Design and Performance Evaluation of Density Based Anomaly Detection Clustering Algorithms using Hadoop and Map Reduce

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

in

Software Engineering

Submitted By

Sourajit Behera

(Roll No. 801431029)

Under the supervision of:

Dr. Rinkle Rani

Associate Professor

Thapar University, Patiala

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

June 2016

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*Design and Performance Evaluation of Density Based Anomaly Detection Clustering Algorithms using Hadoop and Map Reduce*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Rinkle Rani* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Sourajit Behera
(Sourajit Behera)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Rinkle 17/06/16

(Dr. Rinkle Rani)
Associate Professor
Computer Sc. and Engg. Department
Thapar University
Patiala

Countersigned by

(Dr. Maninder Singh)
Head
Computer Sc. and Engg. Department
Thapar University
Patiala

(Dr. S. S. Bhatia)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

First of all I would like to thank the Almighty, who has always guided me to work on the right path of the life. It is a great privilege to express my gratitude and admiration towards my respected supervisor **Dr. Rinkle Rani,** Associate Professor Computer Science & Engineering Department. She has been an esteemed guide and great support behind achieving this task. This work would not have been possible without the encouragement and able guidance of her. I also thank my supervisor for her time, patience, discussions and valuable comments. Her enthusiasm and optimism made this experience both rewarding and enjoyable. I am truly grateful to her for extending her total co-operation and understanding whenever I needed help and guidance from her. I am also heartily thankful to **Dr. Maninder Singh**, Head, Computer Science & Engineering Department and **Rupali Bhardwaj**, PG coordinator, for motivation and providing uncanny guidance and support throughout the preparation of the thesis report.

I will be failing in my duty if I do not express my gratitude to **Dr. S. S. Bhatia**, Senior Professor and Dean of Academic Affairs, for making provisions of infrastructure such as library facilities, computer labs equipped with net facilities, immensely useful for the learners to equip themselves with the latest in the field.

I am also thankful to the entire faculty and staff members of Computer Science and Engineering Department for their direct-indirect help, cooperation, love and affection, which made my stay at Thapar University memorable. Last but not least, I would like to thank my family for their wonderful love and encouragement, without their blessings none of this would have been possible.

Sourajit Behera

Sourajit Behera

(801431029)

Abstract

Advancement in technology in the last few decades has created loathes of data emerging from different source ranging from social media, customer centric data, online transactions to mention a few. So, companies and individuals are keen to analyze the data which is constantly increasing in both volume and complexity using effective data mining algorithms to take better decision based on the analysis. Various approaches are followed under data mining to meet present day demands of data analysis. Clustering approach is one such technique used to find instances in a data set which are more similar to each other and form groups while being different from other instances in other groups. Using the approach helps to detect data instances which do not follow an idea of well defined instance and raises suspicion of being generated externally due to some process.

DBSCAN and OPTICS have been classified under the clustering approach for data mining. Because of high volume and complexity of data, hadoop framework has been in demand to perform operations on the data. Much of the work has already been done in proposing the individual algorithms and implementing them on smaller data sets. In this thesis we focus on creating multi node cluster of hadoop framework and perform a performance comparison of the algorithms DBSCAN and OPTICS by implementing them on real data sets on multi node clusters using Map Reduce programming approach.

In this thesis, the implementation of DBSCAN and OPTICS on multi node cluster shows that OPTICS algorithm is slower than DBSCAN by a factor of 1.5-1.6. It is also observed that increasing the number of nodes in the cluster, leads to reduction in execution time of the algorithms on real data set.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is Big Data?

With advancement of technology in recent years and embedding up of the technology in the day-to-day lives of people has created lump sum amount of data in magnitudes of terabytes. With such huge amounts and increased complexity up of data summed as **Big Data** available to individuals, companies, government analyzing them and taking effective falls under a broader term called **Data Mining**. The introduction up of Big Data is due to the fact that with increase in complexity of data the process of analyzing the data cannot be achieved using traditional data processing applications. As cited by **Wikipedia**, the term Big Data is coined for data sets which are large in size plus the complexity is huge so that traditional data analyzing applications are unable to perform effective computations on the respective data.

Let us understand the concept of Big Data by taking an example. Consider a train having thousands of sensors attached to keep track of things like the condition of the individual mechanized apparatus of the train, the GPS based data used for the shipping of goods and items etc. After many cases of train accidents which led to loss of many precious lives, the government ordered that these types of data for every train need to stored and analyzed to prevent such train accidents in future. The rail bogies are fitted with sensors and have been made intelligent by adding more processors to find meaning of the sensor data on mechanized parts that are known to wear out such as ball bearings. The analysis of such data can find out the faulty parts before they are worn out and prevent rail disasters. If we add to the sensor data, timings like arrival and departure of the train, logistic timings and goods size then we will fast approach the Big Data problem. Despite some of the above data being relational, most of it was still in a raw, unstructured format which cannot be processed by traditional data mining applications.

1.1.1 Characteristics of Big Data

Big Data revolves around **3 V's** referred as data variety, data velocity and data volume.

1) **Huge data volume** is generated at a continuous rate which applies both for companies and individuals as well. Take for example a text file has size in range of few Kilo Bytes, an audio file in range of few Mega bytes which a movie lies in range of few Giga bytes. Now a day's data is generated by employees, partners, customers, as well as by machines in ranges of terabytes while earlier it was just generated by employees of the companies. Such terabytes of information is an issue for people looking to analyze the data rather than just letting them go to waste. As cited by Large Synoptic Survey Telescope, around 30 terabytes of image files will be generated every night over the following decade. Thinking about the data that will be generated for the entire year is out of mind and for a decade it will be just overwhelming for any data center to store and process the data to find patterns within them. As per stats by Google around 72 hours i.e. 3 days time of videos which range from some Megabytes to several Gigabytes is being uploaded on the video sharing website YouTube every minute. These two examples effectively sum up of the huge data constraint up of Big Data.
2) **Huge data Velocity,** summed as the batch processes used earlier to process the data in earlier days is breaking up now a days as the time in which new data pops up from varied sources like social networking or websites is faster than the batch processing time of data. As published by Website Twitter, around 140 million tweets are posted on the micro blogging site on an average which testifies the velocity with which the posts are made on the site.
3) **Huge data Variety,** owes its attributes to the varied sources of data ranging from excel, spreadsheets, word documents, images, video, audio files, portable document format items etc. In short the new generation data has no fixed format i.e. structure-less. New file formats are introduced as new applications are introduced into market. For example, Google makes usage of smart phones a way of sensors to determine the traffic patterns at large. Most probably they read the

position up of cars to construct the best possible route to reach a destination asked by passengers. This sort of data had no existence some years ago.

