm should be able to figure out quality solutions in the complex environment; Third is the principle of diversity, the Swarm's operational performance should not be concentrated in a narrow region, however, random exploitation of resources in all directions. Forth is the principle of adaptability: the potentials of the swarm to adapt to environmental fluctuations when required in a flexible model. The Fifth is the principle of stability: The Swarm exhibits fault tolerance and at the same time stabilizes the system processes (Scott, 2018). Therefore, an extensive review of literature was done in this work to produce an efficient model for big data analytics using Dynamic Multi-Swarm Optimization (DMSO) algorithm and Unsupervised machine learning algorithms.

Furthermore, this work explored the applicability of SI bio-inspired approaches mentioned earlier which will be hybridized Reference Distance Weighted (RDW) of dissimilarity measure, Euclidean Distance Measure (ECM), and Chi-square relative frequency of dissimilarity measure to develop an improved algorithm for clustering mixed dataset. The mechanism of the proposed hybridized system of unsupervised learning and intelligent swarm optimization algorithm in this work should provide accurate analytics results of big data analytics. It can achieve the optimal solution in multi-objective optimization problems of Big Data analytic. Also, it would produce well-separated, connected, compact clusters of high-dimensional big data attributes for decision making.

1.2 Problem Statement

Social media such as Facebook, LinkedIn, WhatsApp, and Twitter have remained an effective and cheap means of communication between individuals and various organizations. The usage scenarios produce huge amounts of data that can be mined for insights but the volume, veracity, and velocity of the social media chats, as with other forms of big data, is challenging for traditional data mining approaches because of the heterogeneous and complex nature of the data. Since criminals and insurgents use social media for their activities too, timely and accurate information is critical for planning and developing strategic preventive measures and response against insurgency and insurgent-related activities at any point in time. This is not currently obtainable to a satisfactory degree because common data analysis tools lack adequate capacity to handle the size, nature, and speed of available social media data.

Big data is said to be the new oil because it brings opportunities for new products and services but at the same time it comes with challenges. One of the issues is that the high dimensionality of Big data has a problem of performance and accuracy with existing data mining models. Most machine learning algorithms which include partitioning/clustering algorithms, metaheuristic algorithms, and classification algorithms converge prematurely because of the influence of large datasets on their computational strength. Furthermore, the variety of unstructured datasets generated from various Big

data processes (Social media, sensors, IoT, etc.) is another problem faced by conventional analytic models.

This problem of converging prematurely during data classification/clustering due to outliers and the heterogeneous nature of big datasets is investigated in this research for the improvement of the reliability of the analytical results for decision-making. The motivation for the research work stems from the shortage of existing tools and techniques for Big data analysis of social media chats to mitigate security breaches and threats to life by criminals and insurgents.

1.3 Aim and Objectives

The aim of this work is to develop an improved model for big data analytics using dynamic multi-swarm optimization, and unsupervised machine learning clustering algorithms. The objectives are to:

i. develop improved clustering algorithms for mining unstructured datasets.

ii. develop a hybridized algorithm based on the improved clustering algorithm and dynamic multi-swarm optimization algorithm for analysis of a big dataset.

iii. scrape real-world dataset of the terrorist attack in Nigeria from social media for implementation and performance analysis of the improved big data analytic model.

iv. carryout a comparative performance analysis of the proposed and similar models for big data analytics.

1.4 Justification of the Study

The accuracy of the mining techniques is very important, because its result gives directives for investment, control, and insights on security control challenges. Big Data poses enormous challenges (such as heterogeneous data type, high dimensionality, validity, and veracity of analytic outcomes) on traditional machine learning algorithms. In Information Technology, Big Data is a collection of datasets so large and complex that they become difficult to compute and process using existing database management tools or traditional data processing applications. Thus, the time to reach decision in critical situations increases with the attendant consequences.

The proposed improved model in this work will be justified as follow:

1. The proposed hybrid of an intelligent characteristic of Dynamic Multi-swarm optimization agents and robust analytic behavior of partitioning clustering algorithms possess the potentials to handle the analytics challenges big data posed on traditional data mining models.
2. It provides an efficient and robust mechanism to overcome the challenges of traditional tools for bigdata analytics.
3. The outcome of this research will benefit national security planning and strategic response to insurgency on one hand and benefit business intelligence on the other hand.

1.5 **Scope of the Study**

This study is aimed to developing an improved model for big data analytics using Dynamic Multi-swarm Optimization, and clustering algorithms. It is concerned with the improvement of the K-means clustering algorithm by designing and eradiating a Dynamic-K-reference clustering algorithm that is proficient for the analysis of big dataset. The developed algorithm was evaluated using real-life social media dataset with the intent to determine strategies to mitigate insurgency attacks within Nigeria, particularly in the North Eastern parts of Nigeria.

CHAPTER TWO

LITERATURE REVIEW

This section will discuss the conceptional literature, theoretical literature, empirical review of big data analysis, machine learning algorithms and summary of highlights of literature review.

2.1 Conceptional Literature

The conceptual frameworks reviewed in this section will guide the study on the applications and implementations of big data tools, processing, and analytics methods. These will help to achieve the objectives of this thesis. The big data and its analytic architecture are represented graphically in Figure 2.1.

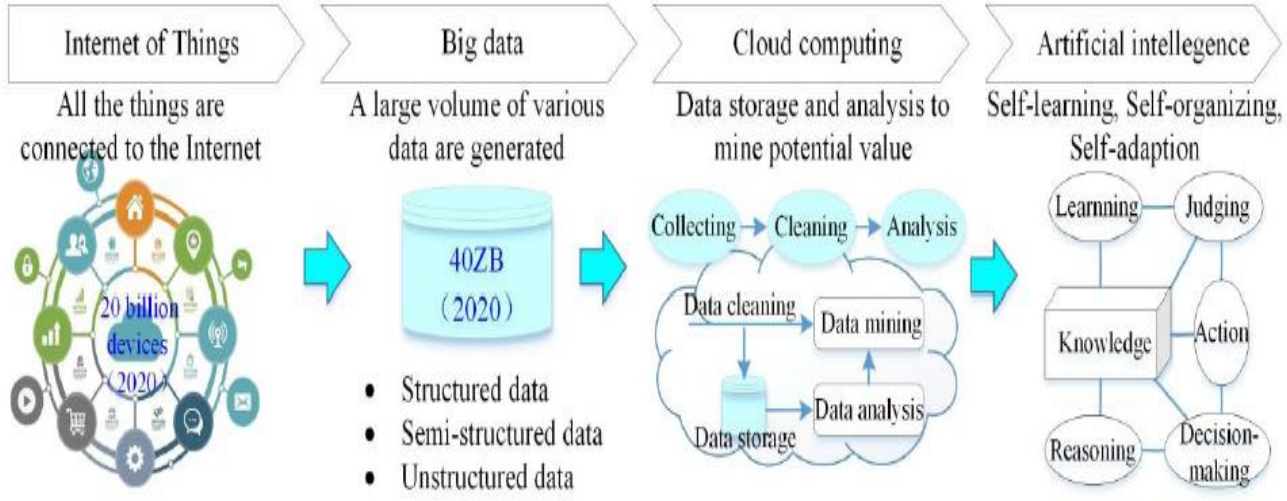

Figure 2.1: Big Data and Analytics architecture (Source: Qinglin & Fei, 2019).

2.1.1 **Review of Big Data**

"Big Data refers to collections of data that are too large for traditional models to extract information. It could either be because they cannot cope with the size or take too long to obtain results. The most characteristics of Big Data are the 5 Vs: volume, variety, velocity, veracity, and value" (Marcu *et al*., 2019).

"Big data is a term for data sets that are so large or complex that traditional data processing applications are inadequate. Challenges include analysis, capture, data curation, search, sharing,

storage, transfer, visualization, querying, and information privacy. The term Big Data was first introduced to the computing world by Roger Magoulas from O'Reilly media in 2005 to define a great amount of data that traditional data management techniques cannot manage and process due to the complexity and size of this data. A study on the Evolution of Big Data as a Research and Scientific Topic shows that the term Big Data was present in research starting with the 1970s but has been comprised in publications in 2008.

Nowadays the Big Data concept is treated from different points of view covering its implications in many fields". Pence (2015) defined Big Data as information that can be interpretated by Big data tools because of the huge volume of data, the variety of data formats, the velocity at which the data is flowing, and/or the complexity of the interpretation of the data. However, more recent articles have added two more Vs to give a better definition for Big Data. The additional Vs include veracity and value (Ishwarappa & Anuradha, 2015; Mehdi & Farshid, 2018).

The five Vs of big data analytics are as follows:

1. Volume: refers to the quantity of data gathered by a company. This data must be used further to obtain important knowledge; It is the quantity of data or information. The big volume of big data reflects the size of the data set, which is typically in exabytes (EB) or ZB (Vajjhala, Strang & Sun, 2015) although many, as a child, might consider that 10 is big as an integer. Nowadays, many data-driven companies are working with petabytes (PB) of data daily. Google processes over 20 PB of data daily (Pence, 2015). Walmart collects more than 2.5 PB of unstructured data from its 1 million customers every hour. Big data repositories for future generation parallel and distributed systems currently exceed EB and are increasing at a rapid speed in size. The volume of big data has been increased from EB or ZB (Vajjhala *et al*., 2015).

To overcome the challenge arising from big data with big volume, a massively parallel computing platform is a practical choice because its underlying principle is a distribution of workload across many processors as well as storage and transportation of underlying data across a set of parallel

storage units and streams (Sathi, 2013). Hadoop has become the de-facto standard for storing, processing, and analyzing the data with a big volume of hundreds of terabytes, and even PB. For example, Hadoop clusters at Yahoo! span over 40,000 servers, and store 40 PB of application data.

2. Velocity: refers to the time in which Big Data can be processed. Some activities are very important and need immediate responses, that is why fast processing maximizes efficiency; Big velocity is related to throughput and latency of data (Sathi, 2013). For throughput, big velocity means that at a big speed, data in and out from the networked systems in a real-time" (Betser *et al*., 2015). In other words, it is the high rate of data and information flowing into and out of interworked systems in real-time. The big velocity of big data is more important than big volume for many real-world applications (McAfee & Brynjolfsson, 2012). In some developing countries, some internetworked systems cannot be running normally because of low speed, although big data are available there. Latency is the other measure of velocity. Low latency is the requirement of modern businesses and individuals. For example, turn (www.turn.com) is conducting its analytics in 10 milliseconds to place advertisements in online advertising platforms (Sathi, 2013). IBM's InfoSphere Streams has been successfully used for low-latency, real-time analytics (Sathi, 2013).
3. Variety: Big variety means that big diversity or big different types of data sources with different structures from which it arrived, and the types of data available to everyone (Betser, and Belanger, 2015). Big data can be classified into three types: structured, semi-structured, and unstructured at a higher level. The data stored in relational database systems like Oracle are structured. The data available on the web are unstructured. 80% of the world's data is unstructured (Sathi. 2013). Blogs and tweets on social media are not structured data, because they contain a large number of slang words, with a mix of languages in a multiethnic, multi-language environment (Sathi. 2013). A big variety exists in the data on the Web. For example, in the WeChat world (https://web.wechat.com/), one can interact with his/her friends anywhere in the world using a variety of media such as text, sound clips, photos, and videos. The big variety brings about big

challenges for storage, mining, and analytics (Kumar, 2015). NoSQL databases have been used to overcome this challenge from a big variety (Kumar. 2015). There are 225+ NoSQL databases on the market.

A big variety of big data is related to big venue and variability. The big venue of big data includes distributed big data, heterogeneous big data from multiple platforms, from different owners' systems, with different access and formatting requirements, private vs. public cloud (Borne, 2014). Big variety of big data is also related to big variability which, e.g. consists of dynamic big data, evolving, spatiotemporal big data, time-series big data, seasonal big data, and any other type of non-static data sources.

4. Veracity: refers to the degree to which a leader trusts the user information to take a decision. So, getting the right correlations in Big Data is very important for the business future. Also, in Gartner's IT Glossary Big Data is defined as high volume, velocity and variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making (Marcu *et al*., 2019). Big veracity refers to the accuracy, truth, and truthfulness of big data. Veracity was introduced by IBM researchers as the 4th V for characterizing big data. The reality is that there exists big ambiguity, incompleteness, the uncertainty of big data (Kumar 2015). This might be the reason why vagueness has been considered as one of the ten challenges of big data (Borne, 2014). Accuracy and reliability are less controllable for many forms of big data, for example, Twitter posts several tweets with hashtags, abbreviations, typos, and colloquial speech. Thus, big veracity is a characteristic of big data, this is particularly true in big data analytics for business decision-making (Sun et al., 2016

5. Big Value: Big value indicates the importance and context of big data (Vajjhala *et al*., 2016). It characterizes the big business value, ROI, and potential of big data to transform an organization to have more competitiveness in the global platform (Borne, 2014). Big data has extremely big value for increasing productivity, efficiency, and revenues, lowering costs, and reducing risk in

businesses and management (Wang, 2012). For example, big data and big data analytics have brought big value for Facebook, Google, Amazon and made them become the top companies in the world. It has also big value for scientific breakthroughs in science and technology that have given and will give us a big number of opportunities to make great progress in many fields. For example, many medical studies are based on big data to find better solutions to cure diabetes and hypertension (Minelli *et al*., 2013).

Big value also implies that big data brings big social value. The big social value means that big data has been revolutionizing society in terms of working, living, and thinking (Mayer-Schoenberger & Cukier 2013). The thinking methods have been changed because of big data, for example, learning heavily relies on books, library, schools, newspapers in the past whereas in the big data age, learning is based on search, learning as a search (LaaS), I learn using LaaS whenever and wherever I am.

2.1.2 **Big data analytics**

Big analytics is a brief representation of big data analytics or big data-based analytics or big data-driven analytics (Minelli *et al*., 2013). Big analytics can be defined as the process of collecting, organizing, and analyzing big data to discover and visualize patterns, knowledge, and intelligence as well as other information within the big data for supporting decision making. The main components of big analytics include big descriptive analytics, big predictive analytics, and big prescriptive analytics (Minelli *et al*., 2013).

1. Big data descriptive analytics (Sun et al., 2016): It addresses the problems such as what happened, and when, as well as what is happening. For example, web analytics for pay-per-click or email marketing data belongs to big data descriptive analytics.
2. Big data predictive analytics (Sun *et al*., 2016): It focuses on forecasting trends by addressing the problems such as what will happen, what's going to happen, what is likely to happen, and why it will happen (Delena *et al*., 2013). For example, big data predictive analytics can be used to predict where might be the next attack target of terrorists.

3. Big data prescriptive analytics (Sun *et al*, 2013): it addresses the problems such as what we should do, why we should do it, and what should happen with the best outcome under uncertainty (Minelli *et al*, 2013). For example, big data prescriptive analytics can be used to provide an optimal marketing strategy for an e-commerce company (Sun *et al*., 2016).

2.1.3 **Big Service**

Services are playing a pivotal role in nations, organizations, businesses, and individuals, service computing, social computing, and cloud computing. Web services have been changing our work, life, and thinking with the healthy development of a service-centered society (Sun, 2016). Big services are big data-driven services or big data based on services. The big in big service means that it can provide big services to at least hundreds of millions of people. For example, big data infrastructure services, cloud services, mobile services, big analytics services (Sun, and Yearwood, 2016) social networking services are big services.

Big services, the web of services (Sun *et al*., 2016), and the Internet of services are interchangeable. The internet of services (IoS) is a big service covering all the services on the Internet-based on big data. For example, online storage services are a kind of big service (Huang, Smith, and Sun, 2014). Big services are an emerging frontier for innovations, competencies, and improving the business performance of governments, organizations, and enterprises.

2.1.4 **Big Data Analysis Frameworks and Platforms**

Various solutions have been presented for big data analytics which can be divided into

1. Processing/Compute: Apache Hadoop, Nvidia Cuda, or Twitter Storm, MongoDB, Cassandra, Zookeeper, Kafka, Sqoop, Apache Spark, ScrapeStorm, etc.
2. Storage: Hadoop Distributed File System (HDFS), and
3. Analytics: Zookeeper, Mahout, etc. “Most of the studies on the traditional data analysis are focused on the design and development of efficient and/or effective “ways” to find the useful things from the data. But when we enter the age of big data, most of the current computer

systems will not be able to handle the whole dataset all at once; thus, how to design a good data analytics framework or platform and how to design analysis methods are both important things for the data analysis process. Hadoop ecosystem and their components are shown in Figure 2.2.

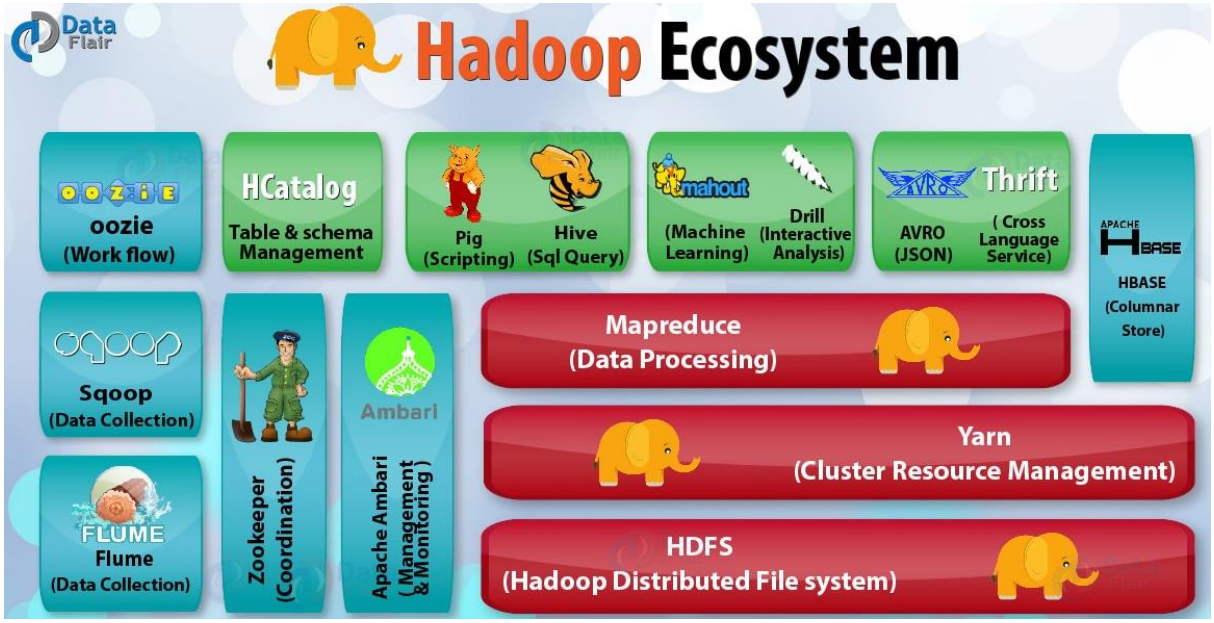

Figure 2.2: Hadoop ecosystem and their components (Source: Premkumar, 2019)

2.1.5 **Apache Hadoop**

Apache Hadoop is an open-source framework which is created by Doug Cutting and Mike Caferella in 2005 which is named after a toy elephant (Ishwarappa *et al*., 2015). Hadoop is initially written in Java (Pence, 2015) and it uses distributed processing through enormous clusters of computers (Ishwarappa *et al*., 2015). Hadoop has the capability of parallel processing of huge data sets, which results in scalable computing. Apache Hadoop is comprised of two major layers: Hadoop distributed file system (HDFS) and MapReduce. In fact, Apache Hadoop is a framework to implement MapReduce programming model (Rehan, and Gangodkar, 2018). Rather than relying on high-end hardware, the resiliency of these clusters comes from the software's ability to detect and handle failures at the application layer.

In addition, Hadoop's cost advantages over legacy systems redefine the economics of data, Legacy systems, while fine for certain workloads simply were not engineered with the needs of Big Data in mind and are far too expensive to be used for general purpose with today's largest data sets. Apache Hadoop has two main subprojects:

i. MapReduce - The framework that understands and assigns work to the nodes in a cluster. Has been defined by Google in 2004 and is able to distribute data workloads across thousands of nodes. It is based on "break problem up into smaller sub-problems" strategy and can be exposed via SQL and in SQL-based BI tools (Ankam, 2016);
ii. Hadoop Distributed File System (HDFS) - An Apache open source distributed file system that spans all the nodes in a Hadoop cluster for data storage; It links together the file systems on many local nodes to make them into one big file system. HDFS assumes nodes will fail, so it achieves reliability by replicating data across multiple nodes".

2.1.5.1 **Hadoop Distributed File System**

HDFS is expected to run on high-performance commodity hardware; it is known for highly scalable storage and automatic data replication across three nodes for fault tolerance. Furthermore, automatic data replication across three nodes eliminates need for backup (write once, read many times) (Ankam, 2016).

Hadoop is supplemented by an ecosystem of Apache projects, such as Pig, Hive and Zoo keeper that extend the value of Hadoop and improve its usability. Due to the cost-effectiveness, scalability and streamlined architectures, Hadoop changes the economics and the dynamics of large-scale computing, having a remarkable influence based on four salient characteristics. Hadoop enables a computing solution that is:

i. Scalable: New nodes can be added as needed and added without needing to change data formats, how data is loaded, how jobs are written, or the applications on top.

ii. Cost effective: Hadoop brings massively parallel computing to commodity servers. The result is a sizeable decrease in the cost per terabyte of storage, which in turn makes it affordable to model all your data.

iii. Flexible: Hadoop is schema-less, and can absorb any type of data, structured or not, from any number of sources. Data from multiple sources can be joined and aggregated in arbitrary ways enabling deeper analyses than any one system can provide.

iv. Fault tolerant: When a node is lost, the system redirects work to another location of the data and continues processing without missing a beat" (Elena, Florina and Anca, 2012; Rehan *et al*., 2018).

"Text mining makes sense of text-rich information such as insurance claims, warranty claims, customer surveys, or the growing streams of customer comments on social networks.

Optimization helps retailers and consumer goods makers, among others, with tasks such as setting prices for the best possible balance of strong-yet-profitable sales. Forecasting is used by insurance companies, for example, to estimate exposure or losses in the event of a hurricane or flood.

Cost will certainly be a software selection factor as that's a big reason companies are adopting Hadoop; they're trying to retain and make use of all their data, and they're expecting cost savings over conventional relational databases when scaling out over hundreds of Terabytes or more. Sears, for example, has more than 2 petabytes of data on hand, and until it implemented Hadoop two years ago, it was observed that the company was constantly outgrowing databases and still couldn't store everything on one platform. Once the application can run on Hadoop it will presumably be able to handle projects with even bigger and more varied data sets, and users will be able to quickly analyze new data sets without the delays associated with transforming data to meet a rigid, predefined data model as required in relational environments.

From architectural point of view, Hadoop consists of the Hadoop Common which provides access to the filesystems supported by Hadoop. The Hadoop Common package contains the necessary JAR

files and scripts needed to start Hadoop. The package also provides source code, documentation, and a contribution section which includes projects from the Hadoop Community.

For effective scheduling of work, every Hadoop-compatible filesystem should provide location awareness: the name of the rack (more precisely, of the network switch) where a worker node is. Hadoop applications can use this information to run work on the node where the data is, and, failing that, on the same rack/switch, reducing backbone traffic. The Hadoop Distributed File System (HDFS) uses this when replicating data, to try to keep different copies of the data on different racks. The goal is to reduce the impact of a rack power outage or switch failure so that even if these events occur, the data may still be readable" (Elena et al, 2012; Rehan *et al*., 2018).

2.1.6 **Gaps Between Big Data Analysis Frame Works and Platform**

Big data analytics can be time-consuming, complicated, and computationally demanding, without the proper tools, frameworks, and techniques. When the volume of data is too high to process and analyze on a single machine, Apache Spark and Apache Hadoop can simplify the task through parallel processing and distributed processing (Balazinska and Dan 2016). To understand the need for parallel processing and distributed processing in big data analytics, it is important to first understand what "big data" is. The high-velocity at which big data is generated requires that the data also be processed very quickly and the variety of big data means it contains various types of data, including structured, semi-structured, and unstructured data (Balazinska *et al*, 2016). The volume, velocity, and variety of big data calls for new, innovative techniques and frameworks for collecting, storing, and processing the data, which is why Apache Hadoop and Apache Spark were created.

2.1.6.1 **Parallel Processing vs. Distributed Processing**

Understanding what parallel processing and distributed processing will help to understand how Apache Hadoop and Apache Spark are used in big data analytics. Since both parallel processing and distributed processing both involve breaking up computing into smaller parts, there can be confusion between the two. The difference between parallel computing and distributed computing is in the

memory architecture. Parallel computing is the simultaneous use of more than one processor to solve a problem (Pierfederici, 2016).

Distributed computing is the simultaneous use of more than one computer to solve a problem" (Pierfederici, 2016). Parallel computing tasks access the same memory space, while distributed computing tasks don't since distributed computing is disk-based, instead of memory-based. Some distributed computing tasks run on one computer, and some on others (Pusukuri, 2019).

2.1.6.2 **Apache Spark vs. Apache Hadoop**

Apache Hadoop and Apache Spark are both open-source frameworks for big data processing with some key differences. Hadoop uses the MapReduce to process data, while Spark uses resilient distributed datasets (RDDs). Hadoop has a distributed file system (HDFS), meaning that data files can be stored across multiple machines. The file system is scalable, since servers and machines can be added to accommodate increasing volumes of data (Ankam, 2016). Spark does not provide a distributed file storage system, so it is mainly used for computation, on top of Hadoop. Spark does not need Hadoop to run, but can be used with Hadoop since it can create distributed datasets from files stored in the HDFS (Ankam, 2016).

2.1.6.3 **Performance Differences**

A key difference between Hadoop and Spark is performance. Researchers from University California Berkeley realized Hadoop is great for batch processing, but inefficient for iterative processing, so they created Spark to fix this (Ankam, 2016). Spark programs iteratively run about 100 times faster than Hadoop in-memory, and 10 times faster on disk. Spark's in-memory processing is responsible for Spark's speed. Hadoop MapReduce, instead, writes data to a disk that is read on the next iteration. Since data is reloaded from the disk after every iteration, it is significantly slower than Spark (Hari, 2019).

2.1.6.4 **Apache Spark Resilient Distributed Dataset**

Apache Spark's fundamental data structure, the Resilient Distributed Dataset (RDD), is a fault-tolerant, immutable, and distributed collection of objects that can be processed in parallel across a cluster. RDDs are resilient due to their fault tolerance, which means that it has recovery from node failure. In the event that a node fails to complete its task, the RDD can be automatically reconstructed on the remaining nodes to complete the task (Dua, Ghotra and Pentreath, 2017). RDDs' immutability helps to prevent and avoid data inconsistency, even when performing transformations on unstructured data (Pusukuri, 2019). Big data is often variable and can include text data, relational databases, videos, audio, photos, and graph data. RDD's make it possible to process both structured and unstructured data efficiently. DataFrames, which were built on top of RDDs, are also immutable, distributed collections of data, but organized into columns and rows, similar to a relational database (Drabas and Lee, 2017). Spark DataFrames make it even easier to process and analyze data from a variety of sources and formats, such as JSON files and Hive tables (Ankam, 2016).

2.1.7 **Spark Streaming**

Apache Spark's stack includes multiple frameworks for big data processing, such as Spark Streaming. The Spark Streaming framework is for stream processing fault-tolerant, live data streams to handle big data's velocity (Ignacio, 2017). Upon receiving live input data, Spark Streaming divides the data into batches, so that the data is processed in batches, and the final result is a stream of processed batches (Drabas *et al*., 2017). Spark Streaming data is known as Discretized Streams (DStreams), which essentially are sequences of RDDs (Dua *et al*., 2017). DStreams are consistent and fault tolerant. Big data is valuable to businesses and decision-makers due to its ability to be processed in real-time or near real-time. An example use-case of Spark Streaming is an investment firm needing to monitor social media trends that could impact their investments and holdings. Staying up to date on what is

being said about the firm's clients or companies they have holdings in can greatly impact business outcomes.

2.1.7.1 **Spark MLlib**

Apache Spark has a machine learning library called MLlib that provides the major machine learning algorithms such as classification, clustering, regression, dimensionality reduction, transformers, and collaborative filtering (Hari, 2019). Some machine learning algorithms can be applied to streaming data, which is useful for applications that use Spark Streaming (Dua *et al*., 2017). An example of a classification algorithm being used on a Spark Streaming application is credit card fraud detection. As soon as a new incoming transaction is processed, the classification algorithm will classify whether or not a transaction is fraud. Identifying potentially fraudulent transactions as soon as they happen is important so that banks can reach out to the customer as soon as possible. The challenging endeavor involved with Spark Streaming is the unpredictability of the volume of the incoming data (Ignacio, 2017). The inconsistent distribution of incoming data over the stream's lifetime can make stream learning for machine learning models rather difficult.

2.1.7.2 **Recommendation Systems**

A popular real-world use for Spark Streaming and MLlib is for building machine learning-based recommendation systems. Recommendations systems have impacted how we shop online, the movies we watch, the songs we listen to, and the news we read. Companies most notable for their recommendation systems include Netflix, YouTube, Amazon, Spotify, and Google. With companies as large as these, generating and collecting as much data as they do, big data analytics is amongst the top of their priorities. MLlib includes an iterative collaborative filtering algorithm, known as Alternating Least Squares to construct recommendation systems (Dua *et al*., 2017). Collaborative filtering systems make recommendations based on similarity measures between users and items

(Ignacio, 2017). Similar items will be recommended to similar users. The system doesn't know details about the items, only that similar users viewed or chose the same item.

2.1.7.3 **Apache Mahout**

MapReduce once had its own machine learning library, however, since MapReduce is inefficient for iterative processing, it quickly lost its compatibility with the library to Apache Spark (Dua *et al*., 2017). Apache Mahout is the machine learning library built on top of Apache Hadoop that started out as a MapReduce package for running machine learning algorithms. Since machine learning algorithms are iterative, MapReduce encountered scalability and iterative processing issues (Dua *et al*., 2017). Because of these issues, Apache Mahout stopped supporting MapReduce-based algorithms, and started supporting other platforms, such as Apache Spark.

2.1.8 **Big Data Input Processing**

The problem of handling a vast quantity of data that the system is unable to process is not a brand-new research issue; in fact, it appeared in several early approaches (Lee, Hong & Lee, 2013), e.g., marketing analysis, network flow monitors, gene expression analysis, weather forecast, and even astronomy analysis. This problem still exists in big data analytics today; thus, preprocessing is an important task to make the computer, platform, and analysis algorithm to be able to handle the input data. The traditional data preprocessing methods (Famili, Shen, Weber & Simoudis, 1997) (e.g., compression, sampling, feature selection, and so on) are expected to be able to operate effectively in the big data age. However, a portion of the literature review in this study will focus on how to reduce the complexity of the input data because even the most advanced computer technology cannot efficiently process the whole input data by using a single machine in most cases. By using domain knowledge to design the preprocessing operator is a possible solution for the big data. In (Zhang, 2013) used the domain knowledge, B-tree, divide-and-conquer to filter the unrelated log information for the mobile web log analysis. Thus, (Dawelbeit & McCrindle, 2014) employed the bin packing

partitioning method to divide the input data between the computing processors to handle these high computations of preprocessing on cloud system. The cloud system is employed to preprocess the raw data and then output the refined data (e.g., data with uniform format) to make it easier for the data analysis method or system to perform the further analysis work.

Sampling and compression are two representative data reduction methods for big data analytics because reducing the size of data makes the data analytics computationally less expensive, thus faster, especially for the data coming to the system rapidly. In addition to making the sampling data represent the original data effectively (Cormode & Duffield, 2014), how many instances need to be selected for data mining method is another research issue (Satyanarayana, 2014) because it will affect the performance of the sampling method in most cases. To avoid the application-level slow-down caused by the compression process, in (Jun, Fleming, Adler and Emer, 2012) attempted to use the FPGA to accelerate the compression process. The I/O performance optimization is another issue for the compression method. For this reason (Zou, Yu, Tang & Chen, 2014) employed the tentative selection and predictive dynamic selection and switched the appropriate compression method from two different strategies to improve the performance of the compression process. To make it possible for the compression method to efficiently compress the data, a promising solution is to apply the clustering method to the input data to divide them into several different groups and then compress these input data according to the clustering information. The compression method described in (Yang, Zhang, Zhong, Liu, Pei, Ramamohanarao, & Chen, 2014) is one of this kind of solutions, it first clusters the input data and then compresses these input data via the clustering results while the study (Xue, Shen, Li, Xu, Zhang, & Shao, 2012) also used clustering method to improve the performance of the compression process.

This implies that to handling the large and fast data input, the research issues of heterogeneous data sources, incomplete data, and noisy data may also affect the performance of the data analysis. The input operators will have a stronger impact on the data analytics at the big data age than it has in the

past. As a result, the design of big data analytics needs to consider how to make these tasks (e.g., data clean, data sampling, and data compression) works well.

2.1.9 **Review of Frameworks and Platforms of Big Data Analysis**

"To date, tools and platforms presented by well-known organizations can easily be found. The cloud computing technologies are widely used on these platforms and frameworks to satisfy the large demands of computing power and storage. Most of the works on Knowledge Discovery in Database (KDD) for big data can be moved to cloud system to speed up the response time or to increase the memory space. With the advance of these works, handling and analyzing big data within a reasonable time has become not so far away. Since the foundation functions to handle and manage the big data were developed gradually; thus, the data scientists nowadays do not have to take care of everything, from the raw data gathering to data analysis, by themselves if they use the existing platforms or technologies to handle and manage the data. The data scientists nowadays can pay more attention to finding out the useful information from the data even thought this task is typically like looking for a needle in a haystack. That is why several recent studies tried to present efficient and effective framework to analyze the big data, especially on find out the useful things (Russom, 2011).

Performance-oriented from the perspective of platform performance, (Huai, and Lee, Zhang, Xia, & Zhang, 2011) pointed out that most of the traditional parallel processing models improve the performance of the system by using a new larger computer system to replace the old computer system, which is usually referred to as scale up. But for the big data analytics, most researches improve the performance of the system by adding more (Russom, 2011) similar computer systems to make it possible for a system to handle all the tasks that cannot be loaded or computed in a single computer system (called "scale out"). To build a scalable and fault-tolerant manager for big data analysis, (Huai *et al*., 2011) presented a matrix model which consists of three matrices for data set (D), concurrent data processing operations (O), and data transformations (T), called DOT. The big data is divided

into n subsets each of which is processed by a computer node (worker) in such a way that all the subsets are processed concurrently, and then the results from these n computer nodes are collected and transformed to a computer node. By using this framework, the whole data analysis framework is composed of several DOT blocks. The system performance can be easily enhanced by adding more DOT blocks to the system. Another efficient big data analytics was presented in, called Generalized Linear Aggregates Distributed Engine (GLADE). The GLADE is a multi-level tree-based data analytics system which consists of two types of computer nodes that are a coordinator and workers. "The simulation results show that the GLADE can provide a better performance than Hadoop in terms of the execution time (Rusu & Dobra, 2012). Because Hadoop requires large memory and storage for data replication and it is a single master, (Essa, Attiya, & El-Sayed, 2013) presented a mobile agent-based framework to solve these two problems, called the Map Reduce Agent Mobility (MRAM). The main reason is that each mobile agent can send its code and data to any other machine; therefore, the whole system will not be down if the master failed. Compared to Hadoop, the architecture of MRAM was changed from client/server to a distributed agent. The load time for MRAM is less than Hadoop even though both of them use the map-reduce solution and Java language. (Wonne, Capobianco, & Bechmann., 2012) considered issues of the user needs and system workloads. They presented a self-tuning analytics system built on Hadoop for big data analysis. Since one of the major goals of their system is to adjust the system based on the user needs and system workloads to provide good performance automatically, the user usually does not need to understand and manipulate the Hadoop system" (Russom, 2011). The study of (Demchenko, Laat & Membery, 2014) was from the perspectives of data centric architecture and operational models to present a Big Data Architecture Framework (BDAF) which includes: big data infrastructure, big data analytics, data structures and models, big data lifecycle management, and big data security.

In the studies of (Demchenko *et al*., 2014) the authors observed that "cluster services, Hadoop related services, data analytics tools, databases, servers, and massively parallel processing databases are

typically the required applications and services in big data analytics infrastructure. Result-oriented (Fisher, DeLine, Czerwinski, & Drucker, 2012) presented a big data pipeline to show the workflow of big data analytics to extract the valuable knowledge from big data, which consists of the acquired data, choosing architecture, shaping data into architecture, coding/debugging, and reflecting works. From the perspectives of statistical computation and data mining, (Ye, Wang, Zhou, Wang, & Zhou., 2013) presented architecture of the services platform which integrates R to provide better data analysis services, called Cloud-Based Big Data Mining and Analyzing Services Platform (CBDMASP). The design of this platform is composed of four layers: the infrastructure services layer, the virtualization layer, the dataset processing layer, and the services layer. Several large-scale clustering problems (the datasets are of size from 0.1 G up to 25.6 G) were also used to evaluate the performance of the CBDMASP. The simulation results show that using map-reduce is much faster than using a single machine when the input data become too large. Although the size of the test dataset cannot be regarded as a big dataset, the performance of the big data analytics using map reduce can be sped up via this kind of testing. Map-reduce is a better solution when the dataset is of size more than 0.2 G, and a single machine is unable to handle a dataset that is of size more than 1.6 G" (Russom, 2011). Furthermore, (Wu, Zhu, Wu, & Ding, 2014) presented a theorem to explain the big data characteristics, called HACE: the characteristics of big data usually are large-volume, Heterogeneous, Autonomous sources with distributed and decentralized control, and we usually try to find out some useful and interesting things from complex and evolving relationships of data. Based on these concerns and data mining issues, (Wu *et al*., 2014) also presented a big data processing framework which includes data accessing and computing tier, data privacy and domain knowledge tier, and big data mining algorithm tier. This work explains that the data mining algorithm will become much more important and much more difficult; thus, challenges will also occur on the design and implementation of big data analytics platform. In addition to the platform performance and data mining issues, the privacy issue for big data analytics was a promising research in recent years.

Furthermore, (Laurila, Gatica-Perez, Aad, Blom, Bornet, Do, Dousse, Eberle, and Miettinen, 2012) explained that the privacy is an essential problem when we try to find something from the data that are gathered from mobile devices; thus, data security and data anonymization should also be considered in analyzing this kind of data. (Demirkan and Delen., 2013) presented a Service-Oriented Decision Support System (SODSS) for big data analytics which includes information source, data management, information management, and operations management.

2.1.10 **Big Data Analytics Algorithms**

(Fan and Bifet, 2013) pointed out that "the terms big data and big data mining were first presented in 1998. Data mining algorithms for data analysis play the vital role in the big data analysis, in terms of the computation cost, memory requirement, and accuracy of the end results. In this section, the perspective of analysis and search algorithms to explain its importance for big data analytics will be discussed brief.

Clustering algorithms in the big data age, traditional clustering algorithms may become even more limited than before because it typically requires that all the data be in the same format and be loaded into the same machine so as to find some useful things from the whole data (Russom, 2011). Although the problem (Chiang, Chiang, Tsai, & Yang, 2011) of analyzing large-scale and high-dimensional dataset has attracted many researchers from various disciplines in the last century. Several solutions have been presented in recent years. The characteristics of big data still brought up several new challenges for the data clustering issues. Among them, how to reduce the data complexity is one of the important issues for big data clustering. (Shirkhorshidi, Aghabozorgi., The, & Herawan, 2014) divided the big data clustering into two categories: single-machine clustering (i.e., sampling and dimension reduction solutions), and multiple-machine clustering (parallel and MapReduce solutions). This means that traditional reduction solutions can also be used in the big data age because the complexity and memory space needed for the process of data analysis will be decreased by using sampling and dimension reduction methods. More precisely, sampling can be regarded as reducing

the “amount of data” entered into a data analyzing process while dimension reduction can be regarded as “downsizing the whole dataset” because irrelevant dimensions will be discarded before the data analyzing process is carried out (Xu, Li, Guo, & Chen, 2012). Xu *et al*. (2012) presented CloudVista, a representative solution for clustering big data which used cloud computing to perform the clustering process in parallel. Sampling method was used in CloudVista to show that it is able to handle large-scale data, e.g., 25 million census records. Using GPU to enhance the performance of a clustering algorithm is another promising solution for big data mining. The Multiple Species Flocking (MSF) was applied to the CUDA platform from NVIDIA to reduce the computation time of clustering algorithm in (Cui, Gao, & Potok, 2006). The simulation results show that the speedup factor can be increased from 30 up to 60 by using GPU for data clustering. Since most traditional clustering algorithms (e.g, k-means) require a computation that is centralized, how to make them capable of handling big data clustering problems is the major concern of (Feldman, Schmidt, & Sohler, 2013) used a tree construction for generating the core sets in parallel which is called the merge-and-reduce approach. Moreover, (Feldman et al., 2013) pointed out that by using this solution for clustering, the update time per datum and memory of the traditional clustering algorithms can be significantly reduced. Classification algorithms Similar to the clustering algorithm for big data mining, several studies also attempted to modify the traditional classification algorithms to make them work on a parallel computing environment or to develop new classification algorithms which work naturally on a parallel computing environment (Russom, 2011). The work done by (Tekin & Schaar, 2013) which designed a classification algorithm took into account the input data that are gathered by distributed data sources which will be processed by a heterogeneous set of learners. The study of (Tekin *et al*., 2013) presented a novel classification algorithm called “Classify or Send for Classification” (CoS). They assumed that each learner can be used to process the input data in two different ways in a distributed data classification system. One is to perform a classification function by itself while the other is to forward the input data to another learner to have them labeled. The information will be

exchanged between different learners. In brief, this kind of solutions can be regarded as a cooperative learning to improve the accuracy in solving the big data classification problem. An interesting solution uses the quantum computing to reduce the memory space and computing cost of a classification algorithm. For example, in the work of (Rebentrost Mohseni, & Lloyd, 2014) that presented a quantum-based support vector machine for big data classification and argued that the classification algorithm proposed can be implemented with a time complexity O(logNM) where N is the number of dimensions and M is the number of training data. There are bright prospects for big data mining by using quantum-based search algorithm when the hardware of quantum computing has become mature. Frequent pattern mining algorithms most of the researches on frequent pattern mining (i.e., association rules and sequential pattern mining) were focused on handling large-scale dataset at the very beginning because some early approaches of them were attempted to analyze the data from the transaction data of large shopping mall. Because the number of transactions usually is more than tens of thousands, the issues about how to handle the large-scale data were studied for several years, such as FP-tree. Han, Pei, & Yin (2002) used the tree structure to include the frequent patterns to further reduce the computation time of association rule mining. In addition to the traditional frequent pattern mining algorithms, of course, parallel computing and cloud computing technologies have also attracted researchers in this research domain. Among them, the map-reduce solution was used for the studies (Lin, Lee & Hsueh, 2012; Leung, MacKinnon, & Jiang, 2014) to enhance the performance of the frequent pattern mining algorithm. By using the map-reduce model for frequent pattern mining algorithm, it can be easily expected that its application to "cloud platform" will definitely become a popular trend in the forthcoming future. The study of (Leung et al., 2012) did not only used the map-reduce model, it also allowed users to express their specific interest constraints in the process of frequent pattern mining. The performance of these methods by using map-reduce model for big data analysis is, no doubt, better than the traditional frequent pattern mining algorithms running on a single machine.

2.1.10.1 **Machine Learning for Big Data Mining**

The potential of machine learning for data analytics can be easily found in the early literature (Mitra, Pal, & Mitra, 2002, p. 4; Xu & Wunsch, 2009). Different from the data mining algorithm design for specific problems, machine learning algorithms can be used for different mining and analysis problems because they are typically employed as the "search" algorithm of the required solution. Since most machine learning algorithms can be used to find an approximate solution for the optimization problem, it can be employed for most data analysis problems if the data analysis problems can be formulated as an optimization problem.

For example, genetic algorithm, one of the machine learning algorithms, can not only be used to solve the clustering problem (Krishna & Murty, 1999), it can also be used to solve the frequent pattern mining problem (Kaya & Alhaji, 2005). The potential of machine learning is not merely for solving different mining problems in data analysis operator of KDD; it also has the potential of enhancing the performance of the other parts of KDD, such as feature reduction for the input operators (Russom, 2011).

A recent study (Ma, Zhang, & Wang, 2014) shows that some traditional mining algorithms, statistical methods, preprocessing solutions, and even the Graphical User Interface (GUI)'s have been applied to several representative tools and platforms for big data analytics. The results show clearly that machine learning algorithms will be one of the essential parts of big data analytics. One of the problems in using current machine learning methods for big data analytics is similar to those of most traditional data mining algorithms which are designed for sequential or centralized computing. However, one of the most possible solutions is to make them work for parallel computing. Fortunately, some of the machine learning algorithms (e.g. population-based algorithms) can essentially be used for parallel computing, which have been demonstrated for several years, such as parallel computing version of genetic algorithm. Different from the traditional GA, the population of island model genetic algorithm, one of the parallel GA's, can be divided into several subpopulations.

This means that the sub-populations can be assigned to different threads or computer nodes for parallel computing, by a simple modification of the GA (Ma *et al*., 2014). For this reason, in work of (Kranthi & Babu, 2014) which explained that the framework for distributed data mining algorithm still needs to aggregate the information from different computer nodes. The common design of distributed data mining algorithm is as follows: each mining algorithm can be performed on a computer node (worker) which has its locally coherent data, but not the whole data. To construct a globally meaningful knowledge after each mining algorithm finds its local model, the local model from each computer node has to be aggregated and integrated into a final model to represent the complete knowledge". Kranthi *et al*., (2014) posed that the communication will be the bottleneck when using this kind of distributed computing framework.

Borkar, Carey, Rosen, Polyzotis, & Condie (2012) found some research issues when trying to apply machine learning algorithms to parallel computing platforms. For instance, the early version of map-reduce framework does not support "iteration". But the good news is that some recent works (Ku-Mahamud, 2014) have paid close attention to this problem and tried to fix it. Similar to the solutions for enhancing the performance of the traditional data mining algorithms, one of the possible solutions to enhancing the performance of a machine learning algorithm is to use CUDA (CUDA: is a parallel computing platform and programming model developed by NVIDIA for general computing on Graphical Processing Units (GPU)), i.e., a GPU, to reduce the computing time of data analysis (p. 10). Hasan, Shamsuddin & Lope (2013) used CUDA to implement the Self-Organizing Map (SOM) and Multiple Back-Propagation (MBP) for the classification problem. The simulation results show that using GPU is faster than using Central Processing Unit (CPU). More precisely, SOM running on a GPU is three times faster than SOM running on a CPU, and MPB running on a GPU is twenty-seven times faster than MPB running. In another view of reasoning (Ku-Mahamud, 2014) attempted to apply the ant-based algorithm to grid computing platform. Since the proposed mining algorithm is extended by the ant clustering algorithm of (Deneubourg, Goss, Franks, N., Sendova-Franks,

Detrainl, & Chretien, 1990). Ku-Mahamud, (2014) modified the ant behavior of this ant clustering algorithm for big data clustering. That is, each ant will be randomly placed on the grid. This means that the ant clustering algorithm then can be used on a parallel computing environment. The trends of machine learning studies for big data analytics can be divided into twofold: one attempts to make machine learning algorithms run on parallel platforms, such as Radoop" , Mahout and software (Bu, Carey, Rosen, Polyzotis, & Condie, 2012); "the other is to redesign the machine learning algorithms to make them suitable for parallel computing or to parallel computing environment, such as neural network algorithms for GPU" (Hasan et al., 2013) and ant-based algorithm for grid (Ku-Mahamud *et al*., 2013). In summary, both of them make it possible to apply the machine learning algorithms to big data analytics although still many research issues need to be solved, such as the communication cost for different computer nodes and the large computation cost most machine learning algorithms require.

2.1.11 **Trends in Global Terrorism**

The unstructured dataset of Boko Haram insurgency menace attacks in Nigeria were selected as the big dataset for the real-world dataset for the analysis of the proposed Bigdata Analytic model called Dynamic-K-reference Clustering Algorithm. Trends of terrorist attacks in the world particular in Nigeria were reviewed to guide the study on the elements of the unstructured datasets of Boko Haram scraped from social media. In 2018, (Start, 2019) there were more than 9,600 terrorist attacks around the world, which killed more than 22,980 people, including 7,290 perpetrators and 15,690 victims. The patterns described here provide a general overview. Start, (2019) "encourage readers to explore the Global Terrorism DatabaseTM and consider contextual information for a comprehensive assessment. The information are as follows:

1. 2018 was the fourth consecutive year of declining global terrorism since terrorist violence peaked in 2014 at nearly 17,000 attacks and more than 45,000 total deaths. The total number of terrorist

attacks worldwide decreased 43% between 2014 and 2018, and the total number of deaths decreased 48%. Regional trends varied substantially (Start, 2019).

2. Global statistical trends were heavily impacted by patterns of terrorism In Iraq, which suffered more terrorist attacks than any other country each year from 2013 to 2017. The number of terrorist attacks in Iraq decreased 46% between 2017 and 2018 and the number of people killed in terrorist attacks decreased 78% (Gannon, 2018; Start, 2019).
3. In particular, from the declaration of the caliphate in June 2014 to the liberation of Mosul in July 2017, Islamic State carried out more than 100 terrorist attacks in Iraq and killed more than 500 victims each month, on average. The group's violence in Iraq subsequently decreased dramatically and continued to decline throughout 2018 and early 2019, but remained deadly. More than 650 Islamic State attacks in Iraq killed more than 800 victims and 300 perpetrators in 2018, and caused hundreds of additional casualties in Syria and other locations (Gannon, 2018).

While Islamic State declined in Iraq, the group's influence continued to expand geographically. Attacks carried out by Islamic State "core" operatives, affiliated organizations, or unaffiliated individuals who indicated allegiance to the group took place in 34 countries in 2018, bringing the total number of countries that have ever experienced Islamic State-related terrorist attacks to 53. Preliminary data for 2019 indicate that at least three additional countries: Democratic Republic of Congo, Mozambique, and the Netherlands experienced Islamic State-related attacks, bringing the total number of countries to 56" (Start, 2019).

In 2015, 2016, and 2017 there were multiple events in Western European countries in which assailants killed more than five people, including mass casualty attacks carried out by jihadists in Paris, Brussels, Nice, Berlin, Manchester, London, and Barcelona. In 2018, there were nine lethal terrorist attacks in Western Europe. The deadliest of these occurred in December, when an assailant who claimed allegiance to Islamic State attacked civilians at a Christmas market in Strasbourg, France,

killing five people and injuring 11 others. The number of terrorist attacks In Western Europe decreased 31% between 2017 and 2018, while the number of deaths decreased 70% (Start, 2019).

In the United States, the number of terrorist attacks remained relatively stable in 2017 and 2018, following an increase from 38 attacks in 2015 to 67 attacks in 2016. Terrorist attacks in the United States killed 45 people (including two perpetrators) in 2018, a 54% decrease compared to the number of deaths in 2017 (largely driven by the October 2017 mass casualty attack in Las Vegas) (Gannon, 2018). A suicide bombing in Balochistan, Pakistan in July was the deadliest single attack in 2018. Assailants targeted an election rally for Siraj Raisani, who was among the 150 people killed in the attack. More than 180 others were injured. Islamic State-Khorasan claimed responsibility for the attack, and authorities identified the bomber as an Islamic State operative (Gannon, 2018; Start, 2019).

The deadliest series of attacks in 2018 occurred in Bandundu, Democratic Republic of Congo in December. Members of the Batende tribe attacked the Banunu community in four towns in Yumbi territory. The exceptionally violent attacks which occurred in advance of an intensely disputed national election and contributed to the delay and suppression of voting were a sudden escalation in a longstanding ethnic rivalry over territory and resources. Some reports indicated that police and military personnel may have been among the attackers. The assailants killed at least 540 people; tactics included arson, decapitation, drowning, maiming, and sexual violence (Miles, 2019; Start, 2019). The Terrorist Attacks and Total Deaths Worldwide, by months, 2010-2019(Q1) is represented in Figure 2.3.

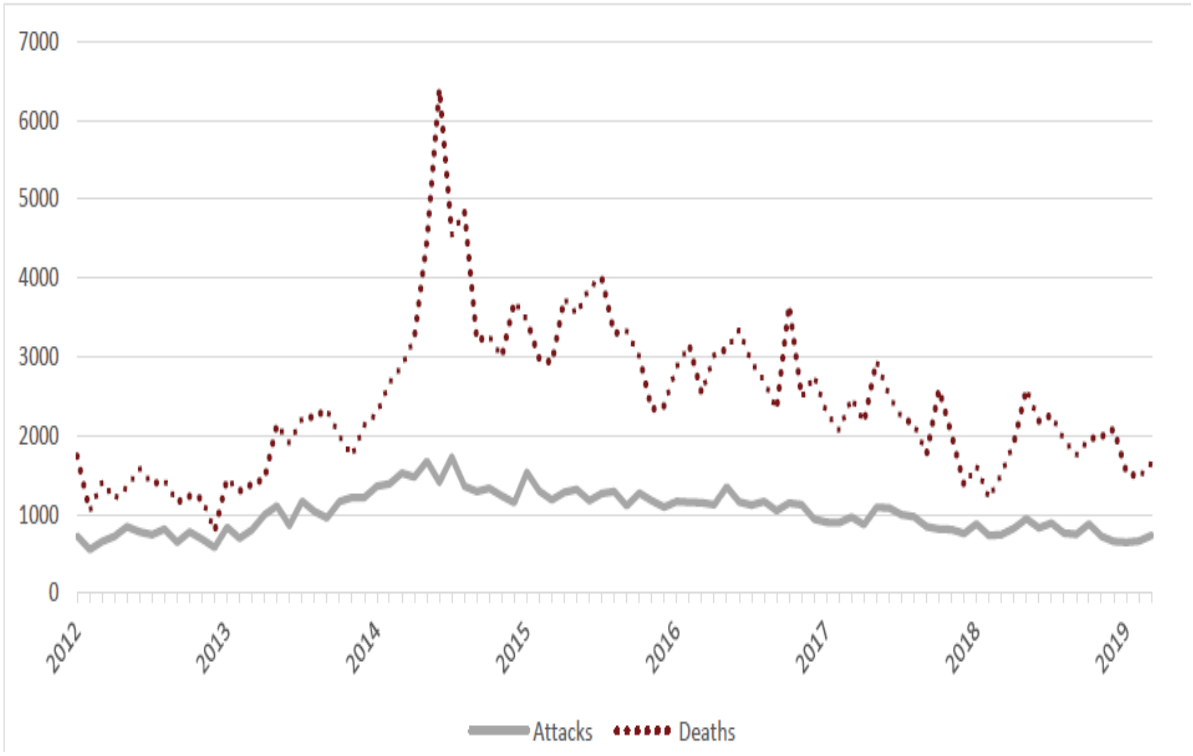

Figure 2.3: Terrorist Attacks and Total Deaths Worldwide, by months, 2010-2019(Q1)
(Source: Start, 2019)

2.1.11.1 **Key Regional Developments of Terrorist Violence**

Terrorist violence remained heavily concentrated in certain locations and coincided with other types of political violence. More than half of all attacks took place in five countries: Afghanistan (18%), Iraq (14%), India (9%), Nigeria (7%), and the Philippines (6%). More than half of all deaths took place in two countries: Afghanistan (43%), and Nigeria (11%) (Start, 2019). The terrorist attacks/deaths, and countries with more than 150 Attacks in 2018 are shown in Table 2.1.

Table 2.1: ***Terrorist Attacks and Deaths, Countries with more than 150 Attacks, 2018***

Country	Total Attacks	% of Total	% Change from 2017	Total Killed*	% of Total	% Change from 2017
Afghanistan	1776	18%	26%	9812	43%	61%
Iraq	1362	14%	-46%	1432	6%	-78%
India	888	9%	-8%	412	2%	-11%
Nigeria	645	7%	33%	2574	11%	43%
Philippines	601	6%	-13%	440	2%	-11%
Somalia	527	5%	-14%	1144	5%	-40%
Pakistan	480	5%	-33%	697	3%	-35%
Yemen	325	3%	43%	829	4%	9%
Cameroon	235	2%	114%	296	1%	21%
Syria	232	2%	-6%	1547	7%	-24%
Colombia	205	2%	72%	132	1%	57%
Thailand	182	2%	2%	69	0%	-4%
Libya	166	2%	-13%	244	1%	-16%
Mali	164	2%	15%	584	3%	61%
Democratic Republic of the Congo	163	2%	14%	993	4%	61%
Worldwide Total	9607	2%	-13%	22987	100%	-13%

(Source: Start, 2019).

The other regions are as follows:

1. Egypt, where the number of terrorist attacks decreased 76% (to 54 in 2018) and the number of deaths decreased 89% (to 98 in 2018) (Bhandari, and Schultz, 2017, Moreno and Rios, 2019).
2. Nepal, where the number of terrorist attacks decreased 60% (to 99 in 2018) and the number of deaths decreased 100% (from 4 in 2017 to 0 in 2018). This followed an isolated increase in mostly non-lethal attacks surrounding critical elections in 2017.5
3. Iraq, where the number of terrorist attacks decreased 46% (to 1,362 in 2018) and the number of deaths decreased 78% (to 1,432 in 2018) (Start, 2019).

Several locations experienced especially large increases in terrorist violence between 2017 and 2018. These include:

1. Cameroon, where the number of terrorist attacks increased 114% (to 235 in 2018), and the number of deaths increased 21% (to 296 in 2018). Violence erupted after Anglophone separatists declared independence in 2017 (Moreno et al., 2019; Start, 2019).
2. Colombia, where the number of terrorist attacks increased 72% (to 205 in 2018) and the number of deaths increased 57% (to 132 in 2018) as the National Liberation Army (ELN) strengthened and peace negotiated between the government and the Revolutionary Armed Forces of Colombia in 2016 faltered (Bhandari et al., 2017; Start, 2019).
3. Saudi Arabia, where the number of terrorist attacks increased 70% (to 92 in 2018), although the number of deaths decreased 45% (to 17 in 2018) (Start, 2019).

2.1.11.2 **Perpetrators**

Information on the perpetrator of the attack was reported for 64% of all attacks worldwide in 2017. In 90 attacks, the individual perpetrator or perpetrators were identified, but they were not known to be affiliated with a particular group or organization. This represents a decline in the number of attacks carried out by unaffiliated individuals, down from 102 in 2017. The lethality of attacks carried out by unaffiliated individuals also decreased, from more than 200 victims killed in 2016 to 85 in 2018 (Start, 2019).

In 2018, 320 groups and organizations carried out terrorist attacks worldwide, fewer than the 372 groups and organizations identified as perpetrators of terrorist attacks in 2017. The perpetrator groups responsible for the most attacks in 2018 are shown in the table 2.2.

Table 2.2: ***Perpetrator Group Responsible for More Than 100 Terrorist Attacks, 2018***

Perpetrator Group	Total Attacks	% Change from 2017	Total Killed*	% Change from 2017
Taliban	1266	40%	8508	73%
Islamic State of Iraq and the Levant (ISIL)	735	-45%	2221	-69%
Al-Shabaab	493	-14%	1149	-39%
Fulani extremists	304	285%	1188	-6%
New People's Army (NPA)	283	-22%	188	-6%
Maoists/Communist Part of India – Maoist (CPI-Maoist)	268	-15%	189	-15%
Houthi extremists (Ansar Allah)	267	68%	659	48%
Boko Haram	243	-29%	1327	-16%
Islamic State-Khorasan Province	155	-21%	1203	-8%
Kurdistan Workers' Party (PKK)	122	-23%	136	-28%
National Liberation Army of Colombia (ELN)	121	95%	106	126%
Separatists (Cameroon)	112	1767%	150	1150%

(Source: Start, 2019).

The Taliban in Afghanistan was responsible for more terrorist attacks in 2018 than any other group by a wide margin, and those attacks resulted in more deaths than the next seven deadliest perpetrator groups combined. Between 2017 and 2018, the number of terrorist attacks carried out by the Taliban increased 40% and the total number of deaths increased 73% (Start, 2019).

"Aside from the Taliban, several of the perpetrator groups that significantly increased their terrorist violence in 2018 were more loosely organized or decentralized movements rather than formally structured organizations. These included Fulani extremists active primarily in Nigeria, Houthi extremists in Yemen, and Anglophone separatists in Cameroon. The number of attacks carried out by Fulani extremists increased 285% and the number of deaths increased 245%. Likewise, the number of attacks carried out by Houthis increased 68% and the number of deaths increased 48%. Perhaps most dramatically, violence carried out by Anglophone separatists in Cameroon increased from a few attacks in 2017 to more than 100 attacks and 150 people killed in 2018.

Islamic State and affiliated groups were among those whose terrorist violence decreased in 2018. These include Islamic State of Iraq and the Levant (attacks decreased 45% and deaths decreased

69%), as well as Boko Haram (attacks decreased 29% and deaths decreased 16%), and Islamic State-Khorasan Province (attacks decreased 21% and deaths decreased 8%). However, these groups remained very deadly and their attacks resulted in thousands of casualties in locations around the world. Attacks by Islamic State operatives in Iraq and Syria killed more than 1,500 victims in 2018. The group also claimed responsibility for deadly attacks in Iran, the Philippines, and Tajikistan.

When considering perpetrator group patterns, it is important to note how groups and organizations evolve over time, often breaking into factions and splinters, sharing members, changing names and aliases, locations and personnel, and forming mergers, alliances, and "franchises." As a result, groups are a problematic unit of analysis that can lead to shortsighted inferences. Combining perpetrator organizations, groups, and individuals with shared goals into "movements" helps capture broader, long-term patterns of perpetrator activity. For example, Global Terrorist Database (GTD) researchers organized the perpetrators associated with the Islamic State movement and al Qaida movement for the purpose of comparison" (Miller, 2016; Start, 2019).

The graph Figure 2.4 indicates the number of countries each year that experienced a terrorist attack carried out by a group or individual affiliated with the broader al Qaida movement (in gray) or Islamic State movement (in red), as well as the cumulative number of countries each movement has impacted.

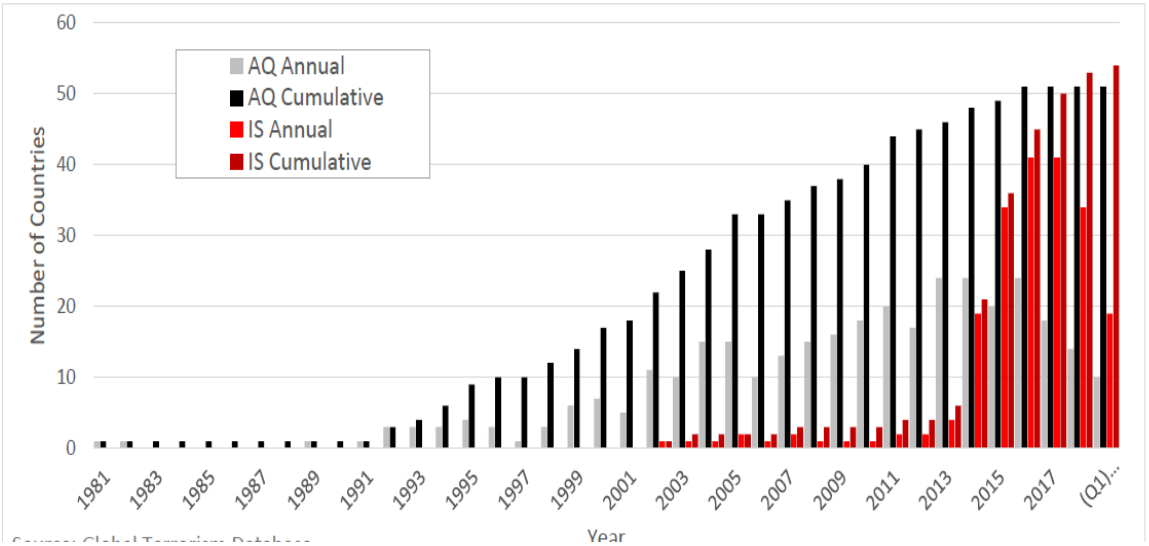

Figure 2.4: Geographic Reach of Qaida and Islamic State-Related Terrorist, 1981 – 2019 (Q1). (Source: Start, 2019)

The fact that Islamic State evolved from al Qaida in Iraq partly explains their more rapid expansion, but we also note that the number of countries impacted by Islamic State-related terrorism continued to expand beyond the dissolution of the caliphate. As violence perpetrated by Islamic State operatives decreased in Iraq, the broader movement continued to expand geographically. Attacks carried out by Islamic State predecessors, "core" operatives, affiliated organizations, or unaffiliated individuals who indicated allegiance to Islamic State took place in 34 countries in 2018, bringing the total number of countries that have ever experienced Islamic State-related terrorist attacks to 53. At least three additional countries—Democratic Republic of Congo, Mozambique, and the Netherlands—experienced Islamic State-related attacks in 2019, bringing the total number of countries to 56 (Start, 2019).

2.1.11.3 **Terrorist Attacks and Total Deaths in Nigeria, By Month, 2012 – 2019 (Q1)**

The number of terrorist attacks and the number of people killed in terrorist attacks in Nigeria increased in 2018 for the first time in four years, following a steady decline from extremely high levels of terrorist violence in 2014 (Start, 2019).

In 2018, there were 645 terrorist attacks in Nigeria, a 33% increase from 2017. There were 2,574 victims and perpetrators killed in these attacks, a 43% increase from 2017. However, most of this increase was driven by victim deaths, which increased 63% from 2017. In 2018, 291 perpetrators were killed in attacks in Nigeria, down from 422 the previous year. Unlike Afghanistan and Iraq, where terrorist conflicts have been dominated by single perpetrator organizations, there are several regional conflicts in Nigeria. Perpetrators were identified in 81% of attacks in Nigeria in 2018 such as:

1. Fulani extremists-active in 15 states but most frequently in Benue, Plateau, and Taraba—continued to engage in violent conflict over land resources. While not a formal organization, Fulani extremists were the most active and deadly perpetrators of terrorism in Nigeria in 2018. They

carried out 299 attacks that killed 1,162 people in Nigeria, in addition to attacks in Central African Republic, Ghana, and Mali (Start, 2019).

2. Boko Haram—including the Shekau faction and al-Barnawi's Islamic State-West Africa Province—was active in six Nigerian states in 2018, most frequently in Borno.11 Boko Haram carried out more than 200 attacks, resulting in 1,095 deaths (including 254 perpetrator deaths), as well as deadly attacks in Cameroon, Chad, and Niger (Start, 2019).
3. Various groups have long engaged in violence rooted in conflict over natural resources in the Niger Delta region. Although there was a wave of attacks by Niger Delta extremist groups in 2016, and two attacks in 2017, none were reported in 2018 following talks between President Buhari and the militants.12 Niger Delta extremists have threatened to resume violence on multiple occasions in 2019, but to date they have not done so (Onyibe, 2019).

Patterns of terrorist tactics also shifted in 2018. While the number of attacks overall increased, they were most frequently armed assaults, a tactic used by both Fulani extremists and Boko Haram. The Patterns of terrorist tactics includes:

i. Bombing: The number of bombings, a tactic not typically used by Fulani militants, but usually the work of Boko Haram or Niger Delta extremist groups, declined sharply, from more than 150 in 2017 to fewer than 90 in 2018. Despite this decline, several hundred people were killed in bombing attacks in Nigeria in 2018 (Start, 2019).
ii. Kidnapping or held hostage: The number of people kidnapped or held hostage by Boko Haram in Nigeria nearly doubled from 2017 (174 victims) to 2018 (346 victims). In particular, Boko Haram's pattern of large-scale kidnappings continued in 2018. Among the attacks with the most kidnap victims were one in February in which more than 100 people, mostly students, were abducted from the Government Girls Science and Technology College in Dapchi. Five students died, and nearly all of the remaining students were released the following month. One student from Dapchi, Leah Sharibu, remained in captivity, as do 112 of the students who were kidnapped

by Boko Haram from the Government Girls Secondary School in Chibok in April 2014 (Egbe, 2019).

iii. Across Multiple Locations: Nigeria continued to experience many attacks that were coordinated across multiple locations. In 2018, more than one-third (35%) of all terrorist attacks in Nigeria were part of coordinated events, compared to 17% worldwide. While this tactic was previously a hallmark of Boko Haram, in 2018 more than two-thirds (72%) of the coordinated attacks in Nigeria were carried out by Fulani extremists (Start, 2019).

The terrorist attacks and total dearth in Nigeria by moths from 2012-2019 (Q1) is representation graphical in Figure 2.5.

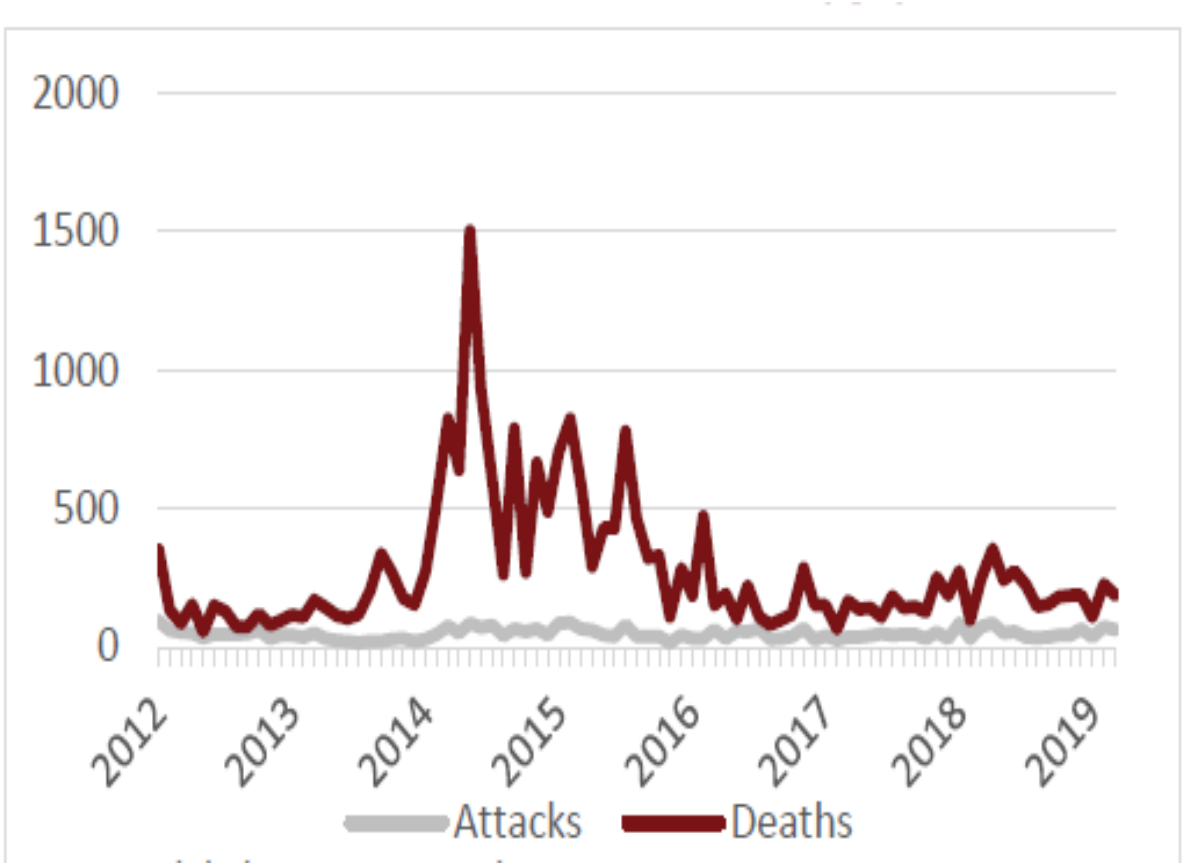

Figure 2.5: Terrorist Attacks and Total Dearth in Nigeria by moths from 2012-2019 (Q1) (Source: Start, 2019)

2.1.11.4 **Statistical Annex Data**

Readers familiar with the U.S. State Department's *Country Reports on Terrorism* may recall that START provided the "Statistical Annex" for that report each year from 2012 through 2017. For

consistency with the definition of terrorism established in the U.S. Code, we used a restricted application of the GTD's inclusion criteria for the analysis in the Statistical Annex.