1.1.2 Big Data Vs Traditional Data

The data in the modern age is emitted from many sources which is already been described in the data variety section above. Most of the data is in unstructured format which can be processed by traditional data processing applications. Traditional data mostly in form of rows and columns use batch processing to process the data as the velocity with which traditional data was emitted from various sources some years ago was very less as compared to the batch processing time. But the scenario has changed drastically with the introduction of new formats which have not been used elsewhere.

A novice solution will be to convert these unstructured data to a well defined format and then use the traditional batch processing applications for analyzing the data. But the cost for such a conversion to take place is very high and in some cases may be unworthy, if the data which is converted is not valuable at all for the analyst. So, the best approach that fits in the context is to first try to find the data which can be worthy using a rough approach and then convert the same into well-defined format and store it in a data warehouse. The data is then obtained from the data warehouse and fed into the traditional data mining tools to find meaningful intended data for companies, individuals & governments to act upon.

As already discussed the cost of converting the data from unstructured to a structured format is very high, same goes for data that has been stored in the data warehouse. The problem of fetching and running the data stored on a traditional data warehouse can be solved by a framework namely **Hadoop.** We take usage up of **Hadoop framework,** where the huge data set is broken down and stored into the common **Hadoop distributed file system (HDFS)** and processed in a parallel fashion using a number of processors **Map Reduce.**

1.2 What are Anomalies?

Anomalies are simply patterns in a data set which violate assumptions of a data point which has been classified as a normal data point. The figure 1.1 shows points in a two dimensional data set. Normal regions N1 and N2 are shown to have the normal data points. The points which are sufficiently farther from these regions like o_1, o_2 and points present in region O_3 are classified to be anomalies.

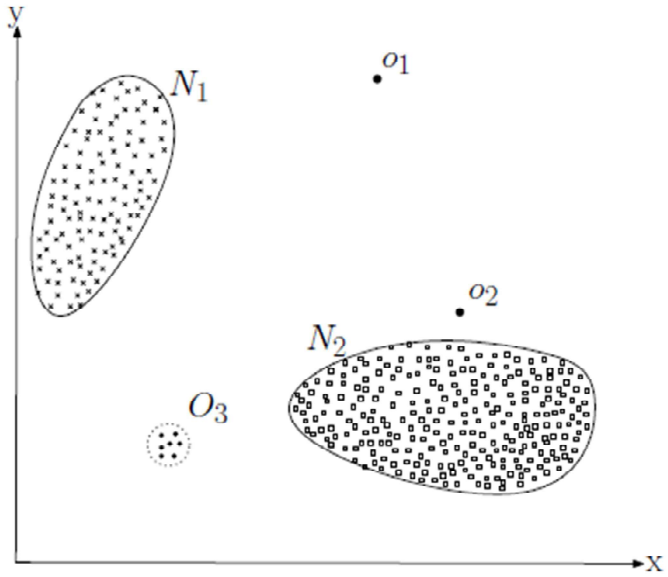

Fig. 1.1 Outliers present in a 2 dimensional data set

As defined by Hawkins: An anomaly is an instance which deviates from other instances by large amounts that it gives a suspicion that the anomaly might have been generated from some other source or mechanism. Anomalies can be inserted into the data from many reasons like malicious activities such as credit card fraud, breaking down of a system, cyber-intrusions or due to some terrorist activities. The key factor is that all these anomalies have one thing in common i.e. it represents some interesting happening in the normal occurring of the application. This interestingness is what drives an analyst for anomaly detection.

1.2.1 Anomaly Detection Vs Noise removal

Anomaly detection has similarity but differs from both Noise removal and accommodation, which are connected to unwanted noise in the data. **Noise** can be stated as occurrence of data in the set not useful to the analyst but stops in proper analysis of the data. Noise removal is connected with the requirement of removing any unwanted data in the data set before any analysis can be done. Noise accommodation relates to creating a statistical model and making it immune to noise.

1.3 Types of Anomalies:

The type of desired anomaly plays an important role in classification of the Anomaly detection technique. Anomalies are categorized into following types:

1) *Point Anomalies:* It is the simplest type of anomaly and most researched anomaly. If a single data point can be termed as anomalous taking into account the rest of the entire data set it is called as a Point Anomaly. In the above figure 1.1, point (s) o_1, o_2 and points present within the boundary of region O_3 are sufficiently away from the normal data set points. And hence they are termed as point Anomalies.
   Taking a real life scenario, consider a scenario of credit card transactions made by a person which will be considered as the data set. And the data is characterized by usage of only one attributes namely the *amount spent.* A particular spending by the person where the transaction is way above than the average spending by the person can be termed as a Point Anomaly.
2) *Contextual Anomalies:* In a particular situation if a data point acts as an anomaly but in general it's not then that data point is called a contextual Anomaly. It needs to be a part of problem speculation to specify the context. Each point is characterized by following set of attributes:
   a) *Contextual Attributes:* It is used to find the neighborhood of the instance. For example in case of spatial data, the contextual attributes which can be considered are latitude and longitude. While in a time-series data, time can

be considered as a contextual attribute which measures the position of the point in the whole data sequence.

b) *Behavioral Attributes:* It says about non-contextual properties of the data instance. If we consider the spatial data which says about average rainfall of the world, then the total amount of rainfall obtained in a particular area can be considered as a Behavioral Attribute.

The anomalous characteristics are judged by the behavioral attributes of any instance in the data set. In a specific context, the data point can be considered anomalous, which in another context can be a normal instance. This particularly characterizes the contextual anomalies to be having both the attributes i.e. contextual and behavioral attributes.