Despite a productive partnership with our colleagues at the Bureau of Counterterrorism and Countering Violent Extremism, in 2018 the State Department did not award START a contract to continue providing data and analysis for the Statistical Annex. Although the *Country Reports on Terrorism*, which is due to Congress each April, has not yet been published by the State Department for 2018, we recognize that any data analysis it may include would not be consistent or comparable with previous years' analysis produced by START. To support analytical continuity, (Start, 2019) "re-produced several key tables using the restricted Statistical Annex version of the GTD. The following tables 2.3 to 2.6 exclude any attacks that do not meet all three GTD inclusion criteria, and any attacks that were classified as "doubt terrorism proper" by GTD analysts" (Start, 2019). The terrorist attacks and casualties worldwide by Month in 2018 is represented in Table 2.3.

Table 2.3: ***Terrorist Attacks and Casualties Worldwide by Month, 2018***

Month	Total Attacks*	Total Deaths*	Total Injured*	Kidnapped/Hostages
January	689	1231	1563	445
February	576	859	986	405
March	586	1093	1036	198
April	666	1421	1399	233
May	772	2079	1922	318
June	614	1430	1389	251
July	711	1676	2085	335
August	602	1356	870	464
September	599	1337	1317	1222
October	702	1313	1317	1222
November	533	1138	1072	561
December	514	1018	821	466
Total	7564	15951	15757	5477

Note. Reprinted from "Trends in global terrorism: Islamic state's decline in Iraq and expanding global impact; fewer mass casualty attacks in western Europe; number of attacks in the united states highest since 1980s". (Source: Start, 2019).

Ten countries with the most terrorist attacks, 2018 is shown in Table 2.4.

Table 2.4: ***Ten countries with the most terrorist attacks, 2018***

Country	Total Attacks		Total Deaths		Deaths per Attacks*		Total Injured*		Injured per Attack		Kidnapped/ Hostages	
	2018	2017	2018	2017	2018	2017	2018	2017	2018	2017	2018	2017
Afghanistan	1443	1168	7379	4653	5.47	4.21	6514	5010	5.12	4.61	1362	835
Iraq	1130	1988	1054	4348	0.97	2.31	1723	4179	1.62	2.23	203	2005
India	750	867	350	384	0.48	0.45	541	602	0.75	0.72	247	223
Nigeria	562	411	2040	1513	3.98	4.04	772	852	2.03	2.64	445	245
Philippines	424	486	297	326	0.71	0.69	343	297	0.83	0.64	55	409
Somalia	286	372	646	1470	2.55	4.18	638	110	2.63	3.25	238	286
Pakistan	368	576	540	852	1.49	1.51	1018	1830	2.85	3.27	23	107
Yemen	228	142	302	379	1.64	2.94	328	418	1.96	3.45	333	107
Cameroon	203	92	208	214	1.28	2.46	180	238	1.22	2.94	335	77
Colombia	166	102	96	59	0.60	0.58	129	95	0.82	0.94	60	79
Worldwide	7564	8695	15951	18859	2.25	2.28	15757	19624	2.34	2.44	5477	9016

(Source: Start, 2019).

Five perpetrator Group with the Most Attacks Worldwide 2018 is shown in Table 2.5.

Table 2.5: ***Five perpetrator Group with the Most Attacks Worldwide 2018***

Perpetrator Group	Total Attacks		Total Deaths*		Total Injured*		Kidnapped/ Hostages	
	2018	2017	2018	2017	2018	2017	2018	2017
Taliban	987	710	6214	3669	4064	3218	1285	722
Islamic State of Iraq and the Levant (ISIL)**	543	909	1358	4438	1302	3379	909	2225
Fulani extremists	297	76	1168	331	234	60	73	44
Al-Shabaab	260	355	673	1467	591	1066	235	316
Maoists/Communist Part of India- Maoist (CPI-Maoist)	145	295	175	205	154	213	135	124

*including perpetrators

**Excludes attacks attributed to branches of ISIS or ISIS-Inspired Individuals (Source: Start, 2019).

The data presented here are drawn from START's Global Terrorism Database (GTD) and reports from news media. The GTD contains information on more than 190,000 terrorist attacks that occurred around the world since 1970. For more information about the GTD, visit www.start.umd.edu/gtd. Targets of Terrorist attacks worldwide, 2017-2018 is shown in Table 2.6.

Table 2.6: ***Targets of Terrorist attacks worldwide, 2017-2018.***

Target Type	Number of Targets	
	2018	**2017**
Private Citizens & Property	3147	3552
Police	1623	1604
Government (General)	977	932
Business	604	603
Military	385	456
Religious Figures/Institutions	251	231
Educational Institutions	185	177
Terrorists/Non-State Militia	153	161
Transportation	140	151
Utilities	140	159
Journalists & Media	105	131
Government (Diplomatic)	89	96
Violent Political Party	83	149
NGO	47	56
Telecommunication	40	33
Airports & Aircraft	21	21
Maritime	21	15
Food or Ware Supply	14	11
Other	13	18
Abortion Related	1	1
Total	8053	8749

(Source: Start, 2019).

2.2 Theoretical Literature

The theories of swarm intelligence and partitioning clustering algorithms was used in this work to develop an improve model for Big Data analytic. The two algorithms are subset of artificial intelligence model. From review of literatures embedded algorithms in artificial intelligence has the potentials to explore and overcome the challenges big data analysis posed on traditional data mining algorithms.

2.2.1 **Swarm Intelligence (SI)**

A swarm is a large number of homogenous, simple agents interacting locally among themselves, and their environment, with no central control to allow a global interesting behaviour to emerge. Swarm-based algorithms have recently emerged as a family of nature-inspired, population-based algorithms that are capable of producing low cost, fast, and robust solutions to several complex problems (Panigrahi, Shi, & Lim, 2011). Swarm Intelligence (SI) can therefore be defined as a relatively new branch of Artificial Intelligence that is used to model the collective behaviour of social swarms in

nature, such as ant colonies, honey bees, and bird flocks. Although these agents (insects or swarm individuals) are relatively unsophisticated with limited capabilities on their own, they are interacting together with certain behavioural patterns to cooperatively achieve tasks necessary for their survival. The social interactions among swarm individuals can be either direct or indirect (Panigrahi *et al*., 2011). Examples of direct interaction are through visual or audio contact, such as the waggle dance of honey bees. Indirect interaction occurs when one individual change the environment and the other individuals respond to the new environment, such as the pheromone trails of ants that they deposit on their way to search for food sources. This indirect type of interaction is referred to as stigmergy, which essentially means communication through the environment.

In the past decades, biologists and natural scientists have been studying the behaviours of social insects because of the amazing efficiency of these natural swarm systems. In the late-80s, computer scientists proposed the scientific insights of these natural swarm systems to the field of Artificial Intelligence. In 1989, the expression Swarm Intelligence was first introduced by (Beni and Wang, 1989) in the global optimization framework as a set of algorithms for controlling robotic swarm. In 1991, Ant Colony Optimization (ACO) (Colorni, Dorigo, Maniezzo, and Trubian, 1994) was introduced by (Dorigo & Stützle, 2004) and colleagues as a novel nature-inspired metaheuristic for the solution of hard Combinatorial Optimization (CO) problems. In 1995, particle swarm optimization was introduced by (Kennedy and Eberhart, 1995) and was first intended for simulating the bird flocking social behaviour. By the late-90s, these two most popular swarm intelligence algorithms started to go beyond a pure scientific interest and to enter the realm of real-world applications. “It is perhaps worth mentioning here that a number of years later, exactly in 2005, Artificial Bee Colony Algorithm were proposed by (Karaboga and Basturk, 2007) as a new member of the family of swarm intelligence algorithms.

Since the computational modeling of swarms was proposed, there has been a steady increase in the number of research papers reporting the successful application of Swarm Intelligence algorithms in

several optimization tasks and research problems. Swarm Intelligence principles have been successfully applied in a variety of problem domains including function optimization problems, finding optimal routes, scheduling, structural optimization, and image and data analysis (Engelbrecht, 2002). Computational modeling of swarms has been further applied to a wide-range of diverse domains, including machine learning, bioinformatics and medical informatics, dynamical systems and operations research; they have been even applied in finance and business (Parsopoulos & Vrahatis, 2010).

2.2.1.1 **Swarm Intelligence Models**

Swarm intelligence models are referred to as computational models inspired by natural swarm systems. To date, several swarm intelligence models based on different natural swarm systems have been proposed in the literature, and successfully applied in many real-life applications. Examples of swarm intelligence models are Ant Colony Optimization, Particle Swarm Optimization, Artificial Bee Colony, Bacterial Foraging, Cat Swarm Optimization, Artificial Immune System and Glowworm Swarm Optimization.

2.2.1.2 **Particle Swarm Optimization Metaheuristic**

Particle Swarm Optimization (PSO) is a heuristic optimization technique introduced by (Kennedy et al., 1995). PSO is a population-based search strategy that finds optimal solutions using a set of flying particles with velocities that are dynamically adjusted according to their historical performance, as well as their neighbors in the search space (Shi & Eberhart, 1998). While ACO solves problems whose search space can be represented as a weighted construction graph, PSO solves problems whose solutions can be represented as a set of points in an n-dimensional solution space. The term ―particles‖ refers to population members, which are fundamentally described as the swarm positions in the n-dimensional solution space. Each particle is set into motion through the solution space with a velocity vector representing the particle ‘s speed in each dimension. Each particle has a memory to

store its historically best solution (i.e., its best position ever attained in the search space so far, which is also called its experience) (Parsopoulos *et al*., 2010).

"The secret of the PSO success lies in the experience-sharing behavior in which the experience of each particle is continuously communicated to part or the whole swarm, leading the overall swarm motion towards the most promising areas detected so far in the search space (Parsopoulos et al., 2010). Therefore, the moving particles, at each iteration, evaluate their current position concerning the problem's fitness function to be optimized and compare the current fitness of themselves to their historically best positions, as well as to the other individuals of the swarm (either locally within their neighborhood as in the local version of the PSO algorithm, or globally throughout the entire swarm as in the global version of the algorithm). Then, each particle updates its experience (if the current position is better than its historically best one) and adjusts its velocity to imitate the swarm 's global best particle (or, its local superior neighbour, i.e., the one within its neighbourhood whose current position represents a better solution than the particle 's current one) by moving closer towards it. Before the end of each iteration of PSO, the index of the swarm's global best particle (or, the local best particle in the neighbourhood) is updated if the most recent update of the position of any particle in the entire swarm (or, within a predetermined neighbourhood topology) happened to be better than the current position of the swarm's global best particle (or, the local best particle in the neighbourhood) (Lazinica, 2009).

PSO draws inspiration from the sociological behaviour associated with bird flocking. It is a natural observation that birds can fly in large groups with no collision for extended long distances, making use of their effort to maintain an optimum distance between themselves and their neighbours. This section presents some details about birds in nature and overviews their capabilities, as well as their sociological flocking behavior (Lazinica, 2009).

2.2.1.3 **Birds in Nature**

Vision is considered as the most important sense for flock organization. The eyes of most birds are on both sides of their heads, allowing them to see objects on each side at the same time. The larger size of bird's eyes relative to other animal groups is one reason why birds have one of the most highly developed senses of vision in the animal kingdom. As a result of such large sizes of bird's eyes, as well as the way their heads and eyes are arranged, most species of birds have a wide field of view (Grosan, Abraham, & Monica, 2006). For example, Pigeons can see 300 degrees without turning their head, and American Woodcocks have, amazingly, the full 360-degree field of view (Jones, Pierce & Ward, 2007). Birds are generally attracted by food; they have impressive abilities in flocking synchronously for food searching and long-distance migration. Birds also have efficient social interaction that enables them to be capable of: (i) flying without collision even while often changing direction suddenly, (ii) scattering and quickly regrouping when reacting to external threats, and (iii) avoiding predators (Kennedy *et al*., 1995).

2.2.1.4 **Birds Flocking Behaviour**

The emergence of flocking and schooling in groups of interacting agents (such as birds, fish, penguins, etc.) have long intrigued a wide range of scientists from diverse disciplines including animal behaviour, physics, social psychology, social science, and computer science for many decades (Parrish, Viscido, & Grunbaum, 2002). Bird flocking can be defined as the social collective motion behaviour of a large number of interacting birds with a common group objective. The local interactions among birds (particles) usually emerge the shared motion direction of the swarm, as shown in Figure 2.6. Such interactions are based on the nearest neighbour principle‖ where birds follow certain flocking rules to adjust their motion (i.e., position and velocity) based only on their nearest neighbours, without any central coordination (Lazinica, 2009). Extensive observation on birds flocking behavior, "three simple flocking rules was derived to implement a simulated flocking behaviour of birds such as: (i) flock centering (flock members attempt to stay close to nearby flock

mates by flying in a direction that keeps them closer to the centroid of the nearby flock mates), (ii) collision avoidance (flock members avoid collisions with nearby flock mates based on their relative position), and (iii) velocity matching (flock members attempt to match velocity with nearby flock mates).

Although the underlying rules of flocking behaviour can be considered simple, the flocking is visually complex with an overall motion that looks fluid yet it is made of discrete birds (Gazi & Passino, 2011). One should note here that collision avoidance rule serves to establish the minimum required separation distance, whereas velocity matching rule helps to maintain such separation distance during flocking; thus, both rules act as a complement to each other. Both rules together ensure that members of a simulated flock are free to fly without running into one another, no matter how many they are. It is worth mentioning that the three aforementioned flocking rules of Reynolds are generally known as cohesion, separation, and alignment rules in the literature" (Lazinica, 2009; Gazi *et al*., 2011). For example, according to the animal cognition and animal behaviour research, individuals of animals in nature are frequently observed to be attracted towards other individuals to avoid being isolated and to align themselves with neighbours (Krause & Ruxton, 2002; Couzin, Krause, James, Ruxton & Franks, 2002; Lazinica, 2009). Reynolds rules are also comparable to the evaluation, comparison, and imitation principles of the Adaptive Culture Model in the Social Cognitive Theory (Kennedy, Eberhart& Shi, 2001; Ahmed. & Glasgow, 2012). The flocking behaviour of a group of birds is shown in Figure 2.6.

Figure 2.6: The flocking behaviour of a group of birds
(Source: Xiong, He, Yang, Kim, & Lin, 2010).

2.2.2 **Genetic Algorithm (GA)**

"GA employs three operators to propagate its population from one generation to another: Selection, Crossover, and Mutation.

1. The selection operator mimics the natural selection principle (Survival of the Fittest), in which the most fitted population individuals are selected for future generations over weaker, less-fit individuals.
2. The crossover operator mimics the reproduction behaviour observed in biological populations. It propagates the good characteristics/chromosomes of the current generation to future ones by allowing fit individuals to produce more offspring than less-fit individuals, which help improve the average fitness of new generations as the algorithm progresses.
3. The mutation operator promotes the exploration ability of the algorithm by introducing useful diversity in population characteristics, which acts as necessary randomness to reduce the probability of getting trapped into local optima. The details of GA are beyond the scope of this paper, but interested readers may refer to these references for more information. The mechanism of GA and PSO has similarities" (Lazinica, 2009; Ahmed *et al*., 2012).

2.2.2.1 **Particle Swarm Optimization and Genetic Algorithm**

The similarities of particle swarm optimization and genetic algorithm includes (Ahmed *et al*., 2012):

1. Initialization Mechanism: Both PSO and GA are stochastic population-based algorithms that start with several randomly generated individuals/particles.
2. Fitness Function: Both PSO and GA use a specific fitness function (that is desired to be optimized) to evaluate the population members (i.e., either individuals' genetic encodings in GA or particles 'positions in PSO), and accordingly assign fitness values to them.
3. Nature-inspired Properties: Both PSO and GA update their population according to several nature-inspired properties" (Abdelhalim and Habib, 2009). For instance, the velocity update

equation in PSO and the arithmetic crossover operator in GA are both nature-inspired properties that can be considered quite analogous to each other (Coello and Lechuga, 2002).

4. Parameter Tuning: Both GA and PSO have several numerical parameters that remarkably affect the convergence process, and therefore need to be carefully selected. For example, population size, crossover, and mutation rates are required to be carefully selected in GA. Also, swarm size, inertia weight, cognitive and social parameters (c1 and c2) need to be cleverly decided upon in PSO. PSO Dissimilarities to GA
5. Different Conceptual Bases: The conceptual bases of PSO and GAs are intrinsically different: GAs is based on the intelligence of natural selection, whereas PSO algorithms are based on the intelligent social behaviour of swarms in nature (Ahmed *et al*., 2012).
6. Cooperation vs. Competition: PSO algorithms choose the path of cooperation‖, i.e., convergence is driven through learning from cooperative peers/particles, while GAs choose the path of competition‖, i.e., the convergence is driven through learning from competitive individuals (following the survival of the fittest principle)" (Lazinica, 2009; Ahmed *et al*., 2012).
7. Selection Mechanism: The objective of the selection mechanism in GA is to apply (Lazinica, 2009) natural selection 's principle (survival of the fittest), in such a way that the best individuals with the highest performance on the optimization problem are selected and individuals with poor performance are discarded. On the other hand, PSO does not explicitly include a selection mechanism for its convergence strategy; rather it relies on each particle's memory of its historically best position and the swarm's global/local best position. It is worth noting that the particle's best position (the individual experience) in PSO largely resembles the parent's role in GA with the distinction that no new individuals in PSO are created, but instead are updated relative to their individual experience, or for analogy purposes, their parents.

8. Population Adapting vs. Population Replacement: In PSO, instead of explicitly using genetic operators like crossover and mutation, each particle adjusts its velocity (and therefore position) according to its own flying experience, as well as the flying experience of its peers, so the changes are driven through learning from peers and not through genetic recombination and mutations. In other words, (Lazinica, 2009; Ahmed *et al*., 2012) PSO iteratively uses a velocity update equation through a process of adapting‖ the current population (so, the convergence is performed by attracting the particles to positions with good solutions), while GAs use crossover and mutation operators through a process of replacing the previous population with a new one (resembling the death and birth of successive generations in nature). In contrast, PSO population is more stable, as its particles are not destroyed or created, but rather they are just influenced by the best performance of themselves and their peers.
9. Conscious Mutation vs. Random Mutation: The position update equation in PSO, which adds the velocity to the current position to generate the new/next position, is quite analogous to the arithmetic mutation operation in GA. However, the "mutation" process in PSO is not randomly performed (as in GA); rather it is guided by the particle's own flying experience and the flying experience of its peers. In other words, the position update equation of PSO performs some sort of conscious mutation, as opposed to the random mutation performed in GA (using a predefined mutation operator and rate) (Ahmed *et al*., 2012).
10. Memory Capabilities: Since the original PSO has a built-in memory capability, each particle in PSO benefits from its previous experience. In contrast, individuals in GA do not benefit from their history because the standard GA has no memory; plus, the population in each iteration of GA's replaces itself, anyway, in several generations that are successively destroyed and created (Ahmed *et al*., 2012).
11. Information Sharing Mechanism: "In GAs, chromosomes mutually share their genetic information through a genetic recombination process known as crossover. In PSO, however,

only the global/local best particle communicates its position information to other particles in a one-way information sharing mechanism" (Valle, Venayagamoorthy, Mohagheghi, Hernandez, & Harley, 2008).

12. "Problems Types: The standard GA is an inherently discrete algorithm, i.e., it encodes its design variables into bits of 0's and 1's, making it generally suitable for discrete/binary problems. In contrast, the original PSO is an inherently continuous algorithm, but it was later modified to handle discrete/binary problems (Hassan *et al*, 2005). It has been observed that the binary PSO is generally faster, more robust, and performs better than GAs, particularly on high dimensional problems" (Hassan et al, 2005; Ahmed *et al*., 2012).

2.2.2.2 **PSO Advantages Over GA**

1. "The key advantage of PSO over GA is that it is algorithmically simpler, yet more robust and generally converges faster than GA. The simplicity of PSO allowed scientists from different backgrounds, not necessarily related to computer science or programming skills, to use PSO as an efficient optimization tool to a wide range of application domains" (Ahmed *et al*., 2012).
2. "PSO is more able to control convergence than GA. Although manipulating rates of crossover and mutation can affect controlling GA's convergence, such controlling effect is not as significant compared to the level of control that can be achieved in PSO through manipulating its inertia weight. For example, it has been shown that the decrease of inertia weight dramatically increases the swarm's convergence" (Ahmed *et al*., 2012).
3. "Because of the various studies available in the literature to address the parameter selection issue in PSO, the PSO parameters are now more easily selected and more robustly tuned/controlled than GA parameters" (Ahmed *et al*., 2012).
4. "PSO has an impressive ability to perform well without having a large swarm size. It has been observed that PSO with smaller swarm sizes performs comparably to GAs with larger populations (Oma. It has also been observed that the PSO performance is not too sensitive to

the population size, as long as the population size is not too small. This observation with first suggested by" (Shi et al., 2008) and then verified by (Ahmed *et al*., 2012).

2.2.2.3 **PSO Discussion**

The summary of the strengths and limitations of PSO are as follows:

1. "PSO uses memory to store the particle's historically best position and the swarm's global best position, which helps not only each particle to keep track of its individual experience but also helps the most superior particle to communicate its social experience to the other particles. This generally directs the convergence to the most promising areas on the search space and accelerates the optimization process towards the optimal solution" (Kennedy *et al*., 2001; Ahmed et al., 2012).
2. "PSO is not only characterized by its fast convergence behaviour, but also by its simplicity. The core mathematical equations of PSO (namely, velocity update, position update, and memory update) are easily calculated. Thus, the implementation of the PSO procedure is simple and generally requires just a relatively few lines of code.
3. PSO has an inherent potential to adapt to a changing environment, which can expand its ability from just locating optima in static environments to further track them in dynamic environments" (Ahmed *et al*., 2012).

2.2.2.4 **PSO Limitations**

Ahmed *et al*. (2012) reviewed "the limitations of PSO are as follows:

1. The typical PSO problems are those whose solutions can be represented as a set of points in an n-dimensional Cartesian coordinate system, as it would be easy, in such problems, to determine the previous and next positions for each point (i.e., particle). On the other hand, PSO fails to work if the problem representation does not offer a clear way to uniquely define what the next and previous particle positions are to help search in the solution space.

2. The “original PSO assumes all particles of the entire swarm are completely homogenous, and therefore employs the same value settings of inertia weight, cognitive and social parameters (c1 and c2) for the entire swarm. This assumption, however, ignores the internal differences among birds of the same swarm in real life, such as ages, catching skills, flying experiences, and muscles' stretching. It also neglects the relative flying position within the swarm, although it provides an important influence on particles. For example, particles flying in the outer side of the swarm often make more choices than those in the swarm center and thus should receive more attention.

3. The original PSO fails to locate multiple optima since the idea of the original PSO was to adjust the swarm direction closer to the swarm’s global best particle to guide the entire swarm to converge to a single optimum. However, many variations of the original PSO have been proposed in the literature to overcome such a limitation. PSO and GA are based on totally different philosophical metaphors (namely, the evolution metaphor for GA, and the bird flocking metaphor for PSO), both PSO and GA share some common features, besides many other different characteristics. The comparison between PSO and GA has generally been popular in the literature, as it highlights what the novelty of the PSO metaphor is and what’s new PSO can offer compared to the other metaphorical models. The comparison also demonstrates the fact that the general SI metaphor is not merely giving new names to existing operations, but there are fundamental differences in the core optimization processes/functions between it and other metaphorical models.

2.2.2.5 **PSO Mathematical Model**

PSO is a population-based stochastic optimization technique that can be used to find an optimal, or near-optimal, a solution to a numerical and qualitative problem. In the PSO algorithm, birds in a flock are symbolically represented as particles. These particles can be considered as simple agents “flying”

through a problem space. A problem space in PSO may have as many dimensions as needed to model the problem space. A particle's location in the multi-dimensional problem space represents one solution for the problem. When a particle moves to a new location, a different solution is generated. This solution is evaluated by a fitness function that provides a quantitative value of the solution's utility (Parsopoulos *et al*., 2010).

The velocity and direction of each particle moving along each dimension of the problem space are altered at each generation of movement. It is the particle's personal experience combined with its neighbors' experience that influences the movement of each particle through a problem space. For every generation, the particle's new location is computed by adding the particle's current velocity V-vector to its location X-vector.

Mathematically, given multi-dimensional problem space, the i^{th} particle changes its velocity and location according to the following equations" (Clerc & Kennedy, 2002; (Parsopoulos *et al*., 2010):

$$v_{id=w*(v_{id}+c_1*rand_1*(p_{id}-x_{id})+c_2*rand_2*(p_{pg}-x_{id}))} \quad (2.1)$$

$$x_{id=}x_{id+}v_{id} \quad (2.2)$$

Where $p_{id}$is the location of the particle (Clerc *et al*., 2002) where it experiences the best fitness value; $p_{gd}$is the location of the particle experienced the highest best fitness value in the whole population; x_{id} is the particle current location; $c_1$and c_2 are two positive acceleration constants; d is the number of dimensions of the problem space; $rand_1$, $rand_2$are random values in the range of (0,1); wish called the constriction coefficient and it is computed by Eq. (2.3) and Eq. (2.4) (Omran, 2005):

$$w = \frac{2}{2 - \vartheta - \sqrt{\vartheta - 4\vartheta}} \quad (2.3)$$

$$\mathcal{G} = c_1 + c_2, \ \mathcal{G} > 4 \quad (2.4)$$

2.2.3 **Multi-Swarm Optimization**

Multi-swarm optimization is a variant of Particle swarm optimization (PSO) based on the use of multiple sub-swarms instead of one (standard) swarm. The general approach in multi-swarm optimization is that each sub-swarm focuses on a specific region while a specific diversification method decides where and when to launch the sub-swarms. The multi-swarm framework is especially fitted for the optimization of multi-modal problems, where multiple (local) optima exist (Zhao, Liang, Suganthan, & Tasgetiren, 2008).

2.2.3.1 **Description of Multi-swarm Optimization**

In multi-modal problems, it is important to achieve an effective balance between exploration and exploitation. Multi-swarm systems provide a new approach to improve this balance. Instead of trying to achieve a compromise between exploration and exploitation which could weaken both mechanisms of the search process, multi-swarm systems separate them into distinct phases. Each phase is more focused on either exploitation (individual sub-swarms) or exploration (diversification method) (Hendtlass, 2005).

The coordination of the sub-swarms depends on the specific diversification method(s) implemented by the multi-swarm system. The wave of Swarm of Particles (WOSP)" (Hendtlass, 2005), for example, bases its diversification mechanism on the collision of particles. When particles get too close, they are expelled by a short-range force into new waves/sub-swarms, avoiding thus a complete convergence. "The Dynamic Multi-Swarm-Particle Swarm Optimizer (DMS-PSO) (Zhao *et al*, 2008) periodically regroups the particles of the sub-swarms (after they have converged) into new sub-swarms, the new swarms are started with particles from previous swarms". Locust swarms (Chen, 2009) "are based on a devouring and move on strategy – after a sub-swarm devours a relatively small region of the search space (to find a local optimum) scouts are deployed to look for new promising regions to "move on".

A distinctive feature of sub-swarms is that their initial positions and initial velocities are not randomly selected as in normal swarms. Instead, they maintain some information from the previous trajectories of the particles. In general, the development of multi-swarm systems leads to design decisions that did not exist during the original development of particle swarm optimization, such as the number of particles to use in each sub-swarm, the optimal value for the constriction factor, and the effects of non-random initial positions and initial velocities. These design decisions have been thoroughly studied and have well-established guidelines – e.g. the use of non-random initial positions and initial velocities leads to improved results in multi-swarm systems, which is not the case for single-swarms (Chen & Montgomery, 2011). Other design decisions, such as which diversification method to use or which specific search strategy will select the initial positions and initial velocities of a sub-swarm, have less established guidelines and constitute open questions in the field of multi-swarm systems.

Some of these design decisions can be addressed by relatively independent sub-components which allow different optimization techniques to be inserted (Röhler & Chen, 2012). Multi-swarm systems thus provide a useful framework for the development of hybrid algorithms. For example, the UMDA-PSO (Röhler *et al*, 2012) multi-swarm system effectively combines components from Particle Swarm Optimization, Estimation of distribution algorithm, and Differential Evolution into a multi-swarm hybrid.

2.2.3.2 **General Advantages of Swarm Intelligence**

1. "Scalability: SI systems are highly scalable; their impressive abilities are generally maintained when using groups ranging from just sufficiently few individuals up to millions of individuals. In other words, the control mechanisms used in SI systems are not too dependent on swarm size, as long as it is not too small" (Ahmed *et al*., 2012).
2. Adaptability: SI Systems respond well to rapidly changing environments, making use of their inherited auto-configuration and self-organization capabilities. This allows them to autonomously

adapt their individuals' behavior to the external environment dynamically on the run-time, with substantial flexibility" (Ahmed *et al*., 2012).

3. "Collective Robustness: SI Systems are robust as they collectively work without central control, and there is no single individual crucial for the swarm to continue to function (due to the redundancy of their individuals). In other words, the fault-tolerance capability of SI systems is remarkably high, since these systems have no single point of failure. A single point of failure is a part of any system that puts the entire system into risk of a complete failure, if it ceased to function" (Ahmed *et al*., 2012).

4. "Individual Simplicity: SI systems consist of several simple individuals with fairly limited capabilities on their own, yet the simple behavioral rules at the individual level is practically sufficient to cooperatively emerge sophisticated group behavior" (Ahmed *et al*., 2012).

2.2.4 Clustering

Clusters are grouping (or partitioning) of similar data items such that similar items are grouped and data items that are not similar are kept aside e.g. arranging of plastic chairs after a meeting (or convocation ceremony) (Jain & Dubes, 2009). The volume of data is on the increase, from the World Wide Web (WWW), electronic mail (e-mail), corporate databases, chat rooms, and digital libraries. These data are stored or represented in some form for further analysis and management. One of the vital means in dealing with these data is to classify or group them into a set of categories or clusters. To learn a new objector, understand a new phenomenon, it is always good to observe the features that can best describe it and further compare it with other known objects (or concepts) based on their similarities or dissimilarities using certain standards or rules. K-means is one of the unsupervised learning algorithms that solve the well-known data clustering problems.

2.2.4.1 **Application of Clustering Algorithm**

The following are some areas where clustering algorithms are proving to be useful:

i. Clustering Algorithms in Marketing**:** Clustering algorithms have been used in marketing in finding groups of customers with similar behaviors given a large database of customer data containing their properties and past buying records.

ii. Clustering Algorithms in Academics: The ability to monitor the progress of students' academic performance has been the critical issue for the academic community of higher learning. Clustering algorithms can be used to monitor the students' academic performance. Based on the student's scores, they are grouped into different clusters, where each cluster denoting the different levels of performance. By knowing the number of students in each cluster, we can know the average performance of a class as a whole (Oyelade, Oladipupo, & Obagbuwa, 2010).

iii. Clustering Algorithms in Search Engines: Clustering algorithm is the backbone behind the search engine. Search engines try to group similar objects in one cluster and dissimilar objects far from each other. It provides results for the searched data according to the nearest similar object which is clustered around the data to be searched.

iv. Clustering Algorithms in Wireless Sensor Network Applications Clustering algorithms can be used effectively in wireless sensor networks-based applications. One application where it can be used is in landline detection. Clustering algorithm plays the role of finding the cluster center which collects all the data in its respective cluster (Oyelade *et al*., 2010).

v. Clustering Algorithms in Earthquakes Studies Clustering: It was observed that earthquake epicenters are used to identify dangerous zones.

vi. Clustering Algorithms in Insurance: This is used in identifying groups of motor insurance policyholders with high average claim cost, identifying frauds.

vii. Clustering Algorithms in Crop Diseases: The agricultural establishments will use a mixed clustering model to investigate the viability of the seed crops before the planting season. It is also a predictive model.

2.2.4.2 **Clustering Challenges of Large Datasets**

Clustering in Big data is required to identify the existing patterns which are not clear at first glance. "The properties of big data pose some challenge against adopting traditional clustering methods (Jain & Verma, 2014):

1. Type of dataset: The collected data in the real world contains both numeric and categorical attributes. Clustering algorithms work effectively either on purely numeric data or categorical data; most of them perform poorly on mixed categorical and numerical data types. Size of the dataset: The size of the dataset affects both the time-efficiency of clustering and the clustering quality (indicated by the precision). Some clustering methods are more efficient than others when the data size is small, and vice versa.
2. Handling outliers/noisy data: Data from real applications suffers from noisy data that pertains to faults and misreported readings from sensors. Noise (very high or low values) makes it difficult to cluster an object thereby affecting the results of clustering. A successful algorithm must be able to handle outliers/noisy data.
3. Time Complexity: Most of the clustering methods must be repeated several times to improve the clustering quality. Therefore, if the process takes too long, then it can become impractical for applications that handle big data.
4. Stability: Stability corresponds to the ability of an algorithm to generate the same partition of the data irrespective of the order in which the data are presented to the algorithm. That is, the results of clustering should not depend on the order of data (Jain *et al*., 2014).
5. High dimensionality (Curse of dimensionality), a term coined by Richard E. Bellman is relevant here. As the number of dimensions increases, the data becomes increasingly sparse, so the distance measurement between pairs of points becomes meaningless and the average density of points anywhere in the data is likely to below. Therefore, algorithms which partition data based on the concept of proximity may not be fruitful in such situations.

6. Cluster shape: A good clustering algorithm should be able to handle real data and their wide variety of data types, which will produce clusters of arbitrary shape. Many algorithms can identify only convex-shaped clusters".

2.2.4.3 **Clustering Different Data Type**

Two types of data can be clustered; namely numerical and categorical data. These are explained below:

1. Numerical Data: They are usually data whose inherent geometric properties can be exploited naturally to define distance functions between data points. These attribute values are ordered. In other words, they are quantities (or objects) that can be measured or quantified e.g. table, chair, door, etc. due to their lengths, L, width, W, and height, H. Other real-world numeric datasets include Wine etc.
2. "Categorical Data: An example of a categorical attribute is shaping whose values include circle, rectangle, eclipse, spherical, banana, orange, soya beans, Iris, mushroom, Yeast species, etc. These attribute values are not ordered and cannot be quantified or measured. Clustering Databases with Categorical Data cannot be ordered singly and, therefore, clustering of such data is a big challenge. We summarize the characteristics of such data in the following list:

   a. Categorical data have no single ordering, therefore, there are several ways in which they can be ordered, but there is no single one that is more semantically sensible than others.

   b. Categorical data can be visualized depending on a specific ordering.

   c. Categorical data define no a priori structure to work with.

   d. Categorical data can be mapped onto unique numbers and, as a consequence, Euclidean distance could be used to prescribe their proximities, with questionable consequences though.

3. Mixed data: This is the combination of numeric and categorical attributes e.g. Soybeans, Iris, Heart Disease, Breast cancer, etc. Due to the differences in the characteristics of these two kinds of data, attempts to develop criteria functions for mixed data have been not very successful. The fundamental issue here is how to put the categorical (or nominal) data in some form to allow input and output for computer processing (Lee & Yang, 2009).

2..2.5 **Clustering Techniques**

A clustering algorithm can be described as partitional or hierarchical depending on the approach it follows to produce a set of clusters (Jain et al., 2009). The analytic models developed in this work are based on Partitioning clustering and Swarm Intelligence algorithms.

2.2.6 **Partitioning Clustering**

Given a database D of n tuples and an input number k that represents the number of desired clusters, a partitional algorithm applies a measure and partitions the data into a flat arrangement of k clusters. A well-known partitioning algorithm is the k-means algorithm (MacQueen,1967). The k-means algorithm is a distance-based algorithm that views data as points in a metric space. Furthermore, k-means represents a cluster by a centroid. A definition of a centroid is given by (Rayward-Smith, 2007) as follows:

"Definition 1: Given any set of points, C, in a metric space, M, with metric, d, a point $\hat{c} \in$ M is called a centroid (San, Huynh, & Nakamori., 2004) if

$$\sum_{x \varepsilon C} d(\overline{c},\ x) \qquad (2.5)$$

is minimized, here d ($\hat{c}$, x) represents the distance between two points. Note that the centroid $\hat{c}$ is not necessarily an element of C".

2.2.6.1 **K-Means Clustering**

The term k-means was first used by James MacQueen in 1967, though the idea goes back to Hugo Steinhaus in 1957. The standard algorithm was first proposed by Stuart Lloyd in 1957 as a technique for pulse-code modulation, though this was not published until" 1982 (San et al., 2004).

2.2.6.2 **K-means Clustering Algorithm**

"K-means clustering Algorithm is aimed at clustering the best center called centroid in a given data set. The idea is to initialize k centroid, one for each cluster, then the distances of all data elements are calculated using Euclidean distance measure such that data sets closer to the centroid are clustered together. This step is repeated until no more changes occur in the cluster. The procedure for k-means clustering algorithm is as follows" (Opara, Eze & Oleji).

INPUT: Number of k initial clusters

Data objects D = $\{d_1, d_2 \ldots d_n\}$

OUTPUT: A set of K clusters

Steps:

1. Select K initial centroid from D

2. Randomly select distinct centroid (new data points as cluster initialization)

3. Calculate the distance between each data objects;

4. Assign D_i (1<= i<=n) to the closest cluster k

4. For each cluster j (1<=j <=k), recalculate the cluster centroid

Until the centroid does not change (Opara *et al*., 2020).

2.2.6.3 **Parameters of K-means**

"The k-means algorithm requires three user-specified parameters: number of clusters k, cluster initialization, and distance metric. The most critical choice is k. While no perfect mathematical criterion exists, several heuristics by (Tibshirani, Walther & Hastie, 2001) and discussion therein are

available for choosing k. Typically, k-means is run independently for different values of k, and the partition that appears the most meaningful to the domain expert is selected. Different initializations can lead to different final clustering because k-means only converge to local minima.

One way to overcome the local minima is to run the k-means algorithm, for a given k, with multiple different initial partitions and choose the partition with the smallest squared error. K-means is typically used with the Euclidean metric for computing the distance between points and cluster centers. As a result, k-means finds spherical or ball-shaped clusters in data. K-means with Mahalanobis distance metric has been used to detect hyper-ellipsoidal clusters (Jain, Flynn & Murty, 2011), but this comes at the expense of higher computational cost".

2.2.6.4 **Mathematical model for the k-means clustering algorithm**

In summary, the number of clusters, K, in a given dataset must always be known in advance. The k-means algorithm can only handle numeric data efficiently but performs poorly with categorical data. The results strongly depend not only on the initial prototypes and sensitivity of noise and outliers but also on the problem of dead prototypes (or empty clusters) that converges to local optima. The proposed hybrid method aims at solving at least one or more of the existing problems mentioned earlier.

"For example, let us consider a given dataset: D = $\{X_1,\ldots, X_n\}$ of n numerical data objects, a natural number $k \leq n$, and a distance measure d, the k-means algorithm aims at finding a partition C of D into k non-empty disjoint clusters

$C_i, \ldots, C_k$ with $C_i \cap C_j = \phi$; and $\cup_{i=1}^{k} C_i = D$

such that the overall sum of the squared distances between data objects and their cluster centers is minimized (Lakshmi *et al*, 2018).

Mathematically, if indicator variables w_{i1} is used, which takes the value 1 if object X_i is in cluster C_l, and 0 otherwise, then the problem can be stated in terms of a constrained non-linear optimization problem as follows (Lakshmi et al, 2018):

Minimize

$$p(W,Q) = \sum_{j=1}^{k}\sum_{i=1}^{n} w_{ij} d(X_j, Q_j) \quad (2.6)$$

Subject to

$$\sum_{l=1}^{k} w_{i,l} = 1, 1 \le i \le n, \quad (2.7)$$

$w_{i1} \in \{0,1\}$, $1 \le i \le n$, $1 \le l \le k$, where $Q = [w_{i1}]_{n\times k}$ is a partition matrix, $Q = \{Q_1, \ldots, Q_k\}$ (San, 2004; Lakshmi *et al*, 2018) is "the set of cluster center, and d(.,.) is the squared Euclidean distance between two objects. As it is well known, the usual method toward the optimization of P in equation (2.6) subject to the constraint in equation (2.7) is to use partial optimization for Q and W. That is, first fix Q and find necessary conditions for W to minimize P. Then, fix W and minimize P according to Q. The k-means algorithm usually iterates through a three-step process until P(W, Q) converges to some local minimum (Lakshmi et al, 2018) as follows:

1. Select an initial $Q^{(0)} = \{Q_1^{(0)}, \ldots, Q_k^{(0)}\}$, and set t = 0.

2. Keep $Q^{(t)}$ fixed and solve $P(W,Q^{(t)})$ to obtain $W^{(t)}$, i.e., regarding $Q^{(t)}$ as the cluster centers, assign each object to the cluster of its nearest cluster center.

3. Keep $W^{(t)}$ fixed and generate $Q^{(t+1)}$ such that $P(W^{(t)}, Q^{(t+1)})$ is minimized, i.e., construct new cluster centers according to the current distribution of objects.

4. In the case of convergence or if a given stopping criterion is fulfilled, output the result and stop. Otherwise, set t = t + 1 and go to Step 2.

In the setting of numerical data clustering, the Euclidean norm ((Morissette & Chartier, 2013; Jain et al., 2009; San *et al*., 2004)

$$d(X,Y) = \sqrt{\sum_{j=1}^{m} |x_j - y_j|^2} \quad (2.8)$$

is often chosen as a natural distance measure in the k-means algorithm.

With this distance measure, the computation of the mean of a cluster's object returns the cluster's center, fulfilling the minimization condition of Step 3. This includes:

$Q_l^{(k+1)}$= ($q_{l,1}^{(k+1)}$, . . ., $q_{l,m}^{(k+1)}$), for i = 1, …, k and

$$q_{i,j}^{(k+1)} = \frac{\sum_{i=1}^{n} w_{i,l}^{(t)} x_{i,j}}{\sum_{i=1}^{n} w_{i,l}^{(t)}} \qquad (2.9)$$

where w represents indicator variables $w_{i,l}$, which takes the value 1 if object X_i is in cluster C_l, and 0 otherwise (San *et al*., 2004)".

2.2.6.5 **Disadvantages of k-Means**

"Despite its efficiency, k-Means has several disadvantages derived from its statistical simplicity. Drawbacks of k-Means variants are well documented in the literature:

1. From an optimization point of view, it often converges to a local optimum of poor quality. k-Means favors hyper spherical clusters and it is sensitive to scaling or similar transformations.

2. Because k central vectors are means of cluster points, they are commonly adopted as representatives of the data points of the clusters. However, it is possible for the arithmetic mean to have no valid interpretation; for example, the average of the coordinates of a group of schools may indicate that the representative school lies in the middle of a lake.

3. k-Means is very sensitive to the presence of noise and outliers, as well as to the initial random clustering. In particular, much effort has been focused on the sensitivity of k-Means on the set of representatives used to initialize the search.

4. The method is statistically biased. For parametric statisticians, this implies that even if provided with the exact number of distributions in a uniform family mixture (e.g., all multivariate normal distributions), and large volumes of noiseless data, k-Means converges to the wrong parameter values. This has favored other statistical methods such as Expectation Maximization and probabilistic clustering. k-Means is also statistically inconsistent" (Lee *et al*., 2009).

2.2.6.6 **Extensions to K-Means**

The K-means algorithm explained at (2.6) is widely been studied by many researchers. As a result, many improvements on its deficiencies have been achieved. The major drawback of this algorithm that leads to these improvements are:

1. Randomly initialized K Clusters
2. Susceptibility to Noise and Outliers
3. Complexity in terms of computational efficiency is O(nkl). Where n is the number of documents, k is the desired number of clusters and l is the number of iterations.
4. Converges sometimes to local optimal.

Considering these draw backs on K-means, let us discuss some extensions made to the traditional k-means in other to increase its usefulness and efficiency.

The summary of some variants of an extension to the k-means algorithm:

1. K-mode is a kind of k-means that uses modes instead of means for the clustering process. It is less susceptible to outliers when compared with k- means.
2. K-medoids run like k-means, but use points called medoids as cluster centroids. Medoids are the most centered point; hence it overcomes the problem of local optimum phased with the traditional K-means
3. Fuzzy C-Means Clustering is a soft version of k-means, where each data point has a fuzzy degree of belonging to each cluster.
4. Gaussian mixture models trained with an expectation-maximization algorithm
5. Expectant Maximization algorithm maintains probabilistic assignments to clusters, instead of deterministic assignments, and multivariate Gaussian distributions instead of means.

Several other methods have also been proposed to choose better starting clusters such as:

(i) The filtering algorithm uses kd-trees to speed up each k-means step.

(ii) Some methods attempt to speed up each k-means step using core sets or the triangle inequality.

(iii) Escape local optima by swapping points between clusters.

(iv) The Spherical k-means clustering algorithm is suitable for directional data.

(v) The Minkowski metric weighted k-means deals with the problem of noise features by assigning weights to each feature per cluster".

2.2.6.7 **K-Modes**

K-modes is an extension of K-means to cluster categorical datasets. K-modes suffer from the effect of outliers due to the use of arithmetic means as measure for calculating centroid. It also computes the distance at each iteration between the centroid and other objects; hence it becomes computationally expensive for massive (or huge) data sets. In addition, there has been a saying by some researchers that K-Means cannot be used on other kind of data except quantitative (or numerical) data, but this is not true. K-means can be used on multivariate data of N-dimension with respect to the dissimilarity measure. These are factors that motivated the development of K-Modes (Oleji *et al*., 2016).

This algorithm was proposed by (Huang, 1997) for the mere reason of handling these drawbacks in K-Means. As a result of using arithmetic means for computing centroid, K-Means sometimes suffer from having centroid that cannot be updated; hence this leads to having empty centroid (or death cluster). This problem is solved by K-Modes, replacing means with modes. It employs some frequency models for updating the modes during clustering processes. This frequency models introduced by K-modes also reduces the computational cost during the process. It also introduces the basic matching techniques for dissimilarity of categorical data objects which helps in eliminating the problems faced with K-Means as a result of the Euclidean distance used.

2.2.6.9 **Formation of "Cluster Centers"**

Two main problems mentioned when applying k-means on categorical data. One of them is the formation of cluster centers. As arithmetic operations are completely absent in the setting of categorical objects, the Cartesian product and union operations were used for the formation of "cluster

centers" based on the notion of means in the numerical setting. The addition and multiplication have been replaced in equation (2.9) by the union and the Cartesian product for categorical data, which is then used in defining the notion of k-representatives for clusters. These equations can be found in chapter two of this work (San *et al*., 2004).

Given a cluster C = $\{X_1, \ldots, X_p\}$ of categorical objects, with

$X_i = (x_{i1}, \ldots, x_{i,m,})1 \leq i \leq p$, denoted by D_j the set formed from categorical values $x_{i1},\ldots x_{pj}$ For example, the set formed from values a, b, a, c is {a, b, c}. Then the representative of C is defined by Q = $(q_1, \ldots, q_m)$, with

$q_j = \{(c_j, rcf_{cj}) \mid c_j \varepsilon D_j\}$, (2.10)

Where rcf_{cj} is the relative cumulative frequency of category c_j within C, i.e., $rcf_{cj} = \frac{n_{cj}}{p}$, where n_{cj} is the number of objects in C having category $c_j$at attribute A_j. Formally, each q_j can be seen as a fuzzy set on D_j with membership grades of elements to be defined by their relative frequencies within the cluster (San *et al*., 2004). It would be also worthwhile to note that the definition of k-representatives in equation (2.10) is applied for numerical data objects by replacing the union and Cartesian product with addition and multiplication to obtain the notion of cluster centers by means as done in equation (2.9).

2.2.6.10 **Dissimilarity Measure**

Due to the modifications in forming representatives for clusters of categorical objects, the dissimilarity between a categorical object and the representative of a cluster is defined based on simple matching as follows (San *et al*., 2004):

"Let C = $\{X_1,\ldots X_p\}$ be a cluster of categorical objects, with

$X_i = (x_{i1}, \ldots, x_{i,m})$, $1 \leq i \leq p$ and X = $(x_1, \ldots, x_m)$ a categorical object.

Note that X may or may not belong to C. Assume that Q = $(q_1, \ldots, q_m)$ with

$q_j = \{(c_j, rcf_{cj}) \mid c_j \, \varepsilon \, D_j\}$,

is a representative of cluster C.

Now, the dissimilarity between object X and representative Q (San *et al*., 2004; Lakshmi *et al*, 2018) is defined by

$$d(X, Q) = \sum_{j=1}^{p} \sum_{c_j \varepsilon D_j}^{m} rcf_{cj} \delta(x_j, c_j) \qquad (2.11)$$

where c = cluster and rcf = relative cumulative-frequency between the clusters.

Under such a definition, the dissimilarity d (X, Q) is mainly dependent on the cumulative frequencies of categorical values within the cluster and simple matching between categorical values. It is also of interest to note that the simple matching dissimilarity measure between categorical objects can be considered as a categorical counterpart of the squared Euclidean distance measure (San et al., 2004; Lakshmi *et al*, 2018).

It is easily seen that

$$d(X, Q) = \sum_{j=1}^{p} \sum_{c_j \varepsilon D_j}^{m} rcf_{cj} \delta(x_j, c_j) \qquad 2.12$$

$$= \sum_{j=1}^{p} \sum_{c_j \varepsilon D_j, c_j \neq x_j}^{m} rcf_{cj} \qquad 2.13$$

$$= \sum_{j=1}^{p} (1 - rcf_{xj}) \qquad (2.14)$$

where rcf_{xj} is the relative cumulative frequency of category x_j within C" (Huang *et al*., 2007).

2.2.6.11 **K-Representatives Algorithm**

With the modifications made in equation (2.14), it is possible now to formulate the problem of clustering categorical data as a partitioning problem in a fashion similar to k-means clustering (San, *et al*., 2004; Jain *et al*., 2009).

"Let us assume that the data set,

$D = \{X_1, \ldots, X_n\}$ as categorical objects to be clustered,

where each object $X_i = (x_{i1}, \ldots, x_{i,m})$, $1 \leq i \leq n$ is described by m categorical attributes.

Then, the problem can be mathematically stated as follows (San *et al*., 2004; Lakshmi *et al*, 2018):

Minimize

$$P(W, Q) = \sum_{j=1}^{k} \sum_{c_j \varepsilon D_j}^{n} w_{i,l}, d(X_j, Q_j) \qquad (2.15)$$

Subject to

$$\sum_{j=1}^{k} w_{i,l} = 1, 1 \leq i \leq n, \qquad (2.16)$$

$w_{i,l} \varepsilon \{0, 1\}, 1 \leq i \leq n, 1 \leq i \leq k,$

where $w = [w_{i,l}]_{n \times k}$ is a partition matrix, $Q = \{Q_1, \ldots, Q_k\}$ is the set of representatives and $d(X_i, Q_l)$ is the dissimilarity between object X_i and representative Q_l defined by equation (2.15). In the same way as in the k-modes algorithm proposed by (Huang, 1997), the following algorithm is introduced for clustering categorical data:

1. Initialize a k-partition of D randomly.

2. Calculate k representatives one for each cluster.

3. For each X_i, calculate the dissimilarities $d(X_i, Q_l), l = 1, \ldots, k$. Then, reassign X_i to cluster C_l (from cluster $C_{l'}$, say) such that the dissimilarity between X_i and Q_l is least. Update both Q_l and $Q_{l'}$ (San, *et al*., 2004; Jain *et al*., 2009).

4. Repeat Step 3 until no object has changed clusters after a full cycle test of the whole data set.

It is worth noting "that the definition of representatives of clusters in the proposed technique is based on the notion of means. That is, the optimized solution of the corresponding numerical problem as in equation (2.9). Thus, the optimization problem (equation 2.14) is reduced to the partial optimization problem, namely, P_1 (Huang, 1998), for weight (W) as specified in the above algorithm".

Equation (2.11) becomes (San *et al*., 2004; Lakshmi *et al*, 2018):

$$d(X_i, Q_1) = \sum_{j=1}^{p} (1 - rcf_{xj,j}) \qquad (2.17)$$

"where $rcf_{xi,j}$ is the relative cumulative frequency of category $x_{i,j}$ within cluster C_l. Thus, in the proposed algorithm, object X_i will be allocated to cluster C_l so that the categories of Xi are most likely to constitute a mode1 of C_l related to the other clusters. It is worthy to note that, by definition, all possible modes of each cluster are taken into account in the proposed algorithm" (Jain et al., 2009).

2.3 **Theoretical Considerations**

Liviu and Mafteiu, (2013) proposes a new measure to evaluating the dissimilarity of two feature-vectors, called Reference Distance Weighted, noted with RDW. The term "reference" shows that the distance is measured from a reference system, ie from the feature-vector specific to a class of problems/objects to the feature-vector of the problem/object to be classified. The term "weighted" has two meanings: the first show that each feature has a specific weight / relevance / importance in final problem to be solved; the second meaning refers to how big is the difference between two features relative to the reference feature value (Liviu *et al*., 2013).

The RDW indicator was designed to use it in systems of equations classification, process that depend on some characteristics of associated matrices, such as: size, sparsity, number of non-zero values on the main diagonal, nonzero elements distribution, symmetry, positivity etc. Some of these features of matrices have been successfully used in other classification processes, relevant examples are given by (Shuting & Jun, 2004).

RDW can be seen as a function: RDW (u, v, α): $R \rightarrow R_+$.

The relation for computing RDW value is (Liviu *et al*., 2013):

$$\text{RDW}(u, v, \alpha) = \frac{\sum_{i=1}^{n} \alpha_i \left|\frac{u_i - v_i}{u_i}\right|}{n} \quad (2.18)$$

$$= \frac{\sum_{i=1}^{n} \alpha_i \left|1 - \frac{v_i}{u_i}\right|}{n} \quad (2.19)$$

$\alpha = \{\alpha 1, \alpha 2, \ldots, \alpha n\} \in R_+^n$;

$u = \{u1, u2, \ldots, un\} \in Rn$,

$u1.\ u2.\ \ldots.\ un \neq 0$;

$v = \{v_1, v_2, \ldots, v_n\} \in Rn$

"with:

-n number of features considered;

-$u = u = \{u_1, u_2, \ldots, u_n\}$, is the feature-vector of reference, associated to a class, vector from whom the distance is measured;

-$v = \{v_1, v_2, \ldots, v_n\}$, is the feature-vector associated to the problem that must be solved or object that must be classified, vector up to which is measured the distance;

- $\alpha = \{\alpha 1, \alpha 2, \ldots, \alpha n\}$, is a vector called relevance vector, whose components αi are parameters specific for each feature in part and assigned to each feature, called relevance factor, proportional to the importance/weight of the respective feature under the conditions of problem to be solved" (Liviu *et al*., 2013).

In relation (5) the case ui = 0 is excluded to avoid dividing by zero.

Remark 1: The generalization of relation (5), i.e. including the situation ui = 0, can be done by introducing a correction factor ε:

$\varepsilon = \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\} \in R^n$, $\varepsilon_i = 10^{-(r_i+1)}$ (2.20)

where r_i represent the magnitude of v_i value. For example, if *vi [1,9]*, we have $r_i = 1$ leading to $\varepsilon i=0.01$, a value that will affect a very little the RDW value in conditions when ui = 0, as can be seen in relation:

$$\text{RDW (u, v, α)} = \frac{\sum_{i=1}^{n} \alpha_i \left|1-\frac{v_i}{u_i}\right|}{n} \quad (2.21)$$

The feature-vector u is regarded as a reference for the feature-vector v, which is a natural approach in conditions which u is attached to a particular class of problems/objects and vector v is attached to the problem/object to be classified and is reported to the known classes of problems/objects (Liviu *et al*., 2013). The approach proposed here, that problem/object relates to class and not vice versa, differentiates the proposed indicator RDW from other indicators used in classification problems.

"The major advantage of using RDW in calculating the degree of dissimilarity is the fact that the value of this indicator depends on the percent in which differ the characteristics of v towards the characteristics of u and not from the absolute values of the differences between these values, such as the Minkowski type distances" (Liviu *et al*., 2013).

"From the mode definition RDW (u, v, α): $R^n \rightarrow R_+$ and for $\forall u, v, w \in R_+^n$ in accordance with relation (6) we have the following properties of the proposed RDW distance:

Property 1: RDW (u, v, α) ≥ 0, axiom of positivity;

Property 2: RDW (u, v, α) = 0 if and only if u = v, i.e. $u_i = v_i$, $\forall_i = \overline{1n}$, axiom of coincidence;

Property 3: RDW (u, v, α) ≠ RDW (u, v, α) if u ≠ v ($\exists I = \overline{1n, u_l \neq v_l}$: non-symmetry;

Property 4: $\exists \gamma \geq 0, \gamma \in R$ such that: $|RDW(v,u,\propto) - RDW(v,u,\propto)| \leq \gamma$:relaxed symmetry (Bhattacharya, Kar & Pal, 2009; Liviu *et al.*, 2013);

Property 5: $\beta . RDW(u,w,\alpha) \leq RDW(u,v,\alpha) + RDW(u,v,\alpha)$

≠

$$RDW(u,v,\alpha) + RDW(u,v,\alpha) \geq \beta * RDW(u,v,\alpha) \quad (2.22)$$

Where $\beta \leq 1$ *is a constant, relaxed triangle inequality* (Bhattacharya *et al*, 2009).

Remark 2: triangle property is verified only in particular cases" (Liviu & Mafteiu, 2013). The situation can be assimilated to a weighted directed graph in which the vectors u, v and w are vertices and RDW values are weighted edges between these vertices, as shown in Figure 2.9.

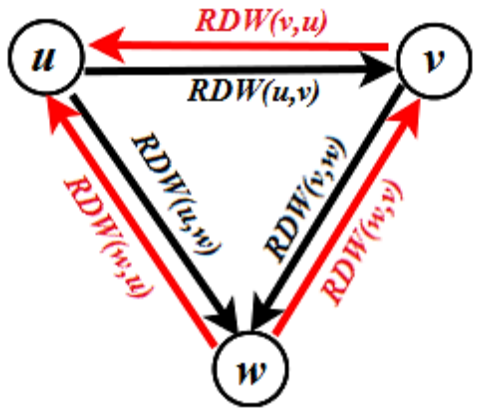

Figure 2.9: The analogy: relaxed triangle inequality - weighted directed graph
(Source: Liviu, & Mafteiu, 2013)

"Remark3: if RDW (u, v, α) = RDW (u, v, α)

and

$$\text{RDW}(u, v, \alpha) \neq 0, \text{RDW}(u, v, \alpha) \neq 0 \quad (2.23)$$

$$\leftrightarrow u = w;$$

If conditions

$$\text{RDW}(u, v, \alpha) \neq 0, \text{and RDW}(u, v, \alpha) \neq 0 \quad (2.24)$$

are eliminated, then:

$$\text{RDW}(u, v, \alpha) = \text{RDW}(u, v, \alpha) \Rightarrow v = w = u \quad (2.25)$$

Theorem: For $\forall v = \{v_1, v_2, \ldots, v_{1n}\} \in R^n$ with strictly monotone sequences, and

$$\forall v = \{u_1, u_2, \ldots, u_n\} \in R^n, u_1, .u_2, \ldots, .u_n \neq 0 \text{ and } \forall \alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_n\} \in R_+^n, \quad (2.26)$$

the function RDW (u, v, α): $R^n \rightarrow R$ is an unimodal function that has a single minimal mode" (Liviu et al., 2013).

2.4 **Empirical Review**

Kumar and Singh (2019) presented a novel clustering technique for efficient clustering of big data in Hadoop Ecosystem. The authors stated that big data analytics and data mining are methods used to analyze data and unveil hidden information. In their analysis, they commented that the existing clustering algorithms, such as k-means and hierarchical algorithms, are not efficient because they produce inaccurate quality clusters. To improve the performance of the k-means clustering algorithm, they hybridized k-means and hierarchical clustering algorithms to overcome the disadvantages of the K-means clustering algorithm. The new hybrid algorithm was compared "with the existing algorithms on the bases of precision, recall, F-measure, execution time, and accuracy of results. The results show that their proposed system produced accurate clusters, and has better precision, recall, and F-measure values more than the existing algorithms. But K-means algorithm converges prematurely and hierarchical clustering is less efficient than K-means clustering in running time. This implies that the dimensionality and massive nature of big data analysis may affect the hybrid of k-means and hierarchical clustering algorithms. And the time complexity of the process would not be efficient.

Rebollo, Puente, Palacious., Piriz, Fuentes, & Jarauta, (2019) proposed detection of Jihadism in social networks using big data techniques supported by Graphs and Fuzzy Clustering algorithm. The work focused on the analysis of Twitter messages to detect how Jihadism leaders orchestrate terrorist networks and their followers. Their proposed big data architecture was used to analyze messages in real-time to classify users according to different parameters like the level of activity, the ability to influence other users, and the contents of their messages. Graphs were used to analyze how the messages propagate through the network, and this involves the study of the followers based on retweet and general impact on other users. Also, Fuzzy clustering techniques were used to classify users' profiles and the algorithm was tested with a public database from Kaggle and other Twitter extraction techniques. Their results provided an efficient decision support system for security agencies, human resources, immigration, and other agencies to detect undesirable clients. Abiodum, Janta, Omolara,

Singh, Abukakar, & Umar (2018) analyzed Big Data: an approach for detecting terrorist activities with people's profiling. The work reviewed advanced knowledge discovery approaches to address terrorism threats which include: Terrorism knowledge portal, Terrorism Expert finder, and Dark web (consisting of internet-based terrorist multilingual resources). It used Linear Regression Analysis (Least-squares method) model as data fitting techniques and develop a mathematical model to detect a person's involvement in terrorism. The research objective was obtained. It recommended that people's profiling analysis had a significant contribution to terrorist detecting if integrated into the system as a solution.

The work of (Acheampong, 2018), which discussed big data, machine learning, and BlockChain technology. He explained how Big Data has impacted the machine learning community. He concluded by analyzing the mechanisms to incorporating BlockChain Technology into Machine Learning.

But they did not implement their proposed system in any case study. A novel technique on class Imbalance big data using Analogous under Sampling Approach was proposed by (Imran, and Rao, 2018). The authors proposed Under Sampled Imbalance Big Data (USIBD) knowledge discovery framework is robust and less sensitive to outliers where the non-uniform distribution of data is applied. Though, the proposed system is fairly adequate to handle outliers but may be inadequate when applied to the unknown problem domain. A simplified analytical model towards Big Data analysis using Ridge Regression Method was presented by (Ali & Site, 2018). The simulation results represented a mapping model of Gaussian data from big data on a sufficient scale. And the proposed model as was claimed presents the new gateway for big data for statistical and mathematical analysis. Ridge regression cannot handle uncertainties and variation in big data analytics dynamic systems. The big data analytics approach using Indexed and Ranking for excellence in higher education was presented by (Shukla and Alim, 2018). The authors used "indexing and ranking of an individual faculty member where the score is derived from big-data and later convert into the index and

thereafter into ranks". The work adapted the index formulae used to measure excellent existing in the higher education system. In the analysis, unstructured and structured data were converted into numerical attributes before using it for the analysis this will affect the accuracy of the analytic results. The work of (Thayananathan & Albeshri, 2018) analyzed big data security issues and quantum cryptography for cloud computing. The authors stated that the enhancement of security in cloud computing is challengeable when big data such as medical and government confidential information is involved in a cloud environment. Without an efficient security encryption key management for every individual user in the mobile cloud, when handing big data, confidential information is not secured. To solve this problem relating to key management, the work suggested that cloud servers need efficient key management that employs quantum cryptography using Grover's algorithm. The results show that the hybridization of Grover's algorithm and Quantum cryptography provided efficient authentication techniques to improve the level of security with minimum complexity. The proposed system can only be applied to the analysis of the supervised approach that requires knowledge of the problem domain. The performance of the proposed system may not be visible in an unknown problem domain.

Qinglin et al. (2018) presented digital twin and big data towards smart manufacturing and industry 4.0: a 360-degree comparison. The authors stated that the developments in new-generation information technologies, especially big data and digital twin, smart manufacturing seems to be the focus of global manufacturing transformation and upgrading.

Kolajo and Daramola (2017) presented big data to combat terrorism in developing countries. The work proposed a model that harnesses data from multiple social media sources to detect terrorist activities using Apache Spark Technology. It described Social Media Analysis for Combating Terrorism (SMACT) model as a plausible approach that leverages big data analytics to address terrorism problems in developing nations. And considered Nigeria's context as a case for illustration for the proposed model to depict its viability as a potential panacea for handling terrorism. Strang and

Sun (2017) analyzed the relationships in terrorism big data using Hadoop and statistics. The work used big data software Hadoop in Google News to collect complex high velocity, high volume terrorism information. And it proposed a hypothesis that there is a significant relationship between terrorist group ideology and terrorist attack type. In addition to the test hypothesis, they developed an asymmetric model to visualize the hidden relationships between terrorist ideology and attack type. They concluded that the finding of a significant relationship between terrorist ideology and attack type may generalize to supply chain operations and national security planning.

In the work of (Palak *et al.*, 2017) that based on Twitter data in the form of tweets where clustering of those tweets is a major part of their study. Clustering analysis was performed using a Genetic Algorithm on Hadoop-based MapReduce and JAVA environment. Their results were promising. For future scope suggestions, they stated that there is a wide scope for the current study soon. Moreover, the nature of its optimization and being a metaheuristic technique in combination with any other technique will help to improve the efficiency of Big Data Analytics. The architecture of the Social Network Generated Big Data Clustering Using Genetic Algorithm is shown in Figure 2.10.

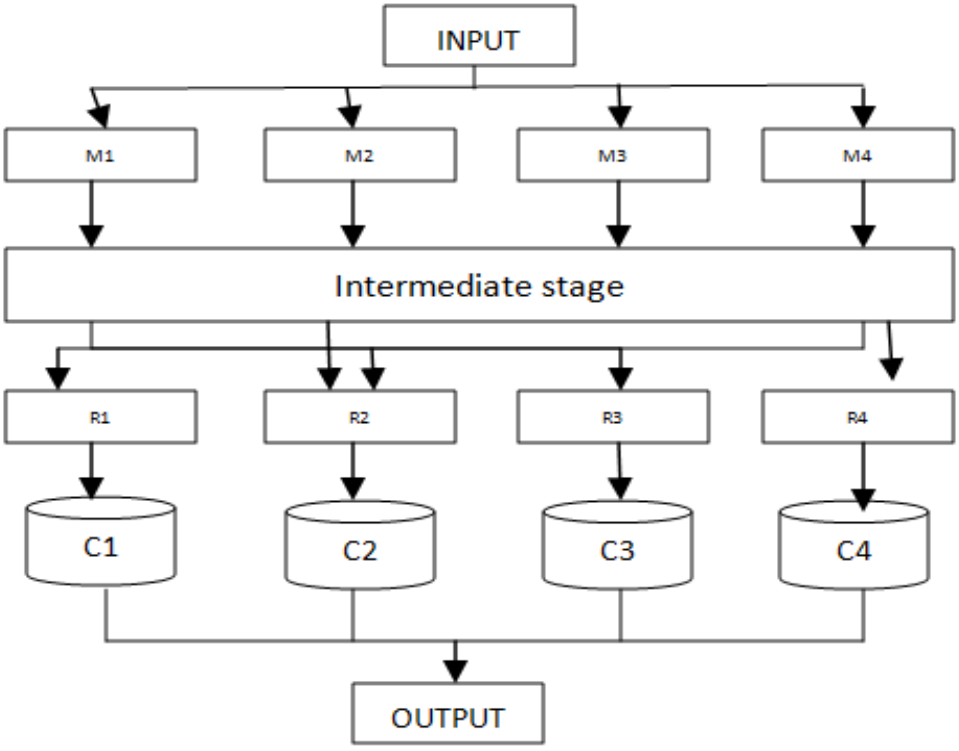

Figure 2.10: Architecture of Social Network Generated Big Data Clustering Using Genetic Algorithm (Source: Palak & Vikas, 2017).

The disadvantages of the social network generated big Data clustering using genetic algorithm are as follows:

1. The position updates of particles are performed randomly. It lacks conscious mutation.
2. The process of searching for the best/global particle is done at the same time this affects the time constrain of big data analysis process.
3. The exploitation and exploration of the process happen at a time instead of indifferent parallel phases. These drawback effects the accuracy and time constrain of big data analytics.

The work of (Aishwaray et al., 2017) presented handing big data analytic using swarm intelligence. The intention was to prove that the big data analytics problem can be solved using swarm intelligence and its application on Hadoop architecture. The authors used a Particle Swarm Optimization (PSO) algorithm to create clusters of a given dataset. The experimental results show that the PSO algorithm is efficient in solving the problems faced in big data analytics. They were not able to develop an effective hybridized algorithm with swarm intelligence to solve big data analytics problems. Big Data Analytics with swarm intelligence was presented by (Shi, Qingyu, & Quande, 2016). The authors described the association between big data analytics and Intelligent Swarm (IS) techniques. The mean and standard deviations of three PSO variants (such as Cooperative coevolving particle swarm Optimization (CCPSO), Competitive Swarm Optimizer (CSO), and Dynamic Multi-swarm Optimizer (DMSO) was compared to solve the problem with 1,000 decision variables. The results show that DMSO produces more accurate results than CCPO2 and CSO algorithms. They concluded that the solution found by PSO variants is good enough for solving challenges of big data analytics with a huge number of decisions. But the dataset they used for the experimental illustrations was formulated instead of using real-world datasets. (Cao, Cui, Shi, and Jiao, 2016) proposed “Big data: A parallel Particle Swarm Optimization-Back Propagation (BP) Neural Network Algorithm Based on Map Reduction”. The authors stated that “back-propagation neural network can solve complicated random nonlinear mapping problems; therefore, it can be applied numerous problems. However, as

the sample size increases, the time needed to train BP neural networks becomes lengthy and the classification accuracy decreases as well. To improve the classification accuracy and the iteration time of its convergences, they proposed a parallel design and realization method for a particle swarm optimization (PSO)-optimized BP neural network based on Map-Reduce on the Hadoop platform using both the OSP algorithm and parallel design. The work POS algorithm to optimize the BP neural network's initial weights and thresholds. And also, to improve the accuracy of the classification algorithm the MapReduce parallel programming model was utilized to achieve parallel processing of the BP algorithm". This solved the problem of hardware and communication overhead when the BP neural network addresses big data. The results showed higher significant classification accuracy and improved time complexity of the process. However, the neural network is a supervised learning model that is applicable in a known domain problem. The proposed system will not be efficient when solving problems that required unsupervised learning in an unknown domain.