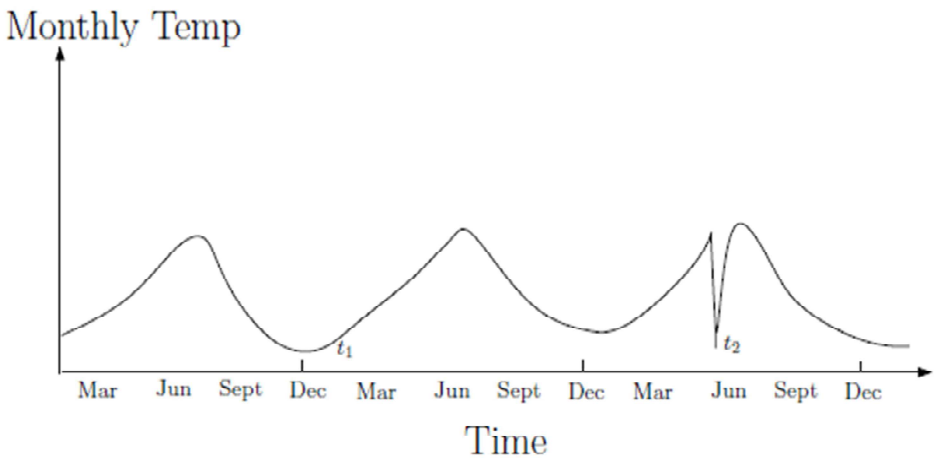

Fig. 1.2 Contextual Anomaly

The above figure 1.2 represents contextual anomaly at time t_2 in time temperature data series. It is important to see that the time at t_1 and t_2 are same but both occur in different situations and therefore t_2 is considered as anomalous whereas t_1 is not. Take for example a credit card fraud detection situation. We can take the Contextual attribute to be the time of purchase in the credit card purchasing domain. The person bearing the card spends around 500$ per month except for

1000$ during Christmas. And if the person spends around 1000$ in the month of May then this can be equate to a contextual anomaly, since it does not follow the rule used for deciding what a normal data point will be.

3) *Collective Anomalies:* If a set of data points is considered to be anomalous when compared with the entire data set, then the set can be considered as Collective Anomalies. The individual data instances may not be anomalous by themselves but their collection may be anomalous altogether.

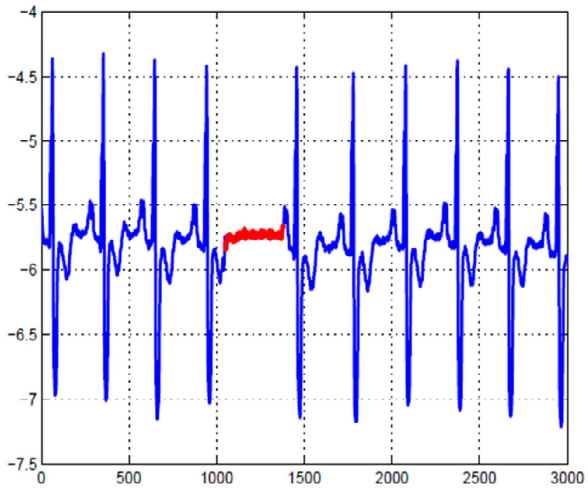

Fig. 1.3 Arterial Premature Condition in a human Electrocardiogram output

The highlighted region in the above figure 1.3 represents an outlier as the low value appears for a considerable amount of time but the low value in itself is not an anomaly.

Point anomalies occur(s) in any data set while collective anomalies can only occur in data set where the data instances are related to each other. Contextual anomalies can only occur when they possess contextual attributes in the data set.

1.4 Why Anomalies need to be detected?

Anomalies need to be seen from the following two perspectives:

1) If the data set contains unusual records which are considered as anomalies but they are actually noise, then they have to be removed before any analysis can be done on the data set. An example for the above requirement can be explained below:
    a) *Time Series Analysis:* Here the observations are collected sequentially over a period of time. For example the number of defects generated by a plant in a day, the height of a plane every minute etc. In all the examples, a measure (number of defects, height, etc.) is connected with a time stamp (day, minutes, etc.) as they are collected. If the observations are collected over a period of time at regular intervals then a time series data is generated. Analysis of this data is needed in many businesses. This becomes very complicated if the curve of the series varies with time. But for most of the time series the values which are closer to each other are more interrelated to each other than those separated by large amounts of time. Time series data are mostly disturbed by unknown events which result in dramatized patterns. These unknown patterns pose a difficulty in understanding the nature of the time series data. Identification of anomalies is important in many areas which deal with time series which contain information that can cause intervention in the working of a process and leads to prevention of failures or abnormal operating conditions of the process. So, it is needed to find anomalies in real time data.
2) If some of the data instances show anomalous behavior and the data points are treated as anomalies which has important information about abnormal properties of the respective system and entities. The finding of the unusual properties provides very large application-specific data. Some examples of the above requirements for detection of anomalies is as possible:
    a) *Intrusion Detection:* It is defined as finding of any suspicious activities in computer related system such as any type of computer abuse like

penetrations, break-ins. These suspicious activities are of specific interest for the analyst from the perspective of system security. This differs from the normal working of the system and so anomaly detection approaches are applicable to this scenario. The major challenges faced in this area are for detecting anomalies is the huge amount of data along with the data is mostly available in a real time which requires real-time analysis. Moreover false alarm rate is also a big problem encountered in this domain. Intrusion detection is further divided into the following subtypes:

i. *Host Based:* These are mainly associated with call traces with operating system. These mostly occur in form of collective anomalies in the traces. These collective anomalies may be occurring due to policy violations or unauthorized behaviors etc. Since all traces will be containing the normal and abnormal behaviors from the same event what is important is the simultaneous occurrence of the events which distinguishes between the normal and abnormal behaviors.

ii. *Network Based:* It deals with finding intrusions in the network traffic and data. These anomalies normally occur as point anomalies while some anomaly detection techniques create a model which detects them as collective anomalies. These mostly occur due to outside intrusion i.e. due to hackers which want to have their hands on private information of the victim or takedown the system. A typical scenario is when many computers in a network are connected to the internet through a router or any connecting device. A key challenge faced in this domain is that the hackers constantly change their mode of working so as to prevent themselves from getting identified by intrusion detection programs.