Tour and Gangopadhyay (2016) proposed real-time big data analytics for predicting terrorist incidents. They developed a real-time terrorist incidents data collection system to gather terrorist incidents data from reliable source. The collected data was modeled to calculate the terrorism risk level of different locations and predicted future terrorist incidents. The prediction method provided a high precision value of 96.30% and high recall values of 100%.

The results of their work will guide terrorism analysts to improve counter-terrorism measures to prevent future attacks.

Einstein (2018) addressed the cybersecurity skills gap through cooperation, education, and emerging technologies. The work sets out examples of existing efforts and initiatives to tackle the gap and demonstrated that new technologies, such as machine learning and AI, can provide efficient mechanisms and measures to mitigate cybercrime threats. Maya (2019) proposed the application of machine learning and deep learning in cybercrime prevention. The author stated that the Threat of cybersecurity has drastically transformed over the past few years due to the large amount of data

produced every second and stored in various forms and on different platforms, which makes it necessary to develop techniques to curbed attacks and theft of critical information. The author discussed techniques for applying machine learning and deep learning to control cybercrime and compared the performance of various machine learning techniques such as Logistic Regression (LR), Classification and Regression Trees (CART), Bayesian Additive Regression Tree (BART), Support Vector Machines (SVM), Random Forest (RF), and Neural Network (NN) using precision, recall and F-measures. The work concluded that LR is a more preferred option among users due to the low false-positive mechanism. Also, it had the highest precision and relatively high recall in comparison with other classifiers under contemplation. Karie, Kenande, and Venter (2019) presented diverging deep learning computing techniques into cybercrime. In cyber forensics, what makes the process even tough for investigators is the fact that Big Data often comes from multiple sources and has different file formats. Forensic investigators often have less time and budget to handle the increased demands when it comes to the analysis of these large amounts of complex data for forensic purposes. The authors proposed a generic framework for diverging Deep Learning (DL) cognitive computing techniques into Cyber Forensics (CF) hereafter referred to as the Deep Learning and Cyber Forensics (DLCF) Framework. Their result was promising and it can be used to mitigate cybercrime.

Sathesh & Hemalatha (2014) proposed an innovative potential on rule optimization using Fuzzy Artificial Bee Colony Algorithms. It was used to optimize rules to get the best classification accuracy of real-world datasets. The proposed system was compared with the traditional bee colony optimization and particle swarm optimization algorithms. Another interesting work of (Marzieh, Ali, & Seyed, 2015) presented a new method of Fuzzy Clustering by using the combination of the Firefly Algorithm (FA) and the Particle Swarm Optimization algorithm. The system was used to compute the maximum distance between the cluster centers and the cost of clustering the datasets to improve the objective function of the fuzzy clustering algorithm. This approach improved the accuracy of the fuzzy clustering algorithm. But there was no clear intelligent procedure for switching the two swarm

optimization algorithms to get the global initial value of k. It will be time taking for both of the optimization algorithms to evaluate the high dimensional dataset before switching to the next optimization algorithm.

An algorithm for classification tasks on big data that is as accurate as ensemble methods such as random forests or gradient boosted trees was presented by (Sambasivan & Sourish, 2017). The results of their work show that their proposed system produced models that are easy to implement than the existing ensemble algorithm. But the mechanism of the random forests tree algorithm is only efficient in a well-known dataset domain to produce an efficient classification.

A new method of Fuzzy Clustering by using the combination of the firefly algorithm and the particle swarm optimization algorithm was presented by (Marzieh et al., 2015). The transitional technique applied in there are as follow; at first, PSO algorithm is performed for a while, and then it is transited to the FA algorithm and it also will be performed for a while, and it will return to PSO and this process will be repeated up to reaching the stopping condition. The proposed system was used to compute the maximum distance between the cluster centers and the cost of clustering the datasets to improve the objective function of fuzzy clustering algorithms. The results improved the accuracy of the fuzzy clustering algorithm. But there was no clear intelligent procedure for switching the two swarm optimization algorithms to the global initial value of k. It will be time taking for both the optimization algorithm to evaluate the high dimensional dataset before switching to the net optimization algorithm.

The work of (Prabha, Natesan, & Visalakshi, 2015) proposed a new variant of binary Particle Swarm Optimization and K-Prototype algorithms to reach a global optimal solution for the clustering optimization problem. Their proposed algorithm was implemented and evaluated on standard benchmarks dataset taken from the University of California Ivory (UCI) machine learning repository. Their comparative analysis proved that Particle Swarm based on K-Prototype algorithm provides better performance than the traditional K-modes and K-Prototype algorithm.

The disadvantages of particle swarm optimization and K-prototype Algorithm are as follows:

1. It took the binary swarm a longer time to explore the high dimensional database. Thought their result produced accurate clustering, but it requires much time for a swarm to quest for X particles moving around whole dimensional search space.
2. The process of searching for the best/global particle is done at the same time. This affects the time constrain of the big data analytic process.
3. K-prototype is not efficient in clustering unstructured mixed datasets.
4. The mechanism lacks conscious mutation to update particles' location; this affects the architecture of Binary Particle Swarm Optimization and K-Prototype algorithm is shown in Figure 2.11.

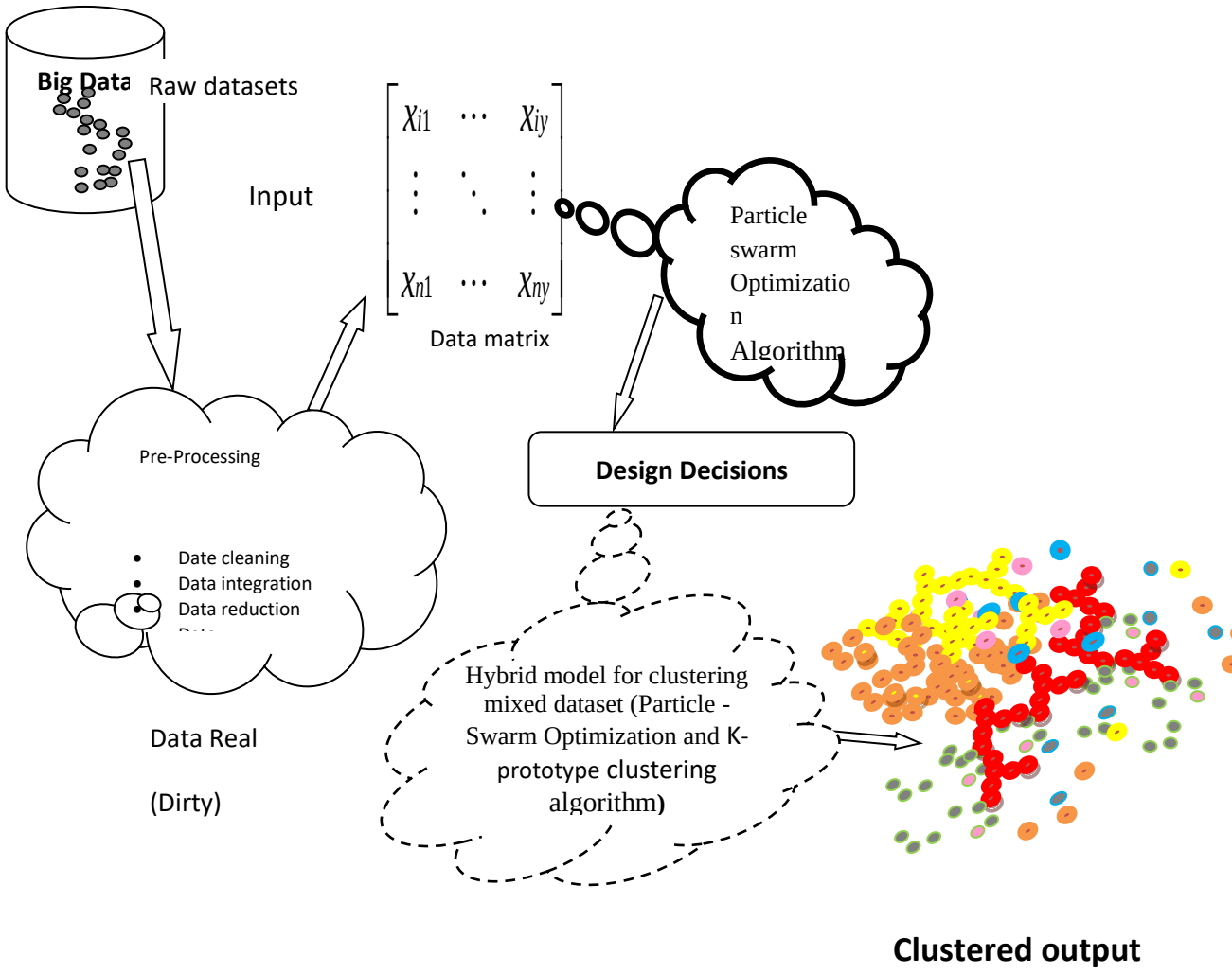

Figure 2.11: Architecture of Particle Swarm Optimization and K-Prototype Algorithm (Prabha *et al.*, 2015).

In the work of (Onuodu *et al*., 2014) the authors developed a hybrid method that made use of two algorithms with a few modifications. Their hybrid method extends the k-means algorithm to categorical domains and mixed-type attributes. Their hybrid system matched different clustering datasets with different algorithms to improve the drawback in the work carried out by (Asadi, Subba, Kisshore, & Raju, 2012). Subsequently, the distance measure used in the existing method was faulted because its distribution was of the frequency-based method. Therefore, a different dissimilarity measure was proposed by (Onuodu *et al*., 2014). They used a relative cumulative frequency-based method (RCF) in clustering objects with mixed datasets.

The Limitations of the Algorithm are as follows;

(i) The method can only guarantee a locally optimal solution.

(ii) There have been no reliable indices (or statistics) that can be used with both k-modes and extended k-modes to ascertain the true K (number of clusters) in the datasets.

(iii). Influence of outliers on creating pure clusters of datasets.

The architecture of the work of Onuodu et al is shown in Figure 2.12.

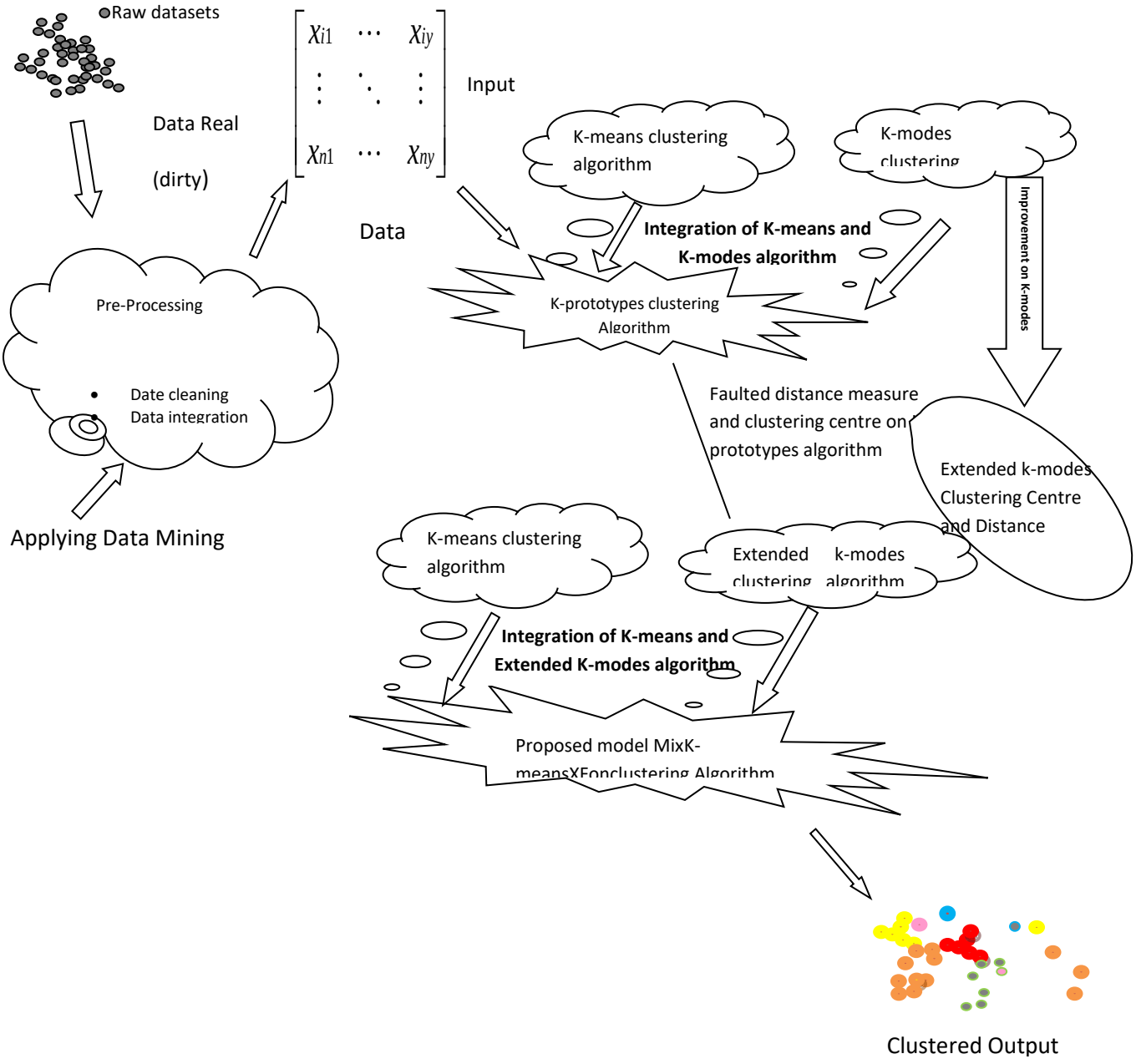

Figure 2.12: Architecture of the "Extension of K-means algorithm" for mixed Dataset.

(Source: Onuodu *et al*., 2014)

Most of the works done from 2009 to 2015 was mainly on the enhancement of the accuracy of unsupervised learning techniques like that of (Mohanavalli & Jaisakthi, 2015) that proposed a chi-square based statistical approach to determine the weight of the attributes. This weight vector is used to derive the distance matrix of the mixed dataset.

The distance matrix is used to cluster the data points using the traditional clustering algorithms. Experiments have been carried out using the UCI benchmark datasets, heart, credit, and vote. Apart

from these data sets they also tested our proposed method using a real-time bank data set. The accuracy of the clustering results obtained was better than those of the existing works". The authors could not extend the proposed system to improve the computation of attribute relevance to be used as the corresponding weights in distance computation in mixed data clustering. Its limitation is the initial selection of k values. This will lead to premature convergence.

An approximate algorithm based on k-means was analyzed by (Jain *et al*., 2014). It is a novel method for big data analysis which is very fast, scalable, and has high accuracy. It overcomes the drawback of k-means of an uncertain number of iterations by fixing the number of iterations, without losing the precision. The efficacy and precision of the algorithm are demonstrated on various real and synthetic datasets. The algorithm presented here cannot handle categorical data well until it is converted into equivalent numerical data. Exploring clustering big data in terms of categorical data could be another possible extension. Deciding primary and secondary attributes is considered in the proposal to be provided as an input by the user (which indicates the viewpoint of study). The proposed system inherited the problem of the initial selection of k values. This will lead to premature convergence. Machine learning concepts can be used to decide the priority of attributes instead of asking from the user. An adaptive K-means clustering technique for data clustering to modify k-means clustering was proposed by (Manjinder, Navjot & Singh., 2013). The proposed system adapts itself according to the image based on color-based clustering. The no. of clusters using the color features are computed based on histogram analysis in gray format. The peak of the histogram is the main source of computation of no. of colors in the image and based on the same, the image data are clustered. In the work of (Chouhan & Chauhan, 2014), which proposed an ameliorated partitioning clustering algorithm for large datasets. Ameliorated k-Medoid clustering algorithm will have accuracy more than the original one. It uses weights of attributes and the edit-distance method to get similarity values between records, and then it detects duplicated records by forming clusters of these similarity values. This algorithm can adjust the number of clusters automatically by comparing similarity value with

the present similarity threshold, and it also avoids a large number of I/O related operations which is used by the conventional "sort/merge" algorithm for sequencing. It initially computes the initial centroids k as per the necessity of the user and then provides an improved and efficient cluster with no sacrifice on accuracy. It generates steady clusters to get better accuracy. It also minimizes the mean square error and improves the quality of clustering, reduces the number of iterations, and works on reducing time complexity. The improved k-medoid clustering algorithm had accuracy greater than the original one. The initial selection of the value of k is a problem inherited by this algorithm.

Mutual information (MI)-based unsupervised feature transformation (UFT) proposed by (Wei, Chow &. Chan, 2015), can transform non-numerical features into numerical features without information loss, was utilized with the conventional k-means algorithm for heterogeneous data clustering. For the original non-numerical features, UFT can provide numerical values that preserve the structure of the original non-numerical features and have the property of continuous values at the same time. Experiments and analysis of real-world datasets showed that the integrated UFT-k-means clustering algorithm outperformed others for heterogeneous data with both numerical and non-numerical features. The authors could not visualization heterogeneous data. By unifying the data to be purely numerical, the UFT can enable Principal Component Analysis (PCA) which can be useful for data visualization of heterogeneous data. In the work of (Ramakrishnan & Jayaraj., 2014), which proposed modified k-means algorithms that can perform clustering very effectively on mixed data sets. To organize the closest pattern vectors efficiently. To perform this, they built a k-d tree that can organize pattern vectors, root representing all the patterns and child nodes representing subsets of patterns completely contained in subspaces. Each node in the tree contains information such as the number of points (m), a linear sum of points (LS), and a square sum of the points (SS). The k-d tree is built with the following choices: (i) Using common dimensions for all the nodes at the same level of the tree. (ii) Another option is to use splitting dimensions. Here dimensions are chosen in a round-robin fashion.

The dimensions can be dividing into two equal parts. Another option is to divide the dimensions so that an equal number of patterns exists on both sides. It is observed that the first approach costs higher than the later one. For each iteration k-d tree is traversed by using the depth-first approach, starting from the root node to all prototypes. The pruning function is applied to each node. If the number of candidate prototypes is equal to one, the traversal below that internal node is not pursued. The cluster information is constantly updated about the points, LS and SS. K-means algorithm is applied on the leaf node if there is more than one candidate prototype. The experimental results show that their algorithm improved the efficiency of the clustering method. The authors could not completely minimize distance calculation. A non-parametric clustering method based on a weighted modified Chi-square test was proposed by (Yawen, Hongchang, Shaomei, Jianpeng, & Chao, 2014). Numerical results show that the proposed method outperforms the Auto-Class algorithm based on classification rate and entropy measure for various simulation settings. The proposed method was most useful when neither a distance function nor a parametric model can be assumed. They could not cluster spatial and temporal data. A novel extension of k-means clustering to mixed divergences. Second, we extend the k-means++ seeding to mixed -divergences and report a guaranteed probabilistic bound was presented by (Nielsen, Nock, & Shun-ichi, 2014). Finally, they describe a soft clustering technique for mixed-divergences. The results are still ongoing.

An efficient algorithm that uses the techniques of Divide and Conquers to cluster large datasets was proposed by (Ahirwar, 2014). The author applied the Squared Euclidean Distance in measuring the similarity between data points. The clusters obtained from each partition of the datasets were merged to get more superior clusters. The idea in the proposed system is that the advantage of using the divide and conquer approach in the work requires a small amount of physical memory to load the datasets into the system compared with the amount of physical memory the large datasets would have required. The reason is that the clustering will be done on parts of the datasets instead of the entire datasets. The authors compared the performance of the proposed algorithm with the F-measure,

purity, and Entropy, and the results show that the approach used was much better than the existing algorithms. Although the approach is very efficient in identifying the data points and assigning the data points to the best clusters, it could not read the entire data files at once.

The work of (Ramesh, Ramar, & Babu, 2013) proposed a parallel k-means algorithm to cluster large datasets using agricultural databases. They experimented with agricultural datasets due to limited researches that have been done in the agricultural field. The authors aimed to improve the time complexity inherent in the k-means algorithm when applied on large datasets since it used to be computationally expensive. Parallel computing is the simultaneous use of multiple computer resources to solve a computational problem, where the problem is broken into discrete parts that can be solved concurrently; and each part is further broken down to a series of instructions. The instruction from each part is executed simultaneously on different CPUs. The work employed "Flynn's taxonomy, which distinguishes multiprocessor computer architectures according to how they can be classified along the two independent dimensions of instruction and data. Each of these dimensions has only one of the two possible states, single or multiple. Flynn's approach uses a matrix, which defines the four possible classifications; Single Instruction, Single Data (SISD), Single Instruction Multiple Data (SIMD), Multiple Instruction, Single Data (MISD), and Multiple Instruction, Multiple Data (MIMD) respectively". Although the performance of the parallel k-means algorithms proved to be better in terms of efficiency and time complexity compared with the normal sequential k-means, the problem is that their processing speed was disturbed with the bandwidth of the network available.

A novel defect segmentation of fruits based on color features with k-means clustering unsupervised algorithm was proposed by (Dubey, Dixit, Singh, & Gupta, 2013).

The authors used color images of apple fruits and this was done in two stages. At first, the pixels were clustered based on their color and spatial features. And secondly, the clustered blocks were merged to a specific number of regions. These two procedures were able to increase the computational

efficiency, at the same time avoiding feature extraction for every pixel in the image of apple fruits. The experimental results show that the proposed k-means-based defect segmentation was able to accurately segment the defected area of fruits present in the image. Although the proposed approach appears promising, it could not automatically determine the number of clusters required to segment the defects more accurately. Kaur & Kaur (2013) presented a K-means clustering algorithm to make groups or clusters of given datasets based on the similarity between them. The author's major aim was how to overcome certain limitations inherent in K-means especially the time of execution. Ranking Method and Query Redirection was used to reduce the time of execution. The work claimed that by using the query redirection approach, users of web engines are sure of getting 90% correct queries that are related to their search in less execution time as compared with the traditional methods. Although the experimental results show that the K-means algorithm is efficient and the users can find better results corresponding to their queries at a decreased execution time, they could not fully apply this method in a distributed environment for large samples of datasets that are also distributed in nature.

Zhang & Fang (2013) introduced the idea of the K-means clustering algorithm analysis. The authors offered two methods of improving the K-means clustering algorithm based on improving the initial focal point and secondly, determining the value of K (where K is the number of clusters). The simulation experiments proved that the improved clustering algorithm was not only stable in the clustering process but reduced or even avoided the impact of the noise data in the data set object. The authors claimed that all of these efforts were to ensure that the final clustering result was more accurate and effective. Although the experimental results appear promising, research on the improvement of K-means clustering algorithms are still not solved completely, rather, it requires further attempt and exploration. A clustering algorithm based on genetic K-means was proposed by (Pinisetty, Valaboju, R & Rao, 2012). The algorithm worked well for both numeric and discrete values in terms of efficiency. The enhanced cost function and the modified cluster center was used in

genetic K-means algorithm to capture cluster characteristics effectively and efficiently. The authors reviewed the additional features encountered in genetic K-means algorithms which include minimizing or avoiding string elimination overhead, mutator operator, and total within-cluster variation (CVs). Notwithstanding their experimental achievements, they could not extend the performance measurement comparison to the number of computations performed by the algorithm and the memory consumption of the cost analysis to the existing analysis on the time by Genetic K-means Clustering Algorithm when compared with GKMODE and IGKA algorithms. Secondly, the algorithms could not be properly enhanced to suit various benchmark datasets, so that the compared results can be analyzed and reported to provide more meaningful insights into the study.

The work of (Kanjanawattana, 2012) proposed a new algorithm called "MixK-Means++" and applied the same to mixed types of datasets. The author was of the opine that most data in the actual world is always mixed of two types, the numeric and categorical one. He accepted that most K-means and K-mean++ could handle numeric datasets only and cannot be applied with real-world data. Therefore, their new algorithm "MixK-means++" would mitigate such setbacks. The advantages of MixK-means++ works well with mixed data types and still retains the strength of K-means and Kmeans++. The experimental results show that MixK-means++ can work better than K-means even though their input is a mixed dataset. Although their experimental results are better in terms of speed and performance than K-means, the approach could not guarantee that the running time and the number of iterations followed its expectation. A cluster of mixed numeric and categorical datasets in an efficient manner was proposed by (Asadi et al., 2012). The authors used a clustering algorithm based on similarity weight and filter method paradigm, that works well for data with mixed numeric and categorical features. Although the approach is very efficient in solving any number of dimensions, reduce time complexity and the irregular boundaries of the filter algorithms, the problem is that they could not mix the different clustering datasets with different algorithms.

The work of (Pham & Suarez, Prostov, 2011) proposed a new algorithm called RANKPRO (an acronym for Random Search with K-prototypes Algorithm) to cluster datasets with mixed numerical and categorical values. The new algorithm proposed combines the advantages of the most recently introduced population-based optimization algorithm called Bees algorithm (BA) and K-prototypes. The work shows that BA works with elite and good solutions, as it continues to look for other possible extreme solutions while keeping the number of testing points constant. BA may be time-consuming because it is based on random neighborhood search. But the introduction of K-prototypes algorithm to a promising solution is very effective since it improves the solution at each iteration. The RANKPRO algorithm explores the search space effectively owing to a random selection of new solutions. Although RANKPRO algorithm is more efficient than K-prototypes algorithm for a specific dataset, the K-prototypes algorithm converges to a local minimum very fast in a few iterations and the algorithms may have approximately equal effectiveness.

A new method based on a combination of supervised and unsupervised learning of clustering data without any preliminary assumption on the cluster shape was implemented by (Kolhe & Zinjore, 2010). This was achieved by extracting the dissimilarity relations directly from the available data. A Feed-Forward Neural Network (FFNNs) was trained with a few known dissimilarity degrees between pattern pairs. The neural dissimilarity relation served as input to the fuzzy clustering algorithm. The experimental results show that the clusters are created correctly when the attributes are petal length verses petal width having an accuracy" of 92.38% for 35% of patterns. Although their results were encouraging and trustable, they could not separate Iris virginica from Iris versicolor when using a typical Iris flowering plant as a case study.

The work of (Sowjanya, Ravindra and Kumar, 2014) proposed a new distance measure based on the weightage, which is automatically generated and applied to the incremental clustering algorithm. The incremental data points were handled in two phases, the first phase used the K-means clustering algorithm, which is employed for initial clustering of the static database. In the second phase, the

designed distance measure was used to generate the appropriate cluster for the incremental data points. The results show that the combination of the two algorithms proved to be more effective in handling mixed datasets consisting of numerical and categorical attributes only. That notwithstanding, the issue of defining the right distance measure and choosing different K-values and thresholds are still open problems. The work of (Yiu-ming and Hong, 2011) proposed a unified metric for categorical and numerical attributes in data clustering, in which the attributes are in either one of the three: numerical, categorical, and mixed. The authors further presented a general clustering framework based on the concept of object-cluster similarity. Finally, they developed an iterative clustering algorithm, which is directly applicable to the three data types in question without any adjustment. Although their experimental results show the efficacy of the proposed approach, there is no user-assigned parameter in the proposed algorithm. From a practical point of view, this is not suitable for real-life applications. A Fuzzy C-means (FCM) clustering algorithm, which allows one piece of data to belong to two or more clusters was proposed by (Dewangan & Ambhaikar, 2010). From the analysis results, it was observed that the traditional fuzzy c-mean type algorithm is limited to numeric data only. Subsequently, the authors presented a modified description of cluster center to overcome the numeric data only limitation of fuzzy c-mean algorithm and provide a better characterization of clusters, which is the fuzzy k-modes algorithm for clustering categorical data. Furthermore, a new cost function and distance measure proposed that is based on co-occurrence of values, which can also take into account the significance of an attribute towards the clustering process. The fuzzy k-modes algorithm for clustering categorical data is extended by representing the clusters of categorical data with fuzzy centroids; which makes it possible to fully exploit the power of fuzzy sets in representing the uncertainty in the classification of categorical data. Although the new fuzzy k-modes algorithm is effective and better than the other existing k-modes algorithms, their algorithm could not combine the two implementations of fuzzy c-means and fuzzy k-modes for a mixture of data items set. Secondly, the authors could not find out the cluster area and center of the

cluster. The work suggested that the Fuzzy C-mean (FCM) algorithm for numeric data should be implemented and the Fuzzy K-modes algorithm for categorical data be also implemented separately. Then combine both implementations for a mixture of data items that is numeric as well as categorical data items to produce the final results. The work of (Ming-Yi, Jheng, & Lai, 2010) presented a new two-step method for clustering mixed (TMCM) categorical and numeric data. In this approach, the items in categorical attributes are processed to construct the similarity or relationships among them based on the ideas of co-occurrence. Then, all the categorical attributes can be converted into numeric attributes based on these constructed relationships. The authors believe that since the categorical data are converted into numeric, the existing distance-based clustering algorithms can be employed to group mix types of the dataset without pain. Subsequently, to overcome the weaknesses of the k-means clustering algorithm, the TMCM algorithm integrates Hierarchical Agglomerative clustering (HAC) and partitioning clustering algorithm (k-means) to cluster mixed types of data. Although the experimental results show that the proposed approach could achieve a high quality of clustering results, they could not set the parameter correctly. That is, how to set this parameter of the number of subsets, which is one-third of the number of objects, is worth more study.

A modified k-means algorithm for clustering mixed data sets was proposed by (Ahmad & Dey, 2007). The authors introduced a new distance measure for categorical attribute values. They also proposed a modified representation for the cluster center. Though similar representations have been used for fuzzy clustering, it was used in a novel manner to compute the distance of an object from a cluster center. The significance of this representation is that it captured cluster characteristics very effectively since it contained the distribution of all categorical values in a cluster. The results obtained with their algorithm over several real-world datasets are highly encouraging. Although their results were encouraging, their methods could not address the discretized numeric valued attributes, which seldom leads to information loss.

The work of (Chaturvedi, Green, & Caroll, 2001) presented "a nonparametric approach of considering clusters for categorical (or nominal scale) data using a new clustering procedure called k-modes, analogous to the traditional k-means procedures for clustering interval scale data. The authors claim that in Monte Carlo simulation, both k-modes and latent class procedures performed with equal efficiency in recovering a known underlying cluster structure. The work reported that k-modes procedures not only optimized a loss function based on the Lo norm defined as the limit of an Lp norm as p approaches zero, but is faster and suffers from fewer problems of local optima than the latent class simulation". Despite their achievements, this approach could not complement the two procedures when dealing with real-life market segmentation applications. The work of (San et al., 2004) presented the notion of cluster centers on datasets of categorical objects and how to use this notion for formulating the clustering problem of categorical objects as a partitioning problem. The results show that the proposed algorithm gives better results, and appears to be more stable than the k-modes algorithm. Despite all these achievements, they could not combine the proposed algorithm with the k-means algorithm in a similar manner as is done in (Huang, 1997; Huang, 1998), using the k-means paradigm to the clustering of mixed datasets.

The work of (Elkan, 2003) "presented an accelerated K-means algorithm by applying the triangle inequality in two different ways, and by keeping track of lower and upper bounds for distances between points and centers. The proposed algorithm is effective when the datasets were up to 1000 dimensions, and becomes very effective as the number k of clusters increases. Although the algorithm is effective, it could not be applied to obtain upper bounds on weights that are close to zero and to obtain approximate soft assignment solutions quickly". "This idea of no best clustering algorithm is partially captured by the impossibility theorem by (Kleinberg, 2002); which states that no single clustering algorithm simultaneously satisfies a set of basic axioms of data clustering". The author is of the opined that algorithms developed may give best results with one type of data set but may fail or give poor results with data set of other types. Although there have been many attempts to

standardize clustering algorithms to obtain algorithms that can perform well under different scenarios, it is still the case that each algorithm has its own merits and demerits and cannot work for all real situations. The fields where clustering algorithms are used are quite diverse. “It is difficult to exhaustively list the numerous science fields and applications that have utilized clustering techniques as well as the thousands of published algorithms”. Two algorithms that extend the k-means algorithm to the categorical domain and mixed values analysis was presented by (Huang, 1998). The k-modes algorithm used a simple matching dissimilarity measure to deal with categorical objects, replaced the means of clusters with modes. Consequently, they used a frequency-based method to update modes in the clustering process to minimize the clustering cost function. Despite their achievements, they acknowledged that they had a priori knowledge of the datasets, which is not common in any practical data mining applications by (Huang, 1998).

“From the viewpoint of target dataset for analysis, existing clustering algorithms can be classified into three categories namely; numeric, categorical, and mixed. Most previous clustering algorithms focus on numerical data whose inherent geometric properties can be exploited naturally to define distance functions between data points”. These algorithms include DBSCAN, BIRTH, C**2**P, CURE, CHAMELEON, WaveCluster (Nanopoulos and Yannis, 2001) reviews that numeric clustering algorithms are not suitable and appropriate for categorical attributes. Therefore, it is easy to conclude that they are also not suitable for the task of clustering mixed-type attributes. A few algorithms have been proposed in recent years for clustering categorical data (Wang, Xu, C., & Liu, 1999). However, all of these algorithms are designed for categorical attributes and their extending capabilities to mixed type attributes are unknown.

For the problem of clustering mixed-type attributes, some research efforts have been done (Li *et al*., 2002). Two algorithms, k-modes and k-prototypes, which extend the k-means paradigm to categorical domains and domains with mixed attributes was presented by (Chiu, Fang, Chen, & Wang, 2001). A new distance measure for categorical attributes based on the total mismatches of the categorical

attributes of two data records is proposed in the k-modes algorithm. For the mixed type of attributes, the k-prototypes algorithm used a weighted sum of Euclidean distance for numeric attributes and the proposed distance measure for categorical attributes. However, the weights have to be determined a priori. This is because improper weights may result in biased treatment of different attribute types. The work of (Chiu *et al*., 2001) "introduced a clustering algorithm that is available commercially in the Clementine 6.0 data mining tool. The distance measure is derived from a probabilistic model between two clusters, which is equivalent to the decrease in log-likelihood function as a result of merging. The clustering algorithm is based on the framework of BIRTH, using the proposed distance measure. The BIRTH algorithm has the drawback that it may not work well when clusters are not 'spherical' and it can be affected by the input order of records". Thus, the same problems exist for the clustering algorithm (Wang *et al*, 1999). Extended K-mean clustering technique using map reduced segmentation clustering for big data analytics. To measure the performance of the scaled k-means algorithms using Hadoop MapReduce, we have executed the algorithms on 10 different samples of data. After execution of the algorithm, we have calculated and measure the inter-cluster and intra-cluster similarity measure. Prasad and Raju (2020) proposed extended K-mean clustering technique using map reduced segmentation clustering for big data. They mechanism applied an optimized repartitioned k-means clustering algorithm applied to large dataset which contain around 10 million objects. The results show that the use of Hadoop and MapReduce framework provides high performance in big data analysis. Saeed, Aghbari & Alsharidah (2020) propose a new taxonomy for the Spark-based clustering methods. The survey aims to present a comprehensive summary of the previous studies in the field of Big Data clustering using Apache Spark. The review also highlighted the new research directions in the field of clustering massive data. Through their survey they found that most existing Spark-based clustering method support the volume characteristic of Big Data ignoring other characteristics. Therefore, future research should focus on other characteristics such as variety and velocity. Additionally, future Spark-based clustering method should investigate new

features: concept drift, scalability, integration, fault-tolerance, consistency, timeliness, load balancing, privacy, etc. (Saeed, Aghbari & Alsharidah, 2020).

The summary of related works is shown in Table 2.7.

Table 2.7: Summary of Related Work

Author	Related Work Done	Result	Limitations
Kumar *et al*. (2019)	presented "a novel clustering technique for efficient clustering of big data in Hadoop Ecosystem	The results show that their proposed system produced accurate clusters, and has better precision, recall, and F-measure values more than the existing algorithms	The dimensionality and massive nature of big data analysis may affect the hybrid of k-means and hierarchical clustering algorithms. And the time complexity of the process would not be efficient.
Palak et al., (2017)	Presented social media generated big data clustering using genetic algorithm.	Their results show that the proposed algorithm is efficient in handling large amount of data.	But the exploration of the high dimensional dataset processes happens at a time instead of in parallel different phases. These drawback effects the accuracy and time constrain of analyzing big data.
Aishwaray et al (2017)	Presented big data analytic and it application in Hadoop architecture using swarm intelligent.	Their experimental results show that PSO algorithm is efficient in solving the problems faced in big data analytics.	They were not able to develop effective hybridized algorithm with swarm intelligent to solve big data analytics problems.
Onuodu, and Nwachukwu, (2014)	proposed a new hybrid method for dissimilarity measure that uses relative cumulative frequency-based method in clustering objects with mixed values.	Their results show a significant improvement than the existing system	The limitation of the new algorithm is that the value of k, which is the number of desired clusters, is still required to be given as input, regardless of the distribution of the data points in clustering mixed data.

2.5 **Highlight Summary of Related Works**

An attempt was made by (Onuodu et al., 2014) to improve the purity of the clustering center of the k-prototype algorithm for mixed datasets using the relative cumulative frequency-based method in

clustering objects with mixed datasets values. In the same manner, as observed in literature reviews, their work inherited the influence of outliers and premature convergency of the existing K-means clustering algorithm. Therefore, it implies that it does not have the analytic intelligence to handle Big data characteristics such as high voluminous, and heterogeneous data types. Another limitation of their algorithm is that the value of k, which is the number of desired clusters, is still required to be given as input, regardless of the distribution of the data points in clustering mixed data".

The work of (Prabha *et al*., 2015) proposed a new variant of binary particle Swarm Optimization and K-prototype algorithms to obtain a better optimal solution for clustering optimization problems. Though, PSO possesses the ability to handle high dimensional datasets of big data but lacks the strength to re-schedule search agents for additional tasks. Also, it took a long time to explore the high voluminous nature of a big dataset because it deploys single swarm instead of multiple swarm (search agents). And it causes it to produce inaccurate results. Also, the K-prototype algorithm lacks the strength to handle high volume and heterogeneous data types. Another attempt was made by (Palak *et al*., 2017) that presented Social media generated Big data clustering using a genetic algorithm. Their proposed system is self-sufficient to handle big data in global space under the paradigm of Hadoop MapReduce. The mechanism applied in the Genetic algorithm manages between exploitation and exploration of the high dimensional space of the datasets. The exploitation and exploration of the process happen at a time instead of different parallel phases. These drawback effects the accuracy and time constrain of big data analytics. The weakness of each component of their proposed hybrid analytical system will cause the algorithm to perform poorly considering the high dimensional dataset of big data and its heterogeneous data types.

The work of (Kumar *et al*., 2019) is titled "A Novel clustering technique for efficient clustering of big data in the Hadoop Ecosystem". They intended to improve the performance of the k-means clustering algorithm. They hybridized k-means and hierarchical clustering algorithms to overcome the disadvantages of the K-means clustering algorithm. But K-means algorithm converges

prematurely and hierarchical clustering is less efficient than K-means clustering in running time. The hybrid of K-means and hierarchic may produce week clusters because of the influence of unique features of Bigdata such as high dimensionality and the massive nature of big data during the analysis. The research gaps observed from various literatures reviewed in this work, on the issue of big data processing, storage and analytics includes: inability of the existing literatures to efficiently handle the high dimensional dataset of big data; heterogenous data types cause the traditional algorithms to converge prematurely and the inability to produce accurate clusters due to influence of outliers and voluminous nature of big dataset.

The procedures and techniques (such as machine learning, optimization, swarm intelligence algorithms and etc.) used by different researchers to solve the problems of big data analytics will be a guide to improve the limitations of the existing models. This work will use the principles of mathematical concepts (such as Reference Distance Weighted (RDW) of dissimilarity measure, Euclidean distance measure and Chi-square relative frequency of dissimilarity measure) to develop an improved robust unsupervised learning algorithm that will be hybridized with swarm intelligent algorithms to solve the problems of big data analytics. The proposed method will consist of partitional clustering algorithms an unsupervised learning model and dynamic multi swarm optimization algorithm. The outcome of this research would produce efficient clustering convergence and minimize the time complexity for clustering large mixed dataset.

The comparison of the performance indicators of the proposed big data analytics model and the existing models is shown in Table 2.8.

Table 2.8: Comparison of the performance indicators of the proposed big data analytics model and the existing models

Proposed Improved Big Data Analytic Model	**Binary Particle Swarm Optimization and K-Prototype algorithms** Prabha *et al*. (2015)	**Genetic Algorithm.** Palak *et al*. (2017)	**K-means and Hierarchical Algorithms** Kumar *et al*. (2019)
Parameters of Dynamic-K-reference clustering algorithm	**Parameters** of Binary Particle Swarm Optimization and K-Prototype algorithms	Meta-Heuristic Algorithm	Unsupervised and Supervised Algorithms
➢ **K-reference Clustering Algorithm** ✓ Reference Distance Weighted (RDW) of dissimilarity measure, (Reference Factor) ✓ Euclidean distance measure and, ✓ K-means Algorithm, ✓ Chi-square relative frequency of dissimilarity measure ✓ Dynamic Multi-Swarm Optimization Algorithm **HYBRID MODEL** ➢ **Dynamic-K-reference Clustering Algorithm.** ➢ Dynamic Multi-Swarm Optimization Algorithm + K-reference Clustering Algorithm	➢ **K-**prototype ✓ K-means Algorithm ✓ Euclidean distance measure and, ✓ K-Mode Algorithm ✓ Particle Optimization Algorithm **HYBRID MODEL** ➢ POS-K-prototype Clustering Algorithm. Particle Swarm Optimization Algorithm + K-prototype Clustering Algorithm	➢ Genetic Algorithm.	➢ K-means and Hierarchical Algorithms

Table 2.8 describes the parameters of the developed Dynamic-K-reference clustering algorithm and the existing algorithms. These results help to describe clearly the mechanisms used in each of the analytic models.

CHAPTER THREE

METHODOLOGY

3.1 **Feasibility Study**

Feasibility study was carried out in this work to assess the practicality of developing the proposed improved model for big data analytics using swarm optimization and unsupervised machine learning algorithms. The outcome of the feasibility study guided the researcher to foresee bottlenecks that may delay the development of this project in a stipulated time. The steps of the feasibility study are as follows:

Step one: Training datasets

The validation of the proposed bigdata analytic model requires a real world unstructured mixed large dataset to verify the veracity. Various sectors of national problems areas were considered as source of data collection in Nigeria such as Education, Industrial productions, Agriculture disease control, Mobile network, Cybersecurity, Terrorist attacks. Boko Haram insurgency attack menace data was considered as the study area, because the information required on it can be accessed on the social media. Also, efforts were made to collect terrorist attack data from START's Global Terrorism Database (GTD) and Statistical Annex Dataset. It was not successful. The reason was that the management of START's Global Terrorism Database does not release information to an individual. The available option to collect big dataset for this project was through the social media web sites using scrapping software (like ScrapeStorm)

Step two: Analytic Features

The researcher had an interactive section with some subject experts on national security challenges to suggest scenarios that can typically lead to the outcome we are trying to analyze. Also, to deepen the understanding of what drives the target outcome of this research, interviews and literature review was carried out on data for Boko Haram Insurgence menace in the

Northern Nigeria. The findings were still unclear on the qualitative understanding of what drives the target outcome. Although a little direction of what data to be scrape from social media for the analysis were achieved.

Step three: Data Source

The previous steps gave directives on the type of data needed to build the proposed model successfully. The divulged knowledge from previous steps helped figure out where the data for each analytic feature will come from. The cost of collecting data from different data sources such as Internal data, external data, data aggregators (Vendors), Web site (Social Media) was reviewed. Moreover, social media sites including Facebook, Twitter, Instagram, etc. was also reviewed for source of data. Decision was made to scrape terrorist attack data from twitter website, considering that scrapping is against the terms of use of most large sites.

Step four: Level of effort

Estimating level of effort for machine learning analytic project is challenging because they usually contain experimental or iterative phases. The stipulated timeline for the project may expand based on the outcome of exploitations of the iterative phases. It was considered breaking the required task into fairly grainy steps, each with level of effect attached considering the researcher's professional skills in analytic models, algorithms and programming languages. With this approach, steps with little uncertainty (such as data scrapping and preprocessing) gets a dependable level of effort, whereas steps with large uncertainty (such as coding and debugging model prototype) can get level of effort range.

Study plan: The researcher designed project study plan that delivers incrementally to meetup the stipulated timeline for project development.

Step five: Model lifecycle

An appropriate consideration was made for the entire project lifecycle based on this feasibility study. There is a strong confidence that the proposed project will be a success within the stipulated timeline.

3.2 System Methodology

System methodology is the framework used to harmonize the structure, plan, and controlling the process of developing Big data analytic model. Object-Oriented Analysis and Design (OOAD) methods were adopted for analysis and design in this project, because it encourages planning and development of system that are truly independent of one another. The basic steps of system designing using Object Modeling (Unified Modeling language) are as follows: System Analysis, System Design, Object Design and Implementation. The diagram is shown in Figure 3.1.

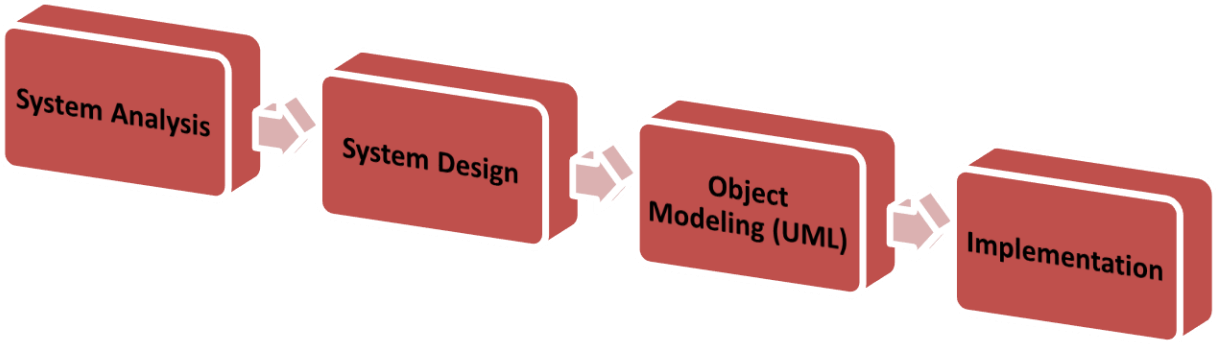

Figure 3.1: Object Oriented Analysis and Design Methodology

3.2.1 System Analysis

In this work, to gain insight on how to develop an efficient model over the challenges big data analysis posed on traditional data mining models such as partitioning clustering algorithms (like k-means) and meta heuristic algorithms (e.g. Swarm Intelligence). This work examined and studied the various mechanism of the existing big data analytic models to identify their goals and purposes and develop systems and methods for optimal performance. This problem-solving technique will help to improve the limitations of the existing models and the implementation of the proposed big data model.

The traditional k-means algorithm comes with some setbacks which has led to the developments of other algorithms (like K-mode, K-medoid, K-representative, K-prototype, etc). These algorithms were developed in order to tackle specifically the setbacks found in k-means. Some of the weaknesses known to be associated with the traditional k-means are as follows: random guess of the starting position, influence of outliers, problem of analyzing heterogeneous data type, circular shaped model, etc. The k-modes algorithm extends the k-means algorithm by using a simple matching dissimilarity measure for categorical objects, modes instead of means for clusters, and a frequency-based method to update modes in the clustering process to minimize the clustering cost function. The extension has removed the numerical limitation of the k-means algorithm and enables the k-means clustering process to be used efficiently to cluster large categorical datasets from real world databases.

In summary, the number of clusters, K, in a given dataset must always be known in advance. The k-means algorithm can only handle numeric data efficiently, but performs poorly with categorical or mixed data. The results strongly depend not only on the initial prototypes and sensitivity of noise and outliers, but also on the problem of dead prototypes (or empty clusters) that converges to local optima. Furthermore, most machine learning algorithms include partitioning clustering algorithms (like k-means, k-mode, k-representative, etc.), metaheuristic algorithms (swarm intelligence), and classification algorithms (like a neural network, tree algorithms) are bottled-up during data classification/clustering due to outliers, voluminous nature of big data, and the heterogeneous nature of a big dataset. These problems affect the Veracity or Reliability of the analytical results for decision making. However, many researchers tend to solve these problems but end up creating more problems to be solved.

3.2.1.1 Users Requirements

The methodology used in this work is based on the following requirements:

Step one: The analysis and design of improve clustering center of the traditional partitioning clustering algorithm using the mathematical notion of Reference Distance Weighted (RDW).

Step two: Integration of the developed clustering center with existing clustering algorithms to develop an improved unsupervised machine learning algorithm for clustering mixed datasets.

Step Three: Analysis and design of the hybrid of the improved unsupervised machine learning algorithm with Swarm Optimization algorithm for bigdata analytics.

Step Four: Design of an application program to test the performance of the proposed big data analytic model using Java programming Language.

Step Five: Design of a visualization application program for knowledge discovery visualization using Apache Spark, Jupyter notebook, and python programming language.

Step Six: Scrapping of Unstructured big data of terrorist attack from social media (Twitter web site) for implementation of the proposed system and performance evaluation. Also extracting machine learning experimental datasets from Repository for performance analysis.

Step Seven: Application of preprocessing Data Mining Model such as selection, cleaning, integration, and storage for veracity of the extracted variables (elements).

Step Eight: Extract datasets from UCI Repository to compare the performance analysis of the improved model and existing models.

This methodology features the unified modeling language (UML): Use Case diagram, high-level model diagram, flow chart diagram, active diagram, and architecture design of the system.

3.2.2. **Analysis of the Proposed System**

This work proposed an improved model for big data analytics using dynamic multi-swarm optimization and unsupervised learning algorithms. The concept of Reference Distance Weighted (RDW) of dissimilarity measure, Euclidean Distance (ED) measure and Chi-square Relative Frequency (CRF) of dissimilarity measure was used to develop an improved clustering algorithm to

efficiently cluster mixed dataset. The improved clustering algorithm was hybridized with Dynamic Multi-Swarm Optimization (DMSO) Algorithm to form a robust and efficient clustering algorithm called "Dynamic-K-reference Clustering Algorithm" for Big Data analytics.

First a simple robust numerical clustering algorithm called RDW-K-means clustering algorithm were developed with Reference Distance Weighted (RDW) and Euclidean distance measure. The RDW calculates distance between feature-vectors with real values (i.e obtaining classified future data-points of clusters) in a given region of the data points. This mechanism improves the traditional K-means clustering algorithm to obtain accurate similarities clusters of attributes in a given region.

Secondly RDW-K-means clustering algorithm were integrated with Chi-square relative frequency dissimilarity measure used in K-representative clustering algorithm to develop improved clustering algorithm called K-reference Clustering algorithm for clustering unstructured (mixed) dataset.

Thirdly, K-reference clustering algorithm was hybridized with Dynamic Multi-swarm Optimization algorithm to develop robust and efficient algorithm called Dynamic-K-reference Clustering algorithm that will overcome the problems that Big Data characteristics posed on traditional data mining techniques on creating clusters of mixed datasets. The intelligent concept of Dynamic Multi-Swarm will guide the proposed clustering algorithm to obtain accurate convergences of the objective function by providing best global initial value of K clusters. Dynamic multi-swarm optimization algorithm was chosen among other meta-heuristic algorithms because it has the capacity to achieve an effective balance in a multimodal problems, instead of trying to reach a compromise between exploration and exploitation which could weaken both mechanisms of the search process like in the case of Artificial Bee Colony and Firefly algorithms etc., Dynamic Multi-swarm system separates them into distinct phases. Each phase is more focused on either exploitation (individual sub-swarms) or exploration (diversification method)". The proposed algorithm is efficient to handle the challenges Big Data analytics posed on traditional data mining models.

3.2.2.1 **Analysis of the Mathematical model for the Developed Dynamic-K-reference Clustering Algorithm**

The analysis of the development of the proposed unsupervised machine learning called Dynamic-K-reference clustering algorithm were analyzed as follows

i. Formation of Dissimilarity Measure of Categorical Domain.
ii. Design of the Categorical clustering algorithm.
iii. Development of the improved RWD-K-means Clustering Algorithm.
iv. Integration of improved RDW-K-means and Categorical Clustering Algorithm to produce K-Reference algorithm.
v. Development of the Hybridized Algorithm of Improved K-Reference Clustering Algorithm using Dynamic Multi-Swarm Optimization Algorithm.

3.2.2.2 **Description of the Dissimilarity Measure of Categorical Domain of the proposed System**

This work developed an improved dissimilarity measure of a categorical domain using Reference Distance Weighted dissimilarity measure and Chi-square Simply Mismatching dissimilarity measure. The categorical domain of mixed dataset analysis from the review of literature is analyzed using the dissimilarity measures between a categorical object and the representative of a cluster, which is defined based on a simple matching approach of chi-square expression. The process is as follows:

Calculate the Reference Distance Weighted (RDW) of the corresponding object representative. Given $C = \{S_1, \ldots, S_P)$ be a cluster of categorical objects, with

$S_i = (s_1, \ldots, s_i)$, $1 \leq i \leq p$, and $S = (s_1, \ldots, s_m)$ a categorical object.

Note that in some cases S may or may not belong to C. Assume that $Q = (q_1, \ldots, q_m)$, with $t_j = \{(c_j, rfc_j)|c \mathcal{E} D_j\}$, is a representative of cluster C.

Now, the dissimilarity between object S and representative T is defined by

$$d(S,Q) = RDW_i\left(\sum_{j=1}^{p}\sum_{c_j \varepsilon D_j}^{m} rf_{c_j}\delta(s_j, c_j)\right) \quad (3.1)$$

Where c = cluster and rf = relative frequency between the clusters. ie. rf represents the relative frequency "relative frequency of category c_j within C".

And RDW is the corresponding reference distance weighted vectors

Under such a definition, the Reference Distance Weight (RDW) determines the dissimilarity of the centroid d(S, Q) with respect to the relative frequencies of categorical values within the cluster and simple matching between categorical values. It is of interest to note that reference factor of the RDW guides the simple matching dissimilarity measure between categorical object and the square Euclidean distance measure to produce pure clustering output.

It is easily seen that

$$d(S,Q) = RDW_i\left(\sum_{j=1}^{p}\sum_{c_j \varepsilon D_j}^{m} rf_{c_j}\delta(s_j, c_j)\right) \quad (3.2)$$

$$= RDW_i\left(\sum_{j=1}^{p}\sum_{c_j \varepsilon D_j, c_j \neq x_j}^{m} rf_{c_j}\right) \quad (3.3)$$

$$= RDW_i\left(\sum_{j=1}^{p} 1 - rf_{x_j}\right) \quad (3.4)$$

where rf_{xj} represents the "relative frequency of category x_j within C".

3.2.2.3 **Proposed Categorical Clustering Algorithm**

Step one: Pick an observation (instance) at random and use that as a cluster (obtain the global best value of clusters k from the result of Dynamic Multi-swarm algorithm).

Step two: Compare each data point in the cluster to each observation data points, any elements that are not equal, +1 is added otherwise nothing is added (Calculate the K representative of each clusters.)

Step three: Assign each individual to the closest centroid

Step four: Each feature should have the mode (most common response) for each centroid

Step five: Repeat steps 2-4 until no changes are made in the assignment of individual to the closest centroid.

3.2.2.4 **RWD-K-means Clustering Algorithms**

The Proposed RWD-K-means Clustering Algorithms are as follow:

Step one: obtain the global best value of clusters k from the result of Dynamic Multi-swarm algorithm.

Step two: Randomly selecting the centroids (v_1, $v_2 \ldots v_k$) in the data set.

Step three: Calculate the Reference Distance Weighted RDW $^{(1)}$ of the corresponding centroids (v_1, $v_2 \ldots v_k$) of the data points in the dataset.

Step four: Calculate RDW (s, t, α) =

$$\frac{\sum_{i=1}^{n} \alpha_i \left|\frac{s_i - t_i}{s_i}\right|}{n} = \frac{\sum \alpha_i \left|1 - \frac{t_i}{s_i}\right|}{n} \qquad (3.5)$$

i = 1, 2, ..,n. (Liviu et al., 2013).

Where: RDW is the corresponding reference distance weighted vectors to the S_i,

$s = \{s_1, s_2 \ldots s_n\}$ represents the feature-vector of reference that associated to each class. It is the vector from which the distance of the inputs data points was measured.

$t = \{t_1, t_2 \ldots t_n\}$ represents the feature-vector that associated to the objects that is to be classified in the dataset (data inputs to the system).

$\alpha = \{\alpha_1, \alpha_2.., \alpha_n\}$, represents the vector called relevance vector. Its 0components are reference factors (α_i) parameters that is allocated and assigned to each feature separately. The relevance vector is proportional to the importance/weight of the respective features under the conditions of the problem (that is the objects or data points of the datasets to be clustered) to be solved.

Step five: Assign datapoints to its clusters based on the Reference Factors of the initial clusters

$x_i \in c_i$ if $x_{i,RDW} \approx c_{iRDW}$ // x_i *are the dataoints,*
// $x_{i,RDW}$ are the Reference Distance Weight of the datapoints
// c_{iRDW} are the Reference Distance Weight of the initial clusters
Assign x_i *in* c_i
Else $x_i \notin c_i$
Assign $x_i \in c_i$ where $x_{i,RDW} \approx c_{iRDW}$

Step six: Find the distance between the centroids (instances of the attributes of the dataset) using the integration of Reference distance weighted and Euclidean Distance equation.

$$d_{ij} = \|RDW_i(s_i - t_k)\|^2 \qquad (3.6)$$

Step seven: Update the centroids Stop the process when the new centroids are nearer to old one. Otherwise go to step-four.

3.2.2.5 **Integration of RDW-K-means and Categorical Algorithms**

The RDW_K-means algorithm were hybridized with categorical algorithms to develop K-reference clustering algorithm.

The Hybridized clustering Algorithm for Mixed Dataset is represented as follow:

Step one: obtain the global best value of clusters k from the result of Dynamic Multi-swarm algorithm.

Step two*: Randomly selecting the centroids (s_1, $s_2 \ldots s_k$) in the data set.*

Step three: *Calculate the Reference Distance Weight RDW $^{(1)}$ of the corresponding centroids (s_1, $s_2 \ldots s_k$).*

Step four*: Calculate RDW (s, t, α)* using equation (3.5)

Step five*: Obtain the distance between the datapoints centroids using the Euclidean Distance equation. (Using equation (3.6))*

Step six*: Update the centroids Stop the process when the computational process of the new centroids is similar to old one. Otherwise go to step-four.*

Step seven*: Initialize a k-partition of D randomly.*

Step eight*: "Calculate k representatives one for each cluster".*

Step nine: Assign datapoints to its clusters based on the Reference Factors of the initial clusters

$x_i \in c_i$ if $x_{i,RDW} \approx c_{iRDW}$ // x_i *are the datapoints,*

// $x_{i,RDW}$ are the Reference Distance weight of the datapoints

// c_{iRDW} is the Reference Distance weight of the initial clusters

Assign x_i *in* c_i

Else $x_i \notin c_i$

Assign $x_i \in c_i$ where $x_i \approx c_{iRDW}$

Step ten: *For each S_i, "compute the dissimilarities of the datapoints such as d (S_i, T_i), i = 1, . . ., k. Then, reassign S_i to cluster C_l (from cluster C_u, say) to obtain minimal dissimilarity between S_i and T_i. Update both T_i and T"*

Step eleven: *Repeat Step ten while no dissimilarity among objects clusters after a full cycle test of the whole data set.*

With the integration of equations (3.1, and 3.6) to develop improve clustering algorithm for mixed dataset between two mixed-type objects S and T, which are described by the attributes V^r_1, V^r_2, . . . V^r_p, V^r_{p+1}, . . . V^r_m, can be measure by:

$$d_2(S,T) = RDW_i\left(\sum_{j=1}^{p}\left(s_j - t_j\right)^2 + \gamma \sum_{j=p+1}^{m} rf\delta\left(s_j, t_j\right)\right) \quad (3.7)$$

Where the numerical algorithms are represented first and followed by the categorical algorithm of the right side of equation (3.7). to avoid favoring either the numerical or categorical attributes the weight γ is used.

The cost function of equation (3.7) for mixed-type objects can be modified for convenient interpretation as follows:

$$P(W,Q) = RDW_i(\sum_{l=1}^{k}(\sum_{I}^{n} w_{i,l} \sum_{j=1}^{p}\left(s_j - t_j\right)^2 + \gamma \sum_{I=1}^{n} w_{i,I} \sum_{j=p+1}^{m} rf\delta\left(s_j, t_j\right))) \quad (3.8)$$

Let

$$P_l^r = RDW_i\left(\sum_{j=1}^{p} w_{ij} \sum_{j=1}^{m}\left(s_{i,j} - t_{i,j}\right)^2\right) \quad (3.9)$$

And

$$P_1^c = RDW_i\left(\gamma \sum_{i-1}^{n} w_{i,l} \sum_{j=p+1}^{m} rf\delta\left(s_{i,j}t_{i,j}\right)^2\right) \quad (3.10)$$

Equation (3.14) can be re-written as:

$$P(W,Q) = RDW_i(\sum_{l=1}^{k}(P_l^r + P_l^c)) \quad (3.11)$$

Therefore, minimizing P(W, Q) is equivalent to minimizing $P^r{}_1$and $P^c{}_1$ for $1 \leq l \leq k$; given that rf relative frequency for the mixed data.

3.2.2.6 **Illustrations of the numerical models of K-reference Clustering Algorithms**

This section illustrates the computations of numerical and categorical clustering algorithm integrated in equation 3.7. The numerical algorithm is at the left side while the categorical algorithm is at the right side.

3.2.2.7 **Illustration of Calculation of Reference Factor for create clusters using RWD in equation (3.5)**

Given the dataset in Table 3.1

Table 3.1: Numerical Sample Dataset of Death Toll of Boko Haram Attacks

Year of attacks	Death Toll
2013	150
2012	159
2012	65
2011	0
2013	12
2014	176
2011	100
2011	50
2012	500
2019	0
2019	100

Step one: Randomly selecting the centroids v_1 = 2013 and v_2 =150 from Table 3.1.

Let $\alpha_i = (1,1)$

Such that S_i = (2013, 150),

Note that:

1. s = {s_1, s_2… s_n} represents the feature-vector of reference that associated to each class. It is the vector from which the distance of the inputs data points was measured.
2. t = {t_1, t_2… t_n} represents the feature-vector that associated to the objects that is to be classified in the dataset (data inputs to the system).

So, t_i = {(2012, 159), (2012, 65), (2011, 0), (2013, 12), (2014, 176), (2011, 100), (2011, 50), (2012, 500), (2019, 0), (2019, 100)}

Applying

Step three: Calculate the Reference Distance Weight RDW $^{(1)}$ of the corresponding centroids (v_1, v_2…v_k) of the data points in the dataset using equation 3.7.

Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2012, 159)

We have

1*(ABS((2013-2012)/2013)) + 1*(ABS((150-159)/150)) = 0.03024

Where the function “ABS( )” is an absolute value formular in excel sheet.

1. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2012, 65)

   We have

   1*(ABS((2013-2012)/2013)) + 1*(ABS((150-65)/150)) = 0.28358

2. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2013, 0)

   We have

   1*(ABS((2013-2011)/2013)) + 1*(ABS((150-0)/150)) = 0.500497

3. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2013, 12)

   We have

   1*(ABS((2013-2013)/2013)) + 1*(ABS((150-12)/150)) = 0.4600

4. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2013, 12)

   We have

   1*(ABS((2013-2013)/2013)) + 1*(ABS((150-12)/150)) = 0.4600

5. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2014, 176)

   We have

   1*(ABS((2013-2014)/2013)) + 1*(ABS((150-176)/150)) = 0.0740

6. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2011, 100)

   We have

   1*(ABS((2013-2011)/2013)) + 1*(ABS((150-100)/150)) = 0.167163

7. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2011, 50)

   We have

   1*(ABS ((2013-2011)/2013)) + 1*(ABS ((50-100)/150)) = 0.33383

8. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2012, 500)

   We have

   1*(ABS ((2013-2012)/2013)) + 1*(ABS ((50-500)/150)) = 1.16691

9. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2019, 0)

   We have

1*(ABS ((2013-2019)/2013)) + 1*(ABS ((0-500)/150)) = 0.50149

10. Let the feature vector of reference be s_1 = (2013, 150) and t_1 = (2019, 100)

    We have

    1*(ABS ((2013-2019)/2013)) + 1*(ABS ((100-500)/150)) = 0.168157

11. 1*(ABS((2013-2012)/2013)) + 1*(ABS((150-159)/150)) = 0.03024

    Tabulating the results in table 3.2.

Table 3.2: Calculation of Reference Factor for create clusters using RWD

Year of attacks	Death Roll	RDW values
2013	150	0.030248
2012	159	0.030248
2012	65	0.28358
2011	0	0.500497
2013	12	0.46
2014	176	0.074
2011	100	0.167163
2011	50	0.33383
2012	500	1.16691
2019	0	0.50149
2019	100	0.168157

The result in table 3.2 shows the RDW values of the datapoints that has the same reference factors indicated with different colors. The datapoint (2012, 65) has similar reference factors with the datapoint (2011, 50). This implies that the datapoint (2012, 65) and the datapoint (2011, 50) should be assigned in the same cluster.

Moreover, the datapoint (2011, 100) has similar reference factors with the datapoint (2019, 100). This indicates that the datapoint (2011, 100) and the datapoint (2019, 100) should be assign to the same cluster.

Also, the datapoints (2011, 0), and (2019, 0) has similar reference factors. This implies that the datapoints (2011, 0) and (2019, 0) should be assigned in the same cluster.

In addition, the datapoint (2014, 176) has similar reference factors with the datapoint (2012, 159). This indicates that the datapoint (2014, 176) and the datapoint (2012, 159) should be assign to the same cluster.

Finally, the datapoint (2012, 500) has reference factors that is dissimilar to the other datapoints, therefore, it has to assign to a separate cluster.

After the Reference Factors of the datapoints were computed using equation (3.5) in equation (3.6). K-means algorithm is applied to create clusters of the data points using the Euclidean Distance measure based on the calculated Reference Factors. The algorithm to assign the similarity/dissimilarity results of K-means to its clusters is shown in section 3.2.2.8.

3.2.2.8 Algorithm to assign the similarity/dissimilarity of datapoints to its clusters

The algorithm is as follows:

Let C_1, C_2...C_5 be the initial clusters with $c_{i,RDW}..c_{n,RDW}$of a given dataset D. where i =1, 2, …n.

Such that D = {x_1, x_2… x_n}

Compute $x_{i\,RDW}$,… $x_{n\,RDW}$ //calculate the reference values of $x_i..x_n$

$x_i \in c_i$ if $x_{i\,\mathbf{RDW}} \approx c_{iRDW}$ //c_{iRDW} $represents\ the\ reference\ factor\ of\ clusters$

Assign $x_i\ in\ c_i$

Else $x_i \notin c_i$

Assign $x_i \in c_i$ where $x_i \approx c_{iRDW}$

The result of the clustered dataset is shown in Tables 3.3.

Table 3.3: **Summary of K-reference Clustering analysis**

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
(2012, 159)	(2011, 100)	(2013, 12)	(2012,65)	(2012, 500)
(2014, 176).	(2019, 100)	(2011,0)	(2011, 50).	
(2013, 150)		(2019, 0)		

3.2.2.9 Illustration of K-means Clustering Algorithm for a small dataset sample of Boko Haram insurgency attack.

Given the dataset in Table 3.4.

Table 3.4: **Sample dataset to Illustration of K-means Clustering Algorithm**

Year of Attack	Death Roll
2012	159
2013	150
2013	12
2011	0
2014	176
2011	100
2011	50
2012	500
2019	0
2019	100
2012	65

Given the dataset D = {(2012, 159), (2012, 159), (2012, 65), (2011, 0), (2013, 12), (2014, 176), (2011, 100), (2011, 50), (2012, 500), (2019, 0), (2019, 100)} tabulated in Table 3.4

Step one: Select 3 initial clusters (k1, k2 and k3) from D.