b) *Fraud Detection:* It means detecting criminal acts in fields relating to financial organizations like banks, credit card issuing organizations, insurance companies, cell phone service providers, stock markets etc. The malicious users can be original customers of the organizations or pretend

to be authentic members of the organization also known by the term *Identity Theft.* The frauds occur when these so-called customers use resources provided by the organization in an unauthorized way leading to financial losses. These types of organizations are interested in immediate detection of these frauds to minimize economic losses. The most common type of anomaly detection used in this domain is to maintain a profile for the so-called customer and then try to monitor the profiles to find any deviation. Some of the fraud detection scenarios are as follows:

i. *Credit Card Fraud:* Here anomaly detection approaches are used to find credit-card fraud cases or searching cards which have already been stolen and used inappropriately by the thief. The data is from multiple dimensions which can be id of the user, the amount spent, the time period between successive transactions of the credit card etc. The frauds are mostly depicted in records of the transactions which are classified as *point anomalies* which have mapping with sudden huge payments made by the user, purchasing items which have never been bought by the user, huge rate of purchasing within any given user session online etc. The credit card issuing companies have all the data available regarding the customers, their profiles etc. Hence mostly labeling and profiling technique is used in this area. The key problem faced in this area is that the fraud needs to be detected pretty soon as the fraudulent transaction has been executed online.

ii. *Mobile Phone Fraud:* It falls under the category of monitoring problem detection for activities. The work is to check all the caller usage profiles and raise an alarm when it detects that an account has been mishandled. Calling activity will be mostly recognized by call records and profiles. Every call record contains information such as *call duration* and *calling city* which have been classified as continuous and discrete data individually. The anomalies in this situation can be having huge volumes of call duration for the caller

profile or dialing destinations which have not yet been seen in the caller's profile yet.

iii. *Insurance Claim Fraud:* One of the important problems for property casualty domain insurance companies is fraud claims curtailing to automobile insurance fraud. Single as well groups of people conspire against the claim processing systems for gaining access to non authorized and claiming illegal claims. Detecting these activities has become a top priority of such industries to overcome huge financial losses.

iv. *Insider Trading:* This mainly occurs in stock markets, where people divulge important information making huge personal profits before the information has been publicly announced. The information given away by the persons can be of many types. It can be the news about a pending merger or acquisition of a company, a terrorist activity occurrence which will affect a particular industry or any data which may hamper the stock prices of a particular organization. Anomalies need to be detected from the data available online to prevent people from making illegal and huge amounts of personal profit.

c) *Medical and Public Health:* It is mainly associated with patient records. Anomalies can exist in the data due to patient problems, instrumentation or recording errors. The data is multi dimensional and may consist of patient's age, blood group type, weight, height, blood pressure measure etc. The data can also have both temporal as well as spatial connection attached to it. Aim is to finding mostly point anomalies. Another form of the time series data like *Electrocardiogram (*ECG) and *Electroencephalogram (*EEG) which need to be handled in this domain. The most challenging aspect in this domain is that the cost of naming an anomalous instance as a normal instance can be very high. Thus anomaly detection in this domain needs to be very accurate.

d) *Industrial Damage:* Machine parts in industries wear out due to normal usage and rust. Such damages need to be detected before these lead to heavy losses. In this domain the data is commonly referred as *Sensor data* as the data is gathered from many sensors for analysis. It can be further subdivided into two categories:

   i. *Fault in Mechanical units:* It monitors the performance of parts of machines in industry such as motor, rotor, turbines, oil flow in the pipes and detect anomalies as early as possible which occurs because of wear and tear as well as due to some unforeseen problems. Outliers are required to be found in an online fashion to apply preventive measures as soon as the problems occur.

   ii. *Structural Defect:* It detects structural anomalies like finding cracking in beams, finding strains in the airframes etc. The information that is collected out here can be a time-series data or be a spatial one which can change over time.

e) *Image Processing:* Analysts in this domain are interested in finding the changes in an image over time or in finding regions in the static image which seem to be abnormal in the context of the static image. This domain contains different areas under its umbrella like images from satellites, recognizing digits, mammographic image analysis, surveillance videos etc. Anomalies in this domain are inserted due to motion or due to inserting of foreign objects in the image or due to instrumentation errors. One of the key challenges in this domain is the huge size of the input file. When dealing with surveillance videos online anomaly detection in a real time manner is required.

f) *Finding Anomaly in Text Data:* This domain is primarily focused on novel topics, news articles in a bunch of news stories or editorials. The information in this area consists of many dimensions and very few to mention. A challenge to be addressed is to handle the huge varieties of data and documents which belong to either one category or topic as needed.

g) *Sensor Networks:* It is a newly research field which has been of interest to the analyst because data from many wireless sensors have several unique properties to name a few. Anomalies in this domain can be about finding faulty sensors or detecting intrusions into the systems. The sensory systems can collect data from various sources like binary, continuous, audio and video etc. Often data is manufactured by online mode. So, anomaly detection techniques to have an online approach. The data being collected from distributed sources and hence distributed data mining approach need to be used by the respective analyst. Moreover, if noise will be present in data collected from several sensors then finding the anomalies will pose a very tedious task.

h) *Earth Science:* Data in this domain can range from weather or climatic changes or concerned with land coverage patterns which are collected from various sources like satellites or through remote sensing. Anomalies which occur in this area show environmental or human trends which may be the cause of such anomalies to detect future calamities with accuracy.

1.5 Noise Vs Anomaly

In most of the real life applications, data is embedded with noise which poses great difficulty for an analyst. It is usually the deviations from normal data which are of interest to the analyst. Consider the examples shown in figures 1.4 (a) and (b).

It is evident from both the figures that the groups of points are same in both the figures while the points outside the groups in both these figures are significantly different. In Fig. 1.4 (a), the point marked as ‘A’ is largely different from the rest of the data instances in the data set and hence can be termed as an anomaly. But in Fig. 1.4 (b), the same point ‘A’ lies in sparsely populated region and it is difficult to state if the point is deviated from the normal stated points. It is very likely that this point can represent the noise which is distributed over the data set. This occurs because the point ‘A’ seems to follow a pattern of other randomly distributed points of the data set.

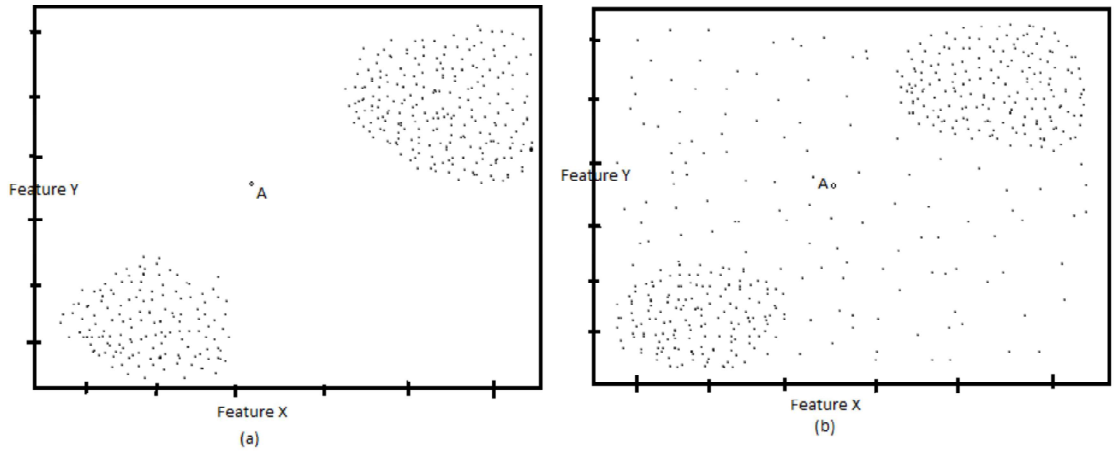

Fig. 1.4 Noise Vs Anomaly

1.6 Anomaly Detection Methods

Depending upon the type of data i.e. labeled or unlabeled, the anomaly detection approaches can be categorized into the following modes:

1) *Supervised Anomaly Detection:* The processes that fall under the *supervised* mode work on the assumption that the data set has known data points of both normal points and also anomalous points. The most common approach is to create a predictive model for normal points versus the anomalous points. Any unknown data point is taken for comparison against this predictive model which will show which group the unseen data point will belong to. There exists two major challenges faced in this type of anomaly detection mode:
    a) The data points considered as anomalies are far less in number than those of normal instances in the training data set.
    b) Obtaining accurate labels for the anomalous data instances in data set is usually very tedious task to perform.

    The supervised Anomaly detection technique is more or less building a predictive model for the data set.

2) *Semi-Supervised Anomaly Detection:* The techniques under this category assume that there exist known instances for the normal data points only. Since they do not require the known instances for anomalous data points, so they are widely applicable than the *supervised anomaly detection* technique. Taking an example a scenario for fault detection in a space craft, an anomalous scenario will be to detect the accident of the space craft which is very difficult to create a model for it. The most common approach in this mode is to create a model for the normal instance points in the entire data set and then compare any unseen point in the data set to this model to find the anomalous data instance in the data set.
3) *Unsupervised Anomaly Detection:* Approaches that fall under this category does not require any training data. Hence are widely used. The detection approach makes an assumption which is that normal data instances are occur more in nature than the anomalous data instances in the given data set. If the above assumption proves to be false then these approaches suffer from a very large false alarm rate.

Semi-supervised approaches are accommodated to be used in unsupervised mode by utilizing a data set which has no distinction for the normal and outlier data points. Such usage depends on the assumption that the data set will have very few anomalous data instances and the model learnt during data set usage is capable of finding fewer amounts of anomalous data points in the given data set. On basis of the assumptions made on the normal and anomalous data instances, the anomalous detection approaches are classified into the following types:

1) ***Classification based:*** This technique focuses on learn a model from the set of known points, often called as the training data. The next step is to take an unknown instance and try to classify the new instance in any one of the classes according to the model. This type of anomaly detection works in a two mode fashion. In the first step, called the training phase, a model is created from the known data points which are present in data set while the second step is called the testing phase, an unknown data point is taken from the data set and classified as either *normal* or *anomalous* according to the model. Depending upon the type

of labels which are available for the data in the training phase, classification based techniques are further divided into two categories:

a) ***Multi-class:*** It works with the idea that the data in training phase contains known data points which belong to multiple normal classes. Such techniques consider that the model created will learn automatically to distinguish one normal class from the other classes residing in data set. A data point is labeled test which is considered to be anomalous, if the model created is not able to identify the data point as part of the various normal classes present in the data set. Some of the techniques which fall under this category assign a confidence score to all the predictions made by the model in this phase. If any of the models present in the categories is not confident enough to say that the data point is normal, then it is automatically tagged as anomalous.

b) ***One-class:*** It works with the idea that all data instances in the training phase are part of one single class. These approaches learn about a boundary present around the normal data points which use a *one-class classification algorithm.* Any considered test data instance in the data set which do not fall within this boundary is classified as outlier.

Multi-class and One-class are represented can be clearly seen in Fig 1.5. The various anomaly detection techniques which use classification based approaches are neural networks, support vector machines, rule based etc.

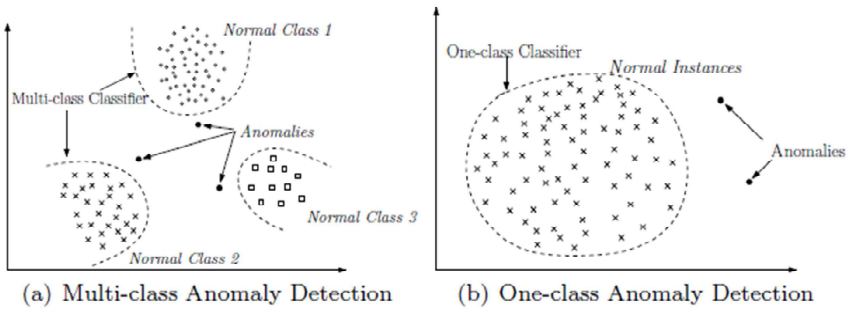

Fig. 1.5 Using Classification based Anomaly Detection Technique

The disadvantages of using classification based anomaly detection techniques are:

a) Many class classification approaches mainly depend upon presence of accurate data for various normal classes which is in general not possible to be present always.
b) These techniques assign information to any data instance in the data set which can be a disadvantage when we try to assign a meaningful relative scale of anomaly score to every instance in the data set.

2) ***Nearest-Neighbor based:*** It relies upon either a distance based approach or similarity based index between two data points. The data instances are mainly provided an anomaly score which is a relative scale to decide the anomalous behavior of the data instance. Nearest neighbor based techniques can be divided into the following two types:
   a) Approach which uses the distance of any data instance to its kth nearest neighbor to assign the same as the anomaly score for the data instance.
   b) Approach which computes the relative density of each data instance to assign the anomaly score for the data instance.