Step two: Randomly select distinct centroid (new data points as cluster initialization)

Table 3.5: Initial Clusters = 3

Cluster	X	Y
K1	2012	159
K2	2013	65
k3	2013	12
K4	2014	176
K5	2011	100

Step two: Obtain the distance between the datapoints centroids using the Euclidean Distance equation. (using K-means algorithm in equation (3.6))

Calculating the Euclidean distance using the given equation

Distance [(x, y), (a, b)] = $\sqrt{(x-a)^2 + (x-b)^2}$ (3.12)

Calculating the Euclidean distance from (2012, 159) to the other data-points using equation (3.12)

Distance from cluster 1 (2012,159) to itself =$\sqrt{(2012-2012)^2 + (159-159)^2} = 0.000$

Distance from cluster 2 (2013,150) to cluster 1=$\sqrt{(2013-2012)^2 + (150-159)^2} = 94.0055$

Distance from cluster 3 (2013,12) to cluster 1 = $\sqrt{(2013-2012)^2 + (12-159)^2} = 147.0034$

Distance from cluster 4 (2014,176) to cluster 1= $\sqrt{(2013-2012)^2 + (12-159)^2} = 17.11724$

Distance from cluster 5 (2011,100) to cluster 1= $\sqrt{(2013-2012)^2 + (100-159)^2} = 59.00847$

Calculating the Euclidean distance from cluster 2 (2012, 65) to the other clusters

Distance from cluster 1 (2012,159) to cluster 2 = $\sqrt{(2012-2012)^2+(159-65)^2} = 94.0055$

Distance from cluster 2 (2012,65) to itself =$\sqrt{(2012-2012)^2+(65-65)^2}$ = 0.0000

Distance from cluster 3 (2013,12) to cluster 2 = $\sqrt{(2013-2012)^2+(12-65)^2} =$ 53.0094

Distance from cluster 4 (2014,176) to cluster 2 = $\sqrt{(2013-2012)^2+(176-65)^2} = 111.018$

Distance from cluster 5 (2011,100) to cluster 2 = $\sqrt{(2011-2012)^2+(100-65)^2} =$ 35.01428

Calculating the Euclidean distance from cluster 3 (2012, 65) to the other clusters using equation (3.12)

Distance from cluster 1 (2012,159) to cluster 3 (2012, 65) = $\sqrt{(2012-2013)^2+(159-12)^2} =$ 147.0034

Distance from cluster 2 (2012,65) to cluster 3 (2012, 65) =$\sqrt{(2012-2013)^2+(65-12)^2}$ = 53.0094

Distance from Cluster 3 (2013,12) to cluster 3 (2012, 65) = $\sqrt{(2013-2013)^2+(12-12)^2} =$ 0.0000

Distance from cluster 4 (2014,176) to cluster 3 (2012, 65) = $\sqrt{(2013-2013)^2+(176-12)^2} =$ 164.0030

Distance from cluster 5 (2011,100) to cluster 3 (2012, 65) = $\sqrt{(2011-2013)^2+(100-12)^2} =$ 88.0227

Calculating the Euclidean from the cluster 4 (2014, 176) to the other clusters

Distance from cluster 1 (2012,159) to cluster 4 (2014, 176) = $\sqrt{(2012-2014)^2+(159-176)^2} =$ 17.1172

Distance from cluster 2 (2012,65) to cluster 4 (2014, 176) =$\sqrt{(2012-2014)^2+(65-176)^2}$ = 111.0180

Distance from Cluster 3 (2013,12) to cluster 4 (2014, 176) = $\sqrt{(2013-2014)^2+(12-176)^2} =$ 53.0094

Distance from Cluster 4 (2014,176) to cluster 4 (2014, 176) = $\sqrt{(2014-2014)^2+(176-176)^2} =$ 0.0000

Distance from cluster 5 (2011,100) to cluster 4 (2014, 176) = $\sqrt{(2011-2014)^2+(100-176)^2} =$ 76.0592

Calculating the Euclidean distance from cluster 5 (2012, 65) to the other clusters

Distance from cluster 1 (2012,159) to cluster 5 (2012, 65) = $\sqrt{(2012-2011)^2+(159-100)^2} =$ 59.0085

Distance from cluster 2 (2012,65) to cluster 5 (2012, 65) $=\sqrt{(2012-2011)^2+(65-100)^2}$ = 35.0143

Distance from Cluster 3 (2013,12) to cluster 5 (2012, 65) = $\sqrt{(2013-2011)^2+(12-100)^2}=$ 88.0227

Distance from Cluster 4 (2014,176) to cluster 5 (2012, 65) = $\sqrt{(2014-2011)^2+(176-100)^2}=$ 76.0592

Distance from cluster 5 (2011,100) to cluster 5 (2012, 65) = $\sqrt{(2011-2011)^2+(100-100)^2}=$ 0.0000

Table 3.6 Assigning datapoints to the initial selected clusters.

Cluster	Centroid					
	v	w	x	y	z	ASSIGNMENT
K1	0.000	94.005	147.003	17.117	59.008	1
K2	94.005	0.000	53.000	111.005	35.0571	2
k3	147.003	53.000	0.000	164.003	88.023	3
k4	111.005	52.998	164.003	0.000	76.059	4
k5	59.008	35.057	88.023	76.059	0.000	5

Table 3.6 show the calculated initial clusters: 1, 2, 3, 4 and 5. The initial clusters will be used to cluster the rest of the dataset.

Calculating Euclidean distance for the next dataset (2011,0) to clusters: 1, 2, 3, 4 and 5 using equation (3.12)

Distance from cluster 1 (2012,159) to the dataset (2011,0) = $\sqrt{(2012-2011)^2+(159-0)^2}=$ 159.003

Distance from cluster 2 (2012,65) to the dataset (2011,0) $=\sqrt{(2012-2011)^2+(65-0)^2}$ = 65.008

Distance from Cluster 3 (2013,12) to the dataset (2011,0) = $\sqrt{(2013-2011)^2+(12-0)^2}=$ 12.166

Distance from Cluster 4 (2014,176) to the dataset (2011,0) = $\sqrt{(2014-2011)^2+(176-0)^2}=$ 176.026

Distance from cluster 5 (2011,100) to the dataset (2011,0) = $\sqrt{(2011-2011)^2+(100-0)^2}=$ 100.000

Table 3.7: Assigning dataset (2011,0) to the cluster with the minimum distance

Dataset	Euclidean Distance					
	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	ASSIGNMENT
(2011,0)	159.003	65.008	12.165	176.026	100.000	3

Table 3.7 shows the Euclidean distance for the next dataset (2011,0) to cluster 1, cluster 2, cluster3, cluster4 and cluster 5. Cluster 3 has the minimum distance therefore, the dataset (2011,0) is assigned to cluster 3

Next is to update the centroid b/w datapoints (2011,0) and (2013,12)

Table 3.8: Update the cluster centroid b/w datapoints (2011,0) and (2013,12)

	Update the cluster centroid	
Cluster	**X**	**Y**
K1	2012	159
K2	2012	65
k3	(2013+2011)/2=2012	(12+0)/2=6
k4	2014	176
k5	2011	100

Calculating Euclidean distance for the next dataset (2014,176) to cluster 1, cluster 2, cluster3, cluster4 and cluster 5 using equation (3.12) is shown in table 3.9.

Table 3.9: Calculating Euclidean distance for the next dataset (2014,176)

	Euclidean distance
Distance from cluster 1(2012,159)	9.055
Distance from cluster 2(2012,65)	85.006
Distance from cluster 3(2013,12)	26.019
Distance from cluster 4(2013,12)	26.019
Distance from cluster 5(2011,100)	50.040

Table 3.10: Assigning dataset (2014,176) to the cluster with the minimum distance

Euclidean Distance					
Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**	**ASSIGNMENT**
9.055	85.006	26.019	26.019	50.040	1

Table 3.10 shows the Euclidean distance for the next dataset (2014, 176) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5. Cluster 1 has the minimum distance therefore, the dataset (2014, 176) is assigned to cluster 1

Next is to update the centroid b/w datapoints (2014, 176) and (2012, 159)

Table 3.11: Update the cluster centroid b/w datapoints (2014, 176) and (2012, 159)

Cluster	X	Y
K1	(2012+2014)/2=2012.5	(159+176)/2=154.5
K2	2012	65
k3	2012	6
k4	2014	176
k5	2011	100

Calculating Euclidean distance for the next dataset (2011,50) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5 using equation (3.12)

Table 3.12: Calculating Euclidean distance for the next dataset (2011,50)

	Euclidean distance
Distance from cluster 1(2012,159)	104.511
Distance from cluster 2(2012,65)	15.033
Distance from cluster 3(2013,12)	44.011
Distance from cluster 4(2013,12)	126.036
Distance from cluster 5(2011,100)	50.000

Table 3.13: Assigning dataset (2011,50) to the cluster with the minimum distance

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	ASSIGNMENT
104.511	15.033	44.011	126.036	50.000	2

Table 3.12 shows the Euclidean distance for the next dataset (2011,50) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5. Cluster 2 has the minimum distance therefore, the dataset (2011,50) is assigned to cluster 2.

Next is to update the centroid b/w datapoints (2011,50) and (2012, 65)

Table 3.14: Update the centroid b/w datapoints (2011,50) and (2012, 65)

Cluster	X	Y
K1	2012.5	154.5
K2	(2012+2011)/2=2011.5	(65+50)/2=57.5
k3	2012	6
k4	2014	176
k5	2011	100

Calculating Euclidean distance for the next dataset (2012,500) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5 using equation (3.12) is shown in table 3.14

Table 3.15: Calculating Euclidean distance for the next dataset (2012,500)

	Euclidean Distance
Distance from cluster 1(2012,159)	345.500
Distance from cluster 2(2012,65)	442.500
Distance from cluster 3(2013,12)	494.000
Distance from cluster 4(2013,12)	324.006
Distance from cluster 5(2011,100)	400.001

Table 3.16: Assigning dataset (2012,500) to the cluster with the minimum distance

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	ASSIGNMENT
345.500	442.500	494.000	324.006	400.001	4

Table shows the Euclidean distance for the next dataset (2012,500) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5. Cluster 4 has the minimum distance therefore, the dataset (2012,500) is assigned to cluster 4.

Next is to update the centroid b/w datapoints (2012,500) and (2014, 176)

Table 3.17: Updating the centroid b/w datapoints (2012,500) and (2014, 176)

Cluster	X	Y
K1	2012.5	154.5
K2	2011.5	57.5
k3	2012	6
k4	(2014+2012)/2=2013	(500+176)/2=338
k5	2011	100

Calculating Euclidean distance for the next dataset (2019,0) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5 using equation (3.12) is shown in table 3.16.

Table 3.18: Calculating Euclidean distance for the next dataset (2019,0)

	Euclidean Distance
Distance from cluster 1(2012,159)	154.637
Distance from cluster 2(2012,65)	282.587
Distance from cluster 3(2013,12)	9.220
Distance from cluster 4(2013,12)	338.053

Distance from cluster 5(2011,100)	100.319

Table 3.19: Assigning dataset (2019, 0) to the cluster with the minimum distance

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	ASSIGNMENT
154.637	282.587	9.220	338.053	100.319	3

Table 3.17 shows the Euclidean distance for the next dataset (2019,0) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5. Cluster 3 has the minimum distance therefore, the dataset (2012,500) is assigned to cluster 3.

Next is to update the centroid b/w datapoints (2019,0) and (2012, 6) as shown in table 3.18.

Table 3.20: Update the cluster centroid b/w datapoints (2019,0) and (2012, 6)

Cluster	X	Y
K1	2012.5	154.5
K2	2011.5	57.5
k3	2015.5	3
k4	2013	338
k5	2011	100

Calculating Euclidean distance for the next dataset (2019,100) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5 using equation (3.12) is shown in table 3.21.

Table 3.21: Calculating Euclidean Distance for the next dataset (2019,100)

	Euclidean Distance
Distance from cluster 1(2012,159)	54.886
Distance from cluster 2(2012,65)	43.157
Distance from cluster 3(2013,12)	97.063
Distance from cluster 4(2013,12)	238.076
Distance from cluster 5(2011,100)	8.000

Table 3.22: Assigning dataset (2019,100) to the cluster with the minimum distance

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	ASSIGNMENT
54.886	67.591	97.063	238.078	8.000	5

Table 3.20 shows the Euclidean distance for the next dataset (2019,100) to cluster 1, cluster 2, cluster 3, cluster 4 and cluster 5. Cluster 5 has the minimum distance therefore, the dataset (2012,500) is assigned to cluster 5.

The clustering results of K-means are summarized in table 3.21

Table 3.23: Summary of K-means Clustering analysis

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
(2012, 159)	(2013, 65)	(2013, 12)	(2014, 176)	(2011, 100)
(2013, 150)	(2011, 50)	(2011,0)	(2012, 500)	(2019, 100)
		(2019,0)		

From the analysis of K-means clustering algorithm in table 3.21, K-means assign the datapoint (2012, 500) to cluster 4 which is not proper. i.e. K-means assumes that the Datapoint (2012, 500) has similarity with the datapoints (2014, 176) while it is not. Comparing these results with the results of RDW distance measure in table3.3, which was used to develop K-reference clustering algorithm in this work. The algorithm assigned the datapoint (2012, 500) in cluster 5 that has dissimilarity to the other cluster because their similarity is detached.

It was also observed that the starting point of K-means clustering algorithm effected its poor convergence. Assuming the datapoint (2012, 500) was chosen among the initial clusters, the K-means algorithm would have produced a better result. These limitations of K-means were corrected in the proposed K-reference clustering algorithm in this work.

3.2.2.10 **Illustration of the Categorical Clustering Algorithm of K-reference Clustering Algorithms**

Given the sample dataset in Table 3.23

Table 3.24: Sample dataset of monthly attack of Boko Haram terrorist

Datapoint	Months of Attack		
1	September	July	December
2	November	September	November
3	January	July	November
4	September	July	November
5	November	July	November
6	January	July	November
7	January	September	November
8	September	November	November
9	December	July	November
10	September	July	December
11	September	January	December
12	January	July	November

Step one: Pick an observation (instance) at random and use that as a cluster.

Table 3.25: Initial clusters observations

c1 (1)	September	July	December
c2(3)	January	July	November
c3(7)	January	September	November

Step two: Compare each datapoint in the cluster to each observation datapoints, any element that are not equal add 1, if they are equal nothing is added.

Table 3.26: Compare each datapoint in the cluster with the each of the initial clusters

Datapoint	Months of Attack			c1	c2	c3
1	September	July	December	0	2	3
2	November	September	November	3	2	1
3	January	July	November	2	0	1
4	September	July	November	1	1	2
5	November	July	November	2	1	2
6	January	July	November	2	0	1
7	January	September	November	3	1	0
8	September	November	November	2	2	2
9	December	July	November	2	1	2
10	September	July	December	0	2	3
11	September	January	December	1	3	3
12	January	July	November	2	0	1

Step three: Assign each datapoint to the closet centroid.

The datapoints are assigned to the clusters with the minimum values as shown in table 3.27

Table 3.27: Assigning datapoints to clusters

				Cluster of Datapoints
c1	September	July	December	cluster1 (1, 4, 8, 10, 11)
	September	July	November	
	September	November	November	
	September	July	December	
	September	January	December	
c2				cluster1 (3, 6, 9, 12)
	January	July	November	
	January	July	November	
	December	July	November	
	January	July	November	
c3	November	September	November	cluster1 (2, 3, 7)
	November	July	November	
	January	September	November	

Step four: Each feature should have the mode (most common response) for each centroid

Table 3.28: Selecting Modes from c1 c2 and c3

C_1	September	July	December
C_2	January	July	November
C_3	November	September	November

Step five: Repeat step 2-4 until no changes are made in the assignment of individuals to the closest centroid

Step two: Compare each datapoint in the cluster to each observation datapoints, any elements that are not equal add 1, if they are equal nothing is added.

Table 3.29: Compare each datapoint in the cluster with the each of the initial clusters

Datapoint	Months of Attack			c1	c2	c3
1	September	July	December	0	2	3
2	November	September	November	3	2	0
3	January	July	November	2	0	2
4	September	July	November	1	1	2
5	November	July	November	2	1	1
6	January	July	November	2	0	2
7	January	September	November	3	1	1
8	September	November	November	2	2	2
9	December	July	November	2	1	2
10	September	July	December	0	2	3
11	September	January	December	1	2	3
12	January	July	November	2	0	2

c1	September	July	December			
c2	January	July	November			
c3	November	September	November			

Step three: Assign each datapoint to the closet centroid.

The datapoints are assigned to the clusters with the minimum values as shown in Table 3.24

Table 3.30: Assigning datapoints to clusters

				Cluster of Datapoints
c1	September	July	December	cluster1 (1, 4, 8, 10, 11, 12)
	September	July	November	
	September	July	December	
	September	January	December	
	September	November	November	
c2	January	July	November	cluster1 (3, 6, 9, 12)
	November	July	November	
	January	July	November	
	December	July	November	
c3	November	September	November	cluster1 (2, 3, 7)
	January	September	November	
	November	July	November	

Comparing Table 3.27 and Table 3.30, it shows that no changes are made in the assignment of individuals to the closest centroid. The algorithm terminates.

3.2.2.11 **Dynamic Multi-Swarm Optimization Algorithm**

Step One: Initialized the sub-swarms

Step Two: The swarms are examined to find the best error (smallest) and position of any of the particles in any of the swarms.

Step Three: The sub-swarms examine each particle position and Current Particle Moving to Different Swarm

Step Four: New Velocity Calculated for Current Particle

Step Five: After a new velocity is calculated, that velocity is used to move the current particle

Step Six: Next, the error associated with the new particle position is determined and checked to see if it's a new best error for the current particle

Step Seven: Checking New Position

Step Eight: Method Solve, finished by returning the best position found

Step Nine: Return the global variable form best variable K of each sub-swarm
Step Ten: stop

3.2.2.12 Algorithm of the Hybrid of K-reference clustering algorithm with Dynamic Multi-Swarm Optimization Algorithms

The algorithm of the hybrid of the K-reference clustering Algorithm with Dynamic Multi-Swarm Optimization Algorithms includes:

"Step 1: Initialize the swarm by randomly identifying each particle's (data points or objects) position and velocity in search space. Such that all attractors are set to randomized particle position; set swarm attractors (the best of the particle attractor) to particles attractor 1(best position obtained by that particle so far) and store all function values to function floor.

Step 2: REPEAT FOR EACH swarm f (f is the number of swarms to be deployed)// Test for change. Compute function at swarm attractor for swarm n (n is the sub-swarm deployed at that region). Note: the swarm attractor enables information sharing between particles while the particle attractor severs as individual particle memories.

Step 3: IF value (best position) of next result is distinct from previous iteration THEN

3.1: Re-compute function value at each particle attractor.

3.2: Bring up-to-date swarm attractor and stock function values.

Step 4: WHILE particle k of swarm f, // Bring up-to-date Attractor.

4.1: Compute function at updated position and store value.

4.2: IF value of next result is better than particle attractor value THEN

4.3: Particle attractor k: = position and value of particle k. (where k is the current position of the particle)

4.4 IF value of next result is better than swarm attractor value THEN

4.5: Swarm attractor: = position and value of particle k.

FOR EACH swarm y ≠f (y is the sub-swarms) //Exclusion

Step 5: IF swarm attractor p_f is within r_{excl} of p_y THEN

5.1: Randomize the swarm with the worse swarm attractor value.

5.2: Re-instruct particle attractors, compute f at each new position, stock these values, and place attractor of swarm to the position of its best particle). UNTIL number of functions evaluates performed > max

Step 6: Obtain the global variable from best variable K of each sub-swarm.

Step 7: Calculate the Reference Distance Weighted of the corresponding object representative. (see equation (3.5)).

Step 8: Calculate similarity and dissimilarity of the particles (see equation (3.7)).

Step 9: Compute dissimilarity measure for the mixed dataset (categorical and numerical)

Step 10: Compute the Relative Frequency (rf) of the dissimilarity matrix of the mixed dataset

Step 11: Proceed to compute the clusters using the rf of the dissimilarity matrix as a parameter.

11.1: For each X_i, calculate the dissimilarities d (X_i, Y_i), i = 1, . . . , k. Then, reassign X_i to cluster C_l (from cluster Cu, say) such that the dissimilarity between Xi and Y_l is least. Update both Y_l and Q.

Step 12: If no object has changed clusters after a full cycle test of the whole data set, go to Step7

Step 14**:** else output results.

Step 13: Stop

The global best value of k the output of intelligent swarm optimization is the input for the new developed clustering algorithm called Dynamic-K-reference Clustering algorithm.

3.2.2.13 **Mathematical model for Dynamic multi-swarm optimization algorithms**

For updating particle *k* in swarm *n*, the following equations are used:

For neutral particles (Backwell, and Branke, 2004):

$$v_{id}(t+1) = w * v_{id}(t) + c_1 * r_1 * (p_{id}(t) - x_{id}(t) + c_2 * r_2 * (p_{gi}(t) - x_{id}(t) + T. \quad (3.13)$$

$$x_{ij}(1+t) = x_{ij}(t) + v_{ij}(t+1) \quad (3.14)$$

where:

c_1 and c_2 are the acceleration coefficients controlling the step size, and

r_1 and r_2 are two independently generated random numbers uniformly distributed between 0 and 1 (Backwell et al., 2004; Wang, Liu, Sun, Qu, & Wei1, 2018).

$x_{id}^{(t)}$ represents the position vector of particle i in dimension d at time t;

$v_{id}^{(t+1)}$ represents the velocity vector of particle i in dimension d at time t,

$p_{id}^{(t)}$ represents the personal best position of particle i in dimension d found from initialization through time t;

p_{gi} represents the global best position of particle i in dimension j situated with respect to time t (i.e. The best position found by all particles in the swarm).

c_1 and c_2 are position of acceleration constants which are used to level the contribution of the cognitive and social components respective;

$r_{1j}^{(t)} and\ r_{2j}^{(t)}$ are random numbers from uniform distribution U (0,1) at time t.

w = inertia weight

note: This term w serves as a memory of previous velocities. The inertia weight controls the impact of the previous velocity: a large inertia weight favors exploration, while a small inertia weight favor exploitation).

And

$$T = m * (\frac{t}{t_{max}} - 0.5) \quad (3.14)$$

where m is an accommodation coefficient with a value between 0 and 1.

The term T helps in preventing a particle from flying out of the boundary of the search space when its velocity is too high in early generations, and in escaping from a local optimal point by increasing its velocity so as to improve the overall searching capacity (Wang et al., 2018, p. 2).

Besides, T increases linearly to avoid oscillation and to keep a relatively stable search direction in the optimization process.

r is relative to the range of the landscape and the width of peaks. In general, set r according to the following equation (Backwell et al., 2004):

$$r = \sqrt{\sum_{i=1}^{n} \left(x_i^u - x_i^l\right)^2 / (W_{min} + c(W_{max} - W_{min}))} \quad (3.15)$$

where x_i^u and x_i^l are the lower and upper bound on the i-th dimension of the variable vector of n dimensions. W_{min} and W_{max} are the minimum and maximum peak width. c is a constant number in the range of (0; 1) (Backwell et al., 2004).

At iteration t, the new updated position and velocity of particle i are determined by (3.12) and (3.18), respectively, in the following:

$$x_{ij}(1 + t) = x_{ij}(t) + v_{ij}(t + 1) \quad (3.16)$$

the personal best particle is found by the following equations:

$$P_{best,i}^{t+1} = \begin{cases} P_{best,i}^{t} & if\ p_i^{t+1} > P_{ibest,i}^{t} \\ x_i^{t+1} & if\ p_i^{t+1} \leq P_{ibest,i}^{t} \end{cases} \quad (3.17)$$

Decision rule

$G_{best} = \min \{P_{best,i}^{t}\}$ (3.18)

Where

$P_{best,i}^{t}$is represents the global position of particle i in dimension jsituated with respect to time t

$x_{id}^{(t)}$ represents the position vector of particle i in dimension d at time t;

$p_{id}^{(t)}$ represents the personal best position of particle i in dimension d found from initialization through time t;

3.2.2.14 Illustration of Swarm Intelligence Algorithms

Using equation (3.13) and (3.14)

Using the RDW values in Table 3.2 as the initial positions of the particles such that

$x_{i10}^{(t)} = \{0.0302, 0.074, 0.1672, 1.1669, 0.1682, 0.5005, 0.46, 0.5015, 0.2836, 0.3338\}$

Step 1: Choose the initial of particles: $x_1 = 0.0302$, $x_2 = 0.074$, $x_3 = 0.1672$, $x_4 = 1.1669$, $x_5 = 0.1682$, $x_6 = 0.5005$, $x_7 = 0.46$, $x_8 = 0.5015$, $x_9 = 0.2836$, $x_{10} = 0.3338$.

The initial population (i.e. the iteration number t = 0) can be represented as t_i^O, i = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10:

$x_1 = 0.0302$, $x_2 = 0.074$, $x_3 = 0.1672$, $x_4 = 1.1669$,

$x_5 = 0.1682$, $x_6 = 0.5005$, $x_7 = 0.46$, $x_8 = 0.5015$,

$x_9 = 0.2836$, $x_{10} = 0.3338$.

Evaluate the objective function values as

$$f_i^{\ j} = \frac{\sum_{i=1}^{n} \alpha_i \left|\frac{s_i - t_i}{s_i}\right|}{n} = \frac{\sum \left|1 - \frac{t_i}{s_i}\right|}{n} \quad (3.19)$$

$f_1^0 = 0.8933$, $f_2^0 = 0.7649$, $f_3^0 = 0.6221$, $f_4^0 = 0.7391$,

$f_5^0 = 0.4105$, $f_6^0 = 0.1772$, $f_7^0 = 3.1149$, $f_8^0 = 0.7684$, $f_9^0 = 0.4070$,

Let $c_1 = c_1 = 1$

Assigning velocities of each particle to zero:

$v_i^0 = 0$, i.e $v_1^0 = v_2^0 = v_3^0 = v_4^0 = v_5^0 = v_6^0 = v_7^0 = v_8^0 = v_9^0 = 0$.

Step 2: Set the iteration number as t = 0 + 1 = 1and go to step 3.

Step 3: Find the personal best particle by

$$P_{best,i}^{t+1} = \begin{cases} P_{best,i}^{t} & if\ f_i^{t+1} > P_{5best,i}^{t} \\ x_i^{t+1} & if\ f_i^{t+1} \leq P_{5best,i}^{t} \end{cases} \quad (3.20)$$

This implies that

$P_{best,1}^{1} = 0.8933$, $P_{best,2}^{1} = 0.7649$, $P_{best,3}^{1} = 0.6221$, $P_{best,4}^{1} = 0.7391$,

$P_{best,5}^{1} = 0.4105$, $P_{best,6}^{1} = 0.1772$, $P_{best,7}^{1} = 3.1149$, $P_{best,8}^{1} = 0.7684$, $P_{best,9}^{1} = 0.4070$,

Step 4: Get the global best by

$G_{best} = \min \{P_{best,i}^{1}\}$ where i = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.

Since, the minimum best is $P_{best,6}^{1} = 0.1772$, thus $G_{best} = 0.1772$.

The $G_{best} = 0.1772$ contains the datapoint $G_{best} = 0.\ 0.074$.

The architecture of the proposed system is shown in Figure 3.2.

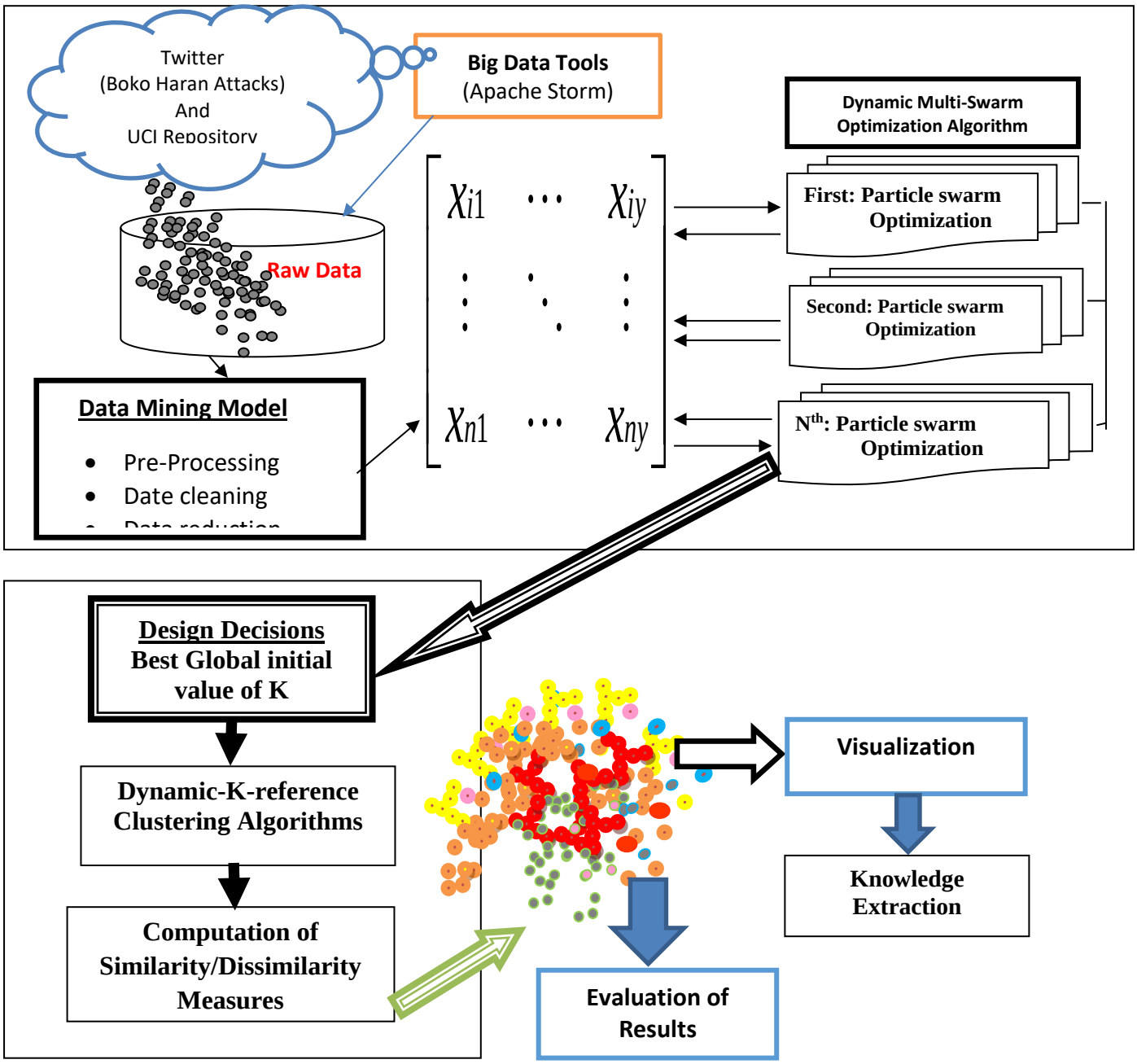

Figure 3.2: Architecture of the Proposed System

The Architecture of the Proposed System in Figure 3.2 shows the development of an improved model for Big Data analytics using dynamic multi-swarm optimization, and unsupervised machine learning algorithms base on the requirements in the previous section. At the first instance the system scraped raw data from the social media and Repository of University of California Ivory (UCI) using scrapping software (ScrapeStorm). The extracted raw (Unstructured) data are preprocessed using data mining model. And the preprocessed data are stored with a file extension of .dat or .data in MATLAB editor for easy computation with the proposed model. At the stage of the analysis of the very large

datasets. The proposed system deployed intelligence search agents to explore the positions of particles in the n-dimension datasets to obtain the optimal variables in each region of the n-dimensional datasets.

The diversification function coordinates the results of the multi search agents to summarized the global optimal positions of the particles in the n-dimensional dataset. The global optimal position of the datapoints guides the improved K-reference clustering algorithms to obtain the initial value of k clusters. The K-reference clustering algorithm creates a Reference Factor to each clusters of k to ensure datapoint with accurate reference factor are assigned to their clusters respectively. The improved model then computes the dissimilarity and similarity of the datapoints thereby assigning attributes with similarity in their clusters and dissimilar attributes in other clusters with respect to the Reference Factor assessment.

The components of the proposed architecture are as followed:

i. Raw Data: Raw dataset may consist of unstructured or structured dataset that are too complex and with high volume to be processed and analyzed. The unstructured raw dataset source may be extracted from web services, social media, agricultural sensor and etc. The detailed description of the data used were stated in section 3.2.5.3. section 3.2.4.3.
ii. Data pre-processing are applied prepare the raw data for analysis purposes. The data mining preprocessing includes selection, cleaning, and reduction as required in this work. The detailed description of the data used were stated in section 3.2.4.3.
iii. Dynamic Multi-swarm optimization: Dynamic Multi-swarm optimization algorithm was developed to enhance optimization limitations of the multi-Swarm optimization algorithm. It has the potential to regroup the sub-swarms for an additional task after the execution of the previous task
iv. Big data analysis requires a great analytic strategy to exploit voluminous nature. The dynamic multi-swarm algorithm has the potentials to use its intelligent principles which includes

principles of proximity, diversity, scalability, adaptability, and robust computational strength to optimize the multi-model problem of analyzing big dataset to unveil hidden pattern, information unknown form its analytical results.

v. The procedures involved scaling the n-dimensional into small sub-regions and deploys subswarm intelligent search agents to exploit the various regions of the problem domain. It also has the potentials to reassign the search agents when the need arises. The activities of the sub-swarms are controlled by the specific diversification method which decides where and when to launch the search agents to exploit the particle's position in each scale of the n-dimensional space to obtain the global value of k that is required to create clusters in the next phase. The result of the exploitation of each of the sub-warm are been explored by the design decision (specific diversification method) to obtain the best global variable of the initial value of K. The output of the Dynamic Multi-swarm algorithm is the input of the K-reference clustering algorithm.

vi. Sub-swarms: Sub-swarms are different sets of swarms deployed to search the high dimensional space specified and assigned for the swarm agent, they search for the best particle position at a given region. Other swarms perform the same task assigned to them in different portions of the high dimensional space of the whole big dataset respectively.

vii. Design decisions: The specific diversification method compares the best particle position from the results of each of the sub-swarms to extract the global best particle position using the design decision.

viii. Dynamic-K-reference clustering algorithm consists of intelligent optimization and clustering algorithms. The intelligent optimization algorithm optimized the high dimensional datasets of big datasets to pave way for better performance of the improved clustering algorithm (K-reference algorithm) which would have been influenced by the characteristics of big data analysis (such as volume, velocity, verity, etc.) on its performance.

ix. Computation of Similarity/Dissimilarity Measures: The improved clustering algorithm is very efficient to create similarity and dissimilarity measures among the attributes and their instance in the dimensional space in the dataset. This implies that similar attributes would be grouped while dissimilar attributes will be grouped in different clusters.

x. Clustering output (Knowledge Discoveries): The clustering results show the knowledge discovered from mining big data analysis for decision making.

xi. Knowledge Extraction: Knowledge extraction is the outcome of big data analytics. The revealed knowledge extracted is useful for organizational decision support systems of product innovations.

xii. Evaluation of Results: The accuracy and clustering errors results of the developed Dynamic-K-reference clustering algorithm were compared with existing MixK-meansXFon clustering algorithm, Binary PSO based K-prototype clustering algorithms. The evaluation results will specify the percentage improvement achieved from the efforts of the developed Dynamic-K-reference clustering algorithm. The evaluation model were described in section 3.2.2.4.

The Block diagram for High-level model of the proposed system is shown in Figure 3.3.

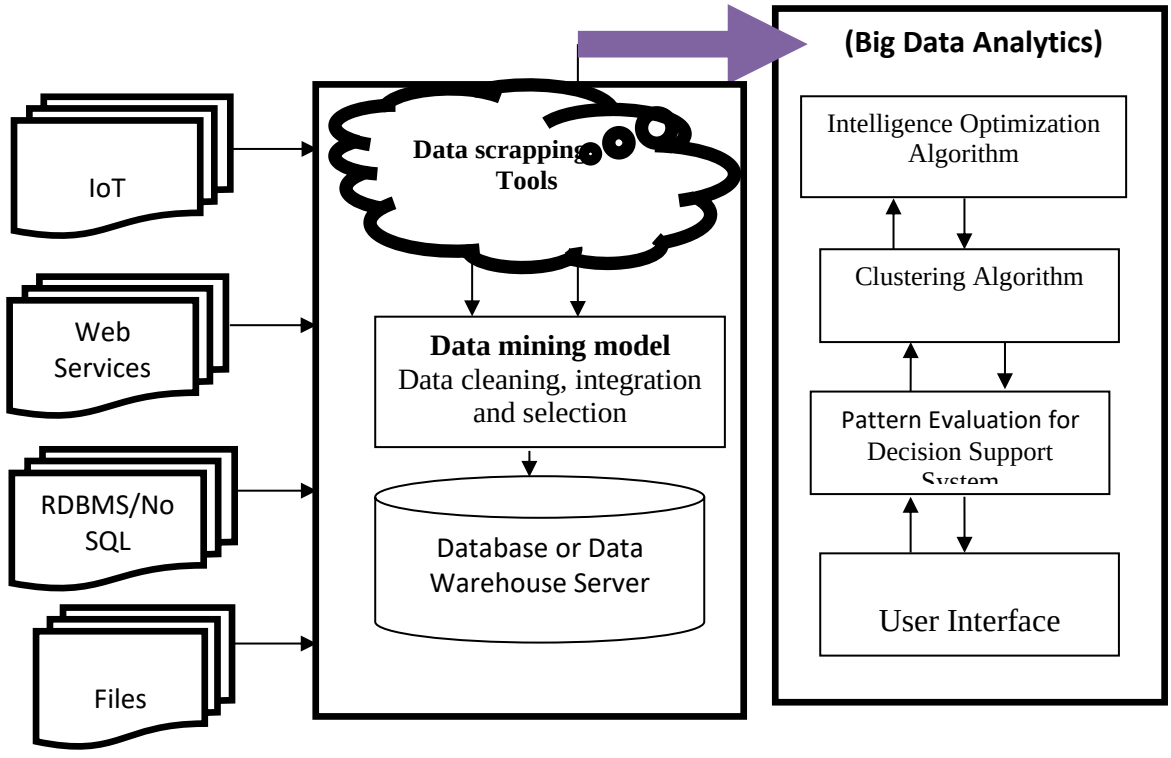

Figure 3.3: High-level model for the proposed system

The High-level model of the proposed system is a generic design that describes the components of the proposed system. It has different sources of big data processes such as the Internet of Things (IoT), Web Services, RDBMS/NoSQL, and Files. Unstructured datasets were generated from these sources and are pre-processed using a data mining model (e.g. data cleanings, selection, and data reduction) to prepare the problem domain in a metric computing form for analytic processes. The pre-processed dataset will be analyzed using the proposed hybridized data mining analytic model called Dynamic-K-reference clustering algorithm for decision making. The High-level model for the proposed system components consists of:

i. **Internet of Things (IoT):** Is the connection of various network devices such as electronic home appliances, vehicles, actuator, software, sensors and connection links that allows these things to interact and exchange data.
ii. **Web services**: These are services provided in the cloud by an electronic device for various electronic devices, communication with each other via the World Wide Web.
iii. **RDBMS/NoSQL**: Big Data and real-time web applications that produce heterogeneous information are store in the NoSQL database that has the storage mechanisms to store unstructured data types.
iv. **Files**: a computer file is a computer resource for recording data discretely in a computer storage device
v. **User Interface**: The user interface consists of a well define directive that describes the process used by users to operate the system.
vi. **Clustering Algorithm**: is an unsupervised learning technique for mining larger mixed datasets.
vii. **Interpretation and Evaluation**: The strength of the proposed model produced an interpretation of the knowledge discovery from the raw data for decision making.

viii. It uses visualization techniques for efficient interpretation. The accuracy of the interpreted information is evaluated to ensure the reliability of the discovered information.

ix. **Data Warehouse**: Data warehouses are most used as central repositories where data of various kinds are deposited. The information in the data warehouse may be stored currently or historical data which are used for creating analytical reports from the knowledge-based decision support system of the problem domain. The various datasets used for the validation of the proposed system in this work were obtained from the University of California Ivory repository. The data stored in the data warehouse supports research validation and verification of solutions to challenges in various areas in our dynamic environment. Some of the data may be processed through the operational data store some computations before it is used in the DW for reports.

x. **Decision Support System**: A knowledge discovery assignment must begin with clear objectives in mind for decision support systems. The output of the analysis of the problem domain will give an appropriate guide for its solutions.

3.2.3 Object Modeling

The observations obtained during system analysis and system design guide the technical approach of design and modeling the new system using Unified Modeling Language tools such as algorithm, activity diagram, use case diagram, high-level model, and architectural design diagram. It entails the design of various mechanisms to explain the connectivity and functionalities of the system components.

3.2.3.1 **Activity diagram used to describe how objective one was achieved**

The Activity Diagram of the proposed RWD_K-means Clustering Algorithms is shown in Figure 3.4. Activity diagram were used to describe the proposed improved clustering algorithms for mining unstructured datasets. To achieve objective one, the concept of Reference Distance Weighted (RDW) of dissimilarity measure, Euclidean Distance (ED) measure and Chi-square Relative Frequency

(CRF) of dissimilarity measure were used to develop improved clustering algorithm to cluster mixed dataset.

First a simple robust numerical clustering algorithm called RDW-K-means clustering algorithm were developed with Reference Distance Weighted (RDW) and Euclidean distance measure. The RDW calculates distance between feature-vectors with real values (i.e obtaining classified future data-points of clusters) in a given region of the data points. This mechanism improves the traditional K-means clustering algorithm to obtain accurate similar clusters of attributes in a given region.

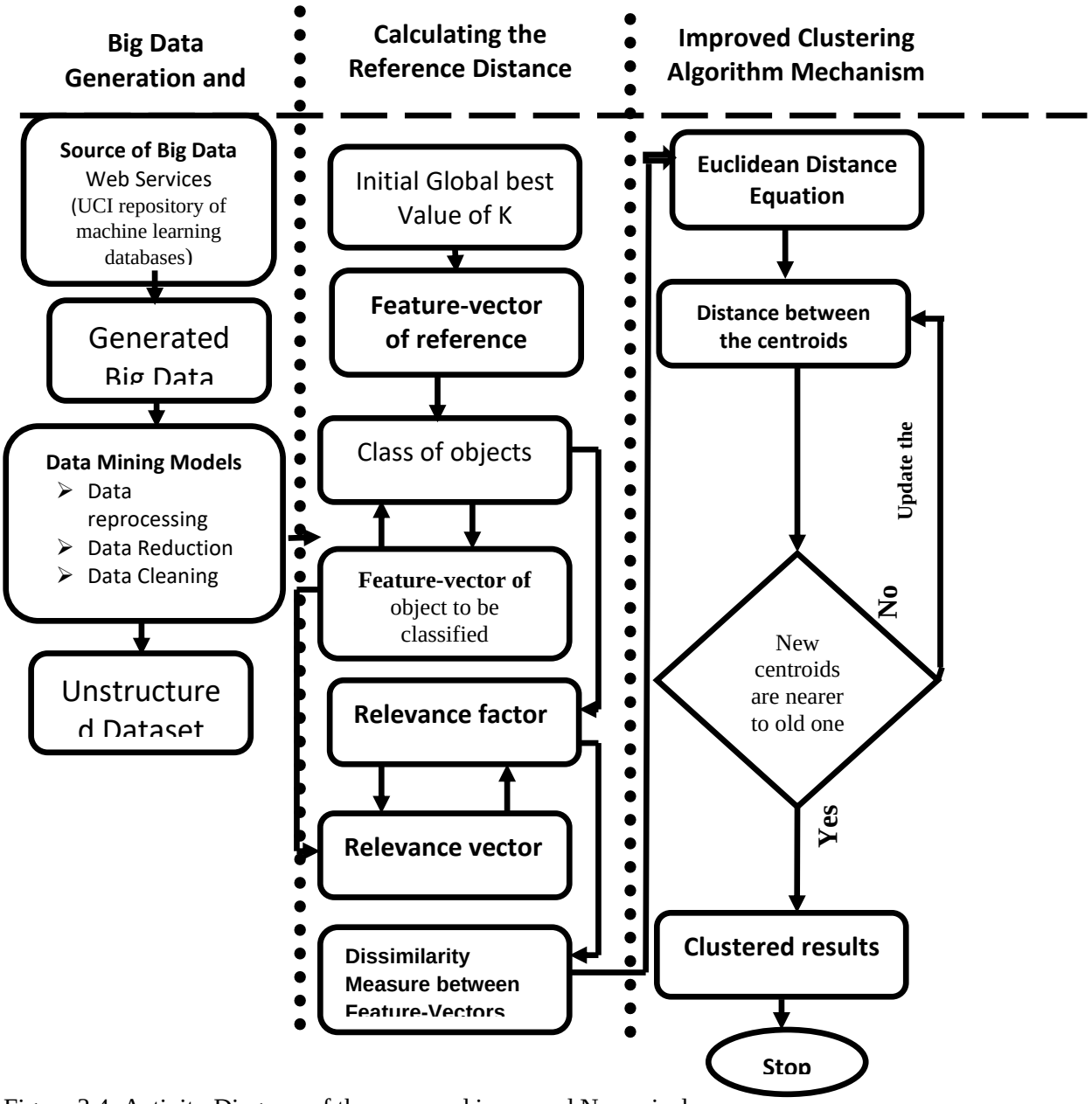

Figure 3.4: Activity Diagram of the proposed improved Numerical Clustering Algorithm.

The components are as follows:

i. Initial Global best Value of K: this is the accurate point or position where the Euclidean distance is used to find the shortest distance between two points (x1, x2) and (x2, y2) in a two-dimensional space.
ii. Feature-vector of Reference: is the parameter that is associated with the classes in the dataset, it is a vector from whom the centroid is measured. It evaluates the similarity between two feature vectors to be grouped.
iii. Class of objects: The class of objects is the features of attributes with instances of the datasets in the nth dimension.
iv. Relevance factor: Relevance factors are used to point out the importance/weight of the respective feature of objects under the conditions of the problem to be solved.
v. Dissimilarity Measure between Feature-Vectors: this shows the clear differences of classes of objects in a given region.
vi. Euclidean Distance Equation: This is the distance used to find the shortest distance between two points (x1, x2) and (x2, y2) in a two-dimensional space.

Secondly, the RDW-K-means clustering algorithm was integrated with Chi-square relative frequency dissimilarity measure used in K-representative clustering algorithm to develop an improved clustering algorithm called K-reference Clustering algorithm for clustering unstructured (mixed) dataset. The proposed K-reference Clustering algorithm (i.e. improved clustering algorithm for mixed dataset) consists of RDW, Euclidean distance measure, and Chi-square relative frequency dissimilarity measure which were used to develop an analytic mechanism for the unstructured dataset. RDW algorithms were developed to guide the clustering algorithm to efficiently perform its task of creating clusters of similar objects and dissimilar objects at any given region of the high dimensional space. This mechanism was achieved by forming a reference factor that uses the relevant vector to access and create similar and dissimilar clusters from every feature-vector of the object to be classified from the class of the objects. (i.e., the relevance factor helps to determine the reference distance). The outcome of the process will be used by the clustering algorithm to create accurate clusters by computing the distance between the centroids to produce clustered outputs. The Activity

Diagram of the proposed improved Clustering Algorithms for the mixed dataset is shown in Figure 3.5

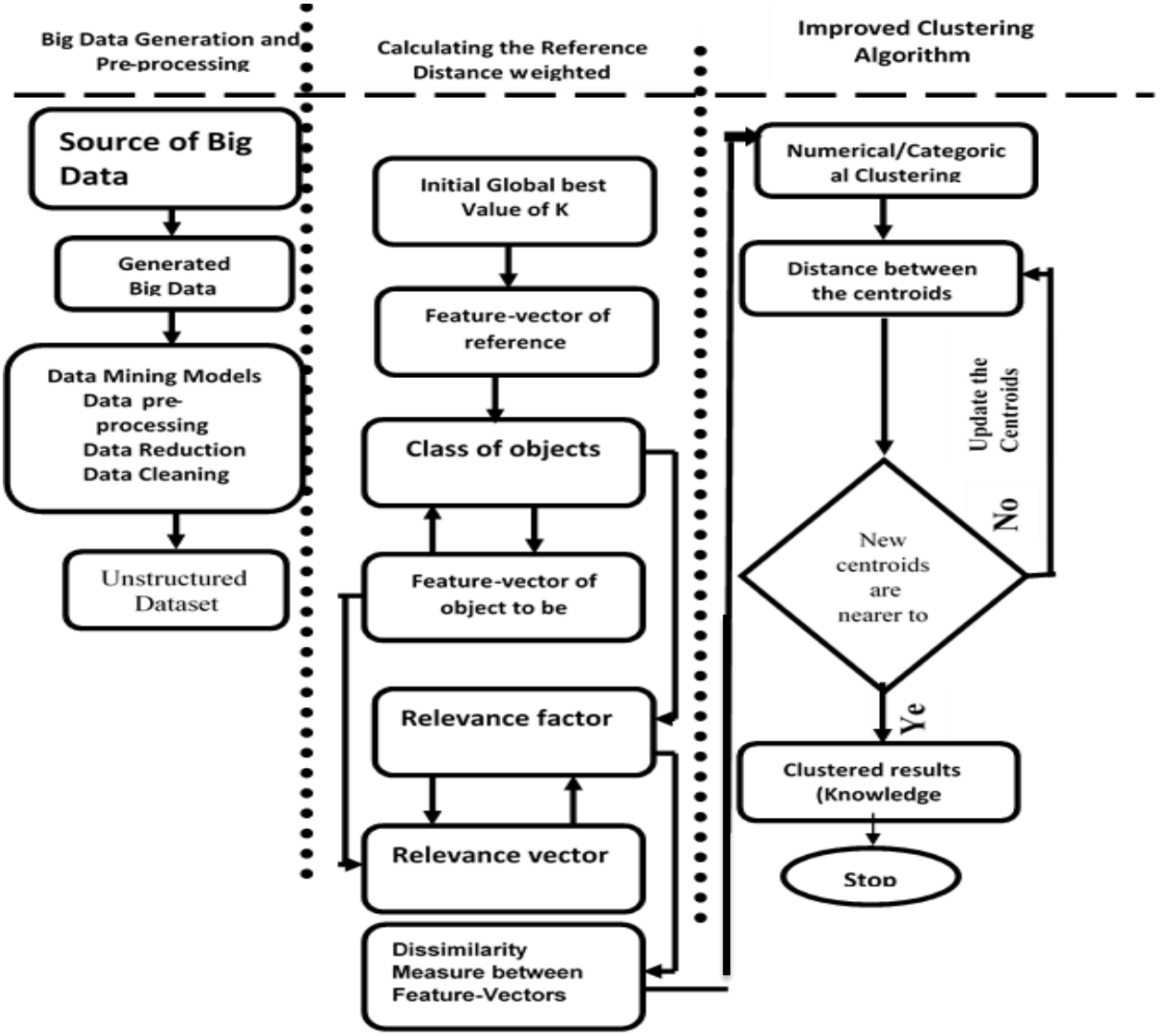

Figure 3.5: Activity Diagram of the proposed improved Numerical and Categorical

3.2.3.2 **Activity diagram to describe how objective two was achieved**

The Activity Diagram of the proposed Hybridization of the improved Clustering Algorithm with Dynamic Multi-Swarm Optimization Algorithms is shown in Figure 3.6.

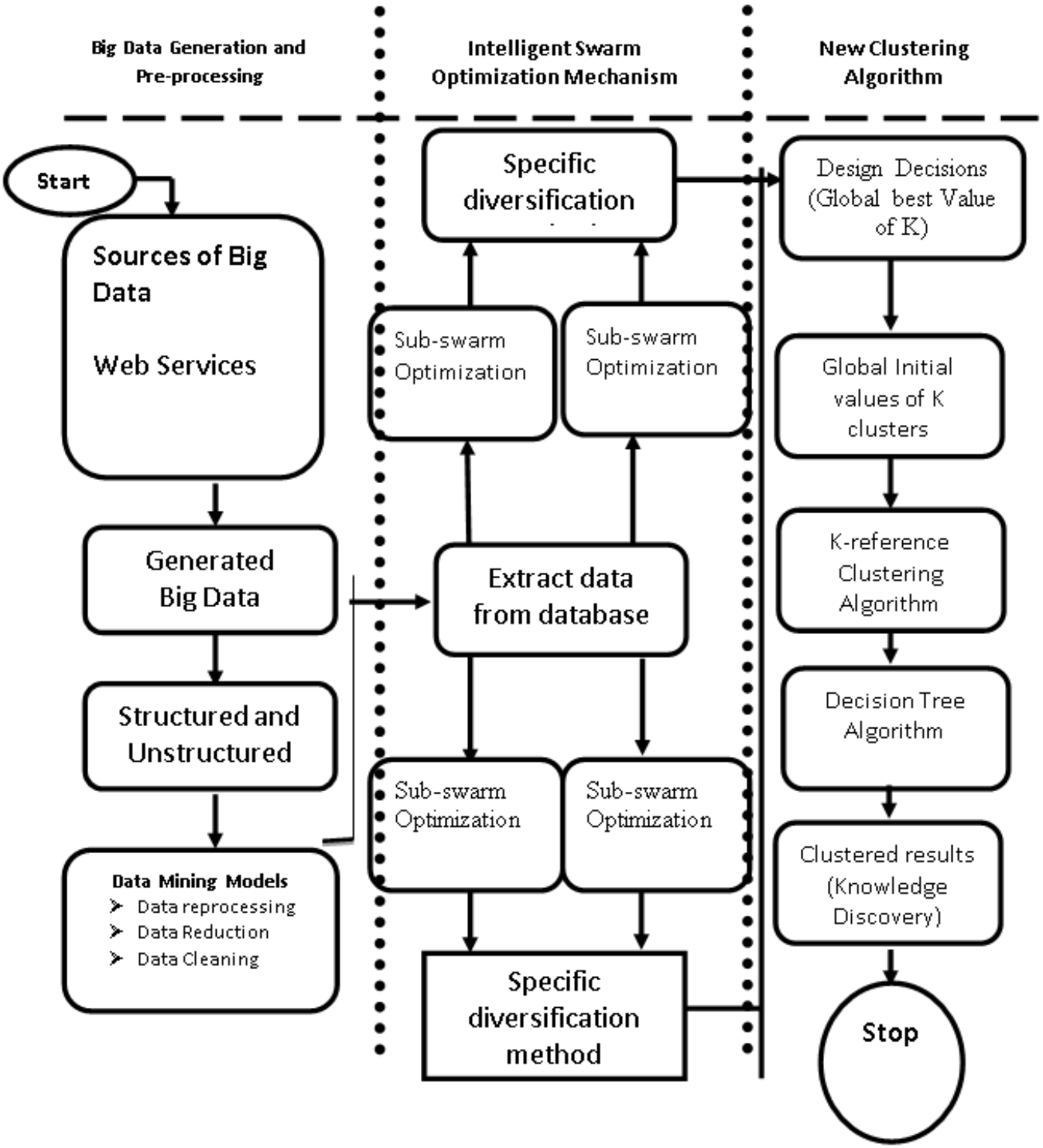

Figure 3.6: Activity Diagram of the Hybridization of the K-Reference Clustering

To achieve objective two which states that "To develop a hybridized algorithm with the improved clustering algorithm and dynamic Multi-Swarm Optimization Algorithm for Analysis of Big Dataset", K-reference clustering algorithm were hybridized with Dynamic Multi-swarm Optimization algorithm to develop a robust and efficient algorithm called Dynamic-K-reference

Clustering algorithm. It will overcome the problems (such as high dimensionality, massiveness, heterogeneous data types, complexity, incomplete, unstructured and noisy, etc.) that big data characteristics posed on traditional data mining techniques on creating clusters of mixed datasets. The intelligent concept of Dynamic Multi-Swarm guides the proposed clustering algorithm to obtain accurate convergences of the objective function by providing the best global initial value of K clusters.

The Dynamic multi-swarm optimization algorithm was chosen among other meta-heuristic algorithms because it can achieve an effective balance in multi-model problems, instead of trying to reach a compromise between exploration and exploitation which could weaken both mechanisms of the search process like in the case of Artificial Bee Colony and Firefly algorithms, etc., But dynamic multi-swarm algorithm handles the multi-model problems by processing them into different phases. At the different phases, such that exploitation (individual sub-swarms) and exploration (diversification method) are handled with different intelligence at the processing time.

The tactic applied in the dynamic multi-swarm optimization algorithm entails that each sub-swarm are assigned to a specific region while the agent, specific diversification method decides where and when to launch and re-assign tasks to the sub-swarms. It uses the mechanism of MapReduce to partition the entire space into various sub-space, where each swarm optimizes the location of the best global particle in a local environment.

3.2.3.3 **Activity diagrams used to describe how objective three was achieve**

Objective three states that “To scrape real-world dataset of the terrorist attack in Nigeria from social media for implementation and performance analysis of the improved big data analytic model”. To achieve objective three, an activity diagram was used to describe the implementation of the proposed model using a real-world dataset and java programming language. Also, Python programming language was used for clustering output results visualization. The activity diagram is shown in Figure 3.7.

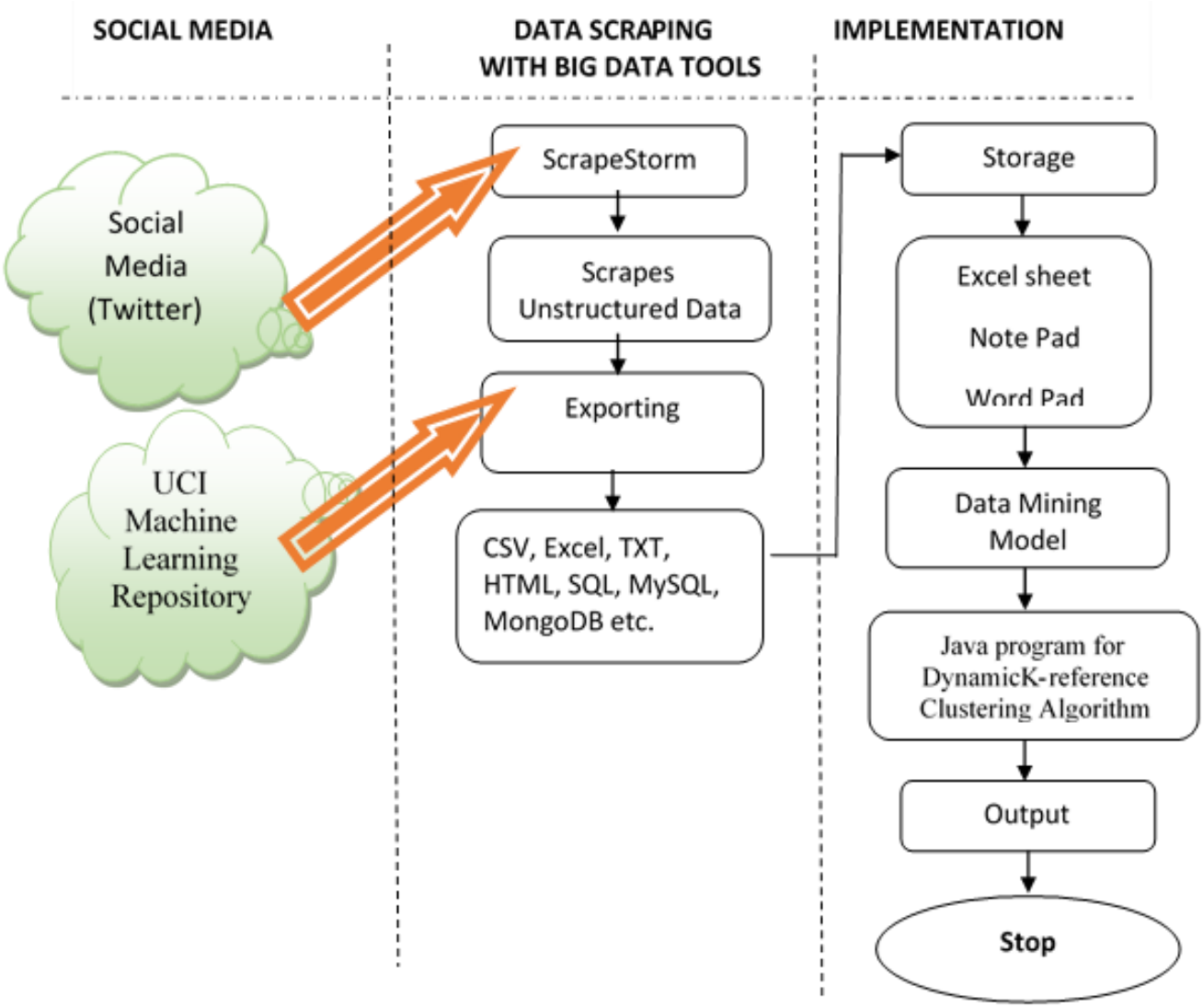

Figure 3.7: Activity diagram of the performance of the proposed model using real world dataset and java programming language.

The social media and UCI machine learning Repository are the two sources of the big dataset used in this work. Apache Spark and ScrapeStorm are big data tools used to extract the Boko Haram insurgency attacks dataset from Twitter. The scraped unstructured dataset was exported and stored in an excel spreadsheet. The raw datasets were preprocessed using a data mining model with a notepad and excel sheet. The data mining model outputs were used as input to compute the proposed dynamic-reference clustering algorithm in the java programming language. The datasets were converted to .dat file extension in MATLAB for smoot analysis with the developed Dynamic-K-representative clustering algorithm.

3.2.3.4 **Evaluation of Clustering Accuracy Model to Achieve Objective Four**

Objective Four States that "Carryout a comparative performance analysis of the proposed and similar models for big data analytics.". To achieve objective five, the performance of the proposed model was evaluated using clustering accuracy and errors.

1. **Clustering Accuracy**

A cluster is called a pure cluster if all the objects belong to a single class.

The clustering accuracy 'r' is defined as (San et al, 2004)

$$r = \frac{1}{n}\sum_{l=1}^{k} a_l \qquad (3.21)$$

Where a_i is the number of data objects that occur in both cluster C_l and its corresponding labeled class. a_i has the maximal value and n is the number of objects in the data set.

3. **Clustering Error**

The clustering error e is defined as (San et al, 2004):

$$e = 1 - \frac{1}{n}\sum_{l=1}^{k} a_l \qquad (3.22)$$

.

3.2.4.5 **Input and Output Designs**

The input design is an interactive user interface for mining large dataset. It consists of select different dataset, click to cluster, and accuracy of the system. Select different dataset option button enables the user to select different datasets from the database. Accuracy box displays the clustering error for the efficiency of the proposed system. Clustered outputs Result display the various distinct clusters of the attributes. The clear results box is used to clean all the outputs results. The Input and Output Designs is shown in Figure 3.8

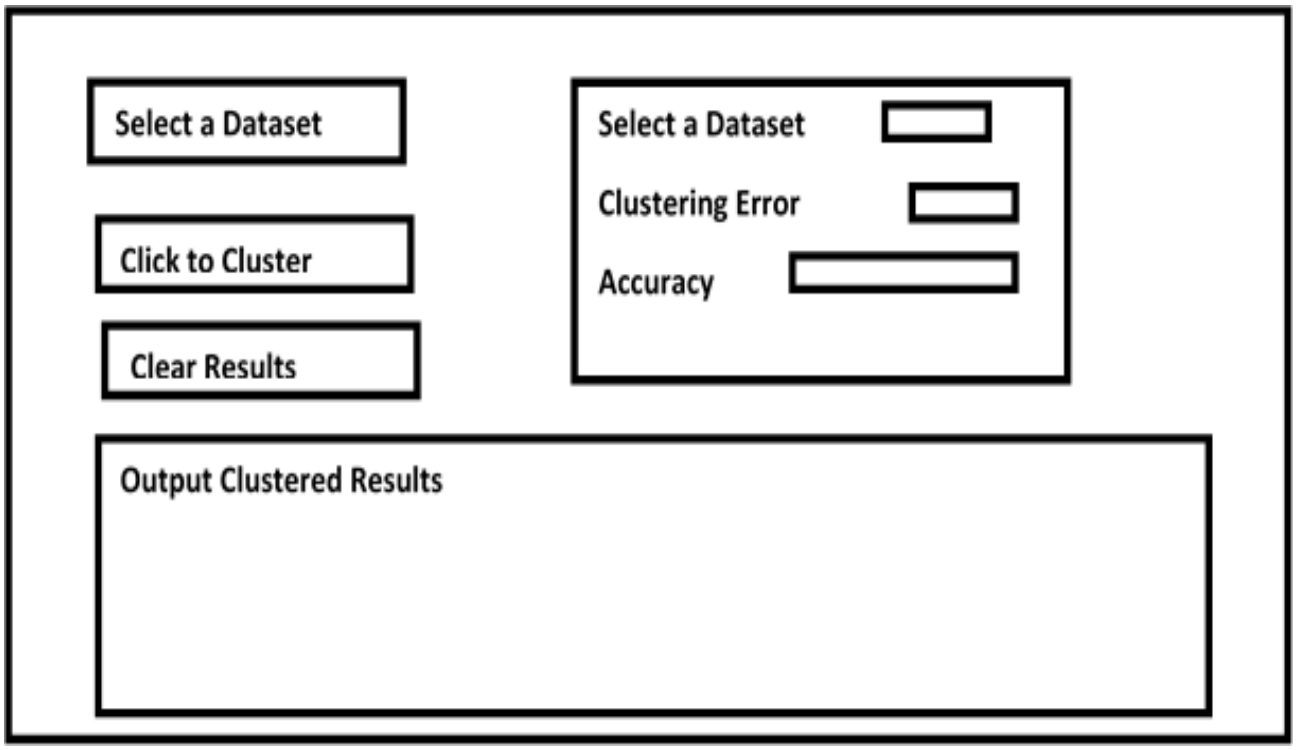

Figure 3.8: Input and Output Designs

3.2.4 **Implementation**

During this phase, the class objects and the interrelationships of these classes are translated and actually coded using the programming language decided upon. The databases are made and the complete system is given a functional shape.

3.2.4.1 **Choose of Programming Languages**

Java and Python Programming Language were used for the development of this project. Java was used to develop the proposed Dynamic-K-reference clustering algorithm analytic results while Python was used to develop visualization app for knowledge discovery. Java and python are developed to be simple, object-oriented, distributed, interpreted, secure, portable, high-performance, and dynamic. Java is a full-featured, general purpose programming language that is capable of developing robust applications. Also, python has flexible integration with Jupyter notebook, Apache Hadoop and vast machine learning libraries.

3.2.4.2 **System Requirements**

The system requirements are as follows: sofware tools and Hardwares specifications.

1. **Software**

The softwares requirements include: Spark Apache version 3.0.0, Jupyter Notebook, Java programming language version 15.0.2, ScrapeStorm, Brower (Chrome), Microsoft spreadsheet, MATLAB version R2018a, NotePad, Netbeans Integrated Development Environment (IDE) version 8.2, Python programming language version 3.8 and Operating System (Windows, and Linux operating systems.

2. **Hardware Specifications**

The Hardware Specifications are as follows:

i. Intel Core i3, Core i5 with 3.5GK, Core i7-7700HE,
ii. At least 4 to 8 GB of RAM
iii. At least 500 GB to 1 Terabyte of Hard Disk
iv. An enhance keyboard
v. Mouse Type: PS2 or USB
vi. Display Device: 14’ to 19’ Inch Monitor

3.2.4.3 Source of Data

Unstructured datasets were extracted from Twitter users in social network web sites to demonstrate the efficiency of the proposed model. ScrapeStorm open source software was used to scrape unstructured dataset of Boko Haram insurgence attack menace from twitter website. The unstructured dataset was preprocessed using data mining models: selection, cleaning, integration to convert the unstructured dataset to structural form for analysis.

Also, to validate the accuracy of the proposed hybrid clustering algorithm, six (6) datasets: Soybean, Hepatitis, Australian Credit Approval, German Credit Data, Statlog Heart and Yeast datasets used by

the existing algorithms was obtained from UCI Machine Learning Repository to verify the performance of the proposed hybrid algorithm.

A. Database Design/ Storage Specification

In this section, the various dataset attributes used to evaluate the performance of the machine learning algorithms in this work were discussed. The dataset's attributes specification shows tables for Hepatitis, German Credit Card dataset, and Boko Haram Insurgency Attacks dataset. MATLAB editor was utilized for the storage of each of the datasets using the file extension ".data". The database storage specification for Boko Haram Insurgency is shown in Figure 3.9.

Figure 3.9 shows the Database Design Attribute Specification for Boko Haram Insurgency Attacks datasets in Nigeria of 6050 tweeters. It attributes like Area of Attack, Month of Attack, Year of Attack, Death-Roll, and Remark. The datasets are stored in MATLAB Editor in a Matrices form for easy upload and computational analysis with a machine learning algorithm. The sample of the dataset is shown in Figure 3.9.

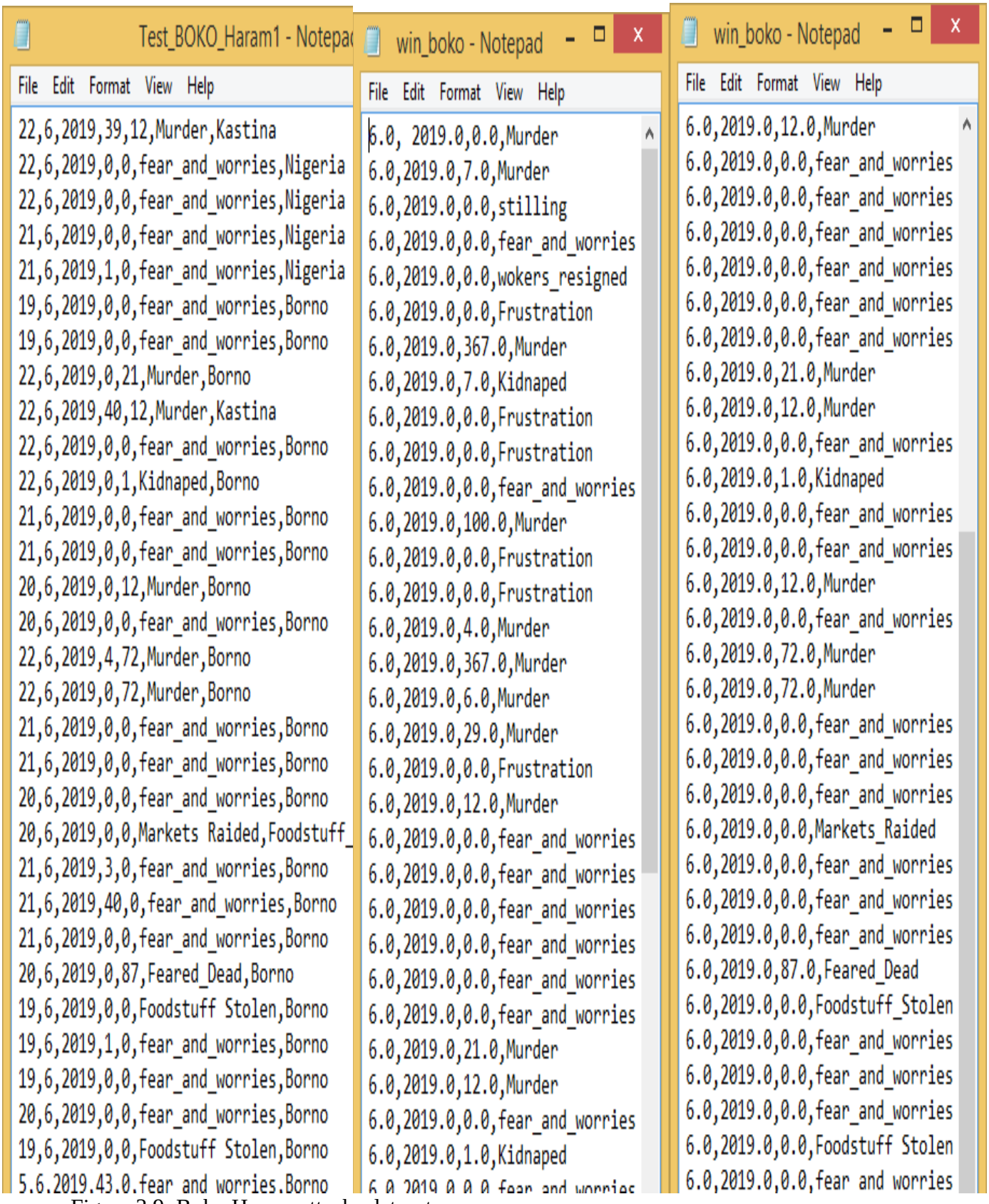

Figure 3.9: Boko Haram attacks dataset

The Datasets Attributes of hepatitis dataset is shown in Table 3.31.