   The limitations of the nearest neighbor based anomaly detection approaches are:

   a) Particularly for unsupervised approaches, if data set contains normal data points which don't have many neighbors which are close or if set has outliers which possess many close neighbors, the result is the technique fails to identify them correctly and outcome is false anomaly detection.
   b) Particularly for semi-supervised approaches, if normal data instances in data set do not have required similar normal data points for testing in the training phase, the false alarm rate for anomalies goes on the rise.

3) ***Clustering based:*** It helps in grouping together similar data points into a cluster(s). It is primarily an unsupervised technique but recently it has been applied to semi-supervised area as well. Clustering based anomaly finding methodologies are divided into following three categories:
   a) Normal data instances form a part of a cluster while anomalous data instances are present and do not form any part of any group.

b) Normal data instances are very near to the centroid of their nearest cluster while anomalous data instances are very distant from the centroid of their nearest cluster.
c) Normal data instances form a part of large and densely populated clusters while anomalous data instances form a part of tiny and sparsely populated clusters.

The demerits of using clustering based anomaly detection approaches are:

a) The performance of the clustering based anomaly detection approaches mainly depends upon the effectiveness of the used clustering algorithm for identifying the cluster containing the normal data instances.
b) Many techniques are used to identify outliers as a byproduct of using clustering and hence are not best suited for detecting outliers.
c) Many techniques under this category are successful only if the clusters do not perform any significant cluster among themselves.

4) ***Statistical based:*** It works under the assumption that, an outlier is an instance which is either partly or totally not required as it is not produced by the stochastic model. It creates a statistical model for the normal data instances and then applies a statistical interference test if any unknown data point is part of this model or not. Data points which have very low probabilities of being created by the learnt model after applying the interference test is termed as anomalous instances.

   The disadvantages of using statistical outlier detection techniques are as follows:

   a) The main disadvantage is that, the techniques work under the idea that the data is produced from some distribution. This particular idea is not real for data sets with multi dimensions.
   b) Even if the assumption for statistical anomaly detection techniques is true, there exists many hypothesis test methods using statistics for detecting outliers but choosing the best method cannot be found out right. Taking the case of creating the hypothesis test which will fit for the high dimensional real data nowadays difficult to achieve.

Chapter 2

Literature Review

2.1 Introduction

Data Mining is the holistic approach to find meaningful patterns in large sources of data stored in large databases. Data Mining helps companies to analyze data and predict future trends for example a product company before releasing a new product in the market, will conduct survey about the competitive product from competitors in the market and on company's customer's behavior on launch of a new product in the market.

Data Mining is effectively summed as a classification of many tasks out of which Outlier Detection is a part. **Anomaly Detection** is defined as finding data points in data set which do not confer to the notion of well formed/defined data. **Clustering** [26] is the technique of finding groups within the data set which are somewhat similar without using any known data structure with in the data set i.e. grouping together points which form part of a logical group and have their similarity to other such groups in the known data set area. It is used primarily in areas of analyzing the statistical data ranging from data mining, machine learning, information retrieval etc. [2].

Anomaly Detection is similar to, but different from [27] and [28]. From the last few decades multiple algorithms have been proposed in the area of clustering which find their implementation in a wide number of fields [3][2]. The clustering algorithms are categorized into many techniques as: **hierarchical** clustering algorithms like Single link, Average Link and Complete Link [3], **Partitioning** based clustering like *k*-Means, *k*-Medoids, EM clustering, *k*-Harmonic means etc. [4], **density** based clustering like DBSCAN, OPTICS, DENCLUE etc. [3] etc., **grid** based clustering like WaveCluster, STING [3] etc., and many algorithms like Affinity Propagation [5].

Data Clustering surveys. Due to the proposals of a lot of clustering algorithms present in literature which vary in both diversity as well as quantity, much of the efforts has been utilized to summarize all these techniques to provide a picturesque view of these

algorithms. In the field of statistics and data mining, a comprehensive survey of outlier detection is presented in [29]. Specifically for symbolic data, a comprehensive review of outlier detection approaches is provided in [30]. Approaches for novelty search in field of neural networks and statistical methods have been specified in [31] and [32]. Regarding the field of Cyber intrusion [33] presents picturesque approaches for anomaly detection.

The usage of anomaly detection approaches in real life scenario has been cited extensively in literature. For instance, an anomalous traffic map in a computer network could possibly lead to notion of hacking [34]. In the MRI image, an outlier can be the presence of a tumor [35]. Credit card frauds have been extensively reviewed in [36]. An anomalous observation in health of a space aircraft sensor may indicate failure of the system has been reviewed in [37]. Outliers in field of time-series data have been presented in [38].

The main focus area in [6] and [7] is Meta-heuristic clustering algorithms. Surveys [8] and [9] provide importance to high dimensional clustering comprising sub-space, projected and correlation based clustering algorithms. An interesting work has been proposed in [10] which focus on the evaluation of experiments, analyzing subspaces and projected clustering based algorithms. A comprehensive tutorial on spectral clustering based algorithms along with their respective properties is presented in [11]. The usage of spectral processes and kernel approaches for clustering has been portrayed in [12]. Majority of the density based algorithms has been the focused in [13].

One emerging research area in data mining is *Multiple Clustering* is proposed in [14]. In [15], a survey about clustering along with instance level constraints has been well proposed. A survey of application of clustering algorithms in areas like wireless and mobile communication is proposed in [16][17]. Focus on usage of clustering algorithms in field of time series data is present in work [18]. A comparison of twelve model-based document clustering approaches is presented in [19]. Many generic approaches for data clustering has been the main focus in [20][2].

Though the advancements in fields of data clustering has been immense and binding all these progress in any book or survey is very cumbersome to achieve. So, a more

systematic approach and much research effort are required to achieve solution to the said problem.

Density based clustering approaches: which are further categorized into various types like *k*-means which works on the idea that the data is produced internally or externally by some probabilistic distribution like for ex- from a group of *k*-Gaussian distributions. Therefore such algorithms are limited data sets where the clusters are generated in spherical form and fails in cases when data set contains clusters of non spherical shapes [2].

In the figure 2.1, due to its strength of identifying clusters having different densities DBSCAN [1] generates results similar to the ground truth. While in case of *k*-means [4] from the diagram it can be inferred that it's not working correctly as it can only find clusters which are spherical in nature.