Table 3.31: Datasets Attributes for Hepatitis Dataset

```
 Attribute information:
     1. Class: DIE, LIVE
     2. AGE: 10, 20, 30, 40, 50, 60, 70, 80
     3. SEX: male, female
     4. STEROID: no, yes
     5. ANTIVIRALS: no, yes
     6. FATIGUE: no, yes
     7. MALAISE: no, yes
     8. ANOREXIA: no, yes
     9. LIVER BIG: no, yes
    10. LIVER FIRM: no, yes
    11. SPLEEN PALPABLE: no, yes
    12. SPIDERS: no, yes
    13. ASCITES: no, yes
    14. VARICES: no, yes
    15. BILIRUBIN: 0.39, 0.80, 1.20, 2.00, 3.00, 4.00
        -- see the note below
    16. ALK PHOSPHATE: 33, 80, 120, 160, 200, 250
    17. SGOT: 13, 100, 200, 300, 400, 500,
    18. ALBUMIN: 2.1, 3.0, 3.8, 4.5, 5.0, 6.0
    19. PROTIME: 10, 20, 30, 40, 50, 60, 70, 80, 90
    20. HISTOLOGY: no, yes
```

Table 3.1 shows the Datasets Attributes Design Specification for Hepatitis disease dataset. The attributes are as follows: Class, Age, Steroid, Antivirals, Malaise, Anorexia, Liver Big, Liver Firm, Spleen, Spiders, Ascites, Varices, Bilirubin, Alk Phophate, Sgot, Albumin, Protime and Histology. Hepatitis Dataset.

The Datasets Attributes for German Credit Card Dataset is shown in Table 3.32.

Table 3.32: Datasets Attributes for German Credit Card Dataset

```
Creditability,Account Balance,Duration of Credit (month),Payment Status of
Previous Credit,Purpose,Credit Amount,Value Savings/Stocks,Length of current
employment,Instalment per cent,Sex & Marital Status,Guarantors,Duration in
Current address,Most valuable available asset,Age (years),Concurrent
Credits,Type of apartment,No of Credits at this Bank,Occupation,No of
dependents,Telephone,Foreign Worker
1,1,18,4,2,1049,1,2,4,2,1,4,2,21,3,1,1,3,1,1,1
1,1,9,4,0,2799,1,3,2,3,1,2,1,36,3,1,2,3,2,1,1
1,2,12,2,9,841,2,4,2,2,1,4,1,23,3,1,1,2,1,1,1
1,1,12,4,0,2122,1,3,3,3,1,2,1,39,3,1,2,2,2,1,2
1,1,12,4,0,2171,1,3,4,3,1,4,2,38,1,2,2,2,1,1,2
1,1,10,4,0,2241,1,2,1,3,1,3,1,48,3,1,2,2,2,1,2
1,1,8,4,0,3398,1,4,1,3,1,4,1,39,3,2,2,2,1,1,2
1,1,6,4,0,1361,1,2,2,3,1,4,1,40,3,2,1,2,2,1,2
1,4,18,4,3,1098,1,1,4,2,1,4,3,65,3,2,2,1,1,1,1
1,2,24,2,3,3758,3,1,1,2,1,4,4,23,3,1,1,1,1,1,1
1,1,11,4,0,3905,1,3,2,3,1,2,1,36,3,1,2,3,2,1,1
,3,31,3,2,1,3,1,1,1
```

Table 3.32 shows the Datasets Attributes Design Specification for the Credit Card dataset. The attributes are as follows Creditability, Account Balance, Duration of Credit, Payment Status of

Previous Credit, Purpose Credit Amount, Value Savings/Stocks, Length of current employment, Instalment percent, Sex & Marital Status, Duration in current Address, Most Valuable available asset, Age, Concurrent credits, Type of apartment, No of credits at this bank, Occupation, No of Dependence, Telephone, Foreign Worker.

The German Credit Data contains data on 20 variables and the classification whether an applicant is considered a Good or a Bad credit risk for 1000 loan applicants.

3.2.4.4 **Use case Diagrams**

The use case diagram was used to depicts how the software was developed. The use case diagram for data clustering analytics are shown in Figure 3.10.

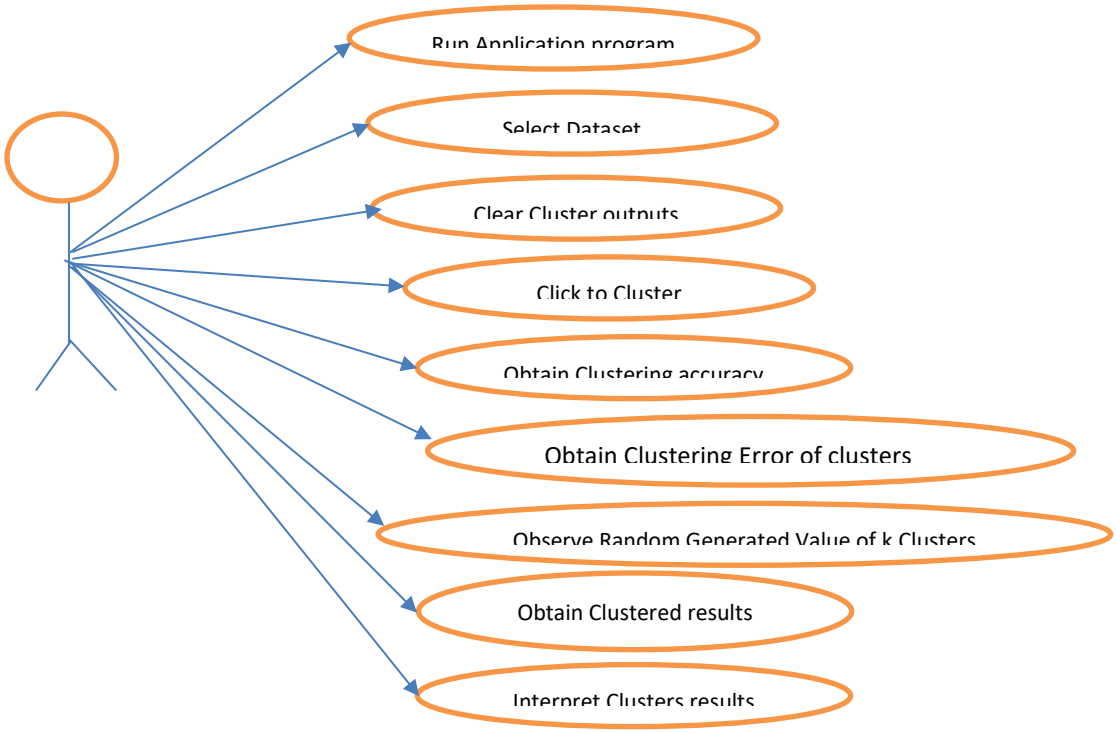

Figure 3.10: Use case Diagram for data clustering analytics

The use case diagram in Figure 12 describes the user's interaction with the system that describes the relationship between the user and the different use case in the developed system. The user runs the application program. Then selects the dataset from the dropdown box for analysis. It clicks on "clear cluster output" to prepare the output box for the next results. A click on "click to cluster" enables the

developed Dynamic-reference algorithm to analyze the selected dataset and outputs the system displays the clustering accuracy, clustering error of clusters, and the clustered datasets. The user then interprets the clusters results. The Use case Diagram for clustered datasets results Visualization is shown in Figure 3.11.

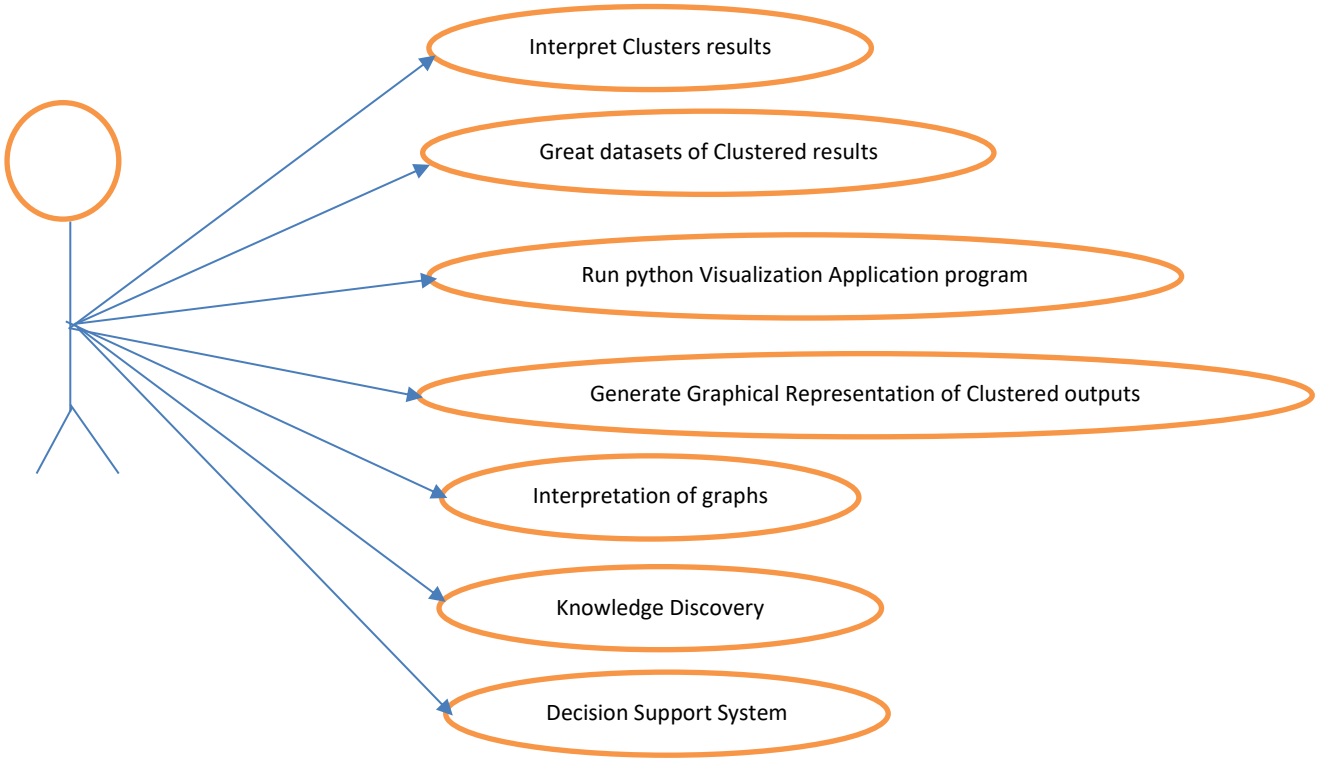

Figure: 3.11 Use case Diagram for clustered datasets results visualization

After the interpretation of the clustered result. The results are used to plot visualization graphs for knowledge discovery. The user inserts the clustered output results that has been interpreted into the python app and runs the python application program. The system generates graphical representation of the clustered result for visualization and knowledge discovery. The discovered knowledges are used for decision support system.

3.2.4.5 **Testing**

System testing is a stage in the implementation, which is aimed at ensuring that the system works accurately and efficiently as per the user need, before the live operation commences. Testing is vital

to the success of a system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. A series of tests are performed before the system is ready for the user acceptance test. The testing steps are:

(i) Unit Testing

Unit testing focuses on the smallest unit of software design. This is known as module testing. The modules are tested separately. The test was carried out during programming stage itself. In this step, each module was and found to be working satisfactorily as regards to the expected output from the module.

(ii) Integration Testing

Data can be lost across an interface. One module can have an adverse effect on sub functions, when combined may not be linked in a desired manner as major functions. Integration testing is a systematic approach for constructing the program structure, while at the same time conducting test to uncover errors associated within the interface. The objective is to take unit tested modules to build program structure. All the modules are combined and tested as a whole.

(iii) Validation

At the culmination of the integration testing, software is completely assembled as a package. Interfacing errors may have been uncovered and corrected, while the final series of software test begin in the validation testing. Validation testing can be defined in many ways, but a simple definition is that the validation succeeds when the software functions in a manner that is expected by the customer. After validation test has been conducted, one of the three possible conditions exist.

a. The function or performance characteristics confirm to specification and are accepted.
b. A deviation from specification is uncovered and a deficiency lists is created.
c. Proposed system under consideration has been tested by using validation test and found to be working satisfactory.

(iv) Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in a specific format. The output format on the screen is found to be correct. The format was designed in the system design time according to the user needs. For the hard copy also; the output comes as per the specified requirements by the user. Hence output testing did not result in any correction for the system.

(v) User Acceptance Testing

User acceptance of a system is the key factor for the success of any system. The system under consideration is tested for the user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes whenever required. This is done in regard to the following point:

a. Input Screen Design

b. Output Screen Design

c. Format of reports and other outputs

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 **Results**

4.1.1: **Graphical User Interface System**

Interactive user interface was designed using Java programing language. The user follows the instructions to select different datasets and clicks on cluster command button to produce the output. The system interface is shown in Figure 4.1.

Figure 4.1: Graphical User Interface of the Proposed System.

The Graphical User Interface of the Proposed System consists of: select datasets, clear records, Click to clusters, Random Generated Values of Cluster, Clustering Error, and Accuracy. The select datasets drop box enables the user to select the dataset for mining. If the user selects a dataset for analysis, he/her will then click on “Click to cluster” to calculate the randomly generated values of clusters, the clustering error of clusters, and the clustering accuracy of the analytic model. Before the next

computation, the user used the clear record button to erase the current results. The simulation result of the soybeans dataset analysis produced with the Dynamic-K-reference clustering algorithm is shown in Figure 4.2.

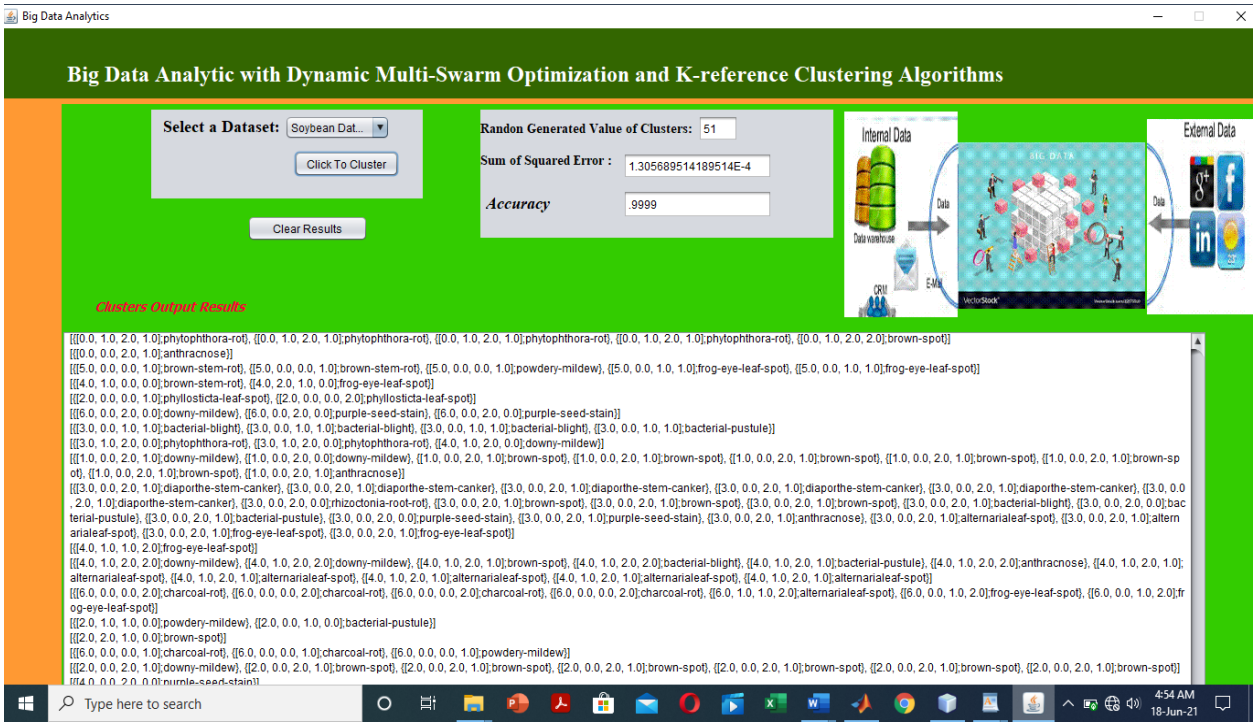

Figure 4.2: Graphical representation of the simulation results of soybeans dataset.

The tabulated results of five (5) simulations analysis are shown in Table 4.1

Table 4.1: **Simulation Result of Clustering Accuracy and Error of Different numbers of Clusters for Hepatitis Disease Dataset**

Performance Measures	Simulation 1	Simulation 2	Simulation 3	Simulation 4	Simulation 5
Number of Clusters	4	14	36	55	80
Clustering Error	0.1398	0.04644	0.0159	0.0083	0.0036
Clustering Accuracy	0.8602	0.9536	0.9841	0.9916	0.9964

Table 4.1 analyzed the Dynamic-K-reference clustering algorithm simulation results of clustering accuracy and error of different numbers of clusters for Hepatitis disease dataset. Table 4.1 consists of six (6) columns. Column 1 shows the performance measures of Dynamic-k-reference clustering

algorithm given as the number of clusters, the clustering error, and the clustering accuracy. Column 2 to 5 shows the results of simulation 1 to simulation 5.

Simulation 1 produced 4 clusters, clustering accuracy of 0.8602 and clustering error of 0.1398. Simulation 2 produced 14 clusters, clustering accuracy of 0.9536 and clustering error of 0.0464. Simulation 3 produced 36 clusters, clustering accuracy of 0.9841 and clustering error of 0.0159. Simulation 4 produced 55 clusters, clustering accuracy of 0.99916 and clustering error of 0.0083. Simulation 5 produced 80 clusters, clustering accuracy of 0.9964 and clustering error of 0.0036. Simulation 5 produced the highest accuracy of 0.9964 with 80 clusters. The results show that the Dynamic-K-reference clustering algorithm has the analytic strength to produce accurate clusters. This shows that the larger the number of clusters the more accurate the performance of the model. The results are represented graphically in Figure 4.3.

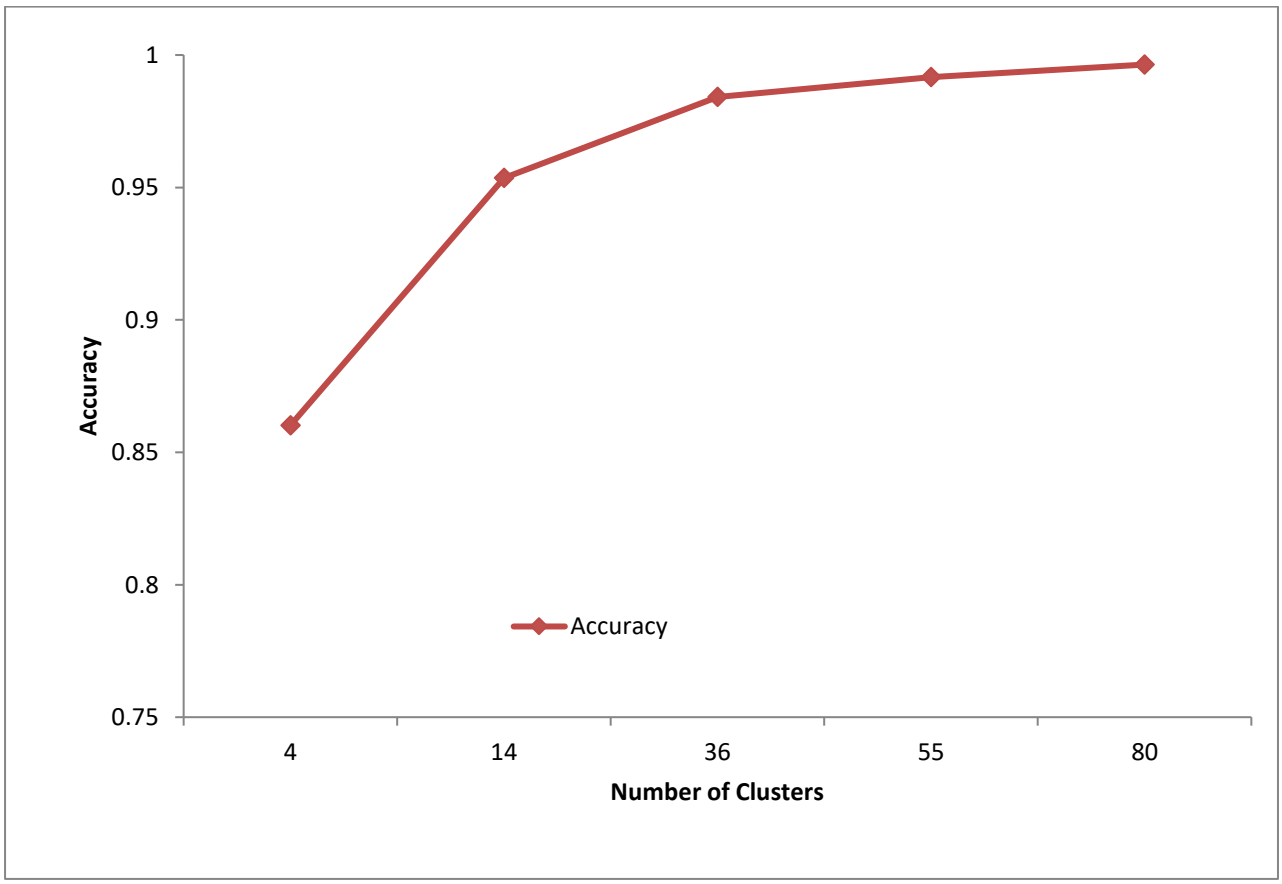

Figure 4.3: Graphical Representations of the clustering accuracy of different numbers of clusters for Hepatitis Dataset.

The clustering accuracy of different number of clusters output for win disease dataset is shown in Figure 4.4.

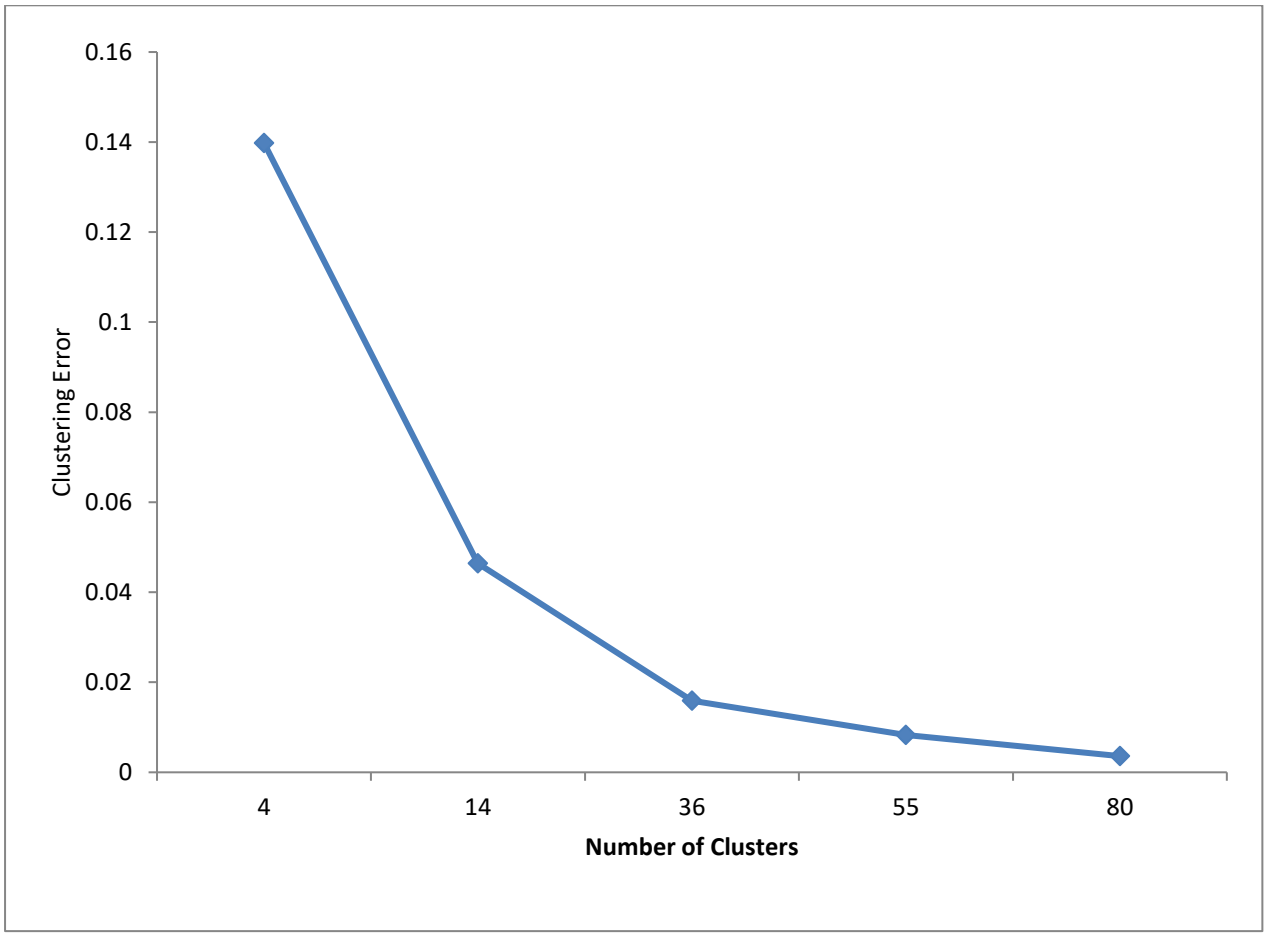

Figure 4.4: Graphical Representations of clustering error of different number of clusters output for hepatitis disease dataset.

The Dynamic-K-reference algorithm's simulation results of the clustering accuracy and error of different number of clusters for Australian disease dataset is shown in Table 4.2.

Table 4.2: **Simulation Result of Clustering Accuracy and Error of Different numbers of Clusters for Australian Disease Dataset**

Australian Credit Card Dataset						
s/n	Performance Measures	Simulation 1	Simulation 2	Simulation 3	Simulation 4	Simulation 5
1	Clusters	4	21	37	57	87
2	Clustering Error	0.1398	0.1194	0.0453	0.0261	0.0132
3	Clustering Accuracy	0.8602	0.8891	0.9546	0.9739	0.9868

Table 4.2 show the simulation result of clustering accuracy and error of different numbers of clusters for Australian disease dataset. It consists of six (6) columns. Column 1 shows the performance measures of Dynamic-k-reference clustering algorithm given as the number of clusters, the clustering error, and the clustering accuracy. Column 3-5 shows the results of simulation 1 to simulation 5. Simulation 1produced 4 clusters, clustering error of 0.1398 and clustering accuracy of 0.8602. Simulation 2 produced 21 clusters, clustering error of 0.1194 and clustering accuracy of 0.8891. Simulation 3 produced 37 clusters, clustering error of 0.0453 and clustering accuracy of 0.956. Simulation 4 produced 57 clusters, clustering error of 0.0261 and clustering accuracy of 0.9739. Simulation 5 produced 87 clusters, clustering error of 0. 0132 and clustering accuracy of 0.9868. Cluster 5 has the most minimum clustering error and accuracy. This implies that the proposed algorithm is very efficient to mine the experimental dataset.

The results are represented graphically in Figure 4.5.

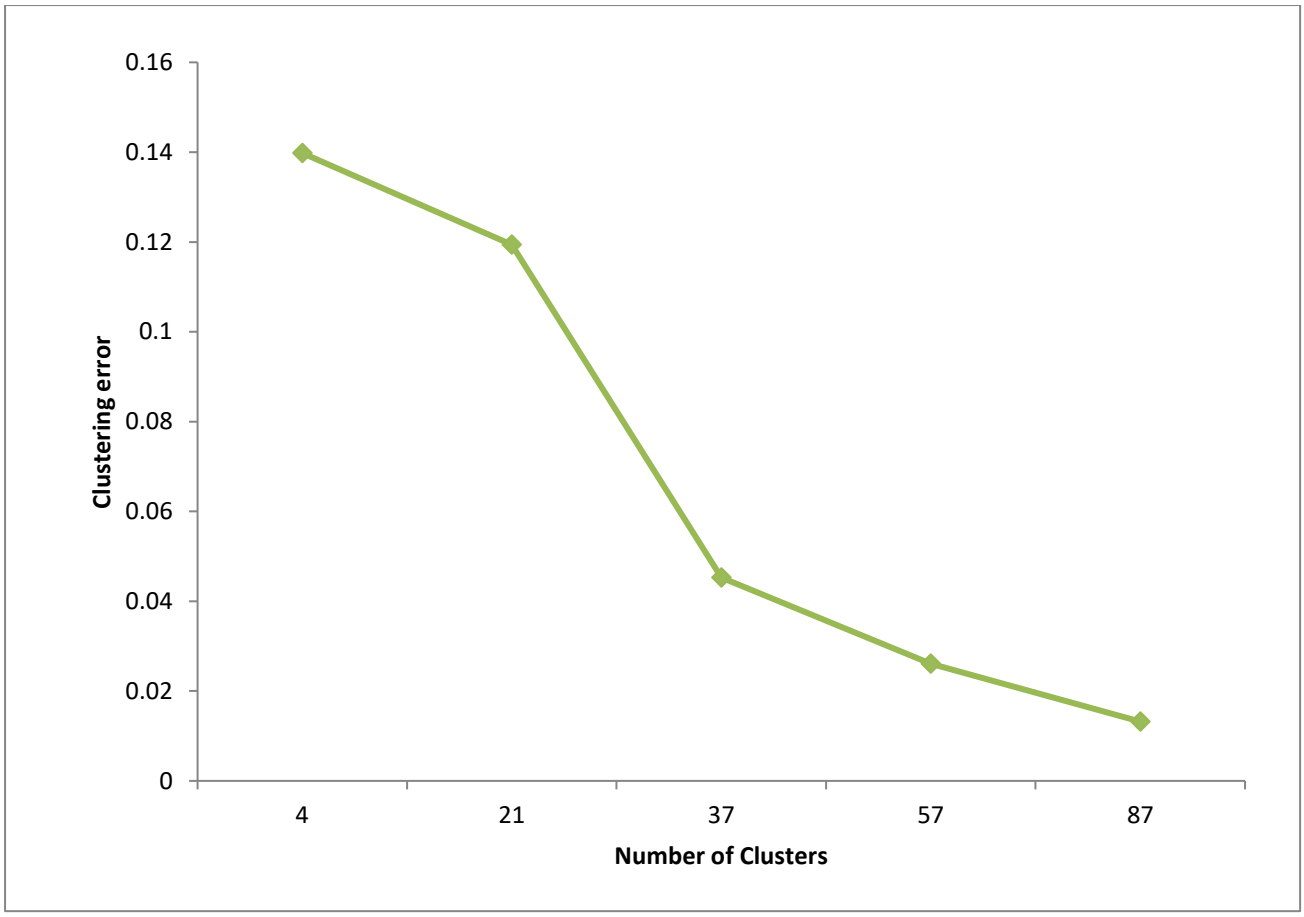

Figure 4.5: Graphical Representations of clustering "error" of different numbers of clusters output for Australian Credit dataset.

The clustering accuracy of different number of clusters output for Australian Credit dataset is represented in Figure 4.6.

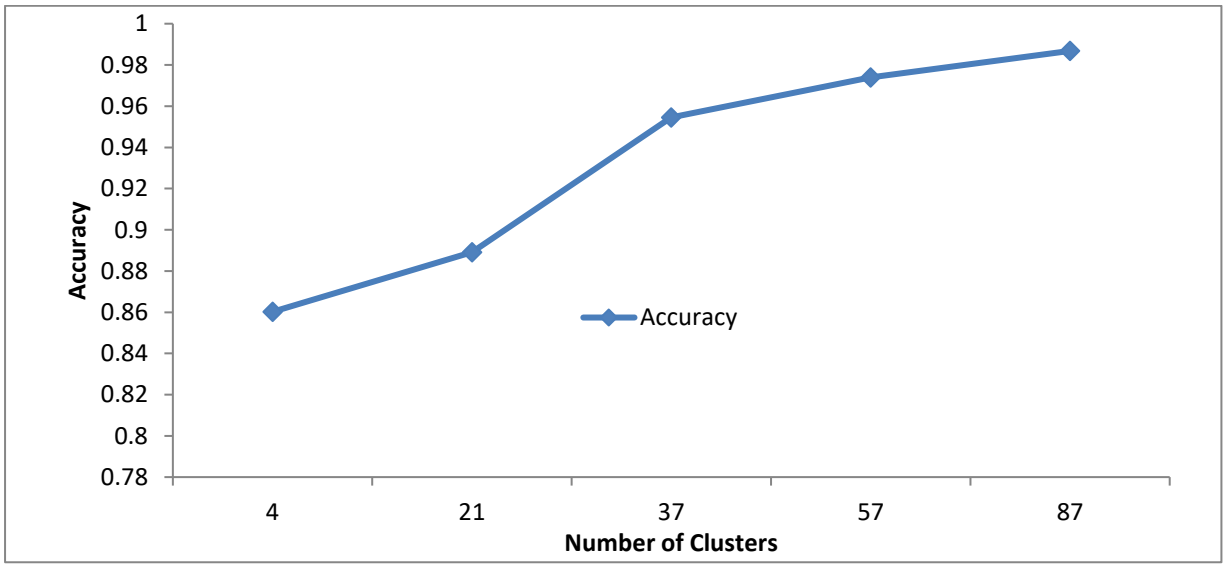

Figure 4.6: Graphical representations of clustering accuracy of different numbers of clusters for Australian Credit dataset.

The simulation results of clustering accuracy and error of different numbers of clusters for Starlog Heart disease dataset is shown in Table 4.3.

Table 4.3: **Simulation Result of Clustering Accuracy and Error of Different numbers of Clusters for Starlog Heart Disease Dataset**

Performance Measures	**Simulation 1**	**Simulation 2**	**Simulation 3**	**Simulation 4**	**Simulation 5**
Clusters	3	13	37	96	98
Clusters Error	0.00023	0.00009	0.000036	0.000014	0.000013
Clusters Accuracy	0.9998	0.9999	1	0.9964	1

Table 4.3 shows the Simulation results of the clustering error of different numbers of clusters of the Starlog Heart disease dataset produced with Dynamic-K-reference clustering algorithm. It consists of six (6) columns. Column 1 shows the performance measures of Dynamic-k-reference clustering algorithm given as the number of clusters, the clustering error, and the clustering accuracy. Column 2-5 shows the results of simulation 1 to simulation 5. Simulation 1 produces 3 clusters, clustering error of 0.00023 and clustering accuracy of 0.9998. Simulation 2 produces 13 clusters, clustering error of 0.00009 and clustering accuracy of 0.9999. Simulation 3 produces 37 clusters, clustering error of 0.0000036 and clustering 1. Simulation 4 produces 96 clusters, clustering error of 0.0000014 and clustering accuracy of 0.9946. Simulation 5 produces 98 clusters, clustering error of 0. 000013 and clustering accuracy of 1. The proposed algorithm is very efficient to mine the experimental dataset.

The results are represented graphically in Figure 4.7.

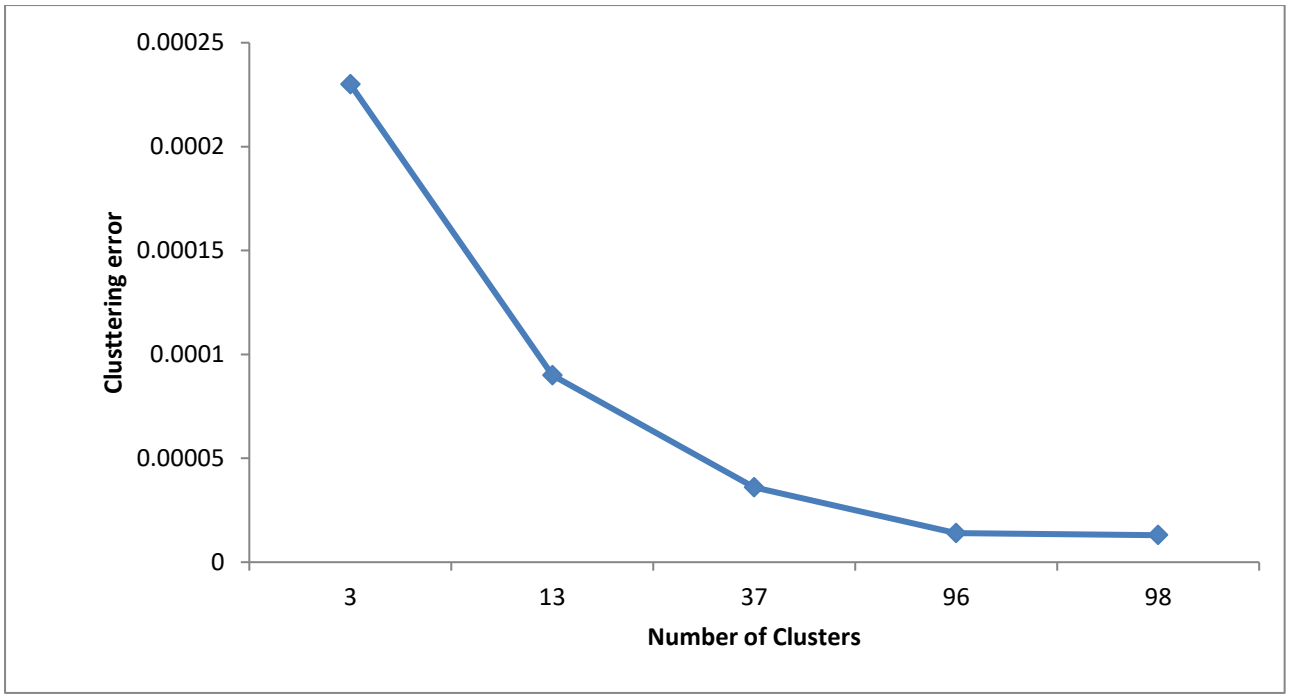

Figure 4.7: Graphical Representations of clustering error of different numbers of clusters output for Starlog Heart disease dataset.

The clustering accuracy of different number of clusters output for Starlog Heart disease dataset is shown in Figure 4.8.

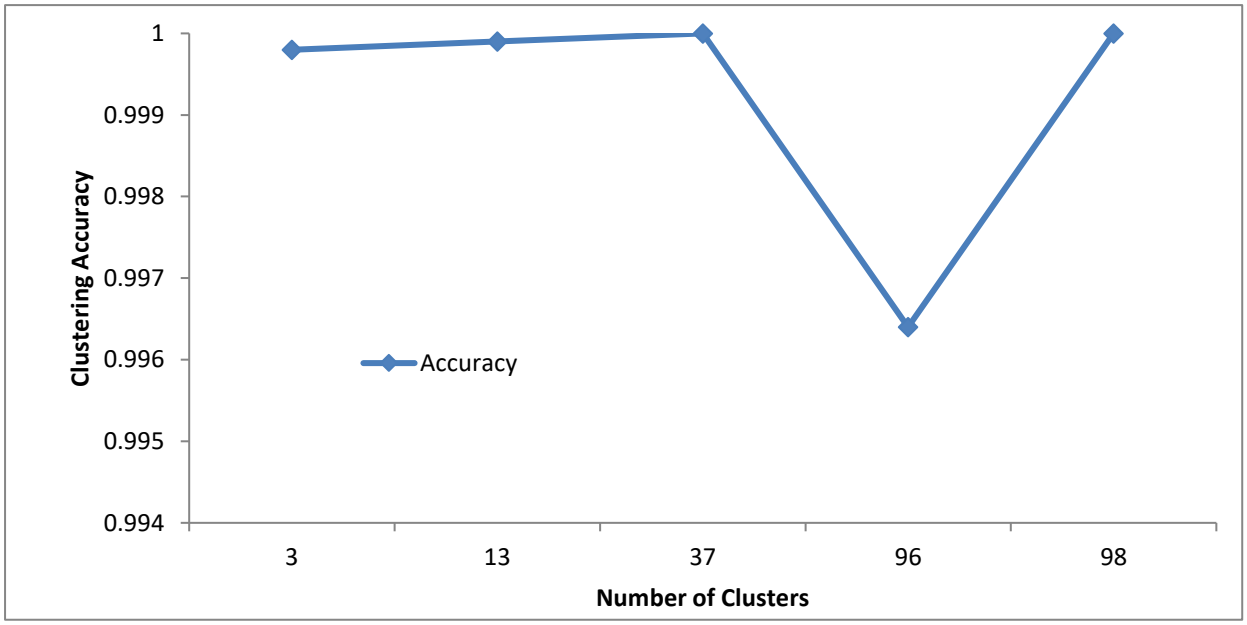

Figure 4.8: Graphical Representations of clustering accuracy of different numbers of clusters for Starlog Heart disease dataset.

The simulation results of clustering accuracy and error of different numbers of clusters for German Credit Card dataset is shown in Table 4.4.

Table 4.4: **Simulation Result of Clustering Accuracy and Error of Different numbers of Clusters for German Credit Card Dataset**

Performance Measures	**Simulation 1**	**Simulation 2**	**Simulation 3**	**Simulation 4**	**Simulation 5**
Number of Clusters	17	27	55	86	99
Clustering Error	0.10534	0.05643	0.02183	0.00499	0.00359
Accuracy	0.8947	0.9436	0.9782	0.995	0.9964

Table 4.4 shows the simulation result of clustering accuracy and error of different numbers of clusters for German Credit Card Dataset with the Dynamic-K-reference clustering algorithm. It consists of six (6) columns. Column 1 shows the performance measures of Dynamic-k-reference clustering algorithm given as the number of clusters, the clustering error, and the clustering accuracy. Column 2-5 shows the results of simulation 1 to simulation 5.

Simulation 1 produced 17 clusters, clustering accuracy of 0.8947 and clustering error of 0.10534. Simulation 2 produced 27 clusters, clustering accuracy of 0.9436 and clustering error of 0.05643. Simulation 3 produced 55 clusters, clustering accuracy of 0.9782 and clustering error of 0.02183. Simulation 4 produced 86 clusters, clustering accuracy of 0.9995 and clustering error of 0.0050. Simulation 5 produced 99 clusters, clustering accuracy of 0.9964 and clustering error of 0.0036. Simulation 5 produced the highest accuracy with clusters 99. The proposed algorithm was very efficient to mine the experimental dataset.

The results are represented graphically in Figure 4.9.

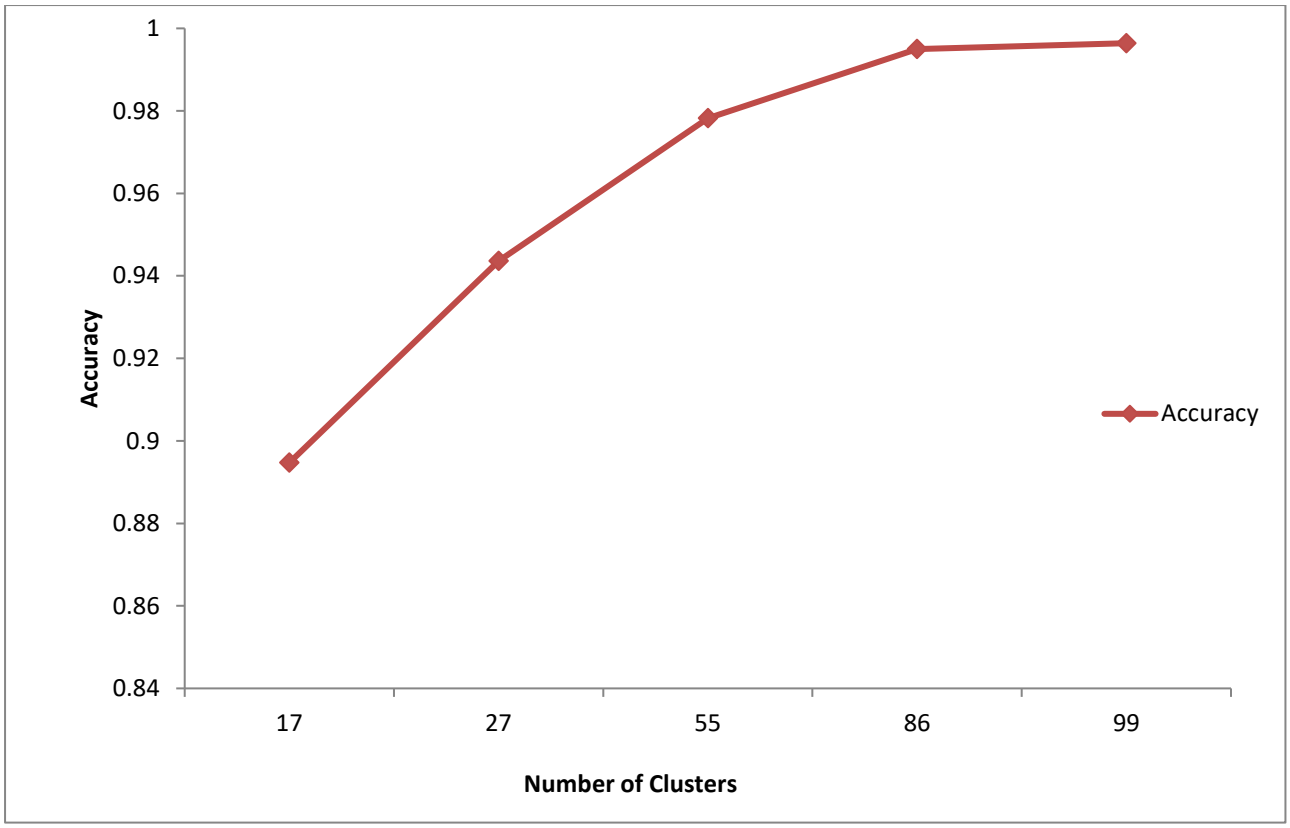

Figure 4.9: Graphical Representations of clustering accuracy of different numbers of clusters for German Credit Card dataset.

The simulation result of clustering error of different number of clusters for German Credit Card dataset is shown in Figure 4.10.

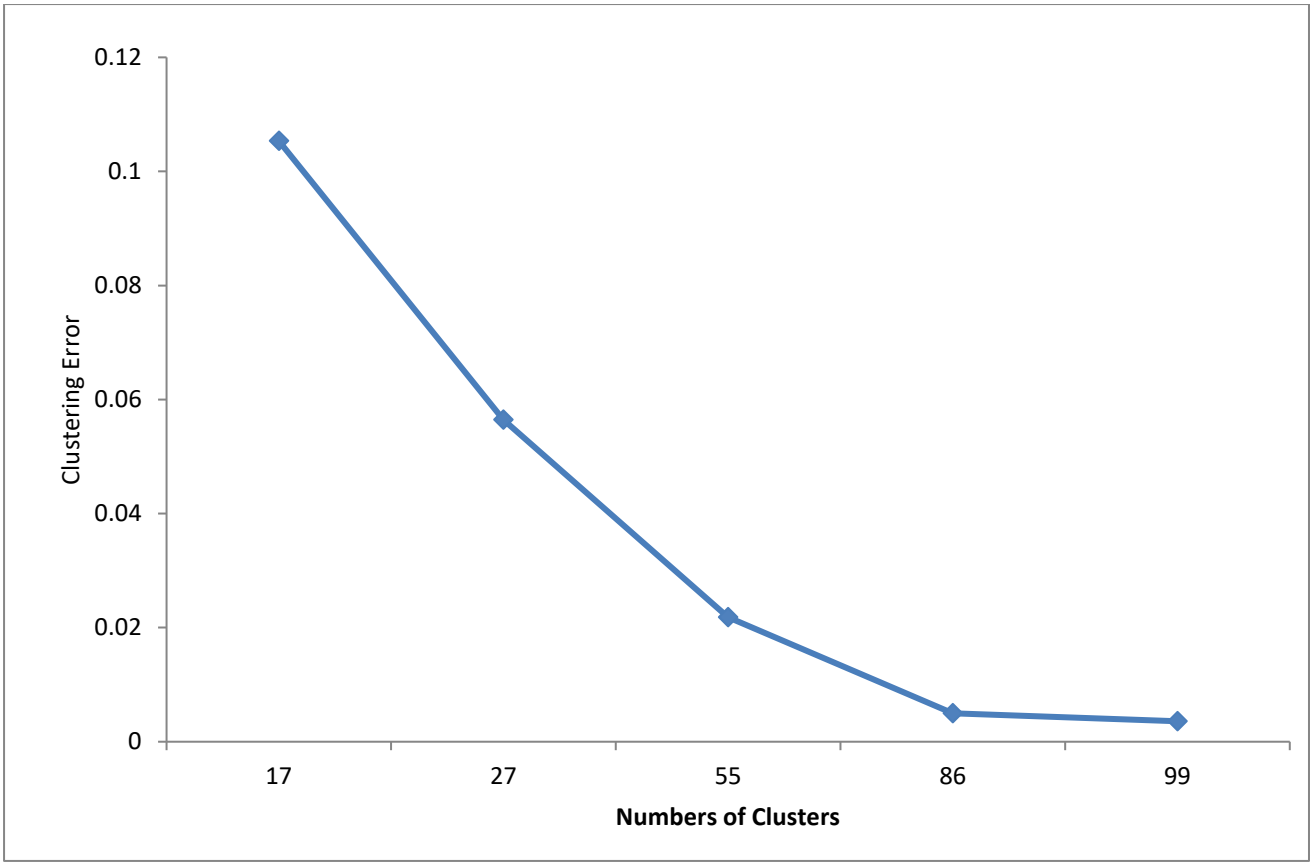

Figure 4.10: Graphical Representations of clustering error of different numbers of clusters output for German Credit Card dataset.

The comparison of the accuracy values of Dynamic-K-reference and Binary Swarm-based k-prototype clustering algorithm is shown in Table 4.5.

Table 4.5: Comparison of the clustering accuracy values of Dynamic-K-reference and Binary Swarm based k-prototype Clustering Algorithms.

S/N	Dataset	PSO based K-prototype algorithm (Prabha et al., 2015)	Proposed Dynamic-K-reference Clustering Algorithm
1	Hepatitis	0.7521	0.99964
2	Australian Credit Approval	0.8229	0.9852
3	German Credit Data	0.6261	0.9964
4	Statlog Heart	0.8387	1.000

Table 4.5 analyzed the comparison of the accuracy values of Dynamic-K-reference and Binary Swarm based k-prototype Clustering Algorithms. Column 2 shows the experimental datasets such as Hepatitis, Australian Credit Approval, German Credit Data, and Statlog Heart disease datasets, for the validation of the accuracy of the various algorithms. Column 3 shows the accuracy of PSO based K-prototype algorithm and column 4 shows the accuracy of Dynamic-K-reference Clustering Algorithm. The result is represented in Figure 4.11.

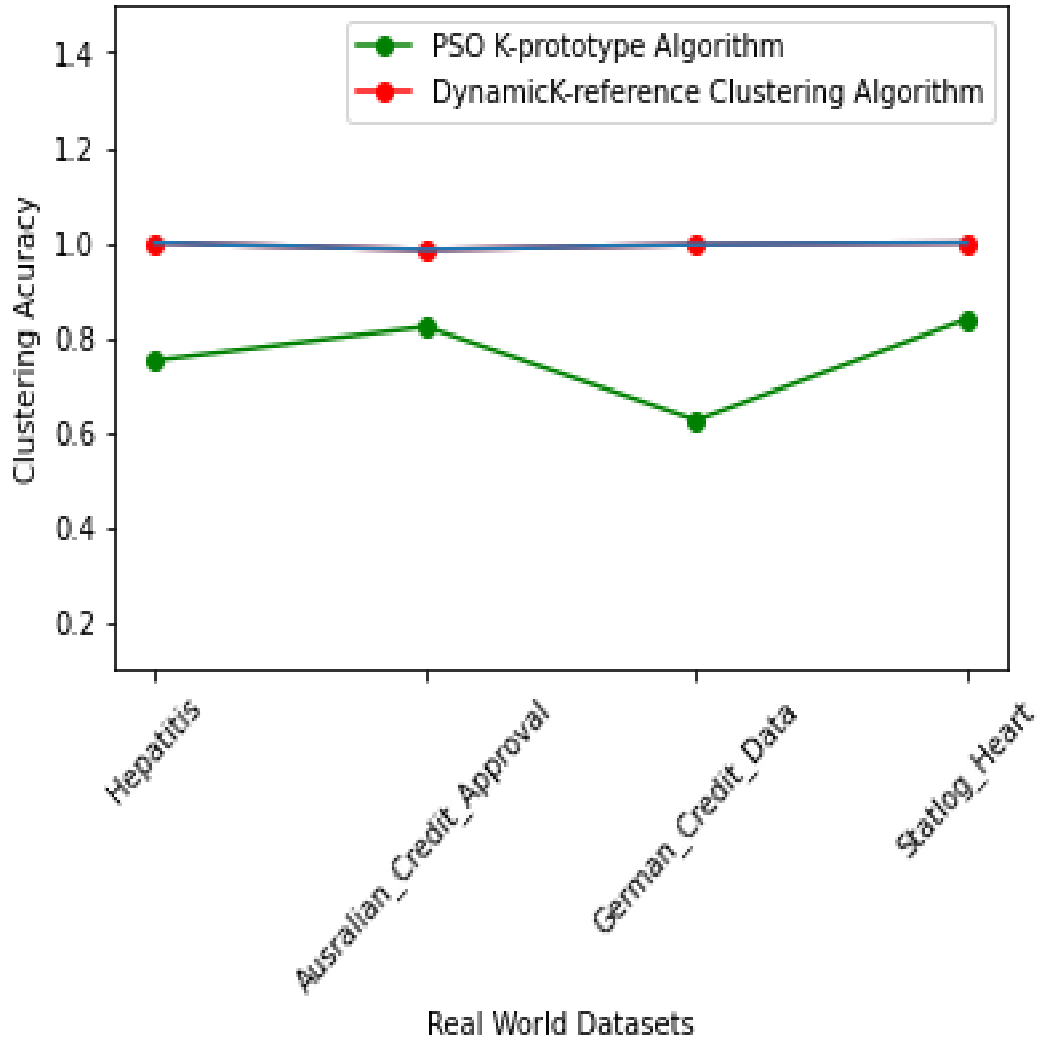

Figure 4.11: Graphical representation of comparison of the accuracy values of Dynamic-K-reference and Binary swarm-based k-prototype clustering algorithms.

The Percentage distribution of the performance improvement of Dynamic-K-reference clustering algorithm on Binary swarm-based k-prototype clustering algorithm. is shown in Table 4.6.

Table 4.6: Percentage distribution of the Performance Improvement of Dynamic-K-reference clustering algorithm on Binary swarm-based k-prototype clustering algorithms

Datasets	Performance Improvement
Hepatitis	22.2
Australian Credit Approval	16.8
German Credit Data	34.8
Statlog Heart	11.8

Table 4.6 shows the percentage distribution of the performance improvement of Dynamic-K-reference clustering algorithm. It consists of two columns. Column 1 shows the datasets (Hepatitis, Australian Credit Approval, German Credit Data and Starlog Heart) used for the experiments. Column 2 shows the percentage distribution of performance improvement of Dynamic-K-reference clustering algorithms. The result is represented in Figure 4.12.

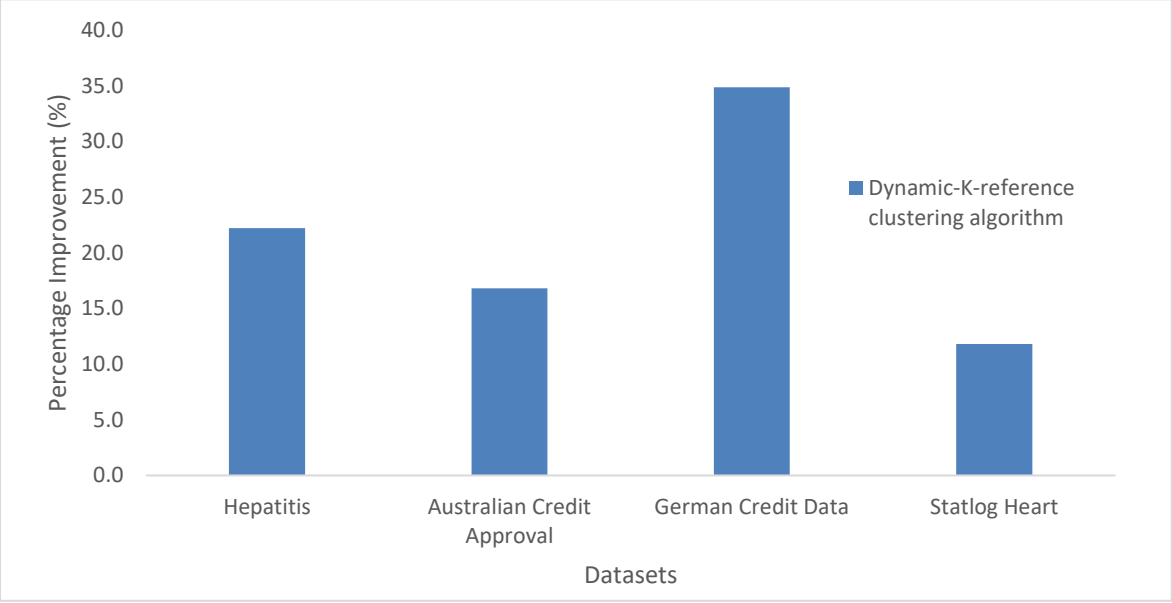

Figure 4.12: Graphical representation the performance improvement of Dynamic-K-reference clustering validated with various datasets (e.g. Hepatitis, Australian Credit Approval, German Credit Data and Starlog Heart).

The comparison of the accuracy values of Dynamic-K-reference and MixK-meanXFon Clustering Algorithms is shown in Table 4.7.

Table 4.7: Comparison of the accuracy values of Dynamic-K-reference and MixK-meanXFon Clustering Algorithms

Datasets	Accuracy	MixK-meanXFon Algorithm	Dynamic-K-reference Algorithm
Soybean	Clustering Accuracy	0.86	0.998
Yeast		0.84	0.973

Table 4.7 shows the comparison of the accuracy values of Dynamic-K-reference and MixK-meanXFon Clustering Algorithms. It consists of four columns. Column 1 contains the datasets which consist of Soybean and Yeast datasets. Column 2 shows the clustering accuracy. Column 3 represents the MixK-meanXFon algorithm's clustering accuracy values of soyabean and yeast datasets. Lastly, Column 4 shows the Dynamic-K-reference clustering algorithm's accuracy values of soyabean and yeast datasets.

The comparison of the accuracy values of Dynamic-K-reference and MixK-meanXFon clustering algorithms are represented in Figure 4.13.

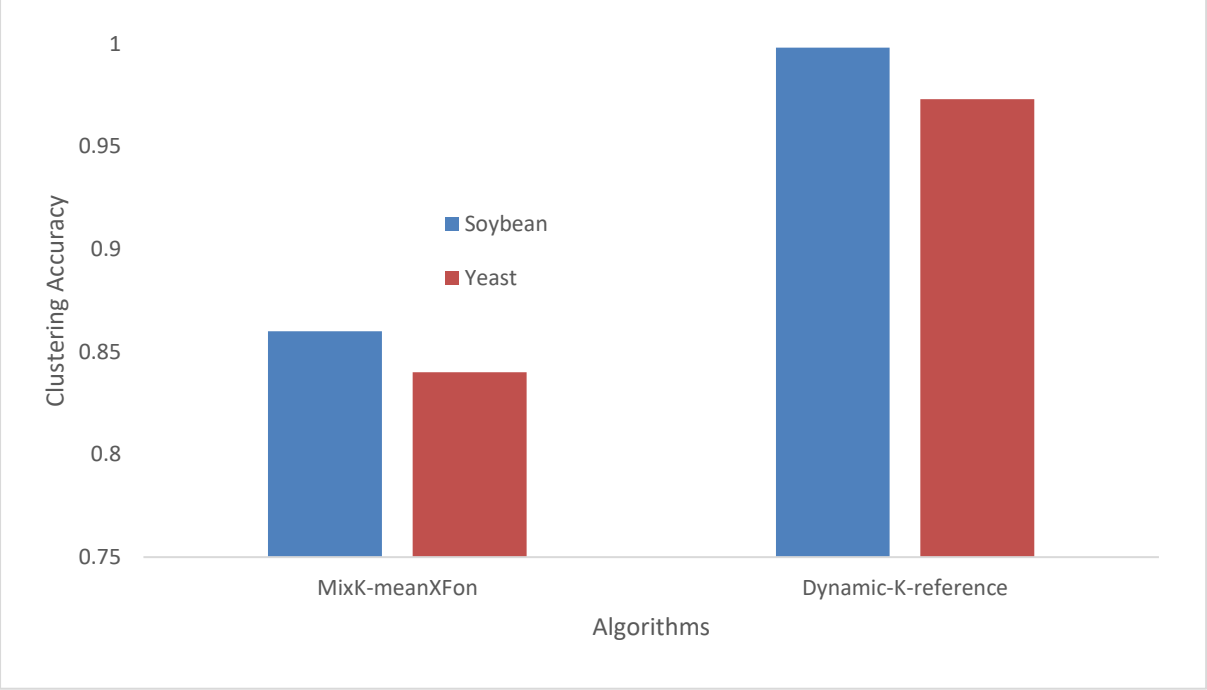

Figure 4.13: Graphical representation of the comparison of the clustering accuracy of Dynamic-K-reference and MixK-meanXFon Clustering Algorithms on Soybean and Yeast datasets.

Table 4.8: Comparison of the Clustering Error values of Dynamic-K-reference and MixK-meanXFon Clustering Algorithms

Datasets	Error of creating Clusters	MixK-meanXFon Algorithm	Dynamic-K-reference Algorithm
Soybean	Clustering Error	0.14	0.002
Yeast		0.16	0.027

Table 4.8 shows the comparison of the clustering error values of Dynamic-K-reference and MixK-meanXFon Clustering Algorithms. It consists of four columns. Column 1 represents the datasets which consist of Soybean and Yeast datasets. Column 2 shows the Clustering Error. Column 3 represents the MixK-meanXFon algorithm's clustering error values of soyabean and yeast datasets. Lastly, Column 4 shows the Dynamic-K-reference clustering algorithm's error values of soyabean and yeast datasets.

The comparison of the clustering error values of Dynamic-K-reference and MixK-meanXFon Clustering Algorithms is represented in Figure 4.14.

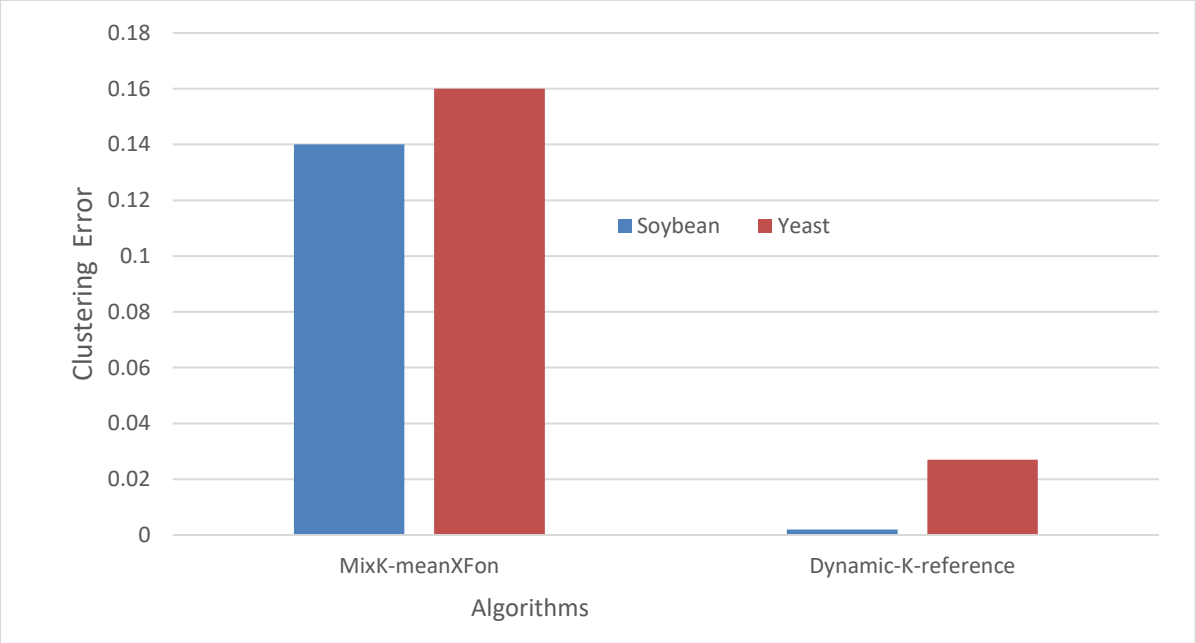

Figure 4.14: Graphical representation of the comparison of the clustering error values of Dynamic-K-reference and MixK-meanXFon clustering algorithms

The percentage distribution of the performance improvement of Dynamic-K-reference Clustering algorithm is represented in Table 4.9.

Table 4.9: Percentage distribution of the Performance improvement of Dynamic-K-reference clustering algorithm on MixK-meanXFon clustering algorithm

Datasets	Performance Improvement (%)
Soybean	13.8
Yeast	13.7

Table 4.9 shows the percentage distribution of the performance improvement of Dynamic-K-reference clustering algorithm. It consists of two columns. Column 1 shows the datasets (soybean and yeast). Column 2 shows the percentage distribution of performance improvement of Dynamic-K-reference clustering algorithms.

The result is represented in Figure 4.15.

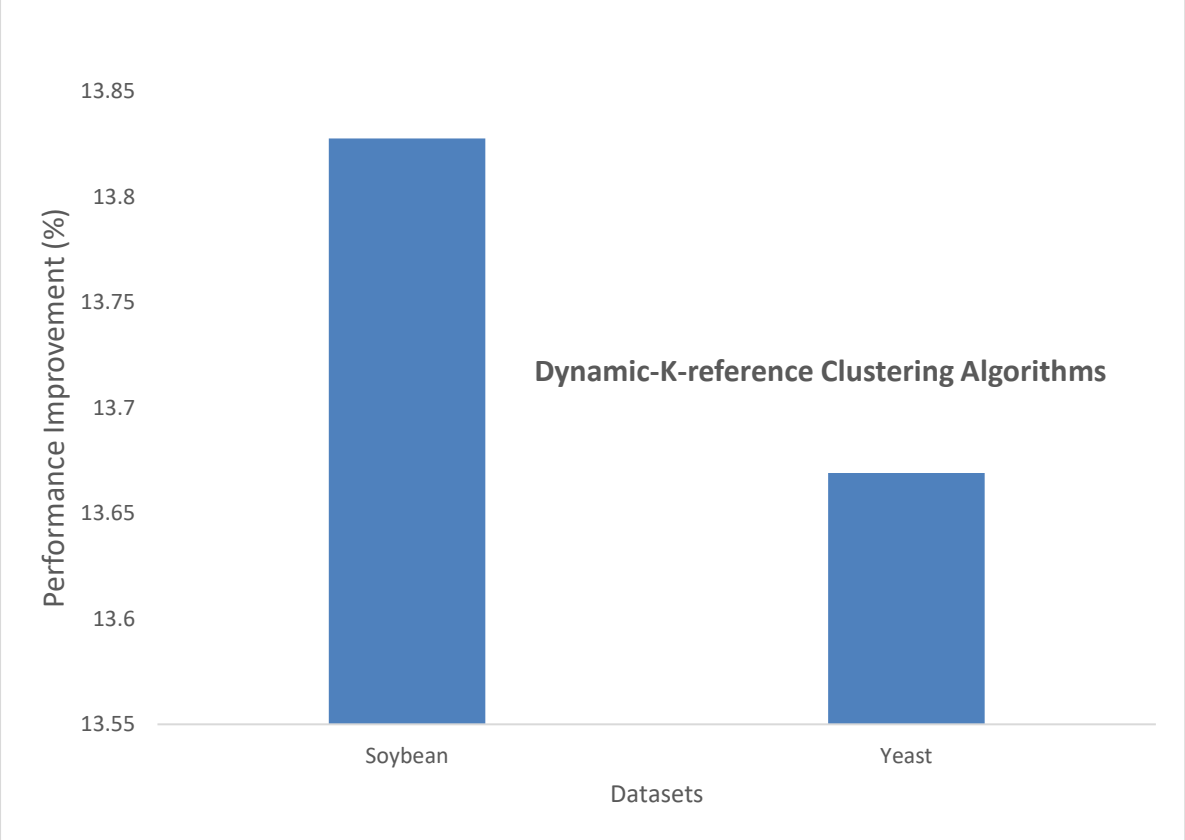

Figure 4.15: Graphical representation of the percentage distribution of performance improvement of Dynamic-K-reference Clustering algorithm

4.1.2 Clustered Output of Boko Haram Insurgency attack datasets

The clustered analysis of attack of Boko Haram insurgency in Nigeria is shown in Figure 4.16.

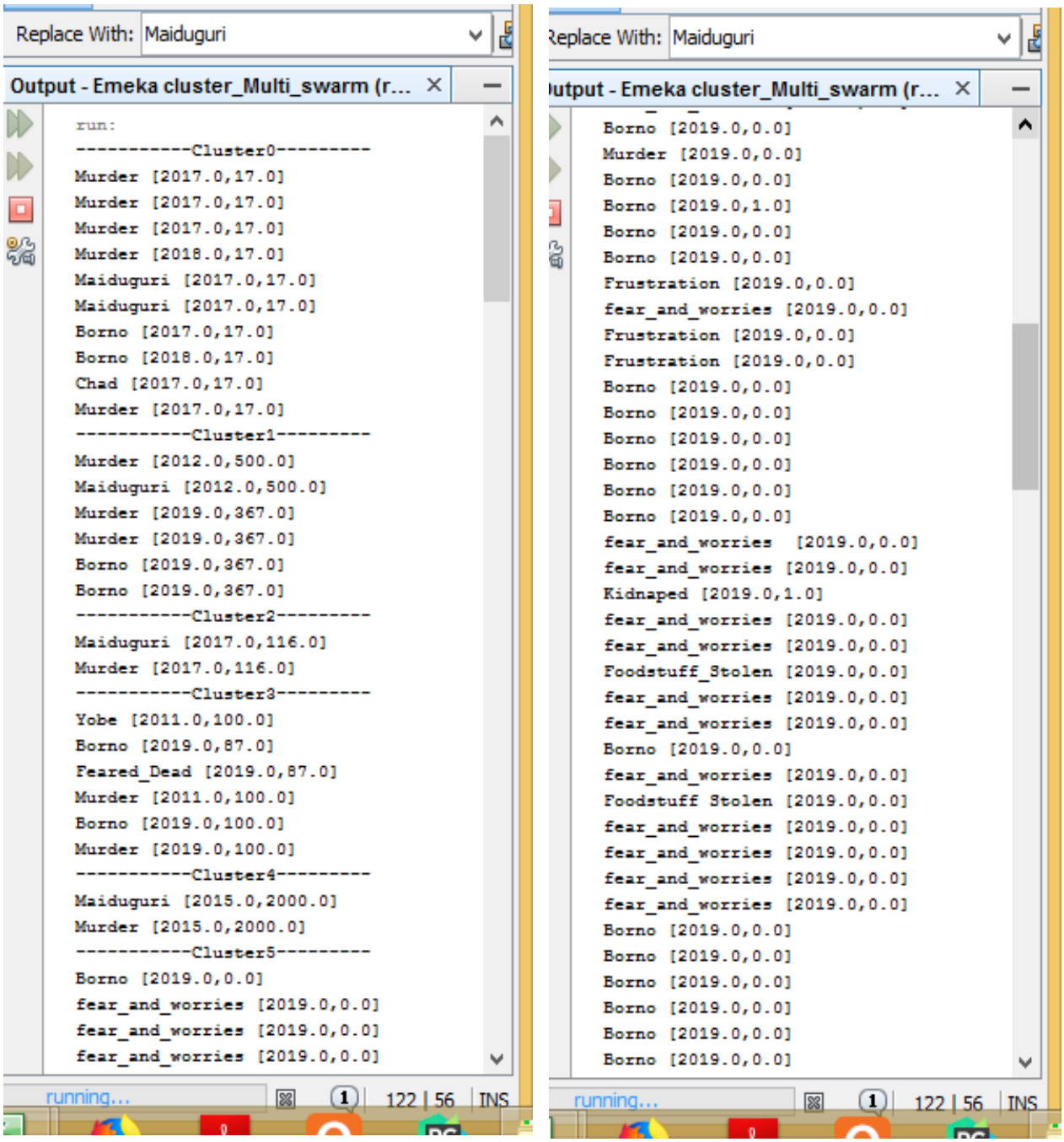

Figure 4.16: Clustered Output of Boko Haram Insurgency from cluster 0 to clusters 5

Figure 4.16 shows the clustered output of Boko Haram insurgency from cluster 0 to clusters 27. The output results of the proposed system gave the different clusters assigned to different group of similarities and dissimilarities. The percentage distribution of clustered result of Boko Haram area of attack is shown in Table 4.10.

Table 4.10: **Percentage distribution of clustered results of Boko Haram area of attack in the study areas.**

Clustered Area of Attack	Frequency	Percentages (%)
Borno	51	63.75
Abuja	1	1.25
Adamawa	1	1.25
Gombe	3	3.75
Kano	2	2.5
Kastina	2	2.5
Maiduguri	16	20
Yobe	4	5

Table 4.10 analyzed the percentage distribution of clustered results of Boko Haram area of attack in the study areas. It consists of three (3) columns. Column 1 shows the clusters of area of attacks such as Borno, Abuja, Adamawa, Gombe, Kano, Kastina, Maiduguri, and Yobe. Column 2 shows the clustered frequency of the areas of attack. Column 3 shows the percentage (%) distribution of the clustered areas. The result is represented graphically in Figure 4.17.

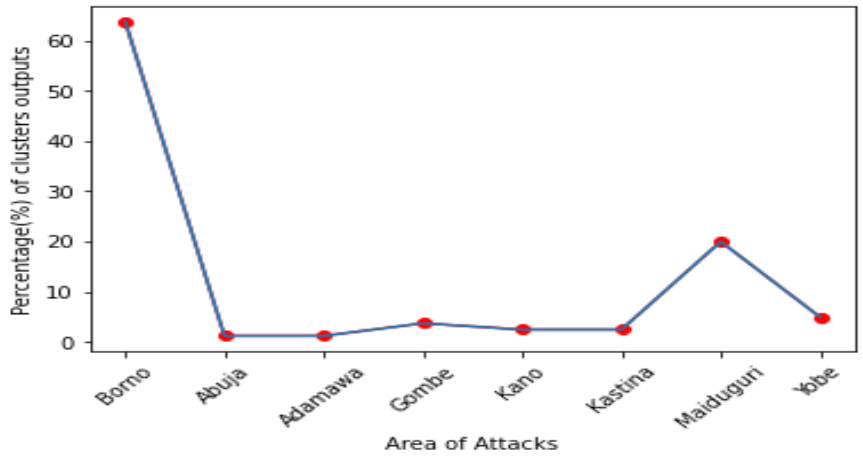

Figure 4.17: Graphical representation of the Percentage Distribution of clustered Boko Haram Attacks in Different Areas in Nigeria.

The percentage distribution of clustered result of death toll of Boko Haram attack is shown in Table

4.11: **Clustering Distribution for Year and Death Tolls of Boko Haram Attack**

Year	Death Toll	Death Toll (%)
2011	191	4.1
2012	733	15.6
2013	162	3.4
2014	283	6.0

2015	2009	42.6
2016	0	0.0
2017	133	2.8
2018	282	6.0
2019	919	19.5

Table 4.11 shows the death tolls of Boko Haram attack in different years. Column 1 shows the years of attack. Column 2 shows the clustered frequency of the number of death tolls and column three 3 shows the percentage (%) of the clustered death tolls. The result is represented graphically in Figure 4.18.

The Clustered distribution of death tolls of Boko Haram attacks in Nigeria is show in Figure 4.18.

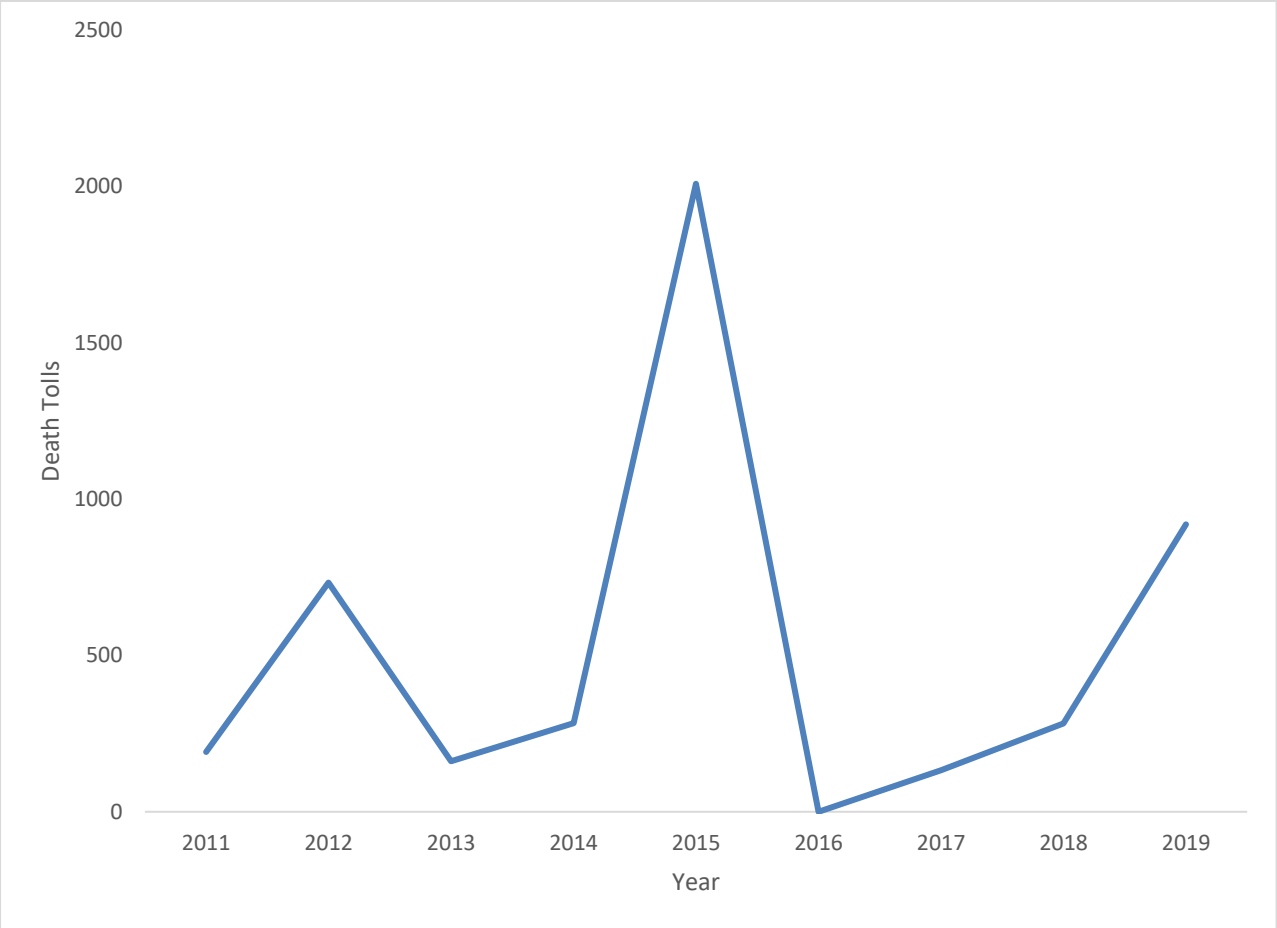

Figure 4.18a: Graphical representation of the clustered distribution of death tolls resulted from Boko Haram attack in Northern Nigeria.

The percentage of death tolls are represented in Figure 4.18b.

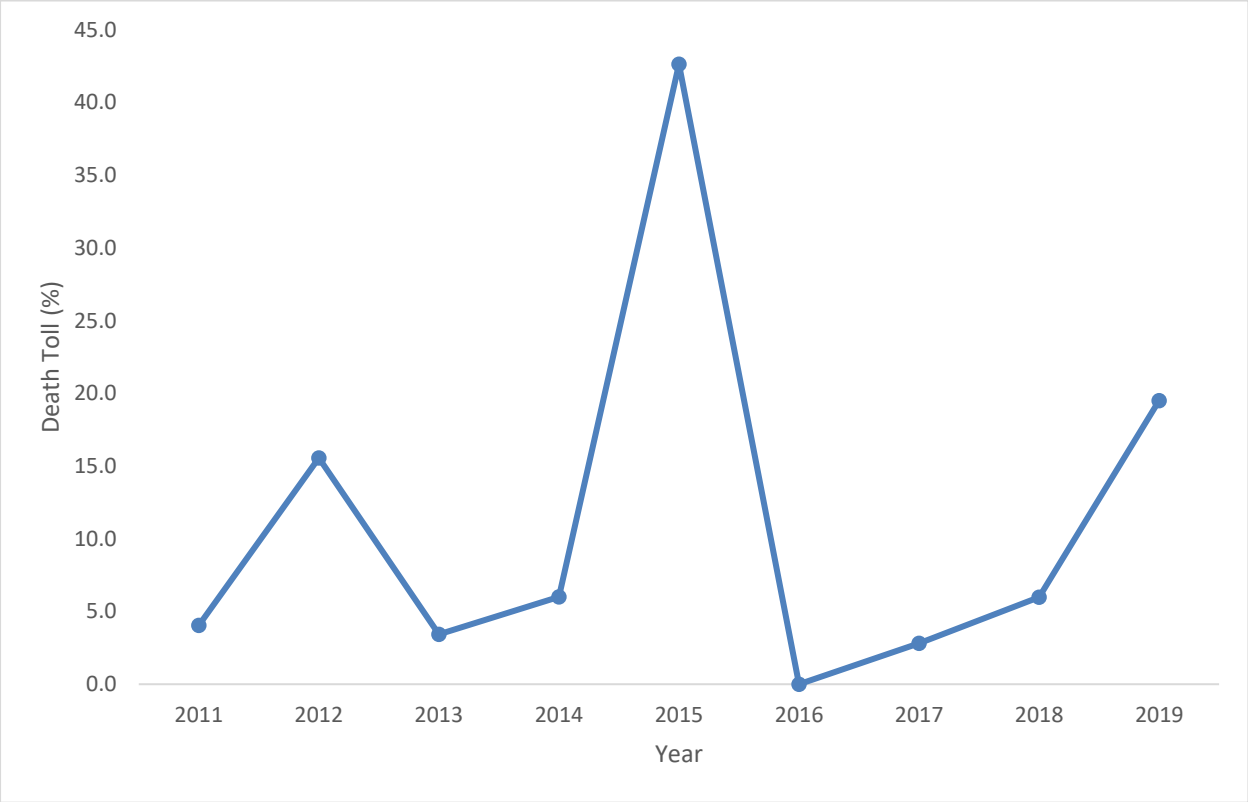

Figure 4.18b: Graphical representation of clustered distribution of year of attack and death tolls in percentage.

The monthly attacks of Boko Haram insurgency in northern Nigeria is represented in Figure 4.19.

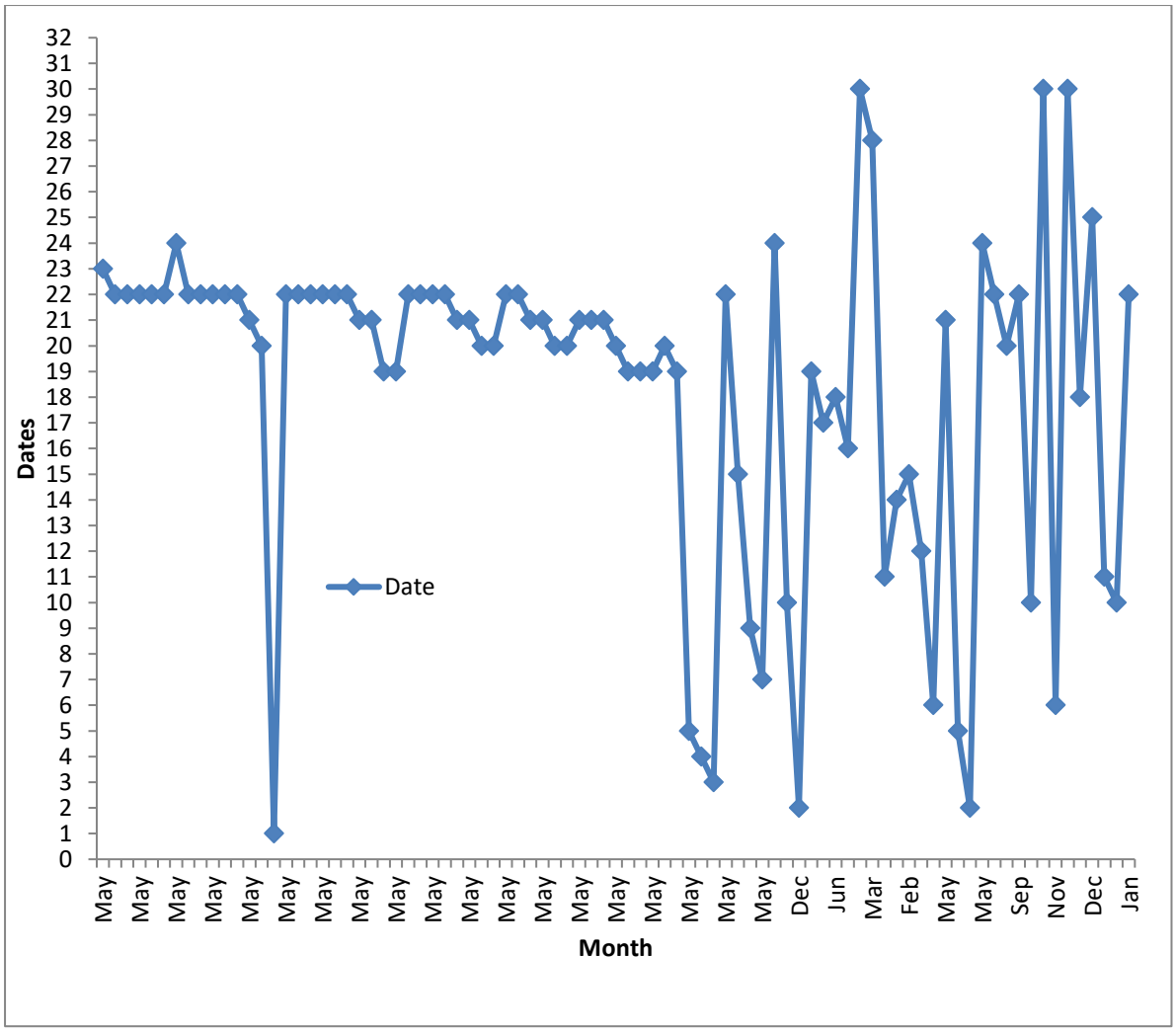

Figure 4.19: Graphical Representation of the clustered distribution of dates and months of Boko Haram attacks in Nigeria.

The clustered distribution of dates and years of Boko Haram attacks in Nigeria is shown in Figure 4.20.

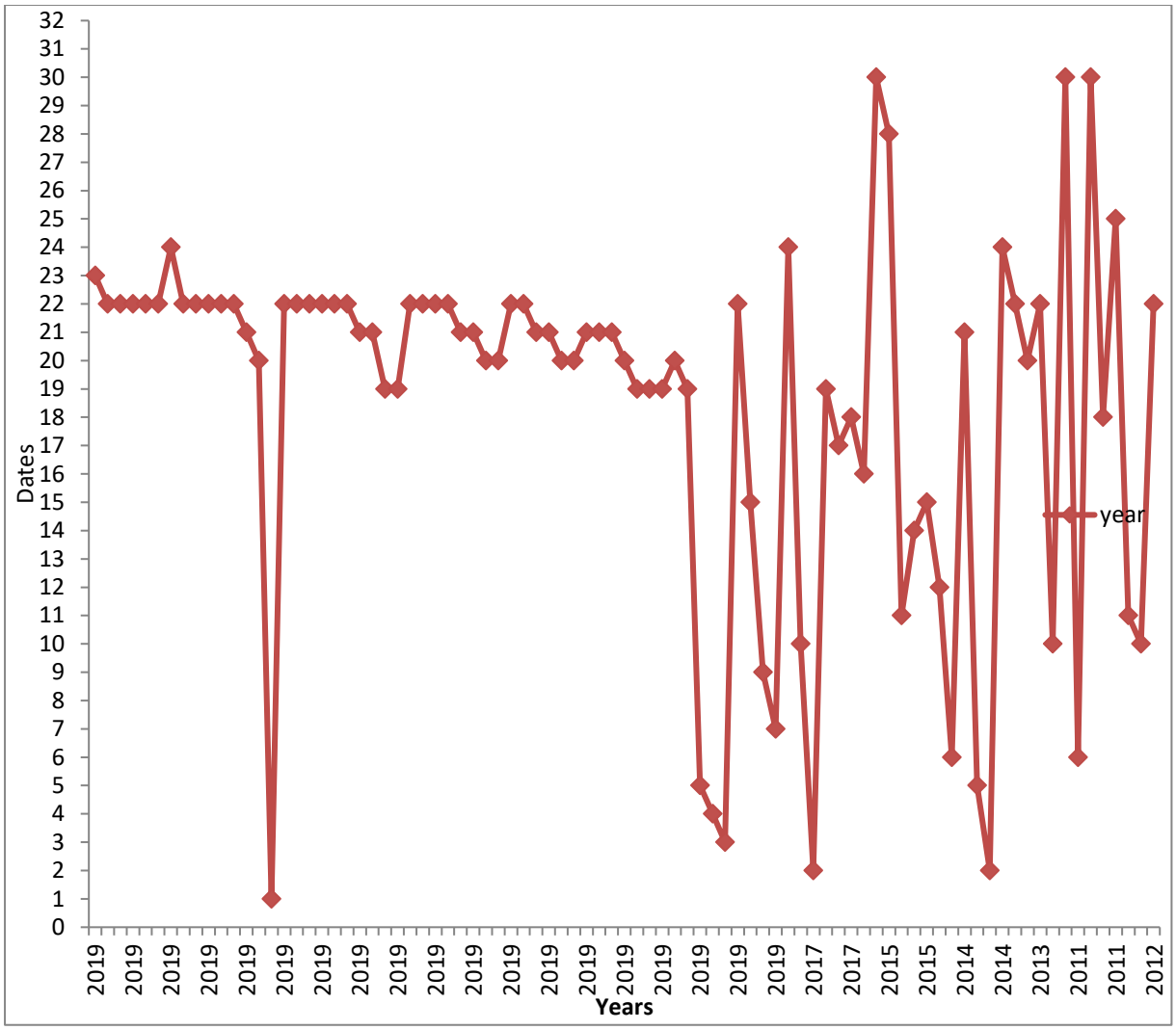

Figure 4.20: Graphical Representation of the clustered distribution of dates and years of Boko Haram attacks in Nigeria.

4.2 **Discussion**

In this work, the concept of unsupervised Machine Learning and Swarm Intelligence algorithms were used to develop an improved model called Dynamic-K-reference clustering algorithm for big data analytics. The Dynamic-K-reference clustering algorithm was developed to solve the problems and bridge the research gaps mentioned in the problem statement in chapter one and the limitations of the existing algorithms that was highlighted in chapter two. The developed hybrid model was implemented with Java programming language and MATLAB Editor, version R2018a for data extraction respectively. Jupyter notebook integrated with Python programming language were used for the visualization of the output clustered results.

The performances of the developed model were verified using real-world datasets. The datasets of Boko Haram insurgency attacks in Nigeria were scraped from twitter in social media to validate the accuracy of the proposed hybrid algorithm. 6050 tweeters comments and reports were extracted from the Twitter website. In other to validate Dynamic-K-references clustering algorithm its performance was compared with the existing algorithms on six datasets from the machine learning repository: Hepatitis, Australian Credit Approval, German Credit Data, Starlog Heart, Soybean, Yeast.

4.2.1 **Developed improved clustering algorithms for mining unstructured datasets**

First, the research work developing an improved clustering algorithm called K-reference algorithm using the concept of Reference Distance Weighted (RDW) of dissimilarity measure, Euclidean Distance (ED) measure and Chi-square Relative Frequency (CRF) of dissimilarity measure to solve the problems of the influence of outliers and heterogeneous nature of the big datasets on the existing partitioning clustering algorithms. A similar existing algorithm called MixK-meanFon clustering algorithm developed by (Onuodu *et al.*, 2014) that utilized the concept of relative cumulative frequency to improve the performance of K-means algorithm were used for comparison.

Two datasets Soybean and Yeast were analyzed to compare the accuracy measures and clustering error for Dynamic-K-reference and MixK-meansXFon clustering algorithms. The analytic results of Soybean and Yeast datasets for Dynamic-K-reference clustering algorithm produced clustering accuracy of: 0.998 and 0.973 while the MixK-meansXFon algorithm produced clustering accuracy of: 0.86 and 0.84. In addition, from the analytic results of soybean and yeast dataset, Dynamic-K-reference clustering algorithm produced clustering error of: 0.002 and 0.027 while MixK-meansXFon produced clustering error of: 0.14 and 0.16. The result of the experiment shows that Dynamic-K-reference clustering algorithm produced good clustering results for soybean and yeast dataset than MixK-meansXFon.

Considering the factors: outliers, heterogenous data types and voluminous nature of big dataset that affects the performance of partitioning clustering algorithms which causes them to converge

prematurely when creating clusters. The k-reference clustering algorithm were developed such that it does not depend absolutely on the mean of the datapoints to create clusters, but on percent which differ the characteristics of the datapoint's centroids. The mechanism of K-reference clustering algorithm does not depend on the absolute values of the differences between these values to obtain the centroid of the datapoints. Therefore, outlies does not have effect on its computational strength when creating clusters. While in the work of (Onuodu *et al, 2014)* the MixK-meansXFon algorithm that depends on Euclidean distance measure to create clusters from the dataset. This implies that outliers in the datasets does not influence the mechanism of K-reference clustering algorithm when calculating centroids to create clusters of datapoints in the large datasets unlike the MixK-meansXFon algorithm that converges prematurely due to influence of outliers.

The analytical results show that the Dynamic-K-reference clustering algorithm produced performance improvement of 13.8% and 13.7% respectively. Based on the comparative analysis, it is concluded that the Dynamic-K-reference clustering algorithm has high analytic accuracy than the other existing algorithms.

4.2.2 **Develop a hybridized algorithm based on the improved clustering algorithm and dynamic multi-swarm optimization algorithm for analysis of a big dataset**

Secondly a hybrid algorithm called Dynamic-K-reference clustering algorithm were developed in this work. It consists of Dynamic multi-swarm optimization algorithm and K-reference clustering algorithm. Another of its kind called Binary PSO based K-prototype clustering algorithm was developed by (Prabha et al., 2015). Four datasets: Hepatitis, Australian Credit Approval, German Credit Data, and Starlog Heart were analyzed to compare the performance of the Dynamic-K-reference and Binary PSO based K-prototype clustering algorithms. The experimental results of Hepatitis, Australian Credit Approval, German Credit Data, and Starlog Heart datasets for Dynamic-K-reference clustering algorithm produced clustering accuracy of: 0.9669, 0.9889, 0.9610, 0.9509 while PSO based K-prototype algorithm produced an accuracy of: 0.7521, 0.8229, 0.6261, and 0.8387

respectively. These results show that the Dynamic-K-reference clustering algorithm is highly proficient for clustering very large datasets than the PSO based K-prototype algorithm.

This achievement was as the results of the improve analytic mechanisms developed in Dynamic-K-reference clustering algorithm including: deploying multi-swarm search agents to explore the high dimensional datasets; divide by conquer approach of exploitation and exploration of the problem domain; regrouping the search agent after the exploitation of the assigned task; previous experiences gained by the particles in their former groups (subswarm) are share when regrouped. While in the case PSO based K-prototype algorithm (Prabha et al., 2015) that: deployed binary swarm search agents, managing between exploitation and exploration of the high dimensional datasets instead of balancing them in different phases; And it does not have the potential to regroup the search agents.

Furthermore, the analysis shows that Dynamic-K-reference clustering algorithm produced performance improvement of: 22.2%, 16.8%, 34.8% and 11.8% respectively.

4.2.3 **Scrape real-world dataset of the terrorist attack in Nigeria from social media for implementation and performance analysis of the Dynamic-K-reference Clustering Algorithm.**

The developed Dynamic-K-reference clustering algorithm were used to analyze the dataset of Boko Haram insurgence attacks in the Northern Nigeria. The attributes of the dataset including the area of attacks, period of attacks, death tolls, and attack strategies were used for the analysis for the period of 2008 to May 2019.

4.2.3.1 **Analysis of Boko Haram insurgency attacks in Different Areas in the Northern Nigeria**

The developed Dynamic-K-reference clustering algorithm were used for analysis of Boko Haram insurgency attacks in different areas in the Northern Nigeria. The results show that there were 63.75% attacks at Borno State and its capital Maiduguri has 20% attacks which gave a total of 83.75% attacks in Borno state. 1.25% attacks on Abuja, 1.25% attacks on Adamawa State, 3.75% attacks on Gombe State, 2.5% attacks on Kano State, 2.5% attacks on Kastina State, and 5% attacks on Yobe State respectively. The highest Boko Haram attacks was at Borno State followed by Yobe state and the others.

This finding is in-line with the results of the work of (Start, 2019) that shows that Terrorist violence remained heavily concentrated in certain locations and coincided with other types of political violence. More than half of all attacks took place in five countries: Afghanistan (18%), Iraq (14%), India (9%), Nigeria (7%), and the Philippines (6%). More than half of all deaths took place in two countries: Afghanistan (43%), and Nigeria (11%) (Start, 2019).

4.2.3.2 Dynamic-K-reference clustering algorithm analysis of Death Tolls of Boko Haram Attacks in the Northern Nigeria

The analytic results of Dynamic-K-reference clustering algorithm on the death tolls of Boko Haram attacks in the study Northern Nigeria produced clusters of attacks in different years from the period of 2008 to 2019. Thus, no information on the attacks was gotten from 2008 to 2010 in Twitter social media. The available information where from 2011 to 2019 when the analysis was carried out. The output of the clustered results produced by Dynamic-K-reference algorithm with the dataset was analyzed.

The output clustered results of death tolls at different years produced 4.1% on 2011, 15.6% on 2012, 3.4% on 2013, 6.0% on 2014, 42.6% on 2015, 0.0% on 2016, 2.8% on 2017, 6.0% on 2018 and 19.5% on 2019 respectively. The year 2015 has the highest death toll. The results show constant attacks of Boko Haram insurgency in the study area, which had led to millions of people currently displaced and killed. It was also observed that more than 2,000 people were killed from 2010 to 2015 during the Boko Haram Insurgency attacks in Northern Nigeria. The number of attacks and killings reduced in 2016 and early 2017. From 2018 to date, there have been constant insurgency attacks in Northern Nigeria.

This finding is in-line with that of (Start, 2019) that analyzed the terrorist attacks and total deaths in Nigeria, by month, from 2012 to 2019. The results show that the number of terrorist attacks and the number of people killed in terrorist attacks in Nigeria increased in 2018 for the first time in four years, following a steady decline from extremely high levels of terrorist violence in 2014.

Moreover, in 2018, there were 645 terrorist attacks in Nigeria, a 33% increase from 2017. There were 2,574 victims and perpetrators killed in these attacks, a 43% increase from 2017. However, most of this increase was driven by victim deaths, which increased 63% from 2017. Furthermore, in 2018, 291 perpetrators were killed in attacks in Nigeria, down from 422 the previous year. Unlike Afghanistan and Iraq, where terrorist conflicts have been dominated by single perpetrator organizations, there are several regional conflicts in Nigeria (Start, 2019).

4.2.3.3 Dynamic-K-reference Clustering Algorithm Analysis of Dates and Months of Boko Haram Attacks in Nigeria

Dynamic-K-reference Clustering Algorithm produced the output clustered distribution of dates and months of Boko Haram attacks in Nigeria. From the results, it was observed that in May 2019 there have been constant attacks of Boko Haram insurgency in Nigeria. There was an attack on the 23rd of May 2019, and more than eight attacks on the 22nd of May, 2019 at different time intervals of that date. Furthermore, there have been various attacks recorded on the 21st, 20th, and 19th of May 2019. On 19th May, 2019, there were five (5) attacks at different time intervals. Also, on 20th May, 2019 there were seven (7) attacks and on 21th of May, 2019, there were ten (10) attacks at different time intervals of that date. This implies that Northern Nigeria borders are extremely porous for insurgency attacks; there was little or no resistance of the military's defense before the attacks took place.

This finding is in line with that of (Start, 2019) on the coordinated terrorist attacks across multiple locations in Nigeria. The results of the work (Start, 2019) show that Nigeria continued to experience many attacks that were coordinated across multiple locations. In 2018, more than one-third (35%) of all terrorist attacks in Nigeria were part of coordinated events, compared to 17% worldwide. While this tactic was previously a hallmark of Boko Haram, in 2018 more than two-thirds (72%) of the coordinated attacks in Nigeria were carried out by Fulani extremists (Start, 2019).

The real-world dataset of Boko Haram attack dataset was used to verify the performance of the clustering accuracy and error of the developed Dynamic-K-reference clustering algorithm. The

Dynamic-K-reference clustering algorithm was resourceful enough to provide clustering accuracy of 0.9820 and clustering error of 0.0018 from the analysis of Boko Haram Insurgency attacks dataset.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

The results of this work have shown that the integration of Swarm Intelligent and Unsupervised Machine Learning Algorithms provides an efficient, robust mechanisms to handle the analytical challenges posed by complex characteristics of big data on traditional algorithms. This work also has proved that the intelligent characteristic of Dynamic Multi-swarm agents and robust analytic behavior of Unsupervised Machine Learning Algorithms are proficient to unveil the hidden patterns of the unstructured dataset of the big data stream. The proposed hybridized clustering algorithm clustered the unstructured datasets of big data analysis efficiently. It possesses the potential to create clusters with a similar internal structure of unstructured (mixed) datasets and produce accurate clustering interpretation for the decision support system.

Considering the analytical challenges and unique characteristics of big data analytical processes, the Dynamic-K-reference clustering algorithm was used to unveil the hidden patterns of the mixed larger datasets of insurgency attacks in Nigeria. The Dynamic-K-reference clustering algorithm was resourceful enough to provide clustering accuracy of 0.9820 and clustering error of 0.0018 from the analysis of Boko Haram Insurgency attacks dataset. The clustered results identified various strategies of Boko Haram attacks in Northern Nigeria. There were constant attacks of the Boko haram insurgency in northern Nigeria. The continued attacks led to millions of people currently displaced and killed in Northern Nigeria most especially in Borno, State.

The repeated Boko Haram insurgency attacks in Northern Nigeria at different time intervals on the same day indicate that the military defense frontline is weak. The security lapses were due to the insufficient commitment of soldiers at the frontline. The setback in operations of soldiers in the ongoing counter-insurgency and counter-terrorism operations in the north-east was also confirmed by the Chief of Army Staff, Lt.-Gen. Tukur Buratai's reports, during the military transformational

workshop in Abuja on June 18th, 2019. The security lapses were because the security agencies are not adequately equipped to counter attacks at the frontline. The results of this work will enable Nigeria's military intelligence to devise efficient strategies to strengthen security control of insurgency attacks.

5.2 Recommendations

This work recommends intelligent automated systems like crime-mapping platforms, gunshot-detection systems, video analytics, smart lights, and robotic detection systems to be implemented at the border frontline for efficient monitoring and communication of attacks. This will eliminate the setbacks due to the insufficient willingness of soldiers to perform assigned tasks. Future applications should focus on Boko Haram member's profiles within the society to identify their hidden members in society.

Furthermore, Other complex datasets should be used to validate the model to ascertain its converging diversity and accuracy. It would be worthwhile to develop self-adaptation mechanisms for the number of swarms and the exclusion radius particles in the whole space. Other optimization Algorithm should be used to solve the limitation of the traditional k-prototype algorithm selection of the value of *k*, which is the number of desired clusters, regardless of the distribution of the data points. Also, future work should design a big data analytic services with the improved Dynamic-K-reference clustering algorithm and other extension of its kinds using service-oriented architecture methodology for real time analysis and prediction.

5.3 **Contributions to Knowledge**

1. The study developed an improved clustering algorithm called "K-reference clustering algorithm" for clustering unstructured mixed large datasets.
2. It developed a hybrid analytic model called Dynamic-K-reference Clustering Algorithm that is highly proficient for big data analysis.

3. The results of the analysis of Boko Haram Insurgence in northern Nigeria unveil numerous mechanisms to mitigate ongoing insurgency and terrorists in Nigeria.

REFERENCES

Abdelhalim, M. B. & Habib, S. E. D. (2009). Particle swarm optimization for HW/SW partitioning. *Intech Open*, 49-76. https://doi.org/*10.5772/6740*

Abiodum, O. I., Janta, A., Omolara, A. E. Singh, M. M., Abukakar, L. Z., & Umar, A. M. (2018). Big data: an approach for detecting terrorist activities with people's profiling. *Proceedings of the International Multi-Conference of Engineering and Computer Science, IMECS*, 1-7.

Acheampong, A. F. (2018). Big data, machine learning and the block chain technology: an overview. *International Journal of Computer Application, 180*(34), 0975-8887. https://doi.org/10.5120/ijca2018916674

Ahirwar, G. (2014). A novel k-means clustering algorithm for large datasets based on divide and conquer techniques. *International Journal of Computer Science & Information Technologies*, *5*(1), 301-305.

Ahmad, A & Dey, L. (2007). A K-means clustering algorithm for mixed numerical and categorical data. *Data & Knowledge Engineering*, *63*(2), 503-527.

Ahmed, H. & Glasgow, J. (2012). Swarm intelligence: concepts, models and applications. *Queen's University, School of Computing Technical Reports.* 1-50.

Aishwaray, M. S., Bhargavi, H. & Jyothi, N. (2017). Handing big data analytic using swarm intelligent. *International Journal of Scientific Development and Research (IJDR), 2*(6), 271-275.

Ali, S. & Site, S. (2019). A simplified analytical model toward big data analysis using ridge regression method. *International Journal of Computer Applications, 180*(258), 41-48.

Ankam, V. (2016). *Big data analytics* (1st ed.). Packet Publishing.

Arora, P., Deepali, & Varshney, S. (2016). Analysis of k-means and k-medoids algorithm for big data. *Procedia Computer Science*, *78*, 507–512. https://doi:10.1016/j.procs.2016.02.095

Asadi, S., Subba, R. C.D.V, Kisshore, C. & Raju, S. (2012). *Clustering the mixed numerical and categorical datasets using similarity weight and filter method. International journal of Computer Science Information Technology and Management*, *5*(1), 1-2.

Backwell, T. & Branke, J. (2004). Multi-swarm Optimization in Dynamic Environments. S*pringer-Verlag B*, 489-500. https://doi.org/10.1007/978-3-540-49774-5_2

Balazinska, M. & Dan, S. (2016). Parallel Databases and MapReduce. Spring, 1-28. Retrieved on 25th October, 2019 from http://pages.cs.wisc.edu/~paris/cs838-s16/lecture-notes/lecture-parallel.pdf.

Beni, G. & Wang, J. (1989). Swarm intelligence in cellular robotic systems. *In NATO Advanced Workshop on Robots and Biological Systems, Il Ciocco, Tuscany, Italy, Springer-Verlag Berlin Heidelberg.* 703-712. https://doi.org/10.1007/978-3-642-58069-7_38

Betser, J. & Belanger, D. (2015). Architecting the enterprise with big data analytics. *In Big Data and Business Analytics, Boca Raton, FL, CRC Press*. 1-20.

Bhandari, R. & Schultz, K. (2017). Violence flares as Nepal heads to landmark elections. The New York Times. Retrieved on 17th December, 2020 from https://www.nytimes.com/2017/11/25/world/asia/nepal-election-violence.html.

Bhattacharya, A. Kar, P. & Pal, M. (2009). On low distortion embeddings of statistical distance measures into low dimensional spaces. *Springer Journal*, 164-172.

Borne, K. (2014). Top 10 Big Data Challenges – A serious look at 10 big data v's. Retrieved on 15th April 2019 from https://www.mapr.com/blog/top-10-big-data-challenges-%E2%80%93-serious-look-10-big-data-v%E2%80%99s.

Bu, Y. Borkar, V. R., Carey, M. J., Rosen, J. Polyzotis, N. & Condie, T. (2012). Scaling data log for machine learning on big data. *ArXiv Journal*, 1-14.

Cao, J., Cui, H., Shi, H., & Jiao, L. (2016). Big data: a parallel particle swarm optimization-back-propagation neural network algorithm based on map-reduction. *PLOS ONE, 11*(6), 1-17. https://doi.org/10.1371/journal.pone.0157551

Chaturvedi, A., Green, P.E., & Caroll, J.D. (2001). K-modes clustering. *Journal of Classification, 18*(1), 35–55. https://doi.org/10.1007/s00357-001-0004-3

Chen, S & Montgomery, J. (2011). Selection strategies for initial positions and initial velocities in multi–optima particle swarms. *In Proceedings of the Genetic and Evolutionary Computation Conference.* 53–60. https://doi.org/10.1145/2001576.2001585

Chen, S. (2009). Locust Swarms – A new multi-optima search technique. *In Proceedings of the IEEE Congress on Evolutionary Computation*. 1745–1752. https://doi.org/10.1109/CEC.2009.4983152

Chiang, M. C., Tsai, C. W., & Yang, C. S. (2011). A time-efficient pattern reduction algorithm for k-means clustering. *Inform Sci, 181*(4):716–31. https://doi.org/10.1145/2001576.2001585

Chiu, T., Fang, D., Chen, J., & Wang, Y. (2001). A robust and scalable clustering algorithm for mixed type attributes in large database environment. *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '01,* 263-268. https://doi.org/10.1145/502512.502549

Chouhan, R & Chauhan, A., (2014). An ameliorated partitioning clustering algorithm for large *datasets*. *International Journal of Advanced Research in Computer and Communication Engineering*, *3*(7), 7617-7621.

Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, *6*(1), 58-73. https://doi.org/10.1109/4235.985692

Coello, C & Lechuga, M. (2002). A proposal for multi objective particle swarm optimization. *Proceedings of the 2002 Congress on Evolutionary Computation*, 1051-1056. https://doi.org/10.1109/cec.2002.1004388

Colorni, A., M. Dorigo, V. Maniezzo, & M. Trubian. (1994). Ant System for Job-shop Scheduling. *Belgian Journal of Operations Research, Statistics and Computer Science*, *34*(1):39-53. https://doi.org/10.4236/jsea.2014.77053

Cormode, G., & Duffield, N. (2014). Sampling for big data: a tutorial. *In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1975–1975. https://doi.org/10.1145/2623330.2630811

Couzin, I. D., Krause, J., James, R., Ruxton, G. D., & Franks, N. R. (2002). Collective memory and spatial sorting in animal groups. *Journal of Theoretical Biology*, *218*(1), 1–11. https://doi.org/10.1006/jtbi.2002.3065

Cui, X., Gao, J. & Potok T. E. A (2006). Flocking based algorithm for document clustering analysis. *Journal for System Architecture*, *52*(89), 505–515. https://doi.org/10.1016/j.sysarc.2006.02.003

Dawelbeit, O. & McCrindle, R. (2014). A novel cloud based elastic framework for big data preprocessing. *Conference: 2014 6th Computer Science and Electronic Engineering Conference (CEEC)*, 1-6. https://doi.org/10.1109/CEEC.2014.6958549.

Delena, V. and Demirkanb, H. (2013). Data, information and analytics as services, Decision Support Systems. *Elsevier B.V*, *55*(1), 359–363. https://doi.org/10.1016/j.dss.2012.05.044

Demchenko, Y., Laat, C. and Membery, P. (2014). Defining architecture components of the big data ecosystem. *In: Proceeding of the International Conference on Collaboration Technologies and Systems*, 104-112. https://doi.org/10.1109/CTS.2014.6867550

Demirkan, H. and Delen, D. (2013). Leveraging the capabilities of service-oriented decision support systems: putting analytics and big data in cloud. Decision Support Syst. *Published by Elsevier B.V*, *55*(1), 412–421. https://doi.org/10.1016/j.dss.2012.05.048

Deneubourg, J.L Goss, S., Franks, N., Sendova-Franks, A., Detrainl, C. & Chretien, L. (1990). The dynamics of collective sorting robot-like ants and ant-like robots. *In: Proceedings of the International Conference on Simulation of Adaptive Behavior on from Animals to Animats*, 356-363.

Dewangan, O. & Ambhaikar, A. (2010). Extended fuzzy c-means clustering algorithm in segmentation of noisy images Omprakash. *International Journal of Science and Research (IJSR)*, *1*(2), 16-19.

Dorigo, M. & Stützle, T. (2004). *Ant Colony Optimization* (1th ed.). MIT Press.

Drabas, T., and Lee, D. (2017). *Learning PySpark* (1th ed.). *Packt Publishing*. Retrieved on 23th October 2019 from https://www.packtpub.com/product/learning-pyspark/9781786463708

Dua, R. Ghotra, M. S., and Pentreath, N. (2017). *Machine learning with spark* (2nd ed.). Packt Publishing.

Dubey, S.R., Dixit, P., Singh, N. and Gupta, P. (2013). Infected fruit part detection using k-means clustering segmentation technique. *International Journal of Artificial Intelligence and Interactive Multimedia*, *2(2)*, 1-8. https://doi.org/10.9781/ijimai.2013.229

Egbe, R. (2019). Chibok abduction: BBOG marks 2,000 days, demands girls' release. *The Nation*, Retrieved on 15th March, 2020 from from https://thenationonlineng.net/chibok-abduction-bbog-marks-2000-days-demands-girls-release/

Einstein, A. (2018). Addressing the cybersecurity skills gap through cooperation, education and emerging technologies. SNAPSHOT, 1-7. Retrieved on 25th November, 2019 from https://cybertechaccord.org/uploads/prod/2019/01/TechAccordWP-AddressingCyberSkillsGap_Jan2019.pdf.

Elena, G., Florina, C., and Anca, M. (2012). Perspectives on big data and big data analytics. *Database Systems Journal*, *3*(4), 3-14.

Elkan, C. (2003). Using the triangle inequality to accelerate-means. *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, 1-8.

Engelbrecht, A. P. (2002). *Computational intelligence: an introduction* (2th ed.). John Wiley & Sons.

Essa, Y. M., Attiya, G., & El-Sayed, A. (2013). Mobile agent based new framework for improving big data analysis. *In: Proceedings of the International Conference on Cloud Computing and Big Data*, 381–386. https://doi.org/10.1109/CLOUDCOM-ASA.2013.75

Famili, A., Shen, W. M., Weber, R. & Simoudis, E. (1997). Data preprocessing and intelligent data analysis. I*ntelligence Data Analysis Journal*, *1*(1), 1–21. https://doi.org/10.1016/S1088-467X(98)00007-9

Fan, W. & Bifet, A. (2013). Mining big data: current status, and forecast to the future. *ACM SIGKDD Explorations Newsletter*, *14*(2), 1–5. https://doi.org/10.1145/2481244.2481246

Feldman, D, Schmidt, M, & Sohler, C. (2013). Turning big data into tiny data: constant-size coresets for k-means, PCA and projective clustering. *In: Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 1434–1453. https://doi.org/10.1137/1.9781611973105.103

Fisher, D., DeLine, R., Czerwinski, M., & Drucker, S. (2012). Interactions with big data analytics. *ACM, Inc*, *19*(3):50–9.

Gannon, K. (2018). IS deadly new front in Pakistan's decades-old terror war. *Associated Press*. Retrieved on 17th December, 2020 from https://www.apnews.com/e5c3dc41cb7b4a058333509a342e34a3

Garbade, M. J. (2018). Understanding k-means clustering in machine learning medium. Retrieved 31th October, 2019 from https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1.

Gazi, V. & Passino, K. M. (2011). *Swarm stability and optimization* (1st ed.). *Springer*.

Grosan. C., Abraham, A & Monica, C. (2006). Swarm Intelligence in Data Mining, Studies in Computational Intelligence, *Springer*, *34*, 1-16.

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data – SIGMOD*, 1-12. https://doi.org/10.1145/342009.335372

Hari, S. (2019) Hadoop vs spark: head to head comparison choosing right framework. Retrieved on 20th April, 2020 from Hackr.io, https://hackr.io/blog/hadoop-vs-spark

Hasan, S. Shamsuddin, S. & Lope, N. (2013). Soft computing methods for big data problems. *In: Proceedings of the Symposium on GPU Computing and Applications*, 235-247. https://doi.org/10.1007/978-981-287-134-3_15

Hendtlass, T. (2005). A multi-optima particle swarm algorithm. *In Proceedings IEEE Congress on Evolutionary Computation*, 727–734. https://doi.org/10.1109/cec.2005.1554755

Huai, Y., Lee, R., Zhang, S., Xia, C. H., & Zhang, X. (2011). DOT: a matrix model for analyzing, optimizing and deploying software for big data analytics in distributed systems. *In: Proceedings of the ACM Symposium on Cloud Computing*, *4*, 1-14. https://doi.org/10.1145/2038916.2038920

Huang, Z. (1997). A fast clustering algorithm to cluster very large categorical data sets in data mining. *Research Issues on Data Mining & Knowledge Discovery*, 1-8.

Huang, Z. (1999). Extensions to the k-means algorithm for clustering large cybernetics 28C. *Data Mining and Knowledge Discovery*, 219-230. https://doi.org/10.1023/A:1009769707641

Huang. C, Smith, P. & Sun, Z. (2014). Secure network solutions for enterprise cloud services. *In: IRMA, Cloud Technology: Concepts, Methodologies, Tools, and Applications, IGI Global*, 68, 1464-1486. https://doi.org/10.4018/978-1-4666-6539-2.ch068

Ignacio, F. G. (2018). Big data processing with apache spark: efficiently tackle large datasets and big data analysis with spark and python. *Packt. EBSCOhost*, Retrieved on 25th October 2020 from search.ebscohost.com/login.aspxdirect=true&db=nlebk&AN=1513368&site=eho st-live.

Imran, M. & Rao, V. S. (2018). A novel technique on class imbalance big using analogous under sampling approach. *International Journal of Computer Application*, *179*(33), 18-21. https://doi.org/10.5120/ijca2018916743

Ishwarappa J., & Anuradha, J. (2015). A brief introduction on big data 5vs characteristics and hadoop technology. *Elsevier Journal*, 319–324.

Jain, A. K. & Dubes, R. C. (2009). Algorith ms for clustering data, new jersey: *Prentice-Hall*, 71–99.

Jain, A. K., Murty, M. N, & Flynn, P. J. (2011). Data clustering: a review. *IEEE Computer Society press*, *31*(3), 264–323.

Jain, M. and Verma, C. (2014). Adapting *k-means for clustering in big data. International Journal of Computer Applications*, *101*(1), 0975 – 8887.

Jones, M., Pierce J. & Ward, D. (2007). Avian vision: a review of form and function with special consideration to birds of prey. *Journal of Exotic Pet Medicine*, *16*(2), 69-87. https://doi.org/10.1053/j.jepm.2007.03.012

Jun, S. W., Fleming, K., Adler, M., & Emer, J. S. (2012). Zip-io: architecture for application-specific compression of big data. *In: Proceedings of the IEEE International Conference on Field-Programmable Technology*, 343–351.

Kanjanawattana, S. (2012), An extended k-means++ with mixed attributes for outlier's detection. *Latest Advances in Information Science and Applications*, 12-16.

Karaboga, D. & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization, Springer Netherlands, 39*(3), 459-471.

Karie, N. M., Kenande, V. R. & Venter, H.S. (2019). Proposed diverging deep learning computing techniques into cyber forensics. *Forensic Science International: Synergy, 1*, 61-67. https://doi.org/10.1016/j.fsisyn.2019.03.006

Kaur, M. & Kaur, U. (2013). A survey on clustering principles with k-means clustering algorithm using different methods in detail. *International Journal of Computer Science and Mobile Computing (IJCSMC), 2*(5), 327-331.

Kaya, M., & Alhajj, R. (2005). Genetic algorithm-based framework for mining fuzzy association rules. Fuzzy Sets and Systems. *Elsevier, B.V. 152*(3), 587–601. https://doi.org/10.1016/j.fss.2004.09.014

Kennedy, J and Eberhart, R. (1995). A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 39–43.

Kennedy, J. and Eberhart, R. C. (1995). Particle swarm optimization. *In proceedings of IEEE International Conference on Neural Networks, Perth, Australia*, 1942-1948.

Kennedy, J. Eberhart, R. C. & Shi, Y. (2001). *Swarm Intelligence* (1th ed.). The Morgan Kaufmann Series in Evolutionary Computation.

Kolajo, T. & Daramola, O. (2017). Leveraging big data to combat terrorism in developing countries. Conference on Information Communication Technology and Society (ICTAS). 1-6. https://doi.org/10.1109/ICTAS.2017.7920662

Kolhe, S. R. & Zinjore, B.S. (2010). Clustering iris data using supervised and unsupervised learning. International Journal of Computer Science and Application, 135-139.

Kranthi, K. B. & Babu, A. V. (2014). A comparative study of issues in big data clustering algorithm with constraint based genetic algorithm for associative clustering. *International Journal of Innovative Research in Computer and Communication Engineering*, *2*(8), 5423-5432.

Krause, J. & Ruxton, G. D. (2002). Living in Groups (1th Edition). *Oxford University Press, Oxford, UK.*

Krishna, K, & Murty, M. N. (1999). Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, *29*(2), 433-439. https://doi.org/10.1109/3477.764879

Ku-Mahamud, K.R. (2013). Big data clustering using grid computing and ant-based algorithm. *In: Proceedings of the International Conference on Computing and Informatics. 6*–14.

Kumar, B. (2015). An encyclopedic overview of 'big data' analytics. *International Journal of Applied Engineering Research*, *10*(3), 5681-5705.

Kumar, S. & Singh, M. (2019). Clustering Techniques for Efficient Clustering of Big Data in Hadoop Ecosystem. *Big Data Mining and Analytics*, *2*(4), 240-247. https://doi.org/10.26599/BDMA.2018.9020037

Lakshmi K. Karthikeyani Visalakshi N., Shanthi S. & Parvathavarthini S. (2018). Clustering Mixed Datasets Using K-Prototype Algorithm Based on Crow-Search Optimization. *ResearchGate*. 1-20. https://doi.org/10.1007/s12046-018-0962-3

Lakshmi K. Karthikeyani Visalakshi N., Shanthi S. & Parvathavarthini S. (2018). Clustering categorical data using k-modes based on cuckoo search optimization algorithm. *ICTACT Journal on soft computing*, *8*(1), 1561-1566. https://doi.org/10.21917/ijsc.2017.0218

Laurila, J.K, Gatica-Perez, D., Aad, I., Blom, J., Bornet, O., Do, T., Dousse, O., Eberle, J. & Miettinen, M. (2012). The mobile data challenge: big data for mobile computing research. *In: Proceedings of the Mobile Data Challenge by Nokia Workshop*, 1–8.

Lazinica, A. (2009). *Particle swarm optimization* (1st ed.). In-Tech. Vienna, Austria. https://doi.org/10.5772/109

Lee, I., & Yang, J. (2009). Common clustering algorithms comprehensive chemometrics in university of western sydney. *Elsevier B.V. A*, 2(27), 577-618. https://doi.org/10.1016/b978-044452701-1.00064-8

Lee, J., Hong, S., & Lee, J. H. (2013). An efficient prediction for heavy rain from big weather data using genetic algorithm. *In: Proceedings of the International Conference on Ubiquitous Information Management and Communication*, *25*(7), 1–25.

Leung, C.S., MacKinnon, R. & Jiang, F. (2014). Reducing the search space for big data mining for interesting patterns from uncertain data. *Proceedings of the IEEE International Congress on Big Data*, 315-322. https://doi.org/10.1109/bigdata.congress.2014.53

Lin, M. Y, Lee, P. Y. & Hsueh, S. C. (2012). A priori-based frequency item set mining algorithms on map reduce. *In: Proceeding of the International Conference on Ubiquitous Information Management and Communication, 76,* 1-8. https://doi.org/10.1145/2184751.2184842

Liviu, O. & Mafteiu, S. (2013). A new dissimilarity measure between feature-vectors. *International Journal of Computer Applications*, *64*(17), 39-44.

Loshin, D. (2013). Developing a strategy for integrating big data analytics into the enterprise, in big data analytics. Journal *Elsevier*, 29-37. https://doi.org/10.1016/B978-0-12-417319-4.00004-1

Ma, C., Zhang, H. H. & Wang, X., (2014). Machine learning for big data analytics in plants. *Elsevier* Journal, *19*(12), 798 – 808. https://doi.org/10.1016/j.tplants.2014.08.004

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observation. *In Proceedings of the Berkeley Symposium on Mathematical Statistics and Probability*, *1*, 281–297.

Manjinder, K., Navjot, K., & Singh, H. (2014). Adaptive k-means clustering techniques for data clustering. *International Journal of Innovative Research in Science, Engineering and Technology*, *3*(9). 15851-15856. https://doi.org/10.15680/IJIRSET.2014.0309009

Marcu, D., Danubianu, M. & Pentiuc, G. (2019). Big data technology – valuable tool for healthcare research. *IJCSNS International Journal of Computer Science and Network Security*, *19*(4), 116-122.

Marzieh, H and Ali, H. A. & Seyed, J. M. A. (2015). A new method of fuzzy clustering by using the combination of the firefly algorithm and the particle swarm optimization algorithm. *WALIA Journal*, *31*(3), 245-252.

Maya, M. K. (2019). Application of machine learning and deep learning in cybercrime prevention- a study. national conference on research trend in big data and intelligent computing organized by department of computer science. *International Journal of Trend in Research and Development*,1-4.

Mayer-Schoenberger, V. & Cukier, K. (2013). Big data: A revolution that will transform how we live, work, and think. *Houghton Mifflin*, *179*(9), 1143–1144. https://doi.org/10.1093/aje/kwu085

McAfee, A. & Brynjolfsson, E. (2012). Big data: The management revolution, Harvard business review. Retrieved on 30th October, 2019 from https://hbr.org/2012/10/big-data-the-management-revolution#.

Mehdi, M., & Farshid, T. (2018). Big Data analytics in oil and gas industry: An emerging trend. Elsevier B. V. Journal, 1-9. https://doi.org/10.1016/j.petlm.2018.11.001

Miles, T. (2019). More than 500 Banunu killed in attacks by rival ethnic group in DR Congo: U.N. investigation. Reuters. Retrieved on 17th December, 2020 https://www.reuters.com/article/us-congo-violence-un/more-than-500-banunu-killed-in-attacks-by-rival-ethnic-group-in-dr-congo-u-n-investigation-idUSKBN1QT148.

Miller, E. (2016). Patterns of islamic state-related terrorism. Start background report. Retrieved on 17th December, 2020 from https://www.start.umd.edu/pubs/START_IslamicStateTerrorismPatterns_BackgroundReport_Aug2016.pdf

Minelli, M., Chambers, M. & Dhiraj, A. (2013). Big data, big analytics: emerging business intelligence and analytic trends for today's businesses. *Wiley & Sons*. Retrieve on 25th April,

2020 from https://www.amazon.com/Big-Data-Analytics-Intelligence-Businesses-ebook/dp/B00AUJ6TV0.

Ming-Yi, S., Jheng, J. & Lai, L (2010). A two-step method for clustering mixed categorical and numeric data. *Tamkang Journal of Science and Engineering*, *13*(1), 11-19.

Mitra, S., Pal, S. K., & Mitra, P. (2002). Data mining in soft computing framework: a survey. *IEEE Transactions on Neural Networks*, *13*(1), 3–14. https://doi.org/10.1109/72.977258

Mohanavalli, S. and Jaisakthi, S.M. (2015). Precise distance metric for mixed data clustering using chi-square statistics. *Research Journal of Applied Sciences*, *10*(12): 1441-1444. https://doi.org/10.19026/rjaset.10.1846

Moreno, J. & Rios, F. (2019). Renewed violence in Colombia: A visit to FARC's jungle lair. Spiegel Online. Retrieved on 3rd October 2020 from https://www.spiegel.de/international/world/farc-preparing-for-renewed-fighting-deep-in-colombian-jungle-a-1286296.html.

Morissette, L. & Chartier, S. (2013). The k-means clustering technique: General considerations and implementation in Mathematica. *Tutorials in Quantitative Methods for Psychology*, *9*(1):15-24. https://doi.org/10.20982/tqmp.09.1.p015

Nanopoulos, A., & Yannis, M. (2001). C2P: Clustering based on Closest Pairs. Retrieved on 29th October, 2019. From http://citeseerx.ist.psu.edu/showciting;jsessionid=2A8586B1F54B5BC067B3C8994A558275?cid=450412

Nielsen, F. Nock, R & Shun-ichi A. (2014). On clustering histograms with k-means by using mixed-divergences. *Entropy*, *16*(6), 3273-3301. https://doi.org/10.3390/e16063273

Oleji, P. C., Nwokorie, N. E., Onuodu, F. & Obinna. O. D. (2016). Clustering mixed dataset with multi-swarm optimization and k-prototype clustering algorithm. *Nigeria Computer Society (NCS), 27*, 205-221.

Omran, M. (2005). Particle swarm optimization methods for pattern recognition and image processing. *PhD Thesis, University of Pretoria*, 1-225.

Onuodu, F. E., Nwachukwu, E. O. & Owolabi, O. (2014). Extension of k-means algorithm for clustering mixed data. *Scientia Africana*, *13*(2), 180-198.

Onyibe, P. (2019). RDNA threatens to resume bombing Niger delta oil installations. new telegraph. Retrieved On 20 April, 2020 from https://www.newtelegraphng.com/2019/09/rnda-threatens-to-resume-bombing-niger-delta-oil-installations-if/

Opara. C. C., Eze, U. F. & Oleji, C. P. (2020). Hybrid Data Mining Model for Knowledge Discovery on Students Academic Performance. *International Journal of Research Publications*, *47*(1), 1-12. https://doi.org/10047122020986

Oyelade, O. J., Oladipupo, O. O. & Obagbuwa, I. C. (2010). Application of K-means clustering algorithm for prediction of Students Academic Performance. *International Journal of Computer Science and Information Security*, *7*(1), 292-295.

Palak, S. & Vikas, K. (2017). Social media generated big data clustering. *International Conference on Computer Communication and Informatics*. 2-7. https://doi.org/10.1109/ICCCI.2017.8117716

Panigrahi, B. K, Shi, Y. & Lim, M. H. (2011). *Handbook of swarm intelligence. series: adaptation, learning, and optimization* (7th ed.). *Springer*.

Park, H., Lee, J & Jun, C. (2006). A k-means-like algorithm for k-medoids clustering and its performance. *International Journal of Research and Analytical Reviews (IJRAR)*, 1-10.

Parrish, J. K, Viscido, S.V & Grunbaum, D. (2002). Self-organization fish schools: an examination or emergent properties. *Biological Bulletin*, *202*(3), 296-305. https://doi: 10.2307/1543482

Parsopoulos, K. E. & Vrahatis, M. N (2010). *Particle swarm optimization and intelligence: advances and applications* (1st ed.). Springer Journal.

Pence, H. E. (2014). What is big data and why is it important? *Journal of Educational Technology Systems*, *43*(2), 159–171. https://doi.org/10.2190/et.43.2.d

Pham, D.-T., Suarez-Alvarez, M. M., & Prostov, Y. I. (2011). Random search with k-prototypes algorithm for clustering mixed datasets. *Proceedings of the Royal Society A*, *467*(2132), 2387–2403. https://doi.org/10.1098/rspa.2010.0594

Prasad1, M. V. S. & Raju, N. O. (2020). Extended K-mean clustering technique using map reduced segmentation clustering for big data. *International Journal of Research and Analytical Reviews (IJRAR)*, *7*(2), 946-952.

Pierfederici, F. (2016). Distributed Computing with Python. *Packt Publishing*. Retrieved on 30th October, 2019 from https://www.packtpub.com/product/distributed-computing-with-python/9781785889691.

Pinisetty, V.N. P., Valaboju, R & Rao, N. R. (2015). Hybrid Algorithm for Clustering Mixed Data Sets. *IOSR Journal of Computer Engineering*, *6*(2), 09-13.

Prabha, A. K., Natesan, K. & Visalakshi, K. K. (2015). Particle swarm optimization-based k-prototype clustering algorithm. *IOSR Journal of Computer Engineering*, *17*(2), 56-62.

Premkumar, B. (2019). Hadoop ecosystem and their components – a complete tutorial. Retrieved on 15th October, 2020 from https://data-flair.training/blogs/hadoop-ecosystem-components/

Pusukuri, K. (2019). CS5412 / Lecture 25 Apache Spark and RDDs. *Spring*, 1-59. Retrieved on 14th March 2020 from http://www.cs.cornell.edu/courses/cs5412/2019sp/slides/Lecture-25.pdf.

Qinglin, Q., & Fei, T. (2018). Digital twin and big data towards smart manufacturing and industry 4.0: 360-degree comparison. *IEEE Translations, 6*, 3585-3593. https://doi.org/10.1109/ACCESS.2018.2793265

Ramakrishnan, M. & Jayaraj, T.D. (2014). Modified k-means algorithm for effective clustering of categorical data sets. *International Journal of Computer Applications, 89*(7), 0975 – 8887. https://doi.org/10.5120/15518-4102

Ramesh, V., Ramar, K., & Babu, S. (2013). Parallel k-means algorithm on agricultural database. *IJCS International Journal of Computer Science, 10*(1), 1694-0784.

Rayward-Smith, V. J. (2007). Statistics to measure correlation for data mining applications. *Computational Statistics & Data Analysis, 51*, 3968–3982. https://doi.org/10.1016/j.csda.2006.05.025

Rebentrost, P., Mohseni, M., & Lloyd, S. (2014). Quantum support vector machine for big data classification. *Physical Review Letters, 113*(13), 1-5. https://doi.org/10.1103/physrevlett.113.130503

Rebollo, C. S., Puente, C., Palacious, R., Piriz, C., Fuentes, J. P. & Jarauta, J., (2019). Detection of jihadism in social network using big data techniques supported by graphs and fuzzy clustering. *Hindawi,* 1-14. https://doi.org/10.1155/2019/1238780.

Rehan, M. & Gangodkar, D. (2015). Hadoop, MapReduce and HDFS: a developer's perspective. Procedia *Computer Science*, *48*, 45–50. https://doi.org/10.1016/j.procs.2015.04.108

Rivinius, J. (2014). Majority of 2013 terrorist attacks occurred in just a few countries, in press release. *National Consortium for the study of Terrorism and Responses to Terrorism*, 1-2.

Röhler, A. B. & Chen, S. (2012). Multi-swarm hybrid for multi-modal optimization. *In Proceedings of the IEEE Congress on Evolutionary Computation*,1759-1766.

Russom, P. (2011). Big data analytics. TDWI Best Practices Report. 1-6.

Rusu, F., Dobra, A. (2012). Glade: a scalable framework for efficient analytics. *In: Proceedings of LADIS Workshop held in conjunction with VLDB*, 1–6.

Sambasivan, R & Sourish, D. (2017). Big data classification using augmented decision trees. *ArXiv*, 1-9.

San, O. M., Huynh, V. & Nakamori, Y. (2004). An alternative extension of the k-means algorithm for clustering categorical data. *International Journal of Applied Mathematics and Computer Science*, *14*(2), 241-247.

Sathesh, K. K. & Hemalatha, M. (2014). An innovative potential on rule optimization using fuzzy artificial bee colony. *Research Journal of Applied Sciences, Engineering and Technology*, *7*(13), 2627-2633. https://doi.org/10.19026/rjaset.7.578.

Sathi, A. (2012). *Big data analytics: disruptive technologies for changing the game* (1th ed.). MC Press.

Satyanarayana, A. (2014). Intelligent sampling for big data using bootstrap sampling and chebyshev inequality. *In: Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering*,1–6.

Saeed, M. M., Aghbari, Z., & Alsharidah, M. (2020). Big data clustering techniques based on Spark: a literature review. Peer J Computer. Science. 2-28. http://doi.org/10.7717/peerj-cs.321

Scott, S. (2018). How swarm intelligence is making simple tech much smarter. *Singularity hub*. Retrieved on 20th October, 2019 from https://singularityhub.com/2018/02/08/how-swarm-intelligence-is-making-simple-tech-much-smarter/.

Shi, C., Qingyu, Z. & Quande, Q. (2016). Big data analytic with swarm intelligent. *Industrial Management & Data Systems*, *116*(4), 646-666. http://dx.doi.org/10.1108/IMDS-06-2015-0222.

Shi, Y., & Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. *Evolutionary Programming VII*, 591–600. https://doi.org/10.1007/bfb0040810

Shirkhorshidi, A. S., Aghabozorgi, S. R., The, Y. W. & Herawan, T. (2014). Big data clustering: a review. *Springer*, 707–720.

Shukla, D. & Alim, A. (2018). Big data analytics approach using indexed and ranking for excellence in higher education. *International Journal of Computer Application*, *180*(35), 41-47.

Shuting, X., & Jun, Z., (2004). A new data mining approach to predict matrix condition number, communication in information and systems. *International Press*, *4*(4), 325-340.

Sowjanya, M., Ravindra, K. & Kumar, R. Y. (2014). Application of concept-based mining model in text clustering. *International Journal of Computer Science and Information Technologies (IJCSIT)*, *5*(5), 6578-6582.

Start (2019). Trends in global terrorism: Islamic state's decline in Iraq and expanding global impact; fewer mass casualty attacks in western Europe; number of attacks in the united states highest since 1980s. *Start*,1-3.

Strang, D. K. & Sun Z. (2017). Analysis relationships in terrorism big data using hadoop and statistics. *Journal of Computer Information Systems*, *57*(1): 67-75. https://doi.org/10.1080/08874417.2016.1181497.

Sun, Z. & Yearwood, J. (2014). A theoretical foundation of demand-driven web services, in Demand-Driven Web Services: Theory, Technologies, and Applications. *IGI-Globa*l, 1-25.

Sun, Z., Sun, L., & Strang, K. (2016). Big data analytics services for enhancing business intelligence. *Journal of Computer Information Systems*, *58*(2), 1-9. https://doi.org/10.1080/08874417.2016.1220239

Tekin, C. & Schaar, M. (2013). Distributed online big data classification using context information. *In Proceedings of the Allerton Conference on Communication, Control, and* Computing,1435–1442.

Thayananathan, V. & Albeshri, A. (2018). Big data security issues and quantum cryptography for cloud computing. *International Journal of Computer Application*, *180*(34), 22-24.

Tibshirani, R., Walther, G., & Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B*, *63*(2), 411–423. https://doi.org/10.1111/1467-9868.00293

Tour, I. & Gangopadhyay, A. (2016). Real time big data analytics for predicting terrorist incidents. *IEEE Symposium on Technologies for Homeland Security*, 1-6. https://doi.org/10.1109/THS.2016.7568906.

Vajjhala, N. Strang, K. & Sun, Z. (2015). Statistical modeling and visualizing of open big data using a terrorism case study. *International Conference on Open and Big Data (OBD)*, 24-26.

Valle, Y., Venayagamoorthy, G., Mohagheghi, S., Hernandez, J. C. & Harley, R. G. (2008). Particle swarm optimization: basic concepts, variants and applications in powers system. *IEEE Trans. on Evolutionary Computation*, *12*(2), 171-195.

Wang, K., Xu, C., & Liu, B. (1999). Clustering transactions using large items. *In: Proceeding of the 10th ACM conference on information and knowledge management*, 483–490.

Wang, F. Y. (2012). A big-data perspective on AI: newton, Merton, and analytics intelligence. *IEEE Intelligent Systems*, 1-4

Wang, L., Liu, X., Sun, M., Qu, j., & Wei1, Y. (2018). A new chaotic starling particle swarm optimization algorithm for clustering problems. *Hindawi*, 1-15. https://doi.org/10.1155/2018/8250480

Wei, M., Chow, T. W. S. &. *Chan*, R. H. M. *(2015)*. Clustering heterogeneous data with k-means by mutual information-based unsupervised feature transformation. *Entropy, 17*, 1535-1548.

Xindong Wu, Xingquan Zhu, Gong-Qing Wu, & Wei Ding. (2014). Data mining with big data. IEEE *Transactions on Knowledge and Data Engineering*, *26*(1), 97–107. https://doi.org/10.1109/tkde.2013.109

Xiong, N. He, J., Yang, H., Kim, T. & Lin, C. (2010). A survey on decentralized flocking schemes for a set of autonomous mobile robots. *Journal of Communications*, *5*(1), 31-38.

Xu, H, Li, Z, Guo, S, & Chen, K. (2012). CloudVista: Interactive and economical visual cluster analysis for big data in the cloud. *Proceedings of the VLDB Endowment*, *5*(12), 1886–1889.

Xu, R. & Wunsch, D. (2009). *Clustering* (1th ed.). Wiley-IEEE Press.

Xue, Z., Shen, G., Li, J., Xu, Q., Zhang, Y. and Shao, J. (2012). Compression-aware I/O performance analysis for big data clustering. *In: Proceedings of the International Workshop on Big Data,*

Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, 45–52. https://doi.org/10.1145/2351316.2351323

Yang, C., Zhang, X., Zhong, C., Liu, C., Pei, J., Ramamohanarao, K., & Chen, J. A. (2014). Spatiotemporal compression-based approach for efficient big data processing on cloud. *Journal for Computer System Science*, *80*(8), 1563–1583. https://doi.org/10.1016/j.jcss.2014.04.022

Yawen, W., Hongchang, C., Shaomei, L., Jianpeng, Z., & Chao, G. (2014). Object tracking by color distribution fields with adaptive hierarchical structure. *The Visual Computer*, *33*, 235–247.