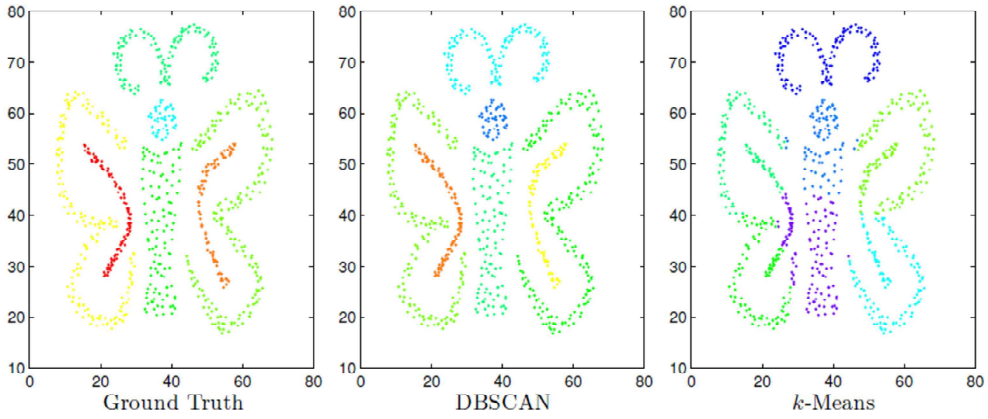

Fig. 2.1 Clustering results on a toy example.

In the field of density based clustering techniques, clusters are defined as regions having large data instance density distant by regions with low data instance density. This idea forms the idea to detect clusters of different shapes by following the densely populated areas of data instances in the data set.

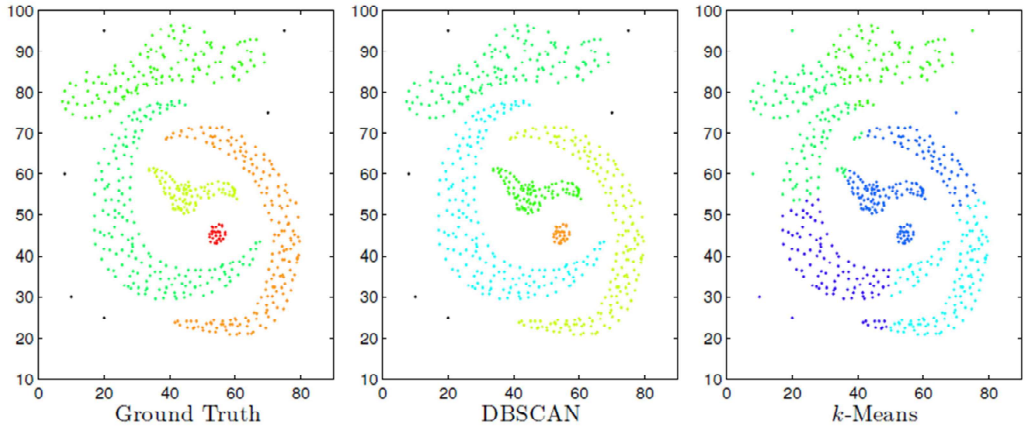

Fig. 2.2 Clustering results for a toy example. Anomalies are marked in black.

In comparison to other traditional approaches for clustering, specifically density-based clustering has the ability to deal with anomalies. In density based approach the anomalies are considered as data instances which belong to sparsely populated data point areas and hence give the idea that such data instances in the data set must have been generated externally in comparison to all other data points present in data set. The above figure 2.2 shows that the DBSCAN [1] algorithm is able to identify effectively the anomalies which are marked as black dots while the traditional *k*-means algorithm is unable to identify these anomalous data points present in data set.

One of the advantages of the density based clustering algorithms is, that it is not mandatory to say the number of *k* clusters before applying the algorithm on the said data set. Since the proposal of the first version of DBSCAN, it has been the focus of many researchers owing to its advantages ranging from gaining robustness against noise or the ability to point out arbitrarily shaped clusters in the data set. There exists many other density based clustering algorithms proposed in the recent past which have their own ideas about the density like the number of neighbors present in neighborhood of each instance in data set [1], finding the influence of a data instance in its neighborhood [21] etc. Among the above said types of density based clustering techniques, the density based idea of DBSCAN algorithm is the most successful idea. Depending on the DBSCAN idea

of density, many other algorithms like GDBSCAN [22], SUBCLU [23] has been proposed.

2.2 DBSCAN

The first version of DBSCAN has been proposed in [1]. DBSCAN gives an approach for the density calculation on the basis of the number of data instances surrounding the considered data instance in the data set. A data instance is said to be part of a group if and only if it has enough data instances above a specified threshold in its neighborhood.

Provided a set of Objects *O* which comprises of *M* data instances, a distance function *d*:

$O \times O \rightarrow R$ and two parameters given by notations $\epsilon \in R^{+}$ and $\mu \in M^{+}$.

Definition 1: *(ϵ −neighborhood) The ϵ −neighborhood of $q \in O$, represented by $M_{\epsilon}(q)$, is represented as $M_{\epsilon}(q) = \{ r \in O \mid d(q,r) \leq \epsilon \}$.*

Each instance in data set *O* is termed one of the three labels namely core point, boundary point or anomaly depending upon the density of the data instance in its neighborhood. A data instance *q* is identified as a core point if the data instance has more than μ data instances in its ϵ −neighborhood. If the data instance *q* has less than μ data instances in its ϵ −neighborhood and if none of the data instances in its ϵ −neighborhood are core points, then the data instance *q* is designated as an anomaly. Else, *q* is designated as a border point.

Definition 2: *(Core point property) An object $q \in O$ is a:*

1. *Core point, represented as core(q), iff $|M_{\epsilon}(q)| \geq \mu$.*
2. *Boundary point, represented as border(q), iff $|M_{\epsilon}(q)| < \mu$ and $\exists_{p} \in M_{\epsilon}(q)$: $|M_{\epsilon}(p)| \geq \mu$.*
3. *Noise point, represented as noise(q), iff it is neither a core or a boundary point.*

A data instance *p* is said to be density reachable from another data instance $q \in O$ if the data instance *q* is a core point and the data instance *p* lies inside ϵ −neighborhood of the

data instance *q*. It is noteworthy that, the instance *p* is density reachable from the instance q necessarily does not imply that the data instance q is also density reachable from the data instance *p*.