Ye, F., Wang, Z. J., Zhou, F. C., Wang, Y. P., & Zhou, Y. C. (2013). Cloud-based big data mining and analyzing services platform integrating R. *In: Proceedings of the International Conference on Advanced Cloud and Big Data*, 147–151. https://doi.org/ 10.1109/CBD.2013.13

Yiu-ming, C., & Hong, J. (2013). Categorical and numerical attribute data clustering based on a unified similarity metric without knowing cluster number, pattern recognition. Sci*enceDirec*t, *46*(8), 2228-2238. https://doi.org/10.1016/j.patcog.2013.01.027

Zhang, C., & Fang, Z. (2013). An Improved K-means Clustering Algorithm. *Journal of Information & Computational Science*, *10*(1), 193-199.

Zhang, H. (2013). A novel data preprocessing solution for large scale digital forensics investigation on big data. *Journal of Information & Computational Science*, 1-150.

Zhao, S. Z., Liang, J. J., Suganthan, P. N., & Tasgetiren, M. F. (2008). Dynamic multi-swarm particle swarm optimizer with local search for Large Scale Global Optimization. *IEEE*

Congress on Evolutionary Computation, 3845-3852. https://doi.org/10.1109/cec.2008.4631320

Zou, H., Yu, Y., Tang, W., & Chen, H. M. (2014). Improving I/O performance with adaptive data compression for big data applications. *In: Proceedings of the International Parallel and Distributed Processing Symposium Workshops*, 1228–1237. https://doi.org/10.1109/IPDPSW.2014.138

APPENDICES A

Program Source Code

```
package MultiSwarm;
import MultiSwarm.Constants;
import MultiSwarm.ParticlePSO;
import net.sourceforge.jswarm_pso.Particle;
import net.sourceforge.jswarm_pso.ParticleUpdate;
import net.sourceforge.jswarm_pso.Swarm;
/*
 * Class Name: PSO Position and Velocity Update
 * Purpose: It is used to update the position and velocity of the given particle
*/
public class ModifiedPSOUpdate extends ParticleUpdate{
	/*
	 * Global Parameters:
	 * obj : is the object used to store the particle as an object.
	 */
	ParticlePSO obj;
	ModifiedPSOUpdate(ParticlePSO particle){
		super(particle);
		this.obj=particle;
	}
		/*
	 * update : provided by JSwarm and used to update the position and velocity
	 *
	 */
	public void update(Swarm swarm,Particle particle) {
				double v[]=particle.getVelocity();
		double x[]=particle.getPosition();
		double pbest[]=particle.getBestPosition();
		double gbest[]=swarm.getBestPosition();
```

```
                    /*
                     * count : stores the number of iterations of each particle has been updated.
                     */
                    obj.count=obj.count+1;
                    int it=obj.count;
                    /*
                     * w : represents inertia weight which has been calculated based on number of iterations
                    * k : represents the constriction factor which has been calculated based on number of terations
                     */
                    double w=0.857143+(1-0.857143)*(1-it/Constants.NoOfIterations);
                    double k=(Math.cos((Math.PI/Constants.NoOfIterations)*it)+2.5)/4.0;
                    for(int i=0;i<Constants.NoOfTasks;i++) {

                    v[i]=0.729844*v[i]+2*Math.random()*(pbest[i]-x[i])+2*Math.random()*(gbest[i]-x[i])+T;
                            // 2.
            //              v[i]=w*v[i]+2*Math.random()*(pbest[i]-x[i])+2*Math.random()*(gbest[i]-x[i]);
                            // 3.
        //                  v[i]=k*(v[i]+2*Math.random()*(pbest[i]-x[i])+2*Math.random()*(gbest[i]-x[i]));
                            // 4.
//                          v[i]=k*(w*v[i]+2*Math.random()*(pbest[i]-x[i])+2*Math.random()*(gbest[i]-x[i]));
                            particle.setVelocity(v);
                            x[i]=(x[i]+v[i]);
                            particle.setPosition(x);
                    }
            }
}
package MultiSwarm;
/*
 * Class Name : Constants
 * Purpose : Serves as a reference for changing the parameters for the PSO Algorithm.
 *
 */
public class Constants {
            /*
```

```
	 *	Here number of tasks represents the number of dimensions in a particle
	 *
	 */
	public static final int NoOfTasks = 5;
	public static final int NoOfSwarms = 6;
	public static final int NoOfParticles = 30;
	public static final int NoOfIterations = 4000;
}
package MultiSwarm;
/**
 *
 *  */
import java.io.*;
import java.util.Scanner;
public class testmain {
	public static void main(String[] args) throws Exception{
		int i,j,k,n,m,func_num;
		double[] x;
		double[] f;
		/*
		 * Change the path in here as well
		 */
		//File fpt = new File("C:\\Users\\admin\\Desktop\\java\\input_data\\shift_data.txt");
		File fpt = new File("C:\\dataset\\\\Australia.data");
		m=2;
		n=10;
		testfunc tf = new testfunc();
		Scanner input = new Scanner(fpt);
		x = new double[n*m];
		for(i=0;i<n;i++){
			x[i]=input.nextDouble();
		}
		for(i=0;i<n;i++){
```

```
                System.out.println(x[i]);
            }
            input.close();

            for(i=1;i<m;i++){
                for(j=0;j<n;j++){
                    x[i*n+j]=0.0;
                    System.out.println(x[i*n+j]);
                }
            }
            f= new double[m];
            for(i=0;i<28;i++){
                func_num=i+1;
                for(k=0;k<1;k++){
                    //tf.test_func(x,f,n,m,func_num);
                    for(j=0;j<m;j++){
                        System.out.println("f"+func_num+"(x["+(j+1)+"])="+f[j]);
                    }
                }
            }

        }
}

package MultiSwarm;
import java.util.Arrays;
import java.io.*;
import net.sourceforge.jswarm_pso.*;
/*
 * Class Name: PSOSubSwarm
public class PSOSubSwarm {
        /*
         * Global Parameters:
         * ff : represents the fitness function used by all particles in the swarm
```

```
	 * swarm : represents the different swarms that are used in multiswarm PSO
	 * particles[i][j] : represents the particles where i is the swarm id and j is particle id
	 * error[][][] : stores the error for all 28 fitness function
	 * ansplot[][][] : stores the fitness value for all 28 fitness function
	 * ansfitness[] : represents the best possible fitness value of the function (minimum)
	 * funcno : stores the function number of CEC function to be checked
	 */
	FitnessFunctionPSO ff ;
	Swarm swarm[] = new Swarm[Constants.NoOfSwarms];
	private static ParticlePSO particles[][];
	static double error[][][] = new double[28][5][11];
	static double ansplot[][][] = new double[28][5][11];
	static double multiply[]={};
	static int minimum=0;
	double ansfitness[]={};
	int funcno;
	int counter;
public PSOSubSwarm(int funcno,int counter)
{
	this.funcno=funcno;
	this.counter=counter;
	ff = new FitnessFunctionPSO(funcno);
	/*
	 * Dividing the particles into subswarms
	 */
	for(int q=0;q<Constants.NoOfSwarms;q++)
	swarm[q] = new Swarm(Constants.NoOfParticles/Constants.NoOfSwarms, new ParticlePSO(), ff);
	initializeParticles();
}
public void initializeParticles() {
	particles = new ParticlePSO[Constants.NoOfSwarms][Constants.NoOfParticles/Constants.NoOfSwarms];
	for(int q=0;q<Constants.NoOfSwarms;q++) {
		for(int i=0;i<Constants.NoOfParticles/Constants.NoOfSwarms;i++) {
```

```
                        particles[q][i]= new ParticlePSO();
                }
        }
}
public double[] run() {
        for(int q=0;q<Constants.NoOfSwarms;q++) {
                /*
                 * Defining the parameters of swarm
                 *
                 */
                swarm[q].setMinPosition(-100);
                swarm[q].setMaxPosition(100);
                swarm[q].setMaxMinVelocity(1.0);
                swarm[q].setParticles(particles[q]);
                swarm[q].setParticleUpdate(new ModifiedPSOUpdate(new ParticlePSO()));
        }
        double anscurr[]= new double[Constants.NoOfSwarms];

        for(int i=1;i<=Constants.NoOfIterations;i++) {
                for(int q=0;q<Constants.NoOfSwarms;q++) {
                        swarm[q].evolve();
                        anscurr[q]=swarm[q].getBestFitness();
                        if(i%10 == 0) {
                        System.out.println("Global best at iteration "+i+" : for swarm :"+q+"
"+swarm[q].getBestFitness());
                        }
                }
                /*
                 * Finding the best particles in all the sub swarms and comparing them
                 *
                 */
                for(int j=0;j<11;j++){
                        if(i==(multiply[j]*Constants.NoOfIterations)){
                                int min=0;
                                for(int w=1;w<Constants.NoOfSwarms;w++)
```

```
                                        {
                                                if(swarm[w].getBestFitness()<swarm[min].getBestFitness())
                                                        min=w;
                                        }
                                        minimum=min;

                                        error[funcno-1][counter][j]=Math.abs(swarm[min].getBestFitness()-
ansfitness[funcno-1]);

                                        ansplot[funcno-1][counter][j] = swarm[min].getBestFitness();

                                }
                        }

                }
                ParticlePSO bestparticle = (ParticlePSO)swarm[minimum].getBestParticle();

                return swarm[minimum].getBestPosition();

                }

/*

 * Printing the best fitness value for all the swarm

 *

 */

public double printBestFitness() {

                System.out.println("The best fitness value is "+swarm[minimum].getBestFitness());

                return swarm[minimum].getBestFitness();

}

public static void main(String args[]) {

                double result[][] = new double[28][5];

                for(int f=0;f<28;f++) {

                        for(int count=0;count<5;count++) {

                                System.out.println("for function fitness :"+ (f+1)+"\t run:"+ (count+1));

                                PSOSubSwarm obj = new PSOSubSwarm(f+1,count);

                                obj.run();

                                result[f][count]=obj.printBestFitness();

                        }

                }

                double maxf[]= new double[28];

                double minf[] = new double[28];

                double mean[] = new double[28];

                double median[] = new double[28];
```

```
for(int i=0;i<28;i++)
{	double maxi=-100000;
	double mini=100000;
	double sum=0;
	double temp[]= new double[5];
for(int j=0;j<5;j++)
{
	sum+=result[i][j];
	maxi=Math.max(maxi, result[i][j]);
	mini=Math.min(mini, result[i][j]);
	temp[j]=result[i][j];
}
maxf[i]=maxi;
minf[i]=mini;
mean[i]=(double)sum/5.0;
Arrays.sort(temp);
median[i]=temp[2];
}
try {
	/*
	 * Change the path of output files
	 */
	//FileWriter fw = new FileWriter("C:\\Users\\admin\\Desktop\\Output Files\\multiswarm\\5-
D_Mean_Median.txt", true);
 //  C:\\dataset\\\\Australia.data
	FileWriter fw = new FileWriter("C:\\dataset\\Australia.data\n" +
"	 ", true);
  BufferedWriter bw = new BufferedWriter(fw);
    PrintWriter out = new PrintWriter(bw);

out.println("max\t\t min\t\t mean\t\t median");
for(int i=0;i<28;i++)
{
```

```
String a = (new Double(maxf[i]).toString())+"\t"+(new Double(minf[i]).toString())+"\t"+(new
Double(mean[i]).toString())+"\t"+(new Double(median[i]).toString());

   out.println(a);

        }

out.close();

}

catch(Exception e)

{

        System.out.println("Error has occured while writing in file");

}

        try {

        /*

         * Change the path of output files

         */

        FileWriter fw = new FileWriter("C:\\Users\\admin\\Desktop\\Output Files\\multiswarm\\5-
D_Error.txt", true);

    BufferedWriter bw = new BufferedWriter(fw);

    PrintWriter out = new PrintWriter(bw);

        for(int i=0;i<28;i++){

                for(int j=0;j<5;j++){

                for(int k=0;k<11;k++){

                                String a = "For function :"+(new Integer(i+1).toString())+ "run :"+(new
Integer(j+1).toString())+" error at:"+(new Double(multiply[k]*Constants.NoOfIterations).toString())+" is :"+(new
Double(error[i][j][k]).toString());

                                out.println(a);
                        }
                }
        }
        out.close();

        }

        catch(Exception e)

        {

                System.out.println("Error has occured while writing in file");

        }

        try {

        for(int i=0;i<28;i++){
```

```
                        /*
                         * Change the path of output files
                         */
                                FileWriter fw = new FileWriter("C:\\Users\\admin\\Desktop\\Output Files\\multiswarm\\f"+(i+1)+".txt", true);
                            BufferedWriter bw = new BufferedWriter(fw);
                            PrintWriter out = new PrintWriter(bw);
                                for(int k=0;k<11;k++){
                                        String p = new Double(multiply[k]*Constants.NoOfIterations).toString()+" ";
                                        for(int j=0;j<5;j++){
                                                p=p+new Double(ansplot[i][j][k]).toString()+" ";
                                        }
                                        out.println(p);
                                }
                                out.close();
                        }
                                        }
                        catch(Exception e)
                        {
                                System.out.println("Error has occured while writing in file");
                        }
                }
}

package MultiSwarm;
import java.util.Random;
/*
 * Class Name: Particle of PSO
 * Purpose: This is the particle whose position and velocity is updated and fitness is calculated.
 *
 */
import net.sourceforge.jswarm_pso.*;
public class ParticlePSO extends Particle{
```

```
/*
 * Global Parameters:
 * count : refers to the number of iterations that has been completed by a particle
 * position[] : represents the position of the particle in n-dimension.
 * velocity[] : represents the velocity of the particle in n-dimension.
 */
public int count=0;
double position[] = new double[Constants.NoOfTasks];
double velocity[] = new double[Constants.NoOfTasks];
public ParticlePSO(){
        super(Constants.NoOfTasks);
        initialize();
}
void initialize() {
        for(int i=0;i<Constants.NoOfTasks;i++) {
                Random random = new Random();
                /*
                 * Search space of CEC-2013 function is bounded from -100.0 to 100.0 along any direction
                 */
                position[i] = (Math.random()-0.5)*200;
                velocity[i] = (Math.random()-0.5)*(0.5);
        }
        setPosition(position);
        setVelocity(velocity);
}
}
 package MultiSwarmPSOCloudTest;
import java.util.Random;
import MultiSwarmPSOCloudTest.Constants;
import net.sourceforge.jswarm_pso.*;
public class ParticlePSO extends Particle{
        /*
         * Global Parameters:
```

```
	 * count : refers to the number of iterations that has been completed by a particle
	 * position[] : represents the position of the particle in n-dimension.
	 * velocity[] : represents the velocity of the particle in n-dimension.
	 */
	int count=0;
	public ParticlePSO(){
	super(Constants.NoOfTasks);
	double position[] = new double[Constants.NoOfTasks];
	double velocity[] = new double[Constants.NoOfTasks];

	for(int i=0;i<Constants.NoOfTasks;i++) {
		Random random = new Random();
		position[i] = random.nextInt(Constants.NoOfVMs);
		velocity[i] = Math.random()*Constants.NoOfVMs;
//		position[i] = (Math.random())*Constants.NoOfVMs;
//		velocity[i] = (Math.random()-0.5)>0?1.0:-1.0;
	}
	setPosition(position);
	setVelocity(velocity);
}
public String toString() {
	String output = "";
	for(int i=0;i<Constants.NoOfVMs;i++) {
		String tasks = "";
		int number_of_tasks = 0;
		for(int j=0;j<Constants.NoOfTasks;j++) {
			if( i== (int)getPosition()[j]) {
				tasks +=(tasks.isEmpty() ? " " : " " ) + j;
				++number_of_tasks;
			}
		}
		if(tasks.isEmpty())
			output += "NO Tasks is in VM "+ i+"\n";
```

```
                else
                        output += number_of_tasks +" Tasks is in VM "+i +" Tasks id = " +tasks +"\n";
        }
        return output;
}
}
“
package PSOCEC2013;
import java.io.*;
/*
 * Class Name: PSO
 * Purpose : Used for running the PSO simulation
 *
 */
import java.util.Arrays;
import net.sourceforge.jswarm_pso.*;
public class PSO {
        /*
         * Global Parameters:
         * ff : represents the fitness function used by all particles in the swarm
         * swarm : represents the swarms that are used in PSO
         * particles[i] : represents the particles where i is particle id
         * error[][][] : stores the error for all 28 fitness function
         * ansplot[][][] : stores the fitness value for all 28 fitness function
         * ansfitness[] : represents the best possible fitness value of the function (minimum)
         * funcno : stores the function number of CEC function to be checked
         */
FitnessFunctionPSO ff ;
Swarm swarm;
private static ParticlePSO particles[];
static double error[][][] = new double[28][7][11];
static double ansplot[][][] = new double[28][7][11];
static double multiply[]={};
```

```
double ansfitness[]={};
int funcno;
int counter;
public PSO(int funcno,int counter)
{
	this.funcno=funcno;
	this.counter=counter;
	ff = new FitnessFunctionPSO(funcno);
	swarm = new Swarm(Constants.NoOfParticles, new ParticlePSO(), ff);
	initializeParticles();
}
public void initializeParticles() {
	particles = new ParticlePSO[Constants.NoOfParticles];
	for(int i=0;i<Constants.NoOfParticles;i++) {
		particles[i]= new ParticlePSO();
	}
}
public double[] run() {
	/*
	 * Defining the parameters of swarm
	 *
	 */
	swarm.setMinPosition(-100);
	swarm.setMaxPosition(100);
	swarm.setMaxMinVelocity(1.0);
	swarm.setParticles(particles);
	swarm.setParticleUpdate(new ModifiedPSOUpdate(new ParticlePSO()));
	for(int i=1;i<=Constants.NoOfIterations;i++) {
		swarm.evolve();
		if(i%10 == 0) {

			System.out.println("Global best at iteration "+i+" :"+swarm.getBestFitness());
		}
```

```
                for(int j=0;j<11;j++){
                if(i==(multiply[j]*Constants.NoOfIterations)){
                        error[funcno-1][counter][j]=Math.abs(swarm.getBestFitness()-ansfitness[funcno-1]);
                        ansplot[funcno-1][counter][j] = swarm.getBestFitness();
                }
                }

        }
        //System.out.println("The best fitness value is "+swarm.getBestFitness()+"\n The Best position value is "+
swarm.getBestParticle().getBestPosition());
        ParticlePSO bestparticle = (ParticlePSO)swarm.getBestParticle();
        printBestFitness();
        return swarm.getBestPosition();
        }
/*
 * Printing the best fitness value for all the swarm
 *
 */
public double printBestFitness() {
        System.out.println("The best fitness value is "+swarm.getBestFitness()+"\n The Best position value is ");
        double[] best = swarm.getBestPosition();
        for(int i=0;i<best.length;i++) {
                System.out.print(best[i]+"\t");
        }
                System.out.println();
                return swarm.getBestFitness();
}

public static void main(String args[]) {
        double result[][] = new double[28][7];
        for(int f=0;f<28;f++) {
        for(int count=0;count<7;count++) {
        System.out.println("for function fitness :"+ (f+1)+"\t run:"+ (count+1));
```

```
PSO obj = new PSO(f+1,count);


obj.run();
result[f][count]=obj.printBestFitness();
}
}
double maxf[]= new double[28];
double minf[] = new double[28];
double mean[] = new double[28];
double median[] = new double[28];
for(int i=0;i<28;i++)
{
        double maxi=-100000;
        double mini=100000;
        double sum=0;
        double temp[]= new double[7];
for(int j=0;j<7;j++)
{
        sum+=result[i][j];
        maxi=Math.max(maxi, result[i][j]);
        mini=Math.min(mini, result[i][j]);
        temp[j]=result[i][j];
}
maxf[i]=maxi;
minf[i]=mini;
mean[i]=(double)sum/7.0;
Arrays.sort(temp);
median[i]=temp[3];
}
try {
        /*
         * Change the path of output files
         */
```

```
FileWriter fw = new FileWriter("C:\\Users\\user\\Desktop\\OutputFiles \\5-D_Mean_Median.txt", true);
BufferedWriter bw = new BufferedWriter(fw);
PrintWriter out = new PrintWriter(bw);

out.println("max\t\t min\t\t mean\t\t median");
for(int i=0;i<28;i++)
{
String a = (new Double(maxf[i]).toString())+"\t"+(new Double(minf[i]).toString())+"\t"+(new Double(mean[i]).toString())+"\t"+(new Double(median[i]).toString());
out.println(a);
}
out.close();
}
catch(Exception e)
{
System.out.println("Error has occured while writing in file");
}
try {
/*
 * Change the path of output files
 */
FileWriter fw = new FileWriter("C:\\Users\\user\\Desktop\\OutputFiles\\5-D_Error.txt", true);
BufferedWriter bw = new BufferedWriter(fw);
PrintWriter out = new PrintWriter(bw);
for(int i=0;i<28;i++){
for(int j=0;j<7;j++){

for(int k=0;k<11;k++){
String a = "For function :"+(new Integer(i+1).toString())+ "run :"+(new Integer(j+1).toString())+" error at:"+(new Double(multiply[k]*Constants.NoOfIterations).toString())+" is :"+(new Double(error[i][j][k]).toString());
out.println(a);
}
```

```
                }
            }
            out.close();
            }
            catch(Exception e)
            {
                System.out.println("Error has occured while writing in file");
            }
        try {
            for(int i=0;i<28;i++){
                /*
                 * Change the path of output files
                 */
                    FileWriter fw = new
FileWriter("C:\\Users\\user\\Desktop\\OutputFiles\\f"+(i+1)+".txt", true);
                   BufferedWriter bw = new BufferedWriter(fw);
                   PrintWriter out = new PrintWriter(bw);
                       for(int k=0;k<11;k++){
                           String p = new
Double(multiply[k]*Constants.NoOfIterations).toString()+" ";
                           for(int j=0;j<7;j++){
                           p=p+new Double(ansplot[i][j][k]).toString()+" ";
                       }
                           out.println(p);
                }
                    out.close();
            }


            }
            catch(Exception e)
            {
                System.out.println("Error has occured while writing in file");
            }
        }
```

```
}
“
“
package PSOCEC2013;
import java.util.Random;
import net.sourceforge.jswarm_pso.*;
public class ParticlePSO extends Particle{
	/*
	 * Global Parameters:
	 * count : refers to the number of iterations that has been completed by a particle
	 * position[] : represents the position of the particle in n-dimension.
	 * velocity[] : represents the velocity of the particle in n-dimension.
	 */
int count=0;
double position[] = new double[Constants.NoOfTasks];
double velocity[] = new double[Constants.NoOfTasks];
ParticlePSO(){
	super(Constants.NoOfTasks);
		initialize();
}
void initialize() {
	for(int i=0;i<Constants.NoOfTasks;i++) {
		Random random = new Random();
		/*
		 * Search space of CEC-2013 function is bounded from -100.0 to 100.0 along any direction
		 */
		position[i] = (Math.random()-0.5)*200;
		velocity[i] = (Math.random()-0.5)*(0.5);
	}
	setPosition(position);
	setVelocity(velocity);
}
}
```

```
package PSOCEC2013;
import net.sourceforge.jswarm_pso.*;
/*
 * Class Name: Fitness Function
 * Purpose: It provides the function used to test the convergence of our PSO Algorithm
 *
 */
public class FitnessFunctionPSO extends FitnessFunction{
	/*
	 * Global Parameters:
	 * funcno : refers to the function number corresponding to different functions in CEC-2013
	 *
	 */
	int funcno;
	FitnessFunctionPSO(int funcno)
	{
		/*
		 * Here false in super denotes that our objective is to minimize the function.
		 */
		super(false);
		this.funcno=funcno;
	}
	/*
	 * evaluate is the function which returns the fitness value of the position
	 * the fitness value of the position is returned in f[0]
	 */
	public double evaluate(double[] position) {
		testfunc tf = new testfunc();
		double f[]= {0.0,0.0};

		try {
			tf.test_func(position,f, Constants.NoOfTasks, 1, funcno);
		} catch (Exception e) {
```

```
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                return f[0];

        }
}
“
“
package PSOCEC2013;
import java.util.Arrays;
import net.sourceforge.jswarm_pso.*;
public class MYPSO {
FitnessFunctionPSO ff ;
Swarm swarm;
private static ParticlePSO particles[];
static double error[][][] = new double[28][51][11];
static double multiply[]={};
double ansfitness[]={};
int funcno;
int counter;
public MYPSO(int funcno,int run)
{
        this.funcno=funcno;
        this.counter=counter;
        ff = new FitnessFunctionPSO(funcno);
        swarm = new Swarm(Constants.NoOfParticles, new ParticlePSO(), ff);
        initializeParticles();
}
public void initializeParticles() {
        particles = new ParticlePSO[Constants.NoOfParticles];
        for(int i=0;i<Constants.NoOfParticles;i++) {
                particles[i]= new ParticlePSO();
```

```
		}
}
public double[] run() {
		swarm.setMinPosition(-100);//minimum value is the minimum value of vm id
		swarm.setMaxPosition(100);//maximum value of vm id
		swarm.setMaxMinVelocity(1.0);
		swarm.setParticles(particles);
//		swarm.setParticleUpdate(new UpdationPSO(new ParticlePSO()));
		swarm.setParticleUpdate(new ModifiedPSOUpdate(new ParticlePSO()));
		double ansprev=0.0;
		double anscurr=10000.0;
		for(int i=0;i<Constants.NoOfIterations;i++) {
			swarm.evolve();
			anscurr=swarm.getBestFitness();
			if(i%10 == 0) {
				System.out.println("Global best at iteration "+i+" :"+swarm.getBestFitness());
			}
			for(int j=0;j<11;j++){
			if(i==(multiply[j]*Constants.NoOfTasks)){
				error[funcno-1][counter][j]=Math.abs(swarm.getBestFitness()-ansfitness[funcno]);
			}
			}
			}
		//System.out.println("The best fitness value is "+swarm.getBestFitness()+"\n The Best position value is "+
swarm.getBestParticle().getBestPosition());
		ParticlePSO bestparticle = (ParticlePSO)swarm.getBestParticle();
		//System.out.println(bestparticle.toString());
		printBestFitness();
		return swarm.getBestPosition();
		}

public double printBestFitness() {
		System.out.println("The best fitness value is "+swarm.getBestFitness());
```

```
		//double[] best = swarm.getBestPosition();
		//for(int i=0;i<best.length;i++) {
		//		System.out.print(best[i]+"\t");
		//}
				System.out.println();
				return swarm.getBestFitness();
}

public static void main(String args[]) {
		double result[][] = new double[28][51];
		for(int f=0;f<28;f++) {
		for(int count=0;count<51;count++) {
		System.out.println("for function fitness :"+ (f+1)+"\t run:"+ (count+1));
		MYPSO obj = new MYPSO(f+1,count);
		obj.run();
		result[f][count]=obj.printBestFitness();
		}
		}
		double maxf[]= new double[28];
		double minf[] = new double[28];
		double mean[] = new double[28];
		double median[] = new double[28];
		for(int i=0;i<28;i++)
		{
				double maxi=-100000;
				double mini=100000;
				double sum=0;
				double temp[]= new double[51];
		for(int j=0;j<51;j++)
		{
				sum+=result[i][j];
				maxi=Math.max(maxi, result[i][j]);
				mini=Math.min(mini, result[i][j]);
```

```
                temp[j]=result[i][j];
        }
        maxf[i]=maxi;
        minf[i]=mini;
        mean[i]=(double)sum/51.0;
        Arrays.sort(temp);
        median[i]=temp[25];
        }
        System.out.println("max\t\t min\t\t mean\t\t median");
        for(int i=0;i<28;i++)
        {
                System.out.println(maxf[i]+"\t"+minf[i]+"\t"+mean[i]+"\t"+median[i]);
        }
        for(int i=0;i<28;i++){
                for(int j=0;j<51;j++){
                        for(int k=0;k<11;k++){
                                System.out.println("For function :"+(i+1)+ "run :"+(j+1)+"error
at:"+(multiply[k]*Constants.NoOfIterations)+" is :"+error[i][j][k]);
                        }
                }
        }
}


}
“


“
package kmeans;
import java.io.File;
import java.util.Random;
import net.sf.javaml.clustering.Clusterer;
import net.sf.javaml.clustering.KMeans;
import net.sf.javaml.clustering.evaluation.ClusterEvaluation;
```

```
import net.sf.javaml.clustering.evaluation.SumOfSquaredErrors;
import net.sf.javaml.core.Dataset;
import net.sf.javaml.tools.data.FileHandler;
/**
 *
 *  */
public class KMeans1 {
      public static void main(String[] args)throws Exception {
       // TODO code application logic here
          // load iris dataset
               Dataset data = FileHandler.loadDataset(new File("C:\\dataset\\heart.data"),4,",");
      //System.out.println(data);
     // export dataset
    //FileHandler.exportDataset(data, new File("output.txt"));
   // perform clustering
  //  System.out.println(data);
System.out.println("----------Swarm Intelligent Clustering Algorithm OUTPUT----------");

       Random random = new Random();
       //int k = 1 + random.nextInt(10);
       System.out.println("The random generated value of k = ");
       //System.out.println(k);
       Clusterer km = new KMeans(6);
             // perform the clustering by calling the cluster method and passing the
       // dataset as a parameter and storing the clusters in the dataset array
             Dataset[] clusters = km.cluster(data);
             // Write out the number of clusters
        for (int i=0; i< clusters.length; i++){
            Dataset tempV = clusters[i];
       // System.out.println("*****************************");
System.out.println("<*><*>><*><*><*><*><*>><*><*><*><*><*><*><*><*><*><*>Cluster"+i+"<*><*>><*><*><*><*>>
```

```
<*><*><*><*>><*><*><*><*>><*><*><*><*>><*><*><*><*>><*><*><*><*>><*><*><*><*>><*><*><*><
*>><*><*><*><*>><*><*><*><*>><*><*><*><*><*>");
      // System.out.println("Cluster count: " + clusters.length);
       // loop through the clusters and output them
       for (Dataset cluster : clusters) {
          System.out.println(cluster);
       }
       }
       /* Create a measure for the cluster quality */
       ClusterEvaluation sse= new SumOfSquaredErrors();
       /* Measure the quality of the clustering */
       double score=sse.score(clusters);

       // output the sum of suare error
       System.out.println(score);

    }

}" "/*  */
4package kmeans;
import java.text.*;
import java.io.File;
import java.io.IOException;
import java.util.Random;
import javax.swing.JOptionPane;
import net.sf.javaml.clustering.Clusterer;
import net.sf.javaml.clustering.KMeans;
import net.sf.javaml.clustering.evaluation.ClusterEvaluation;
import net.sf.javaml.clustering.evaluation.SumOfSquaredErrors;
import net.sf.javaml.core.Dataset;
import net.sf.javaml.tools.data.FileHandler;

/**
```

```
 *
 * @author Oleji Philips
 */
public class UserInterface extends javax.swing.JFrame {

    private Dataset data;

    /**
     * Creates new form UserInterface
     */
    public UserInterface() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jPanel3 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        outputArea = new javax.swing.JTextArea();
        jLabel7 = new javax.swing.JLabel();
        jPanel4 = new javax.swing.JPanel();
        dataBox = new javax.swing.JComboBox();
```

```
        jLabel2 = new javax.swing.JLabel();
        clusterButton = new javax.swing.JButton();
        jPanel5 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        kField = new javax.swing.JTextField();
        jLabel5 = new javax.swing.JLabel();
        ssError = new javax.swing.JTextField();
        jLabel9 = new javax.swing.JLabel();
        entropy = new javax.swing.JTextField();
        jLabel8 = new javax.swing.JLabel();
        jButton1 = new javax.swing.JButton();
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Big Data Analytics");
        setResizable(false);
        jPanel1.setBackground(new java.awt.Color(255, 153, 51));
        jPanel2.setBackground(new java.awt.Color(51, 102, 0));
        jPanel2.setForeground(new java.awt.Color(255, 255, 255));
        jLabel3.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N
        jLabel3.setForeground(new java.awt.Color(255, 255, 255));
        jLabel3.setText("Big Data Analytic with Dynamic Multi-Swarm Optimization and K-reference Clustering
Algorithms");
        jLabel4.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
        jLabel4.setForeground(new java.awt.Color(255, 255, 255));
        javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(
            jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addGap(73, 73, 73)
                        .addComponent(jLabel3))
                    .addGroup(jPanel2Layout.createSequentialGroup()
```

```
                .addGap(305, 305, 305)
                .addComponent(jLabel4)))
            .addContainerGap(637, Short.MAX_VALUE))
    );
    jPanel2Layout.setVerticalGroup(
        jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel2Layout.createSequentialGroup()
            .addContainerGap(35, Short.MAX_VALUE)
            .addComponent(jLabel3)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jLabel4)
            .addContainerGap())
    );
    jPanel3.setBackground(new java.awt.Color(51, 204, 0));
    jPanel3.setForeground(new java.awt.Color(255, 255, 255));
    outputArea.setColumns(20);
    outputArea.setLineWrap(true);
    outputArea.setRows(100);
    jScrollPane1.setViewportView(outputArea);
    jLabel7.setFont(new java.awt.Font("Tahoma", 3, 14)); // NOI18N
    jLabel7.setForeground(new java.awt.Color(255, 0, 51));
    jLabel7.setText("Clusters Output Results");
    dataBox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Iris Dataset", "Soybean Dataset",
"Wine Dataset", "Yeast Dataset", "Hepititis Dataset", "Heart Dataset", "German_Credit_Dataset", "Australia Dataset",
"Statlog Heart", "Vote Dataset", "Boko Dataset" }));
    dataBox.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            dataBoxActionPerformed(evt);
        }
    });

    jLabel2.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
    jLabel2.setText("Select a Dataset:");
```

```
        clusterButton.setText("Click To Cluster");
        clusterButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                clusterButtonActionPerformed(evt);
            }
        });
        javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);
        jPanel4.setLayout(jPanel4Layout);
        jPanel4Layout.setHorizontalGroup(
            jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()
                .addContainerGap(13, Short.MAX_VALUE)
                .addComponent(jLabel2)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(jPanel4Layout.createSequentialGroup()
                        .addGap(10, 10, 10)
                        .addComponent(clusterButton))
                    .addComponent(dataBox, javax.swing.GroupLayout.PREFERRED_SIZE, 115,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(26, 26, 26))
        );
        jPanel4Layout.setVerticalGroup(
            jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(dataBox, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel2))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(clusterButton)
                .addContainerGap(23, Short.MAX_VALUE))
        );
```

```
        jLabel1.setFont(new java.awt.Font("Times New Roman", 1, 14)); // NOI18N
        jLabel1.setText("Randon Generated Value of Clusters:");
        kField.setEditable(false);
        jLabel5.setBackground(new java.awt.Color(0, 0, 0));
        jLabel5.setFont(new java.awt.Font("Times New Roman", 1, 14)); // NOI18N
        jLabel5.setText("Sum of Squared Error :");
        ssError.setEditable(false);
        jLabel9.setFont(new java.awt.Font("Times New Roman", 3, 18)); // NOI18N
        jLabel9.setText("Accuracy");
        javax.swing.GroupLayout jPanel5Layout = new javax.swing.GroupLayout(jPanel5);
        jPanel5.setLayout(jPanel5Layout);
        jPanel5Layout.setHorizontalGroup(
            jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addGroup(javax.swing.GroupLayout.Alignment.LEADING, jPanel5Layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(kField, javax.swing.GroupLayout.PREFERRED_SIZE, 44,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(javax.swing.GroupLayout.Alignment.LEADING, jPanel5Layout.createSequentialGroup()
                        .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
                            .addGroup(javax.swing.GroupLayout.Alignment.LEADING, jPanel5Layout.createSequentialGroup()
                                .addContainerGap()
                                .addComponent(jLabel9, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                            .addComponent(jLabel5))
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                            .addComponent(ssError, javax.swing.GroupLayout.DEFAULT_SIZE, 163, Short.MAX_VALUE)
                            .addComponent(entropy))))
                .addGap(0, 67, Short.MAX_VALUE))
        );
```

```
        jPanel5Layout.setVerticalGroup(
            jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel1)
                    .addComponent(kField, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(ssError, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel5))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel9, javax.swing.GroupLayout.PREFERRED_SIZE, 28,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(entropy, javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(21, Short.MAX_VALUE))
        );
        jLabel8.setBackground(new java.awt.Color(255, 153, 153));
        jLabel8.setForeground(new java.awt.Color(255, 51, 102));
        jLabel8.setIcon(new javax.swing.ImageIcon(getClass().getResource("/kmeans/Big Data.gif"))); // NOI18N
        jLabel8.setText("Swarm Intelligent");
        jButton1.setText("Clear Results");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });
        javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(
```

```
        jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel3Layout.createSequentialGroup()
            .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanel3Layout.createSequentialGroup()
                    .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addGroup(jPanel3Layout.createSequentialGroup()
                            .addGap(36, 36, 36)
                            .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED_SIZE, 284,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGroup(jPanel3Layout.createSequentialGroup()
                            addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addGroup(jPanel3Layout.createSequentialGroup()
                                    .addGap(98, 98, 98)
                                    .addComponent(jPanel4, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGroup(jPanel3Layout.createSequentialGroup()
                                    .addGap(203, 203, 203)
                                    .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 132,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                            .addGap(63, 63, 63)
                            .addComponent(jPanel5, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jLabel8, javax.swing.GroupLayout.PREFERRED_SIZE, 590,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 1253,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addContainerGap(215, Short.MAX_VALUE))
    );
    jPanel3Layout.setVerticalGroup(
        jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel3Layout.createSequentialGroup()
            .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanel3Layout.createSequentialGroup()
                    .addContainerGap()
```

```
                .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

                    .addComponent(jPanel5, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

                    .addGroup(jPanel3Layout.createSequentialGroup()

                        .addComponent(jPanel4, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

                        .addGap(18, 18, 18)

                        .addComponent(jButton1)))

                .addGap(55, 55, 55)

                .addComponent(jLabel7, javax.swing.GroupLayout.PREFERRED_SIZE, 29,
javax.swing.GroupLayout.PREFERRED_SIZE))

              .addComponent(jLabel8, javax.swing.GroupLayout.PREFERRED_SIZE, 236,
javax.swing.GroupLayout.PREFERRED_SIZE))

            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

            .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 579,
javax.swing.GroupLayout.PREFERRED_SIZE)

            .addContainerGap(93, Short.MAX_VALUE))

        );



        javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

        jPanel1.setLayout(jPanel1Layout);

        jPanel1Layout.setHorizontalGroup(

            jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

            .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                .addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

                .addContainerGap())

        );

        jPanel1Layout.setVerticalGroup(

            jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

            .addGroup(jPanel1Layout.createSequentialGroup()

                .addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 523, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE, 702,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 0, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void clusterButtonActionPerformed(java.awt.event.ActionEvent evt) {

//results cluster1 = new results();
//cluster1.setDefaultCloseOperation(WIDTH);
//cluster1.setSize(275, 100);
//cluster1.setVisible(true);
//Heart Dataset selection processing

        if (dataBox.getSelectedItem().equals("Vote Dataset")){
```

```
String newline = "\n";
try{
 data = FileHandler.loadDataset(new File("C:\\dataset\\voters.dat"),4,",");
}
catch(IOException e){
  JOptionPane.showMessageDialog(this," Could not resolve dataset address");
}

outputArea.append("--------------------------------------------Swarm Intelligent Clustering Algorithm OUTPUT-----
-----------------------------------------"+newline);
//outputArea.setText(data.toString());
Random random = new Random();
    Integer k =  random.nextInt(100) + 1;
   kField.setText(k.toString());

  Clusterer km = new KMeans(k);
    // perform the clustering by calling the cluster method and passing the
// dataset as a parameter and storing the clusters in the dataset array
    Dataset[] clusters = km.cluster(data);
    // Write out the number of clusters
 for (int i=0; i< clusters.length; i++){
   Dataset tempV = clusters[i];
   int count = 1;
   count = count + i;

outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);

for (Dataset cluster : clusters) {
  outputArea.append(cluster.toString() + newline);
}
outputArea.append(newline);
}
/* Create a measure for the cluster quality */
```

```
ClusterEvaluation sse= new SumOfSquaredErrors();
/* Measure the quality of the clustering */
Double score=sse.score(clusters);
score = score/1000000000;
        // output the sum of suare error
// System.out.println(score);
ssError.setText(score.toString());
 Double accuracy = 1- score;
DecimalFormat f = new DecimalFormat("##.0000");
        entropy.setText(f.format(accuracy).toString());
outputArea.setCaretPosition(outputArea.getDocument().getLength());
 }

 if (dataBox.getSelectedItem().equals("Iris Dataset")){
    String newline = "\n";
try{
 data = FileHandler.loadDataset(new File("C:\\dataset\\iris.data"),4,",");
}
catch(IOException e){
   JOptionPane.showMessageDialog(this," Could not resolve dataset address");
}
outputArea.append("----------------------------------------------Swarm Intelligent Clustering Algorithm OUTPUT----------
----------------------------------------"+newline);
//outputArea.setText(data.toString());
Random random = new Random();
        Integer k =  random.nextInt(100) + 1;
       kField.setText(k.toString());

  Clusterer km = new KMeans(k);
      // perform the clustering by calling the cluster method and passing the
// dataset as a parameter and storing the clusters in the dataset array
        Dataset[] clusters = km.cluster(data);
        // Write out the number of clusters
```

```
 for (int i=0; i< clusters.length; i++){
     Dataset tempV = clusters[i];
     int count = 1;
     count = count + i;
outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);
     for (Dataset cluster : clusters) {
   outputArea.append(cluster.toString() + newline);
 }
 outputArea.append(newline);
 }
 /* Create a measure for the cluster quality */
 ClusterEvaluation sse= new SumOfSquaredErrors();
 /* Measure the quality of the clustering */
 Double score=sse.score(clusters);
 score = score/1000000000;
        // output the sum of suare error
// System.out.println(score);
 ssError.setText(score.toString());
  Double accuracy = 1- score;
DecimalFormat f = new DecimalFormat("##.0000");
        entropy.setText(f.format(accuracy).toString());
 outputArea.setCaretPosition(outputArea.getDocument().getLength());
 }
 if (dataBox.getSelectedItem().equals("Statlog Heart")){
    String newline = "\n";
 try{
  data = FileHandler.loadDataset(new File("C:\\dataset\\statlog.dat"),4,",");
}
catch(IOException e){
   JOptionPane.showMessageDialog(this," Could not resolve dataset address");
}
```

```
        putArea.append("---------------------------------------------Swarm Intelligent Clustering Algorithm OUTPUT---------
--------------------------------------"+newline);

        //outputArea.setText(data.toString());

        Random random = new Random();

              Integer k =  random.nextInt(100) + 1;

              kField.setText(k.toString());

          Clusterer km = new KMeans(k);

              // perform the clustering by calling the cluster method and passing the

        // dataset as a parameter and storing the clusters in the dataset array

               Dataset[] clusters = km.cluster(data);

               // Write out the number of clusters

         for (int i=0; i< clusters.length; i++){

              Dataset tempV = clusters[i];

              int count = 1;

              count = count + i;

outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);


        for (Dataset cluster : clusters) {

           outputArea.append(cluster.toString() + newline);

        }

        outputArea.append(newline);

        }

        /* Create a measure for the cluster quality */

        ClusterEvaluation sse= new SumOfSquaredErrors();

        /* Measure the quality of the clustering */

        Double score=sse.score(clusters);

        score = score/1000000000;

              // output the sum of suare error

       // System.out.println(score);

        ssError.setText(score.toString());

         Double accuracy = 1- score;

      DecimalFormat f = new DecimalFormat("##.0000");
```

```
            entropy.setText(f.format(accuracy).toString());

        outputArea.setCaretPosition(outputArea.getDocument().getLength());

        }

        if (dataBox.getSelectedItem().equals("Australia Dataset")){

            String newline = "\n";

        try{

         data = FileHandler.loadDataset(new File("C:\\dataset\\Australia.dat"),4,",");

       }

       catch(IOException e){

          JOptionPane.showMessageDialog(this," Could not resolve dataset address");

       }

            outputArea.append("---------------------------------------------Swarm Intelligent Clustering Algorithm
OUTPUT-----------------------------------------------"+newline);

        //outputArea.setText(data.toString());

        Random random = new Random();


        Integer k =  random.nextInt(100) + 1;

              kField.setText(k.toString());

           Clusterer km = new KMeans(k);

                // perform the clustering by calling the cluster method and passing the

        // dataset as a parameter and storing the clusters in the dataset array

                Dataset[] clusters = km.cluster(data);

                // Write out the number of clusters

         for (int i=0; i< clusters.length; i++){

              Dataset tempV = clusters[i];

              int count = 1;

              count = count + i;

outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);

              for (Dataset cluster : clusters) {

           outputArea.append(cluster.toString() + newline);

        }

        outputArea.append(newline);
```

```
    }
    /* Create a measure for the cluster quality */
    ClusterEvaluation sse= new SumOfSquaredErrors();
    /* Measure the quality of the clustering */
    Double score=sse.score(clusters);
    score = score/1000000000;
          // output the sum of suare error
   // System.out.println(score);
    ssError.setText(score.toString());
     Double accuracy = 1- score;
  DecimalFormat f = new DecimalFormat("##.0000");
          entropy.setText(f.format(accuracy).toString());
    outputArea.setCaretPosition(outputArea.getDocument().getLength());
    }
    //Heart Dataset selection processing
    if (dataBox.getSelectedItem().equals("German_Credit_Dataset")){
       String newline = "\n";
    try{
     data = FileHandler.loadDataset(new File("C:\\dataset\\German_Creadit_Card.dat"),4,",");
  }
  catch(IOException e){
     JOptionPane.showMessageDialog(this," Could not resolve dataset address");
  }
          outputArea.append("----------------------------------------------Swarm Intelligent Clustering Algorithm
OUTPUT----------------------------------------------"+newline);
    //outputArea.setText(data.toString());
    Random random = new Random();
    Integer k =  random.nextInt(100) + 1;
        kField.setText(k.toString());
    Clusterer km = new KMeans(k);
          // perform the clustering by calling the cluster method and passing the
    // dataset as a parameter and storing the clusters in the dataset array
```

```
Dataset[] clusters = km.cluster(data);

// Write out the number of clusters
 for (int i=0; i< clusters.length; i++){
      Dataset tempV = clusters[i];
      int count = 1;
      count = count + i;

outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);

for (Dataset cluster : clusters) {
   outputArea.append(cluster.toString() + newline);
}
outputArea.append(newline);
}
/* Create a measure for the cluster quality */
ClusterEvaluation sse= new SumOfSquaredErrors();
/* Measure the quality of the clustering */
Double score=sse.score(clusters);
score = score/1000000000;
       // output the sum of suare error
// System.out.println(score);
ssError.setText(score.toString());
 Double accuracy = 1- score;
DecimalFormat f = new DecimalFormat("##.0000");
        entropy.setText(f.format(accuracy).toString());
outputArea.setCaretPosition(outputArea.getDocument().getLength());
}
      //Heart Dataset selection processing
if (dataBox.getSelectedItem().equals("Heart Dataset")){
    String newline = "\n";
try{
 data = FileHandler.loadDataset(new File("C:\\dataset\\heart.data"),4,",");
```

```
       }
       catch(IOException e){
          JOptionPane.showMessageDialog(this," Could not resolve dataset address");
       }
 outputArea.append("--------------------------------------------Swarm Intelligent Clustering Algorithm OUTPUT----------
------------------------------------"+newline);
        //outputArea.setText(data.toString());
        Random random = new Random();

        Integer k =  random.nextInt(100) + 1;
            kField.setText(k.toString());

        Clusterer km = new KMeans(k);
             // perform the clustering by calling the cluster method and passing the
        // dataset as a parameter and storing the clusters in the dataset array

        Dataset[] clusters = km.cluster(data);
               // Write out the number of clusters
         for (int i=0; i< clusters.length; i++){
             Dataset tempV = clusters[i];
             int count = 1;
             count = count + i;

outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);
            for (Dataset cluster : clusters) {
          outputArea.append(cluster.toString() + newline);
        }
        outputArea.append(newline);
        }
        /* Create a measure for the cluster quality */
        ClusterEvaluation sse= new SumOfSquaredErrors();
        /* Measure the quality of the clustering */
        Double score=sse.score(clusters);
```

```
 score = score/1000000;
        // output the sum of suare error
// System.out.println(score);
 ssError.setText(score.toString());
  Double accuracy = 1- score;
DecimalFormat f = new DecimalFormat("##.0000");
        entropy.setText(f.format(accuracy).toString());
 outputArea.setCaretPosition(outputArea.getDocument().getLength());
 }
 //iris selection processing
 if (dataBox.getSelectedItem().equals("Hepititis Dataset")){
    String newline = "\n";
 try{
  data = FileHandler.loadDataset(new File("C:\\dataset\\hepatitis.dat"),4,",");
}
catch(IOException e){
   JOptionPane.showMessageDialog(this," Could not resolve dataset address");
}
        outputArea.append("----------------------------------------------Swarm Intelligent Clustering Algorithm
OUTPUT----------------------------------------------"+newline);
 //outputArea.setText(data.toString());
 Random random = new Random();
        Integer k =  random.nextInt(100) + 1;
      kField.setText(k.toString());
  Clusterer km = new KMeans(k);
        // perform the clustering by calling the cluster method and passing the
 // dataset as a parameter and storing the clusters in the dataset array
    Dataset[] clusters = km.cluster(data);
   // Write out the number of clusters
 for (int i=0; i< clusters.length; i++){
      Dataset tempV = clusters[i];
      int count = 1;
      count = count + i;
```

```
outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);

      for (Dataset cluster : clusters) {
         outputArea.append(cluster.toString() + newline);
      }
      outputArea.append(newline);
      }
      /* Create a measure for the cluster quality */
      ClusterEvaluation sse= new SumOfSquaredErrors();
      /* Measure the quality of the clustering */
      Double score=sse.score(clusters);
      score = score/10000000;

      // output the sum of suare error
     // System.out.println(score);
      ssError.setText(score.toString());
       Double accuracy = 1- score;
     DecimalFormat f = new DecimalFormat("##.0000");
             entropy.setText(f.format(accuracy).toString());
      outputArea.setCaretPosition(outputArea.getDocument().getLength());
      }

      // soyabean selection processing
      if (dataBox.getSelectedItem().equals("Soybean Dataset")){
       String newline = "\n";
      try{
       data = FileHandler.loadDataset(new File("C:\\dataset\\soybean.data"),4,",");
    }
    catch(IOException e){
       JOptionPane.showMessageDialog(this," Could not resolve dataset address");
    }
```

```
        outputArea.append("-----------------------------------------------Swarm Intelligent Clustering Algorithm OUTPUT-----
-------------------------------------------"+newline);

        //outputArea.setText(data.toString());

        Random random = new Random();


        Integer k =  random.nextInt(100) + 1;



       kField.setText(k.toString());



        Clusterer km = new KMeans(k);


        // perform the clustering by calling the cluster method and passing the

        // dataset as a parameter and storing the clusters in the dataset array


        Dataset[] clusters = km.cluster(data);


        // Write out the number of clusters

         for (int i=0; i< clusters.length; i++){

             Dataset tempV = clusters[i];

             int count = 1;

             count = count + i;


outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);

             for (Dataset cluster : clusters) {

           outputArea.append(cluster.toString() + newline);

        }

        outputArea.append(newline);

        }

        /* Create a measure for the cluster quality */

        ClusterEvaluation sse= new SumOfSquaredErrors();

        /* Measure the quality of the clustering */
```

```
 Double score=sse.score(clusters);
 score = score/100;
        // output the sum of suare error
// System.out.println(score);
 ssError.setText(score.toString());
  Double accuracy = 1- score;
DecimalFormat f = new DecimalFormat("##.0000");
entropy.setText(f.format(accuracy).toString());
 outputArea.setCaretPosition(outputArea.getDocument().getLength());
 }
        else if (dataBox.getSelectedItem().equals("Wine Dataset")){
          String newline = "\n";
 try{
  data = FileHandler.loadDataset(new File("C:\\dataset\\wine.data"),4,",");
}
catch(IOException e){
   JOptionPane.showMessageDialog(this," Could not resolve dataset address");
}
 outputArea.append("----------------------------------------------Swarm Intelligent Clustering Algorithm OUTPUT-----
-------------------------------------------"+newline);
 //outputArea.setText(data.toString());
 Random random = new Random();

 Integer k =  random.nextInt(100) + 1;

kField.setText(k.toString());
 Clusterer km = new KMeans(k);
        // perform the clustering by calling the cluster method and passing the
 // dataset as a parameter and storing the clusters in the dataset array

 Dataset[] clusters = km.cluster(data);

 // Write out the number of clusters
```

```
 for (int i=0; i< clusters.length; i++){
      Dataset tempV = clusters[i];
      int count = 1;
      count = count + i;

outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);

    for (Dataset cluster : clusters) {
       outputArea.append(cluster.toString() + newline);
    }
    outputArea.append(newline);
    }
    /* Create a measure for the cluster quality */
    ClusterEvaluation sse= new SumOfSquaredErrors();
    /* Measure the quality of the clustering */
    Double score=sse.score(clusters);
    score = score/1000000;
     // output the sum of suare error
   // System.out.println(score);
    ssError.setText(score.toString());
     Double accuracy = 1- score;
  DecimalFormat f = new DecimalFormat("##.0000");
         entropy.setText(f.format(accuracy).toString());
    outputArea.setCaretPosition(outputArea.getDocument().getLength());
    }
    else if (dataBox.getSelectedItem().equals("Yeast Dataset")){
                        String newline = "\n";
    try{
     data = FileHandler.loadDataset(new File("C:\\dataset\\yeast.data"),4,",");
   }
  catch(IOException e){
     JOptionPane.showMessageDialog(this," Could not resolve dataset address");
   }
```

```
        outputArea.append("----------------------------------------------Swarm Intelligent Clustering Algorithm OUTPUT-----
------------------------------------------"+newline);

        //outputArea.setText(data.toString());

        Random random = new Random();

                Integer k =  random.nextInt(100) + 1;

              kField.setText(k.toString());

           Clusterer km = new KMeans(k);

                // perform the clustering by calling the cluster method and passing the

        // dataset as a parameter and storing the clusters in the dataset array

                Dataset[] clusters = km.cluster(data);

                // Write out the number of clusters

         for (int i=0; i< clusters.length; i++){

              Dataset tempV = clusters[i];

              int count = 1;

              count = count + i;

outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*><*>>>>"+newline);

           for (Dataset cluster : clusters) {

           outputArea.append(cluster.toString() + newline);

        }

        outputArea.append(newline);

        }

        /* Create a measure for the cluster quality */

        ClusterEvaluation sse= new SumOfSquaredErrors();

        /* Measure the quality of the clustering */

        Double score=sse.score(clusters);

        score = score/100;

                // output the sum of suare error

       // System.out.println(score);

        ssError.setText(score.toString());

        Double accuracy = 1- score;

      DecimalFormat f = new DecimalFormat("##.0000");

                entropy.setText(f.format(accuracy).toString());
```

```
outputArea.setCaretPosition(outputArea.getDocument().getLength());
}
else{
   JOptionPane.showMessageDialog(this, "Dataset has been selected");
}

if (dataBox.getSelectedItem().equals("Boko Dataset")){
                String newline = "\n";
try{
 data = FileHandler.loadDataset(new File("C:\\dataset\\BHaram.data"),4,",");
  // data = FileHandler.loadDataset(new File("C:\\dataset\\voters.dat"),4,",");
 //data = FileHandler.loadDataset(new File("C:\\dataset\\yeast.data"),4,",");
}
catch(IOException e){
  JOptionPane.showMessageDialog(this," Could not resolve dataset address");
}
 outputArea.append("----------------------------------------------Swarm Intelligent Clustering Algorithm OUTPUT----
----------------------------------------------"+newline);
//outputArea.setText(data.toString());
Random random = new Random();

Integer k =  random.nextInt(100) + 1;

kField.setText(k.toString());

Clusterer km = new KMeans(k);
      // perform the clustering by calling the cluster method and passing the
// dataset as a parameter and storing the clusters in the dataset array
       Dataset[] clusters = km.cluster(data);

// Write out the number of clusters
 for (int i=0; i< clusters.length; i++){
     Dataset tempV = clusters[i];
```

```
            int count = 1;

            count = count + i;

outputArea.append("<*>><*><*><*><*>><*><*><*><*><*><*><*><*>Cluster"+count+"<*><*>><*><*><*><*>
><*><*><*><*><*><*><*><*><*><*>>>>"+newline);

            for (Dataset cluster : clusters) {

          outputArea.append(cluster.toString() + newline);

        }

        outputArea.append(newline);

        }

        /* Create a measure for the cluster quality */

        ClusterEvaluation sse= new SumOfSquaredErrors();

        /* Measure the quality of the clustering */

        Double score=sse.score(clusters);

        score = score/100;


        // output the sum of suare error

       // System.out.println(score);

        ssError.setText(score.toString());

        Double accuracy = 1- score;

      DecimalFormat f = new DecimalFormat("##.0000");

             entropy.setText(f.format(accuracy).toString());

        outputArea.setCaretPosition(outputArea.getDocument().getLength());

        }

        else{

           JOptionPane.showMessageDialog(this, "Dataset has been selected");

        }

    }


    private void dataBoxActionPerformed(java.awt.event.ActionEvent evt) {

        // TODO add your handling code here:

    }


    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        // TODO add your handling code here:
        outputArea.setText("");
        entropy.setText("");
        kField.setText("");
        entropy.setText("");
        ssError.setText("");

    }
    public static void main(String args[]) {
        /*
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(UserInterface.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(UserInterface.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(UserInterface.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(UserInterface.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
        }
        //</editor-fold>

        /* Create and display the form */
```

```
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new UserInterface().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton clusterButton;
    private javax.swing.JComboBox dataBox;
    private javax.swing.JTextField entropy;
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JPanel jPanel5;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextField kField;
    private javax.swing.JTextArea outputArea;
    private javax.swing.JTextField ssError;
    // End of variables declaration
}
“
```

“

```
/*----------Centroid.java----------*/

package org.c4s.algorithm.cluster;

/**
 * This class represents the Centroid for a Cluster. The initial centroid is calculated
 * using a equation which divides the sample space for each dimension into equal parts
 * depending upon the value of k.

 */

class Centroid {
    private double mCx, mCy;
    private Cluster mCluster;

    public Centroid(double cx, double cy) {
        this.mCx = cx;
        this.mCy = cy;
    }

    public void calcCentroid() { //only called by CAInstance
        int numDP = mCluster.getNumDataPoints();
        double tempX = 0, tempY = 0;
        int i;
        //caluclating the new Centroid
        for (i = 0; i < numDP; i++) {
            tempX = tempX + mCluster.getDataPoint(i).getX();
            //total for x
            tempY = tempY + mCluster.getDataPoint(i).getY();
            //total for y
        }
        this.mCx = tempX / numDP;
```

```
        this.mCy = tempY / numDP;
        //calculating the new Euclidean Distance for each Data Point
        tempX = 0;
        tempY = 0;
        for (i = 0; i < numDP; i++) {
            mCluster.getDataPoint(i).calcEuclideanDistance();
        }
        //calculate the new Sum of Squares for the Cluster
        mCluster.calcSumOfSquares();
    }

    public void setCluster(Cluster c) {
        this.mCluster = c;
    }

    public double getCx() {
        return mCx;
    }

    public double getCy() {
        return mCy;
    }

    public Cluster getCluster() {
        return mCluster;
    }

}
“

“
/*-----------------Cluster.java----------------*/
```

```
package org.c4s.algorithm.cluster;

import java.util.Vector;

/**
 * This class represents a Cluster in a Cluster Analysis Instance. A Cluster is associated
 * with one and only one JCA Instance. A Cluster is related to more than one DataPoints and
 * one centroid.

 * @see DataPoint
 * @see Centroid
 */
class Cluster {
    private String mName;
    private Centroid mCentroid;
    private double mSumSqr;
    private Vector mDataPoints;
// private Cluster mCluster;
    public Cluster(String name) {
        this.mName = name;
        this.mCentroid = null; //will be set by calling setCentroid()
        mDataPoints = new Vector();
    }

    public void setCentroid(Centroid c) {
        mCentroid = c;
    }

    public Centroid getCentroid() {
        return mCentroid;
    }

    public void addDataPoint(DataPoint dp) { //called from CAInstance
```

```
        dp.setCluster(this); //initiates a inner call to
 dp.calcEuclideanDistance();
        this.mDataPoints.addElement(dp);
        calcSumOfSquares();
    }

    public void removeDataPoint(DataPoint dp) {
        this.mDataPoints.removeElement(dp);
        calcSumOfSquares();
    }
    public int getNumDataPoints() {
        return this.mDataPoints.size();
    }
    public DataPoint getDataPoint(int pos) {
        return (DataPoint) this.mDataPoints.elementAt(pos);
    }
    public void calcSumOfSquares() { //called from Centroid
        int size = this.mDataPoints.size();
        double temp = 0;
        for (int i = 0; i < size; i++) {
            temp = temp + ((DataPoint)
this.mDataPoints.elementAt(i)).getCurrentEuDt();
        }
        this.mSumSqr = temp;
    }
    public double getSumSqr() {
        return this.mSumSqr;
    }
    public String getName() {
        return this.mName;
    }
    public Vector getDataPoints() {
        return this.mDataPoints;
```

```
    }

}
“

“
/*----------------DataPoint.java----------------*/
package org.c4s.algorithm.cluster;
public class DataPoint {
    private double mX,mY;
    private String mObjName;
    private Cluster mCluster;
    private double mEuDt;
    public DataPoint(double x, double y, String name) {
        this.mX = x;
        this.mY = y;
        this.mObjName = name;
        this.mCluster = null;
    }
    public void setCluster(Cluster cluster) {
        this.mCluster = cluster;
        calcEuclideanDistance();
    }
    public void calcEuclideanDistance() {
    //called when DP is added to a cluster or when a Centroid is recalculated.
        mEuDt = Math.sqrt(Math.pow((mX - mCluster.getCentroid().getCx()),
2) + Math.pow((mY - mCluster.getCentroid().getCy()), 2));
    }

    public double testEuclideanDistance(Centroid c) {
        return Math.sqrt(Math.pow((mX - c.getCx()), 2) + Math.pow((mY - c.getCy()), 2));
    }
```

```
    public double getX() {
        return mX;
    }

    public double getY() {
        return mY;
    }

    public Cluster getCluster() {
        return mCluster;
    }

    public double getCurrentEuDt() {
        return mEuDt;
    }

    public String getObjName() {
        return mObjName;
    }

}
“

“
package org.c4s.algorithm.cluster;
import java.util.Vector;
public class JCA {
    private Cluster[] clusters;
    private int miter;
    private Vector mDataPoints = new Vector();
    private double mSWCSS;
//int k= Integer.parseInt(best.text1.getText());
    public JCA(int k, int iter, Vector dataPoints) {
```

```
    clusters = new Cluster[k];
    for (int i = 0; i < k; i++) {
        clusters[i] = new Cluster("Cluster" + i);
    }
    this.miter = iter;
    this.mDataPoints = dataPoints;
}
private void calcSWCSS() {
    double temp = 0;
    for (int i = 0; i < clusters.length; i++) {
        temp = temp + clusters[i].getSumSqr();
    }
    mSWCSS = temp;
}

public void startAnalysis() {
    //set Starting centroid positions - Start of Step 1
    setInitialCentroids();
    int n = 0;
    //assign DataPoint to clusters
    loop1: while (true) {
        for (int l = 0; l < clusters.length; l++)
        {
            clusters[l].addDataPoint((DataPoint)mDataPoints.elementAt(n));
            n++;
            if (n >= mDataPoints.size())
                break loop1;
        }
    }
    //calculate E for all the clusters
    calcSWCSS();
    //recalculate Cluster centroids - Start of Step 2
    for (int i = 0; i < clusters.length; i++) {
```

```
            clusters[i].getCentroid().calcCentroid();
        }

        //recalculate E for all the clusters
        calcSWCSS();
        for (int i = 0; i < miter; i++) {
            //enter the loop for cluster 1
            for (int j = 0; j < clusters.length; j++) {
                for (int k = 0; k < clusters[j].getNumDataPoints(); k++) {
                    //pick the first element of the first cluster
                    //get the current Euclidean distance
                    double tempEuDt = clusters[j].getDataPoint(k).getCurrentEuDt();
                    Cluster tempCluster = null;
                    boolean matchFoundFlag = false;

                    //call testEuclidean distance for all clusters
                    for (int l = 0; l < clusters.length; l++) {

                    //if testEuclidean < currentEuclidean then
                        if (tempEuDt > clusters[j].getDataPoint(k).testEuclideanDistance(clusters[l].getCentroid())) {
                            tempEuDt = clusters[j].getDataPoint(k).testEuclideanDistance(clusters[l].getCentroid());
                            tempCluster = clusters[l];
                            matchFoundFlag = true;
                        }
                        //if statement - Check whether the Last EuDt is > Present EuDt
                        }
//for variable 'l' - Looping between different Clusters for matching a Data Point.
//add DataPoint to the cluster and calcSWCSS
        if (matchFoundFlag) {
                            tempCluster.addDataPoint(clusters[j].getDataPoint(k));
                            clusters[j].removeDataPoint(clusters[j].getDataPoint(k));
                        for (int m = 0; m < clusters.length; m++) {
                            clusters[m].getCentroid().calcCentroid();
```

```
                    }
//for variable 'm' - Recalculating centroids for all Clusters


                    calcSWCSS();
                }
//if statement - A Data Point is eligible for transfer between Clusters.
            }
            //for variable 'k' - Looping through all Data Points of the current Cluster.
        }//for variable 'j' - Looping through all the Clusters.
    }//for variable 'i' - Number of iterations.
  }
  public Vector[] getClusterOutput() {
    Vector v[] = new Vector[clusters.length];
    for (int i = 0; i < clusters.length; i++) {
      v[i] = clusters[i].getDataPoints();
    }
    return v;
  }
  private void setInitialCentroids() {
    //kn = (round((max-min)/k)*n)+min where n is from 0 to (k-1).
    double cx = 0, cy = 0;
    for (int n = 1; n <= clusters.length; n++) {
      cx = (((getMaxXValue() - getMinXValue()) / (clusters.length + 1)) * n) + getMinXValue();
      cy = (((getMaxYValue() - getMinYValue()) / (clusters.length + 1)) * n) + getMinYValue();
      Centroid c1 = new Centroid(cx, cy);
      clusters[n - 1].setCentroid(c1);
      c1.setCluster(clusters[n - 1]);
    }
  }
  private double getMaxXValue() {
    double temp;
    temp = ((DataPoint) mDataPoints.elementAt(0)).getX();
    for (int i = 0; i < mDataPoints.size(); i++) {
```

```
        DataPoint dp = (DataPoint) mDataPoints.elementAt(i);
        temp = (dp.getX() > temp) ? dp.getX() : temp;
    }
    return temp;
}
private double getMinXValue() {
    double temp = 0;
    temp = ((DataPoint) mDataPoints.elementAt(0)).getX();
    for (int i = 0; i < mDataPoints.size(); i++) {
        DataPoint dp = (DataPoint) mDataPoints.elementAt(i);
        temp = (dp.getX() < temp) ? dp.getX() : temp;
    }
    return temp;
}
private double getMaxYValue() {
    double temp = 0;
    temp = ((DataPoint) mDataPoints.elementAt(0)).getY();
    for (int i = 0; i < mDataPoints.size(); i++) {
        DataPoint dp = (DataPoint) mDataPoints.elementAt(i);
        temp = (dp.getY() > temp) ? dp.getY() : temp;
    }
    return temp;
}
private double getMinYValue() {
    double temp = 0;
    temp = ((DataPoint) mDataPoints.elementAt(0)).getY();
    for (int i = 0; i < mDataPoints.size(); i++) {
        DataPoint dp = (DataPoint) mDataPoints.elementAt(i);
        temp = (dp.getY() < temp) ? dp.getY() : temp;
    }
    return temp;
}
```

```
    public int getKValue() {
        return clusters.length;
    }
    public int getIterations() {
        return miter;
    }
    public int getTotalDataPoints() {
        return mDataPoints.size();
    }
    public double getSWCSS() {
        return mSWCSS;
    }
    public Cluster getCluster(int pos) {
        return clusters[pos];
    }
}
“
private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JTextField put;
    // End of variables declaration
  public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
```

```
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(vivi.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(vivi.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(vivi.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(vivi.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new vivi().setVisible(true);
        }
    });
  }
}
```

APPENDICES B

Python Codes

```
# libraries
import numpy as np
import matplotlib.pyplot as plt

# dataset
clusters = [68.75,5.25,3.65,6.75,5.5,3.6,33,5]
area_attacks = ('Borno', 'Abuja', 'Adamawa', 'Gombe', 'Kano','Kastina','Maiduguri','Yobe')
clusters_percentage = [63.75,1.25,1.25,3.75,2.5,2.5,20,5]
bars = ('Borno', 'Abuja', 'Adamawa', 'Gombe', 'Kano','Kastina','Maiduguri','Yobe')
y_pos = np.arange(len(bars))

# Create bars and choose color
#plt.bar(y_pos, height, color = (0.5,0.1,0.5,0.6))

# Add title and axis names
#plt.title('')
#plt.xlabel('Area of Attacks')
#plt.ylabel('Percentage(%) of clusters outputs')

# Limits for the Y axis
plt.ylim(0,80)

# Create names
#plt.xticks(y_pos, bars)

# Show graphic
#plt.xticks(rotation=45)
#plt.show()

#next ploit graph
plt.plot( clusters_percentage, 'go-')
#plt.plot(clusters, 'b*--')
plt.xlabel('Area of Attacks')
plt.ylabel('Percentage(%) of clusters outputs')
plt.xticks(rotation=45)
#plt.show()
#plt.plot(height, 'ro-')
plt.plot(bars, height)
```

#PYTHON CODE for Comparison of existing machine learning algorithms and the proposed Dynamic-K-reference algorithms

```
# libraries
import numpy as np
import matplotlib.pyplot as plt

# Fake dataset
```

```
Iris = [0.0572,0.0429,0.0318,0.0159,0]
Wine = [0.0341,0.0254,0.0193,0.0104,0.0073]
Voting = [0.04791, 0.03468,0.02909,0.02135,0.0039]
#area_attacks = ('Borno', 'Abuja', 'Adamawa', 'Gombe', 'Kano','Kastina','Maiduguri','Yobe')

bars = ('KNN', 'POS', 'ABC', 'FABCO', 'DynamicK-reference')
# Limits for the Y axis
y=plt.ylim(0,0.07)

#legend plotting
plt.plot(Iris, 'go-', label='Iris')
plt.plot(Wine, 'ro-', label='Wine')
plt.plot(Voting, 'b*--', label='Voting')
plt.legend(loc='upper right')
plt.plot(bars, Voting)
#next ploit graph
plt.xlabel('Algorithms')
plt.ylabel(Clustering Error')
plt.xticks(rotation=45)
```

APPENDICES C

PROCEDURES OF DATA SCRAPING

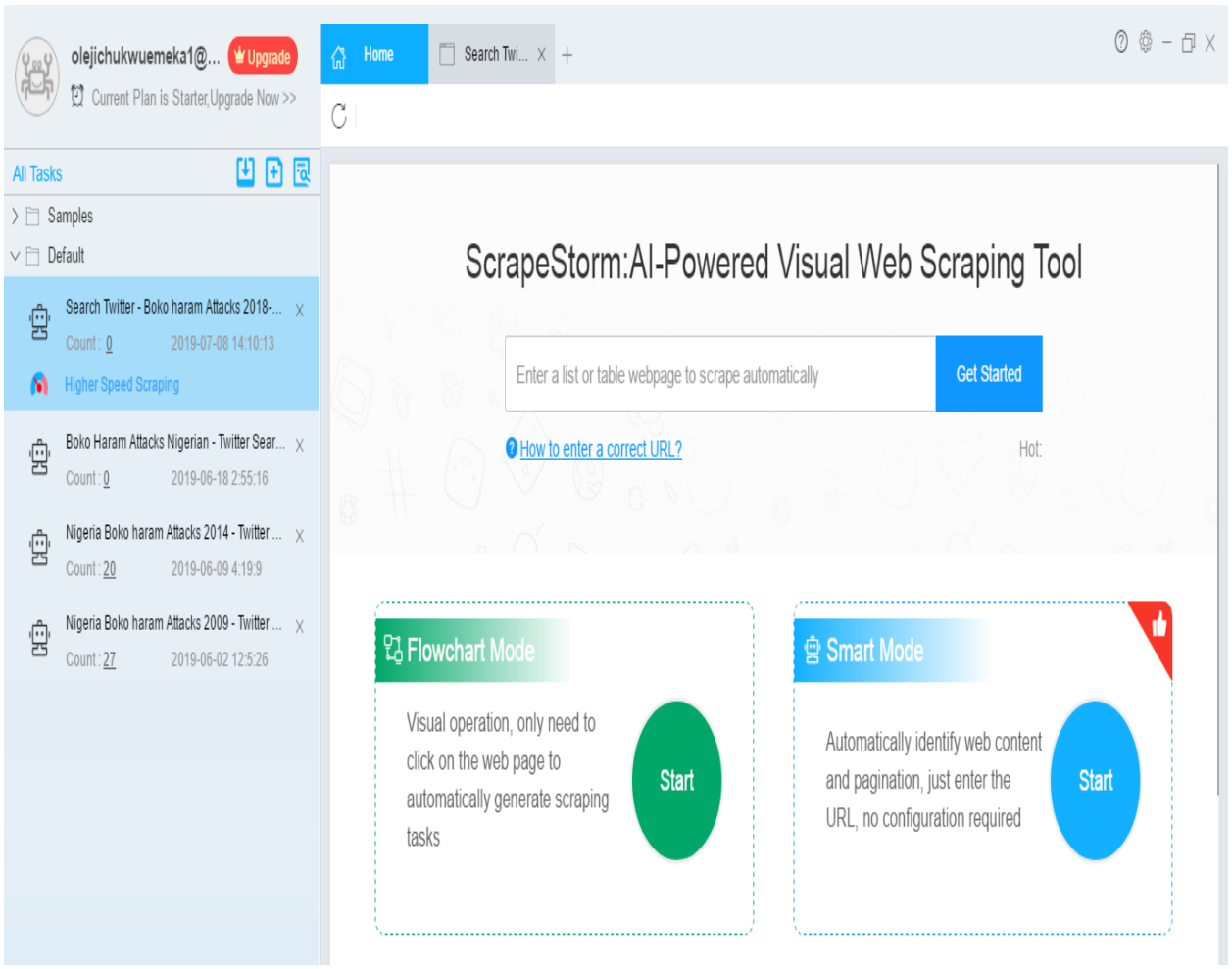

Figure C1: User Interface of Data Scraping from Social Media

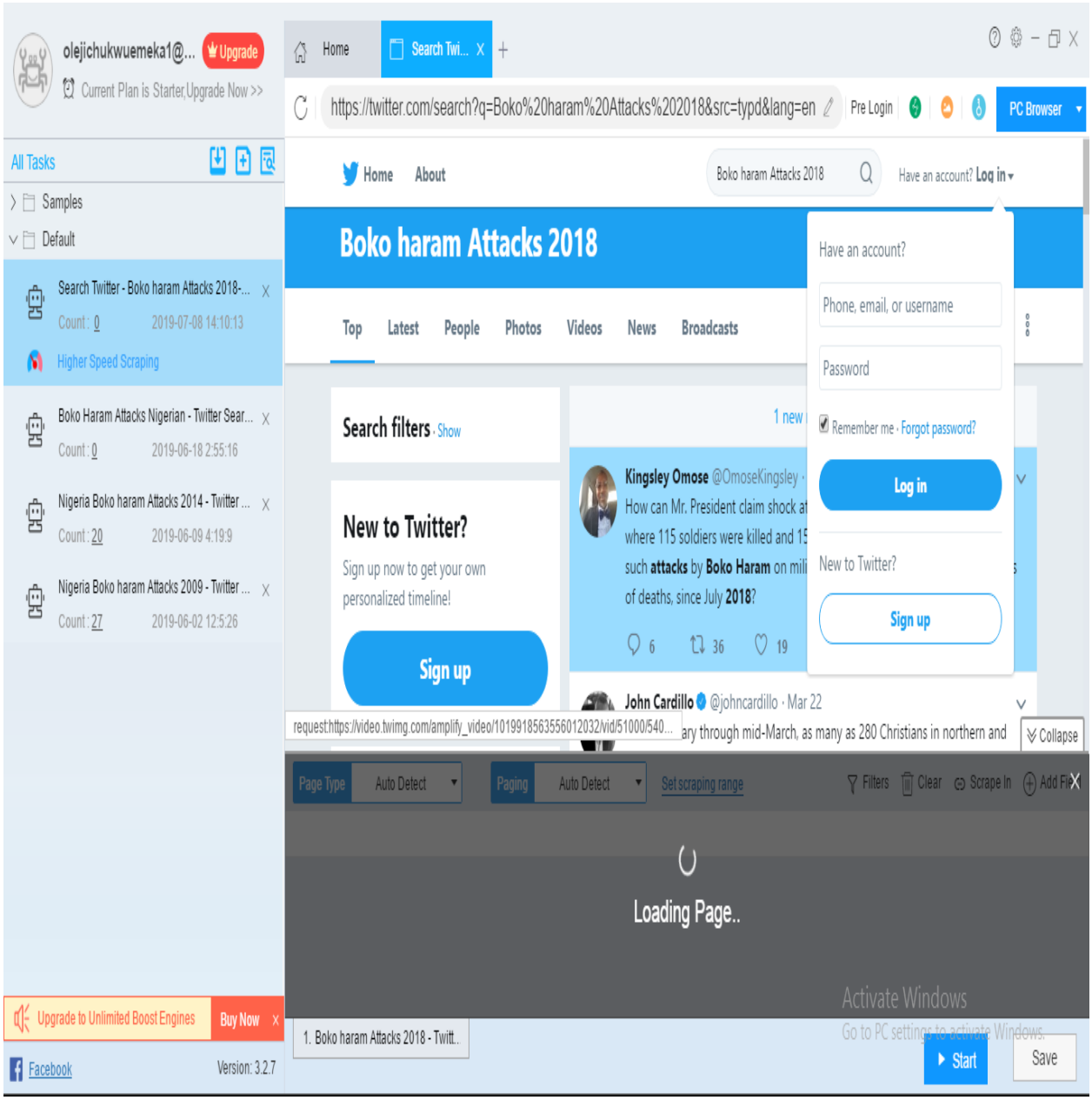

Figure C2: Data Scraping of Boko Haram Insurgency Attack on 2018 from Twitter Web Site

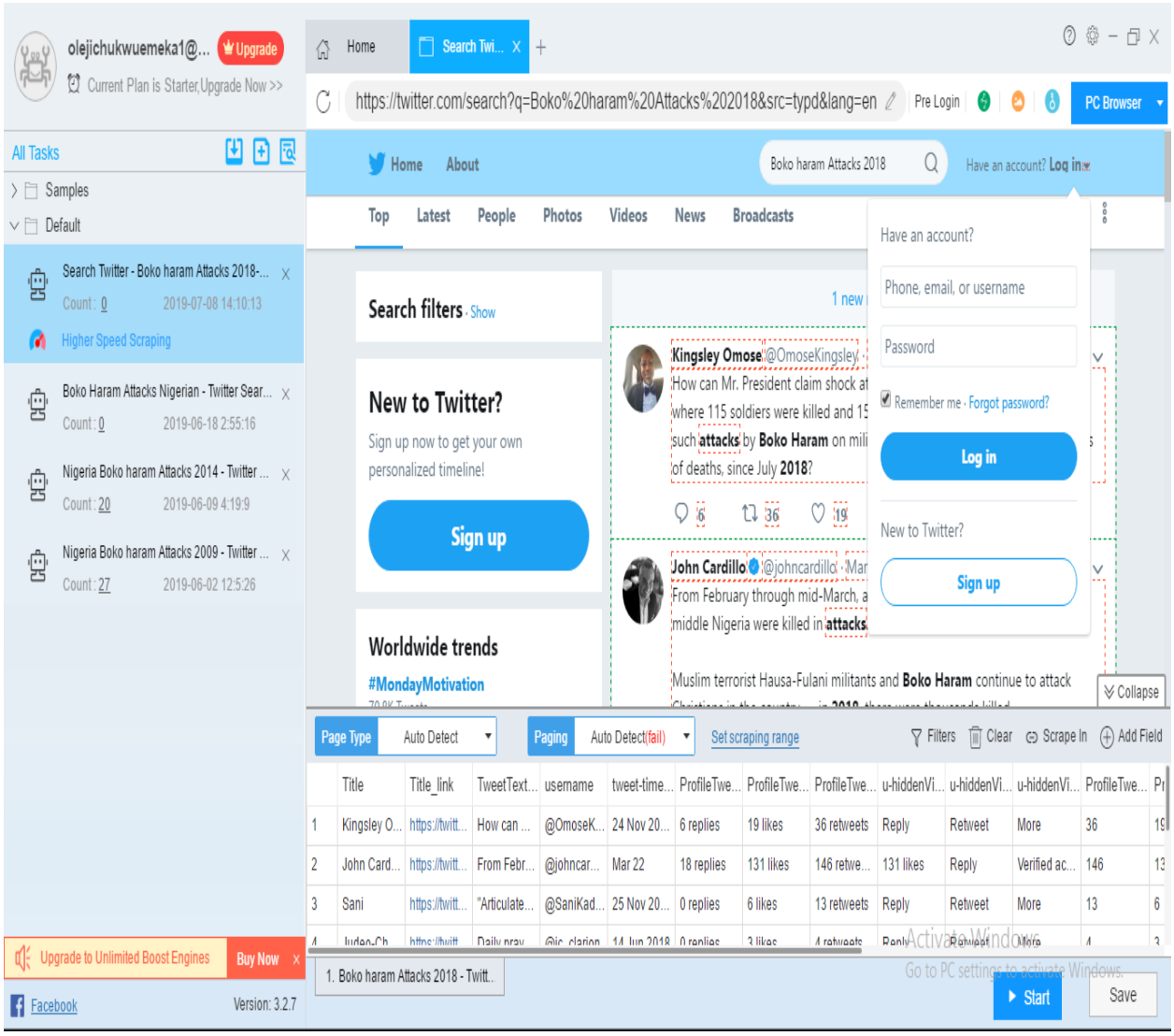

Figure C3: Data Scraping of Boko Haram Insurgency Attack on 2018 from Twitter Web Site showing the field of the records

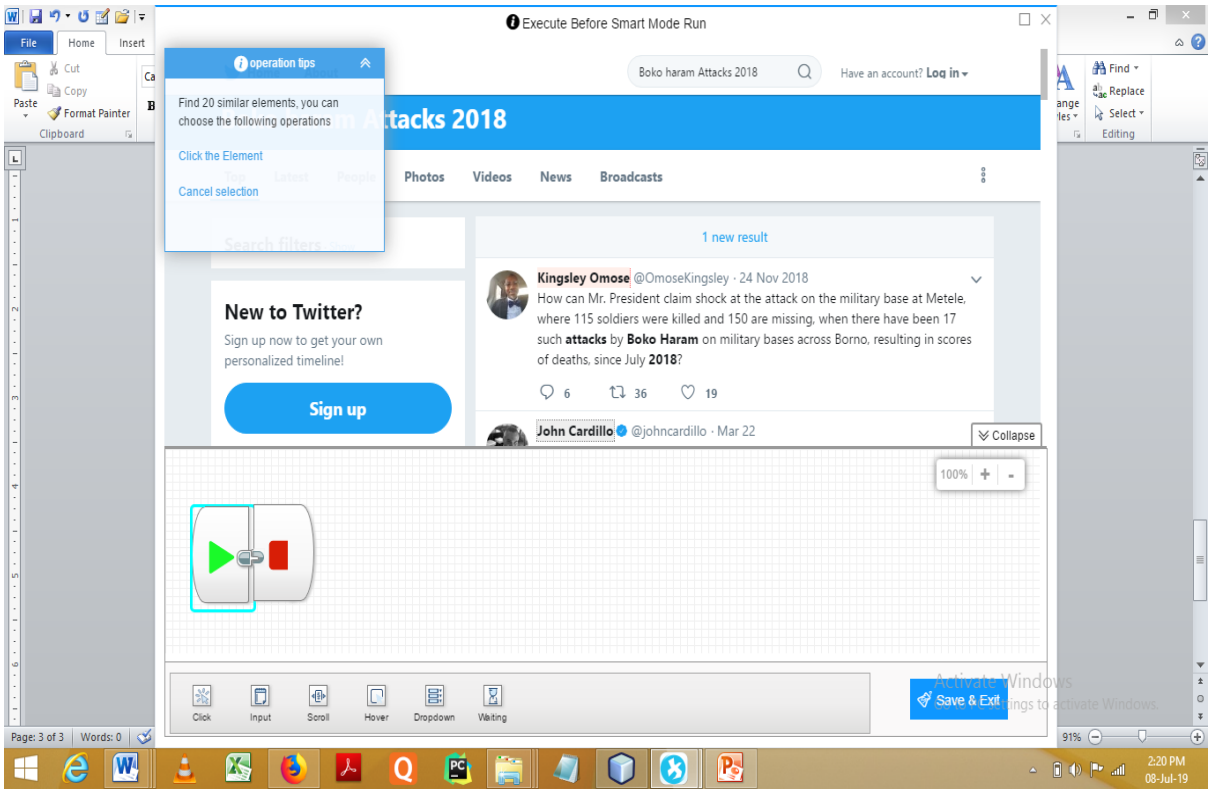

Figure C4: Adding elements to Data Scraping

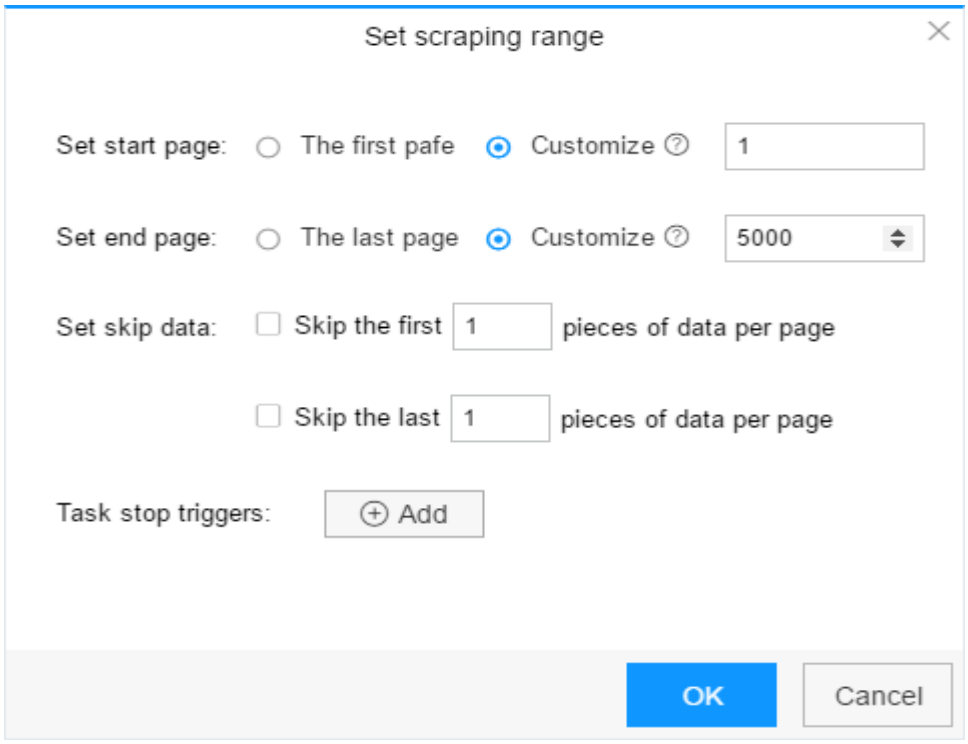

Figure C5: Data Scraping Range

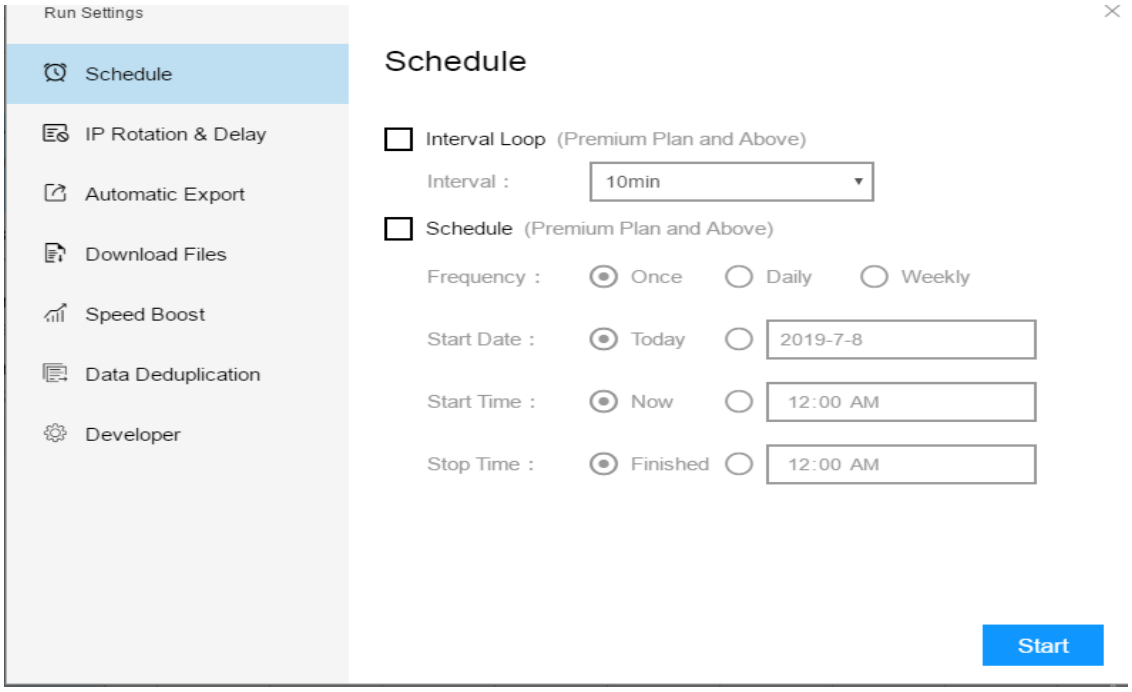

Figure C6: Data Scraping Schedule

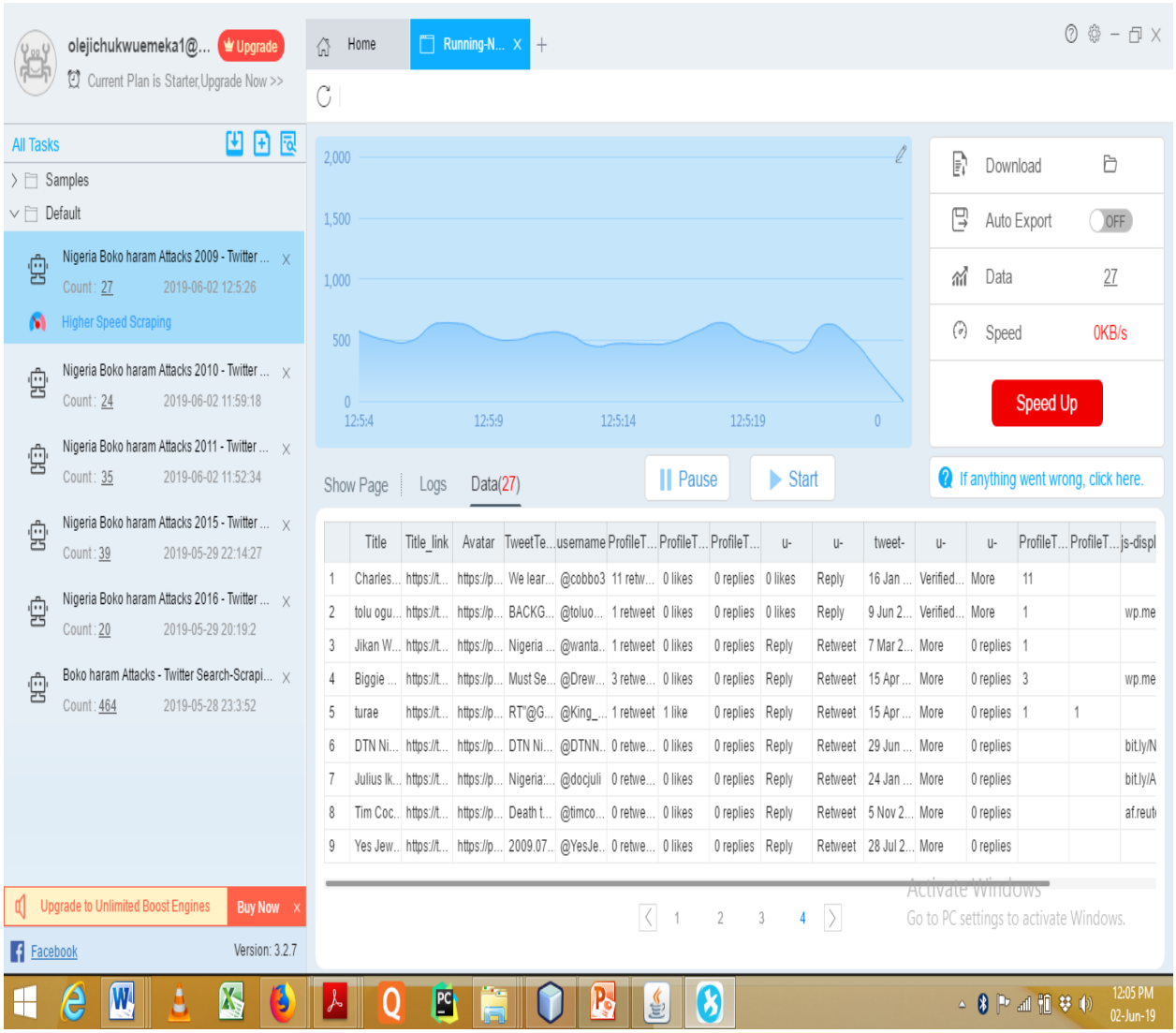

Figure C7: Data Scraping in progress

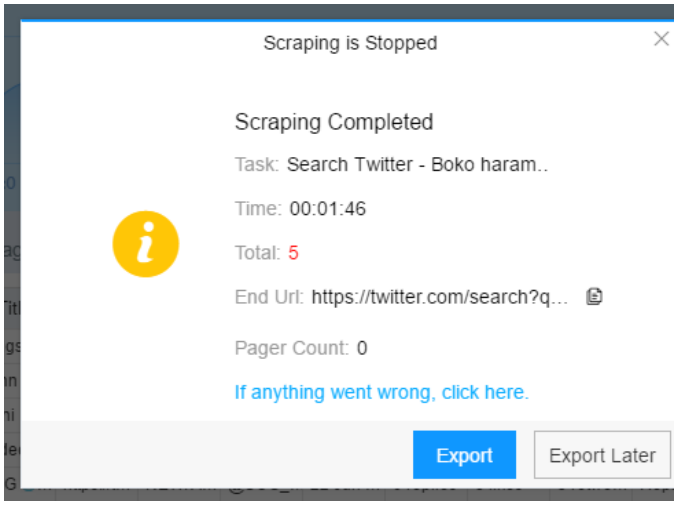

Figure C8: Exporting Scraping data

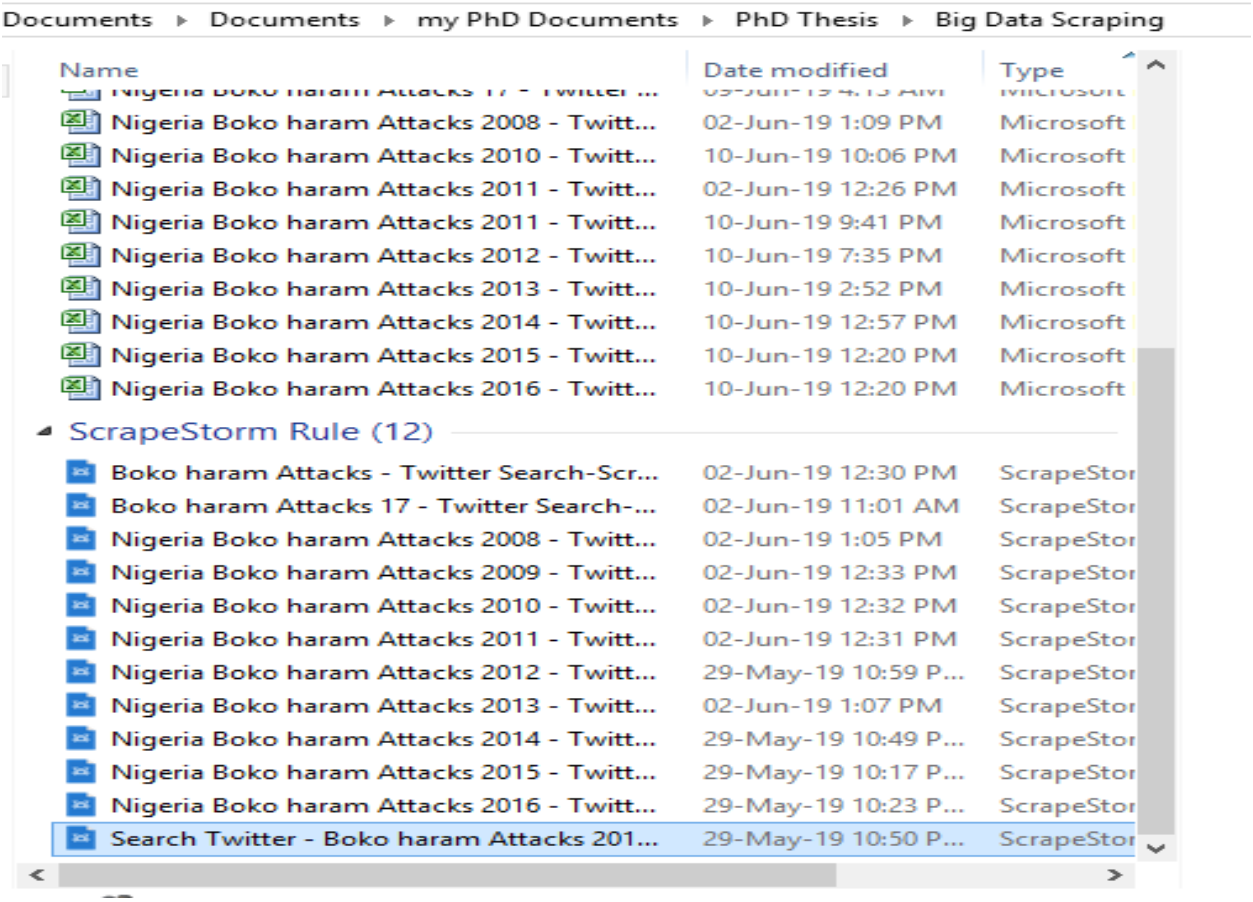

Figure C9: Exports of Data Scraping to Excel.

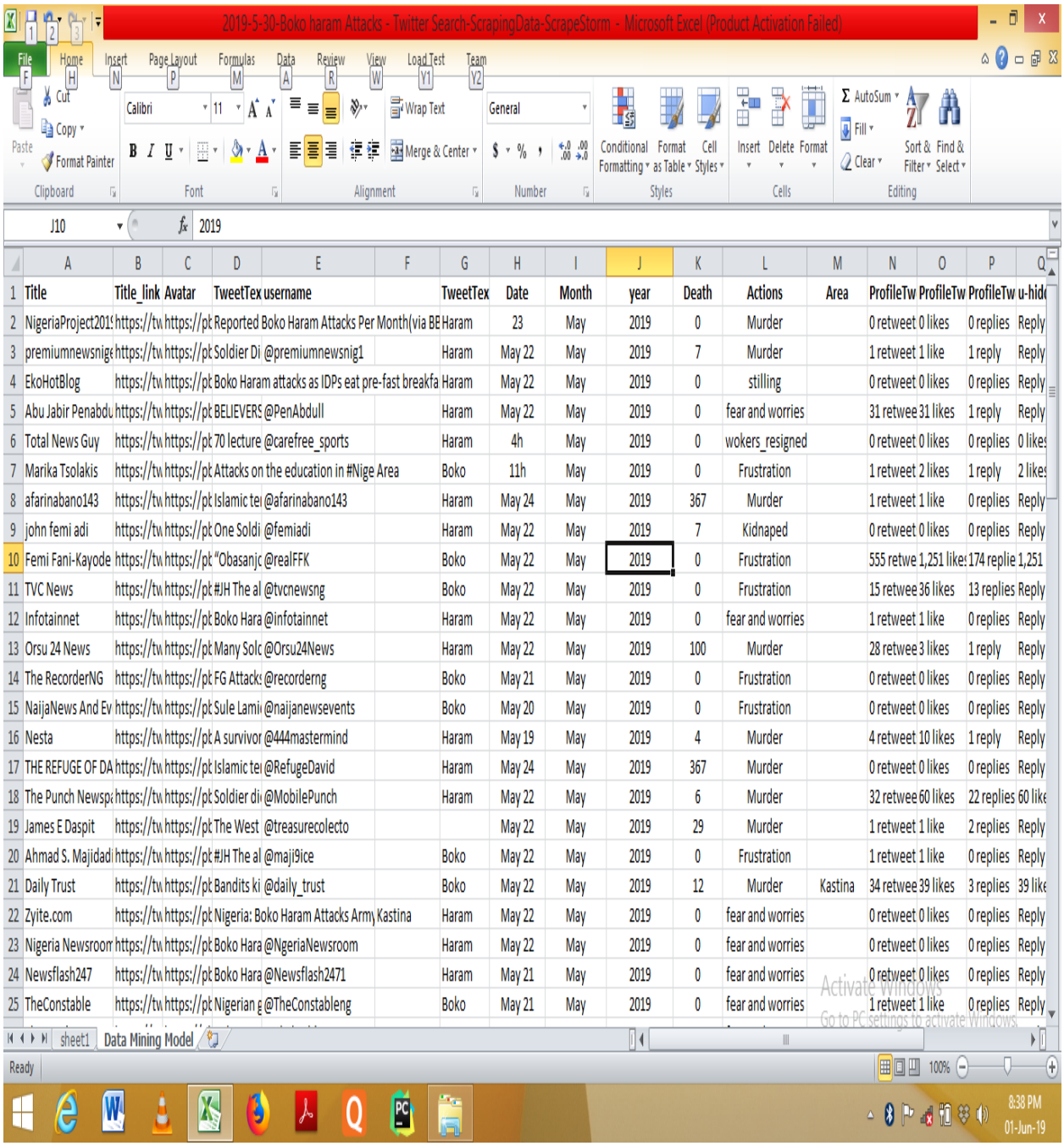

Figure C10 Extracted Raw

C11: Clustered Output of Boko Haram Insurgency from cluster 26 to clusters 27

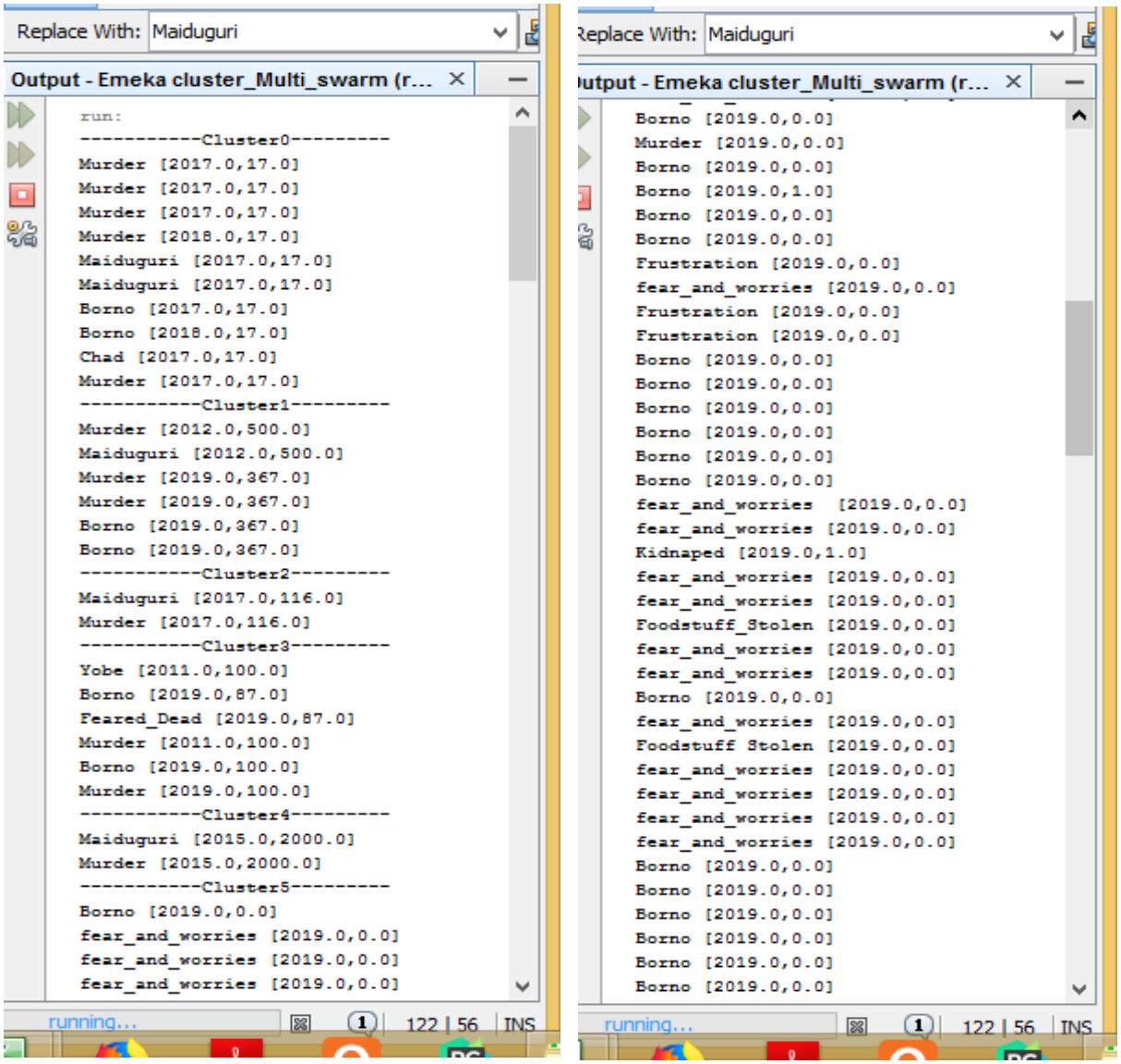

Figure C12: Clustered Output of Boko Haram Insurgency from cluster 0 to clusters 5

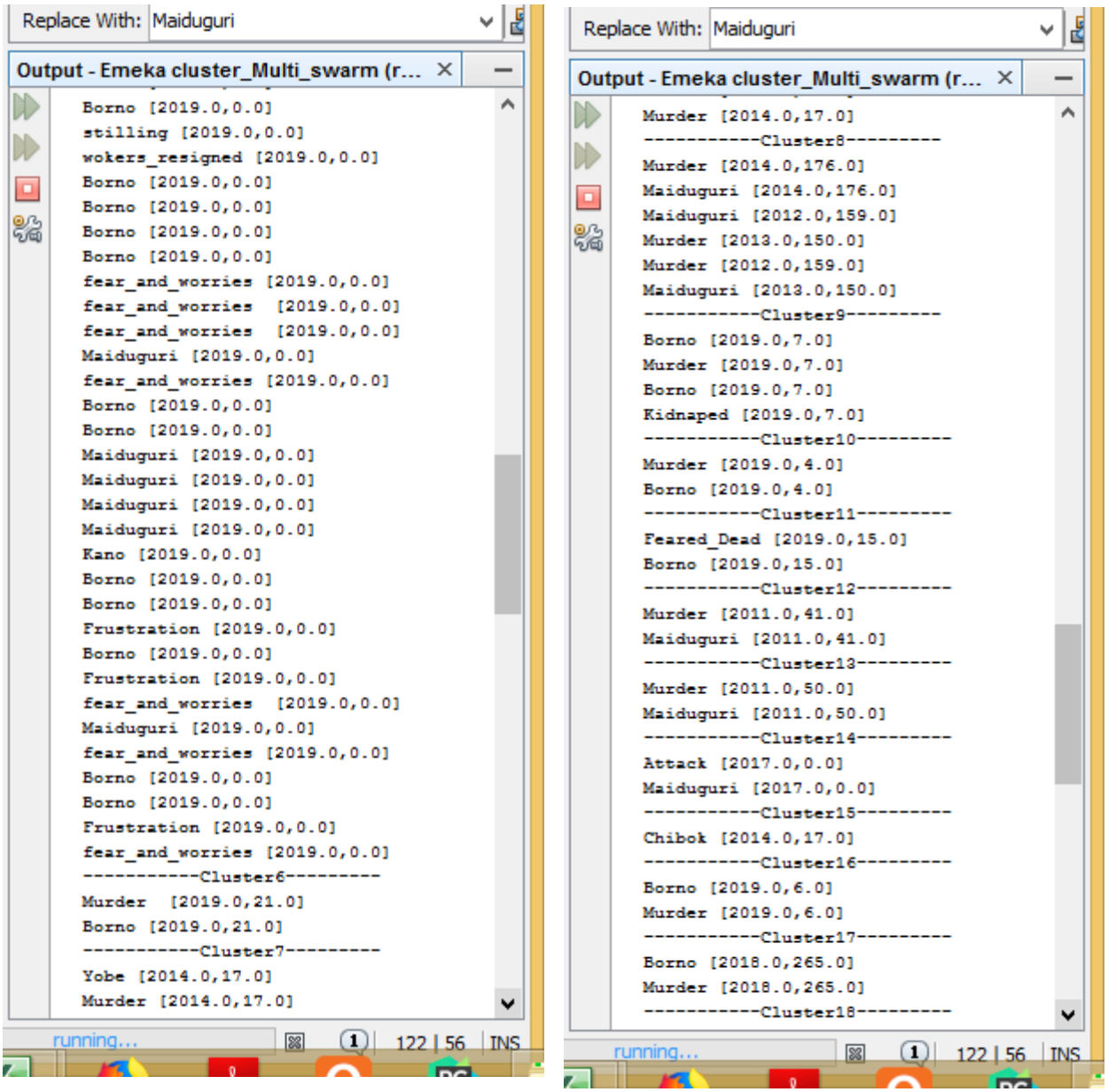

Figure C13: Clustered Output of Boko Haram Insurgency from cluster 5 to clusters 17

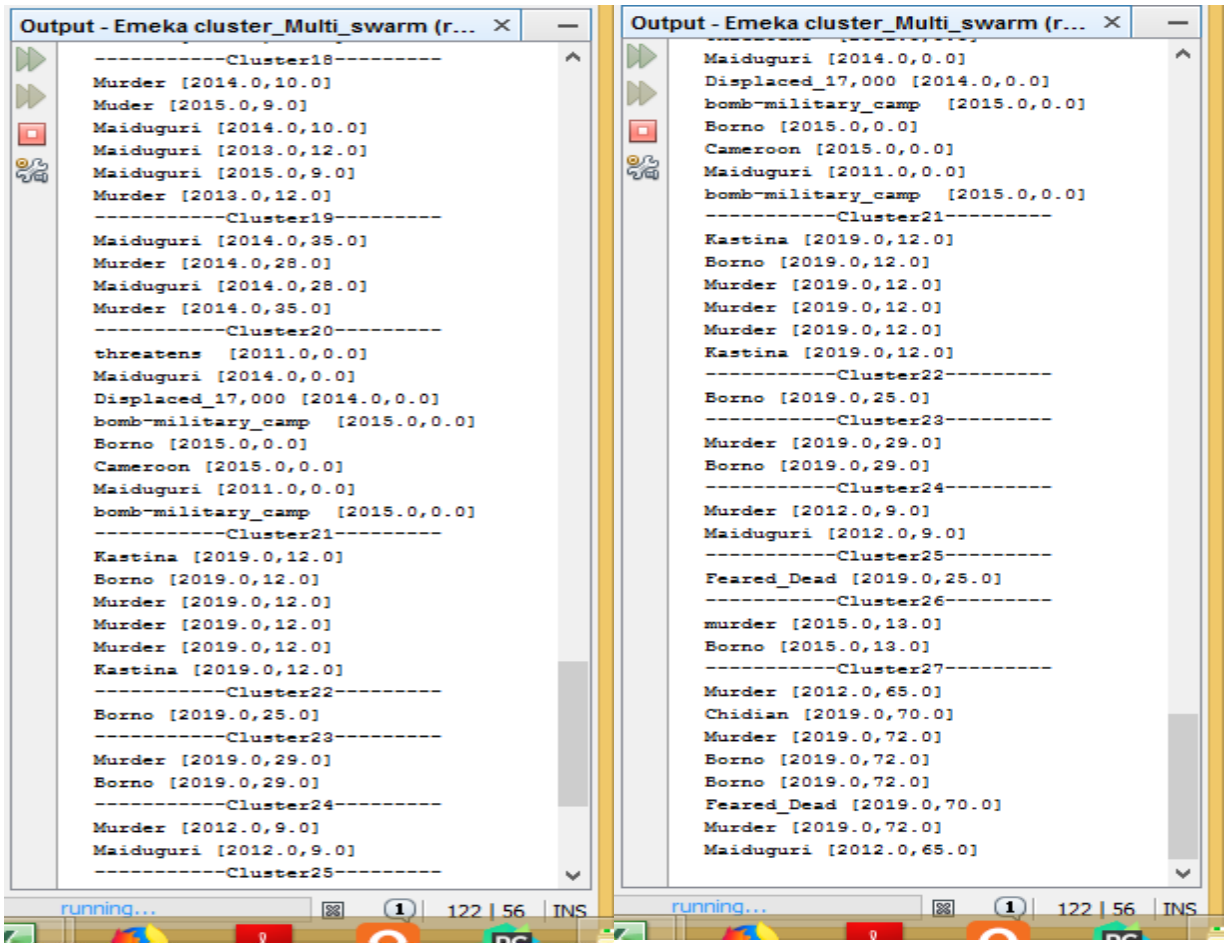

Figure C14: Clustered Output of Boko Haram Insurgency from cluster 18 to clusters 27

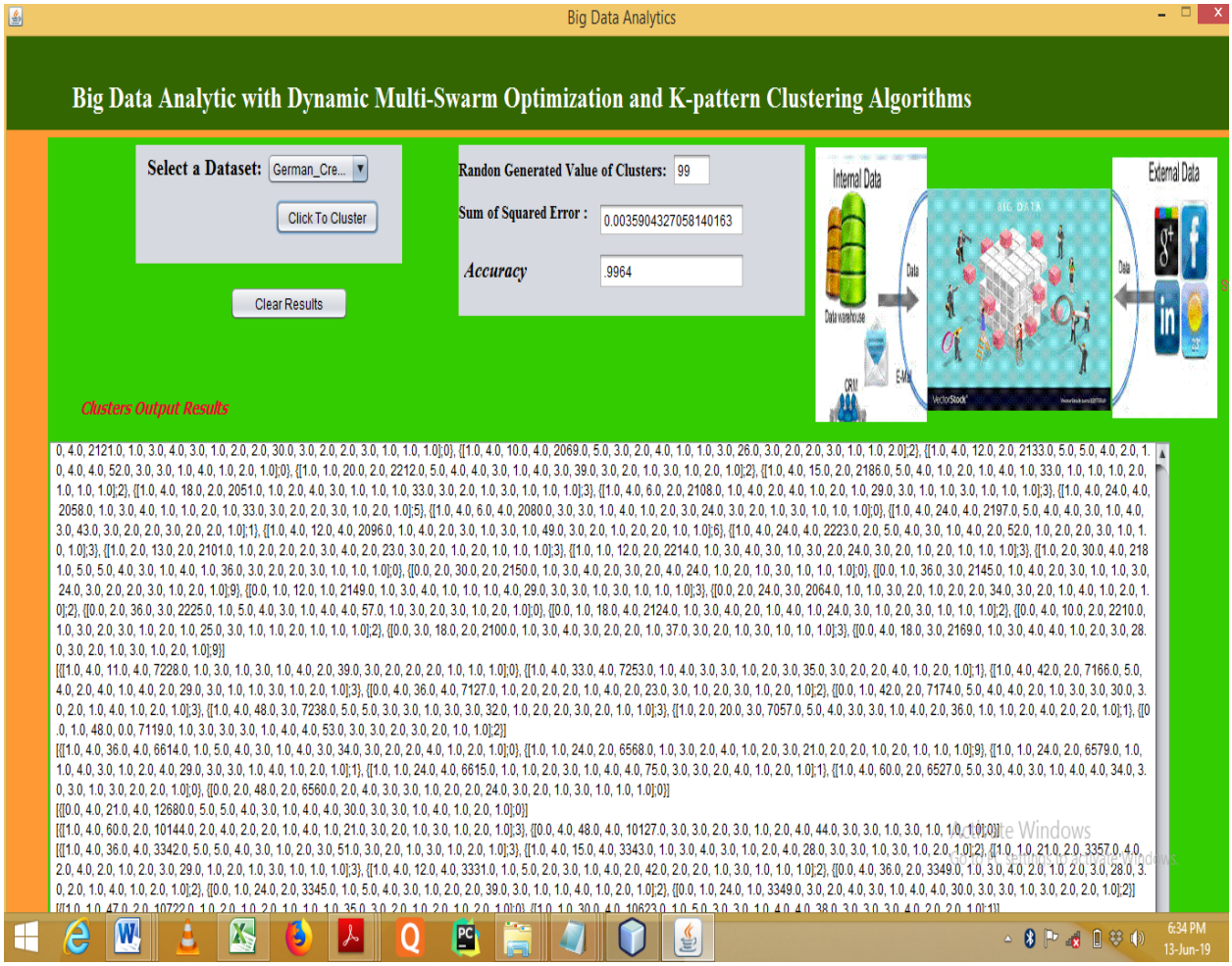

Figure C15: Clustered Output of German Credit Card dataset

APPENDICES D

Summary of Clustered Output

s/n	Date	Month	years	Actions	Deaths	Area	weet-actic
1	23	May	2019	Murder	0	Borno	87
2	22	May	2019	Murder	7	Borno	79
3	22	May	2019	stilling	0	Borno	45
4	22	May	2019	fear and worries	0	Borno	31
5	22	May	2019	wokers_resigned	0	Borno	0
6	22	May	2019	Frustration	0	Borno	2
7	24	May	2019	Murder	367	Borno	1
8	22	May	2019	Kidnaped	7	Borno	0
9	22	May	2019	Frustration	0	Borno	1251
10	22	May	2019	Frustration	0	Borno	36
11	22	May	2019	fear and worries	0	Borno	1
12	22	May	2019	Murder	100	Borno	3
13	21	May	2019	Frustration	0	Borno	0
14	20	May	2019	Frustration	0	Borno	0
15	1	May	2019	Murder	4	Borno	10
16	22	May	2019	Murder	6	Borno	60
17	22	May	2019	Murder	29	Borno	1
18	22	May	2019	Frustration	0	Borno	1
19	22	May	2019	Murder	12	Kastina	39
20	22	May	2019	fear and worries	0	Nigeria	0
21	22	May	2019	fear and worries	0	Nigeria	0
22	21	May	2019	fear and worries	0	Nigeria	0
23	21	May	2019	fear and worries	0	Nigeria	1
24	19	May	2019	fear and worries	0	Borno	0
25	19	May	2019	fear and worries	0	Borno	0
26	22	May	2019	Murder	21	Borno	0
27	22	May	2019	Murder	12	Kastina	40
28	22	May	2019	fear and worries	0	Borno	0
29	22	May	2019	Kidnaped	1	Borno	0
30	21	May	2019	fear and worries	0	Borno	0
31	21	May	2019	fear and worries	0	Borno	0
32	20	May	2019	Murder	12	Borno	0
33	20	May	2019	fear and worries	0	Borno	0
34	22	May	2019	Murder	72	Borno	4
35	22	May	2019	Murder	72	Borno	0
36	21	May	2019	fear and worries	0	Borno	91
37	21	May	2019	fear and worries	0	Borno	77
38	20	May	2019	fear and worries	0	Borno	54
39	20	May	2019	ed, Foodstuff Stolen	0	Borno	38
40	21	May	2019	fear and worries	0	Borno	75
41	21	May	2019	fear and worries	0	Borno	40
42	21	May	2019	fear and worries	0	Borno	23
43	20	May	2019	Feared Dead	87	Borno	12
44	19	May	2019	Foodstuff Stolen	0	Borno	78
45	19	May	2019	fear and worries	0	Borno	1
46	19	May	2019	fear and worries	0	Borno	53
47	20	May	2019	fear and worries	0	Borno	37
48	19	May	2019	Foodstuff Stolen	0	Borno	22
49	5	May	2019	fear and worries	0	Borno	43
50	4	May	2019	Feared Dead	15	Borno	195
51	3	May	2019	Feared Dead	25	Borno	33
52	22	May	2019	fear and worries	0	Borno	824
53	15	May	2019	Feared Dead	70	Chidian	51
54	9	May	2019	fear and worries	0	Nigeria	35
55	7	May	2019	fear and worries	0	Nigeria	23
56	24	Nov	2018	Murder	265	Borno	36
57	10	Aug	2018	Murder	17	Borno	5
58	2	Dec	2017	Murder	17	Gombe	333
59	19	Jun	2017	Murder	17	Maiduguri	36
60	17	Jun	2017	Murder	17	Chad	37
61	18	Jun	2017	Murder	17	Borno	7
62	16	Feb	2017	Attack	0	Kano	1
63	30	July	2017	Murder	116	Bauchi	78
64	28	Mar	2015	murder	13	Borno	115
65	11	Jan	2015	Murder	2000	Nigeria	53
66	14	Dec	2015	Muder	9	Kano	96
67	15	Feb	2015	bomb military camp	0	Borno	57
68	12	Feb	2015	bomb military camp		Cameroon	70
69	6	April	2014	Murder	17	Yobe	2
70	21	May	2014	Murder	17	Chibok	23
71	5	NOV	2014	Displaced17,000	0	Nigeria	69
72	2	Mar	2014	Murder	35	Nigeria	47
73	24	May	2014	Murder	28	Abuja	21
74	22	Sep	2014	Murder	10	Yobe	13
75	18	Jul	2014	Murder	176	Nigeria	12
76	22	Sep	2013	Murder	150	Maiduguri	75
77	30	30	2013	Murder	12	Gombe	45
78	30	Jan	2012	Murder	9	Nigeria	5
79	22	Jan	2012	Murder	500	Nigeria	7
80	10	July	2012	Murder	65	Adamawa	3
81	20	Sep	2012	Murder	159	Nigeria	229
82	25	Dec	2011	Murder	41	Nigeria	1
83	11	Nov	2011	Murder	100	Yobe	54
84	10	Nov	2011	Murder	50	Nigeria	16
85	6	Nov	2011	threatens	0	Nigeria	7

APPENDICES E

MATERIALS USED

The materials used in the research study include: sofware tools and Hardwares specifications. The softwares include: Spark Apache version 3.0.0, Jupyter Notebook, ScrapeStorm, Brower (Chrome), Microsoft spreadsheet, MATLAB version R2018a, NotePad, Netbeans Integrated Development Environment (IDE) version 8.2, and Operating System.

The following materials was used for the implementation of the proposed model:

Apache Spark: is a fast and general-purpose cluster computing system. It provides high-level APIs in Java, Scala, Python and R, and an optimized engine that supports general execution graphs. It also supports a rich set of higher-level tools including Spark SQL for SQL and structured data processing, MLlib for machine learning, GraphX for graph processing, and Spark Streaming. Many common machine learning and statistical algorithms have been implemented and are distributed with MLlib which simplifies large scale machine learning pipelines, including:

i. summary statistics, correlations, stratified sampling, hypothesis testing, random data generation.
ii. classification and regression: support vector machines, logistic regression, linear regression, decision trees, naive Bayes classification, Decision Tree, Random Forest, Gradient-Boosted Tree
iii. collaborative filtering techniques including alternating least squares (ALS)
iv. cluster analysis methods including k-means, and latent Dirichlet allocation (LDA)

Though, Apache Spark has a machine learning library called MLlib that provides the major machine learning algorithms such as classification, clustering, regression, dimensionality reduction, transformers, and collaborative filtering. Most of the machine learning algorithms in Spark are not advance enough for proper benchmarking with our proposed model. Therefore, other big data analytic

tools such as, IBM SPSS modeler or R statistics will also be used for appropriate standard matrix benchmarking with the results of the proposed model.

The apache spark is shown in Figure E1

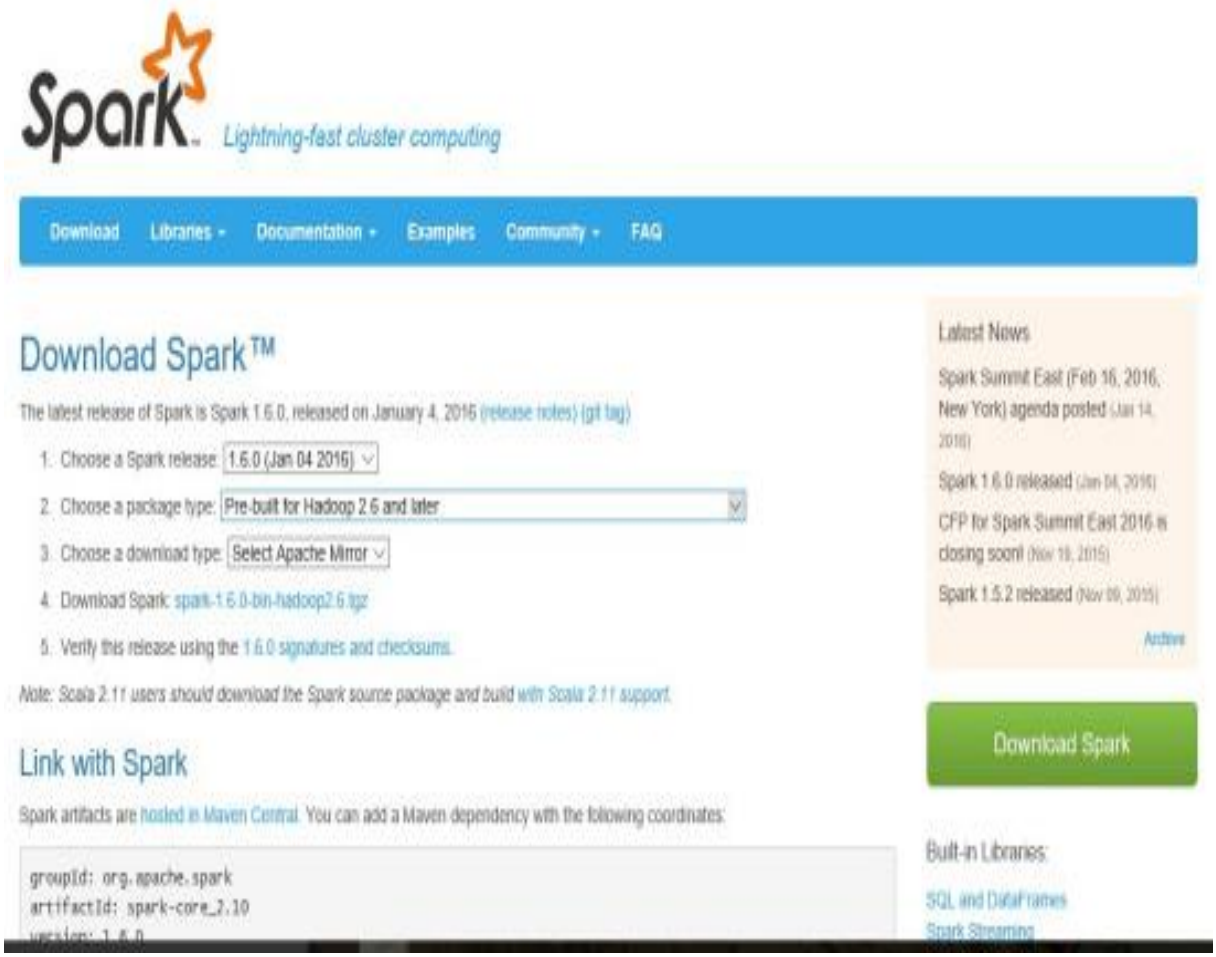

Figure E1: Apache Spark

Apache Jupyter notebook: is a web-b0ased notebook that enables interactive data analytics. With Jupyter, a data-driven benchmark, interactive and collaborative documents with a rich set of pre-built language back-ends (or interpreters) such as Scala and Python (with Apache Spark), SparkSQL, Hive, Markdown, Angular, and Shell will be accessed and used to visualized the results of our proposed system. The diagram for Jupyter notebook is shown in Figure E2.

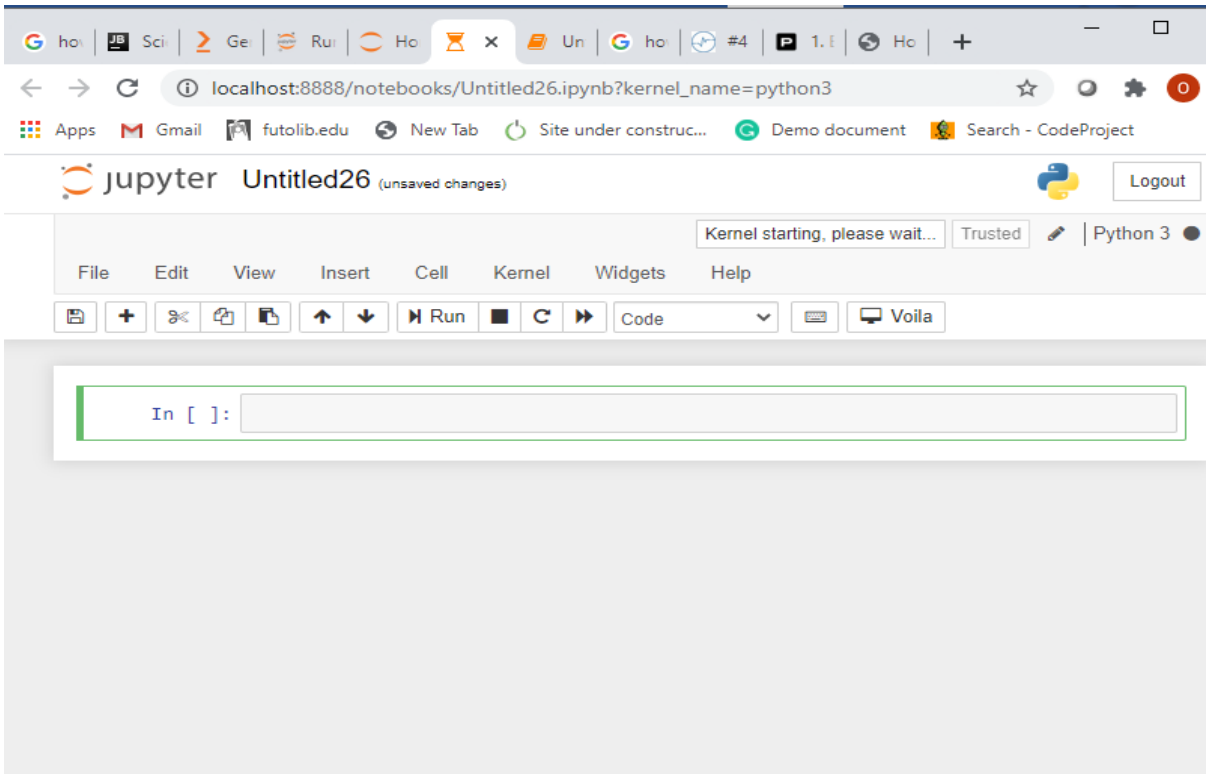

Figure E2: Apache Jupyter notebook

ScrapeStorm: is a scraping software data used to scrape unstructured data from social media and other web contents. It was selected in this work to scrape unstructured Boko Haram insurgency attack historical big data from social media to analyze the efficient of the application of the proposed big data model in this work. This will enable the researcher select required attributes and instance for the unstructured dataset for the evaluation of the improved analytic model. It is built base on machine learning intelligence and has astonishing features for mining. It is easy and user friendly. It enables the users to input the URL of the data/information to be scrape. ScrapeStorm has different interfaces that scrape data in columns, scrapping range settings, adding additional fields, and exporting the scraped data to different computational tools such as spreadsheet, notepad, SQL database, Access database, etc. The diagram for ScrapeStorm is shown in Figure E3.

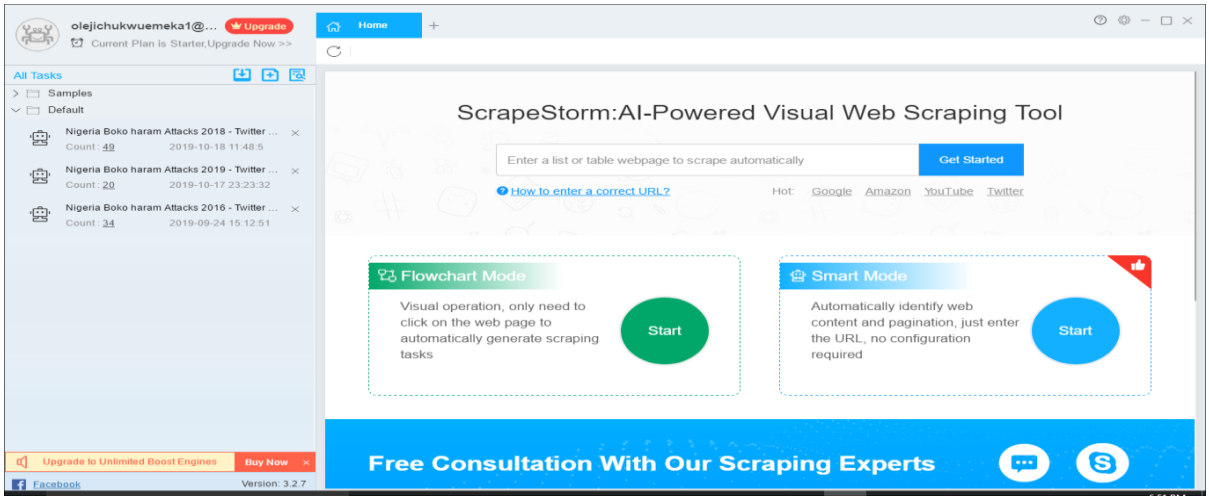

Figure E3: ScrapeStorm.

The Web browser (Chrome): The URL of the problem domain to be extracted from social media can be generated using different web browsers such as Internet explorer, Mozilla or chrome. The Boko Haram attack events was inserted in the search engine of chrome to generate information on Boko Haram attacks of any specific period as requested by the user. The search engine provides the information and its URL that will be used in the ScrapeStorm to extracted the required elements for the analysis.

The Microsoft spreadsheet: The spreadsheet is a computational tool among other such as SQL, MySQL, MongoDB used by big data tools (Apache spark among others are SQL, ScrapeStorm, AZeppelin, MongoDB) to export the unstructured data scraped form social media to create .csv file extension of the unstructured dataset scraped from social media. The scraped data from twitter website using ScrapeStorm big data tools were exported into excel sheet for pre-processing. The unstructured datasets in the excel sheet must be pre-processed using data mining models such as selection, reduction, and cleaning to convert the unstructured high voluminous datasets to structured mixed large dataset for the next phase of the analysis.

MATLAB Editor: After the pre-processing of the raw dataset exported to Microsoft Excel Sheet. The selected, reduced and cleaned structured mixed dataset will be exported into MATLAB Editor version R2018a for further analysis. As the pre-processed mixed large dataset in excel sheet are

exported to MATLAB Editor. The data in columns are converted to comma speared values (CSV) file extension in MATLAB Editor. This will enable the java code of the proposed analytic model to import the large mixed dataset for simulation analysis. The directors of the dataset store in the MATLAB Editor will be upload for simulation, modeling and result analysis of the proposed big data analytic model. The diagram for the notepad is shown in Figure E4.

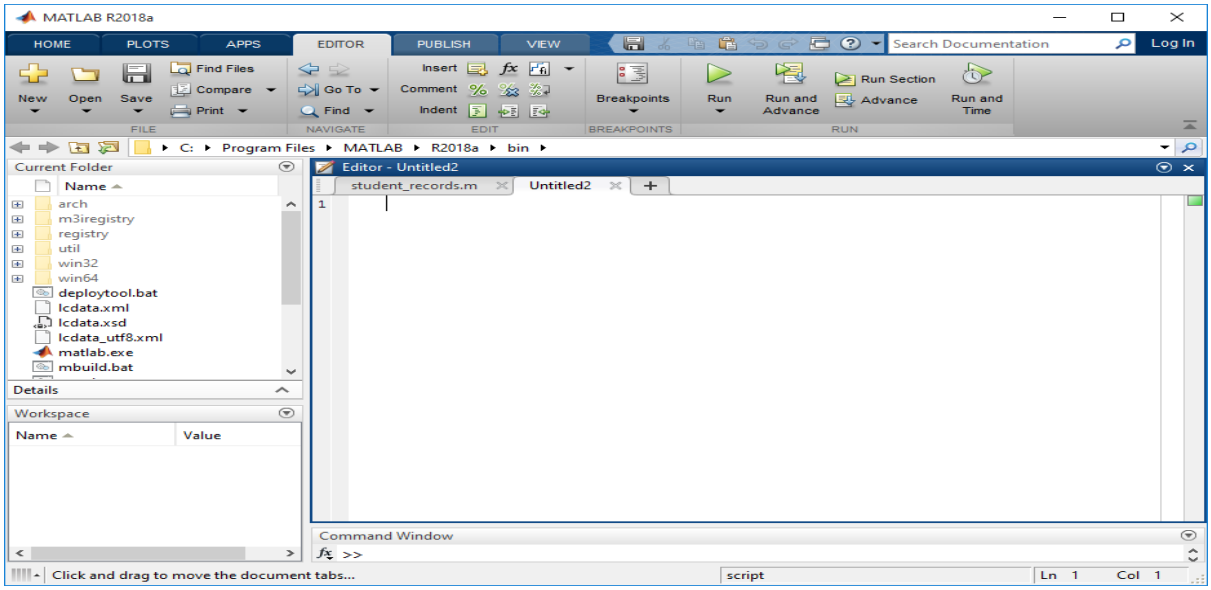

Figure E4: MATLAB Editor

Java Development Environments: Program is a set of instruction written by a programmer in a particular programming language, which enables particular processes to be performed by a computer. The choices of programming languages in this work are JAVA and MATLAB. Java has inbuilt packages of optimizing the techniques to create and manipulate dynamic data structures, such as linked, stack, queue and trees. It has flexible methods such as recursion methods, generic method, Object Oriented Inheritance, file and streams manipulations and graphical user interface components.

NetBeans Development Integrated Environment (IDE): NetBeans IDE is an integrated development environment available for Windows, Mac, Linux, and Solaris. It enables developers to rapidly create Web, enterprise, desktop and mobile applications primarily for the Java and HTML5 platforms, as well as for PHP and C/C++, together with other languages and technologies provided by plugins and add-ons. The combination of technologies, ease of use, and "out-of-the-box"

experience is widely recognized as making NetBeans, Development Integrated Environment (IDE) unique. The proposed model was coded with java programing language in NetBeans development environment. NetBeans development environment diagram is shown in Figure E5.

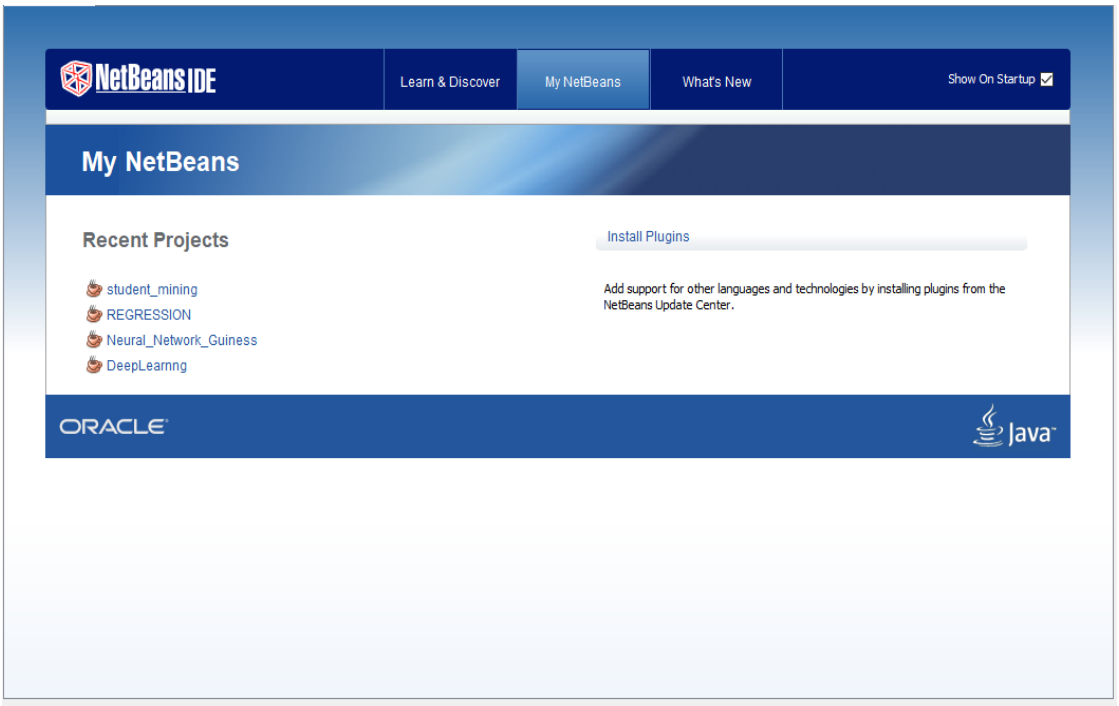

Figure E5: NetBeans Development Integrated Environment (IDE)