Definition 3: *(Directly density-reachable) A data instance $p \in O$ is said to be directly density reachable from another data instance $q \in O$, which is represented as $p \rhd q$, iff $|M_\epsilon(q)| \geq \mu$ and $p \in M_\epsilon(q)$.*

Two data instances *p* and q are said to be density connected to each other if there is present a chain of density reachable core data instances x_i such that the data instance *p* is density reachable the core distance x_i and the data instance *q* is already density reachable from the core data instance from x_i. It is important to note that the data instances *p* and q need not be necessarily core data instances.

Definition 4: *(Density-connected) Two data instances q and $p \in O$ are said to be density connected to each other represented by $p \bowtie q$, iff there is a sequence $(x_i,\ldots,x_n)$ of data instances present, such that $\forall x_i : |M_\epsilon(x_i)| \geq \mu$ and $q \lhd x_1 \lhd \ldots \rhd x_m \rhd p$.*

A cluster is termed as maximal collection of density connected data instances which comprise of core and boundary points. In case of DBSCAN, the boundary point can be a part of many such clusters which depends upon the order of the data instances. A data instance which is not part of any such cluster is termed an anomaly.

Definition 5: *(Cluster) A subset $B \subseteq O$ is called a cluster iff the following two conditions have been met:*

1) *Maximality: $\forall q \in O, \forall p \in O \setminus C : \neg q \bowtie p$*
2) *Connectivity: $\forall q,p \in C : q \bowtie p$*

The DBSCAN algorithm uses a specific data structure termed as Seed list *S* which in turn consists of a set of data instances used for expanding a cluster(s). In order to start its functioning, the algorithm first marks all the data instances as *Unclassified.* Following this, an unclassified instance is chosen and inserted inside the empty Seed list *S.* Then the algorithm continuously takes out an instance q from the seed list and does the ϵ −check to find out the number of neighbors of the concerned instance. If there exist, data

instances in the data set which are directly density reachable from the data instance *q,* then such instances are inserted into *S* if they had been marked as *Unclassified.* The above process continues until *S* is empty and all instances have been classified as either as *core, boundary or outlier* instances.

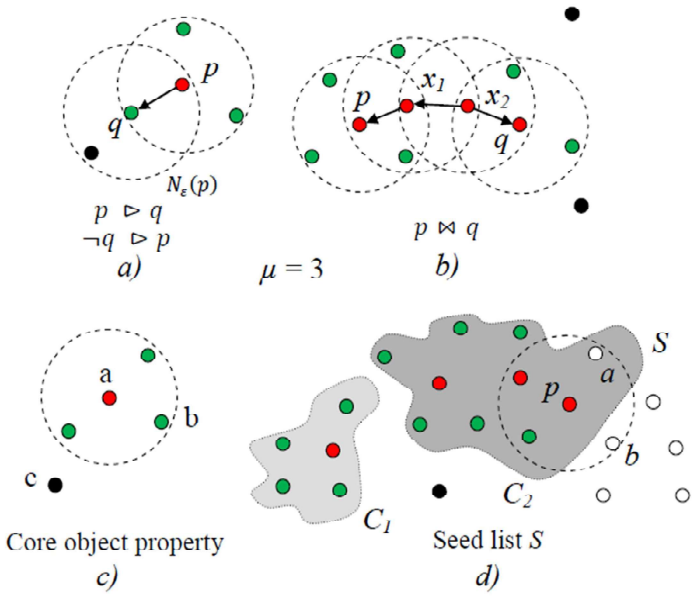

Fig. 2.3 The idea behind DBSCAN:(a) data instance *q*,directly reachable from data instance *p.* (b) data instances *p* and *q* is density connected. (c) Data instance *a (red)* is a core point, *b (green)* is a boundary point, *c (black)* is an anomaly. (d) The seed list *S* used for expanding the cluster.

The DBSCAN algorithm in the above figure 2.3 is building the cluster C_2. Data instance *p* is being taken from the seed list *S* which is used for expansion and is being examined. Data instances *a* and *b* which are present inside ϵ −neighborhood of the data instance *p* have not been processed and therefore are being inserted into the seed list *S.*

The complexity of the DBSCAN algorithm is $O(M^2)$ where M is the total number of data instances in the set of data. But when we use another algorithm like *R*-Tree [24] is used for performing the ϵ −range check for all the data instances then the complexity drop to *O(M logM).*

The figure. 2.3 above gives some idea about the DBSCAN algorithm which includes the direct density reach ability (a), the density connected idea (b), the core point classifier of a data instance (c) and the process of cluster expansion (d).

2.3 OPTICS

One of the major disadvantages of DBSCAN is, it uses the single parameter ϵ to measure the density around each point of the data set. It is well known that the parameter's choice in real time applications is a big problem. Along with that the clustering structure can't be identified by a global clustering approach. The different clusters can only be distinguished using different choice of parameters on the data set. One approach is to apply the DBSCAN algorithm again and again along with different arguments in order to find the different clusters. This approach leads to degradation in performance and is also not considered an apt approach to solve the problem. OPTICS [25] has been proposed to cope up the above mentioned problem.

In comparison to DBSCAN, OPTICS does not create groups explicitly. Instead, OPTICS builds an ordering of instances in the given data set which holds the information about many clusters with respect to different values of input parameters i.e. ϵ which are less than a specified threshold value ϵ^*. The output of OPTICS is a graphical structure called the *Reach ability plot* which helps in easy analysis of the cluster structure as shown in Fig. The concept of OPTICS is based upon on two measures: *core distance* and *reach ability distance* on any data instance *q* in the data set.

Given a set of data instances *O* consisting of M instances, a distance function $d: O \times O \rightarrow R$ and two parameters ϵ^* R^+ and $\mu \in M^+$. The core distance of a data instance *p* can be represented as:

core-distance $(\epsilon^*,\mu)[q]$ and defined as:

1) UNDEFINED if $|M_\epsilon(q)| < \mu$
2) k-distance(q) otherwise

Where the *k*-distance (*q)* is referred as distance between the data instance *q* and its *k*-th nearest data instance. The reach-ability distance of a data instance *q* w.r.t another data instance *o* can be represented as:

1) UNDEFINED if $|M_{\epsilon^*}(o)| < \mu$
2) *max*(*core-distance* (ϵ^*,μ)*[o], d*(*o,q*)) *otherwise*

OPTICS algorithm creates an algorithm of objects, and at the same time stores the core distance and reach-ability distance with respect to the previous instance in data set. The reach-ability graph is built from these distances for each instance to understand the creation of clusters.

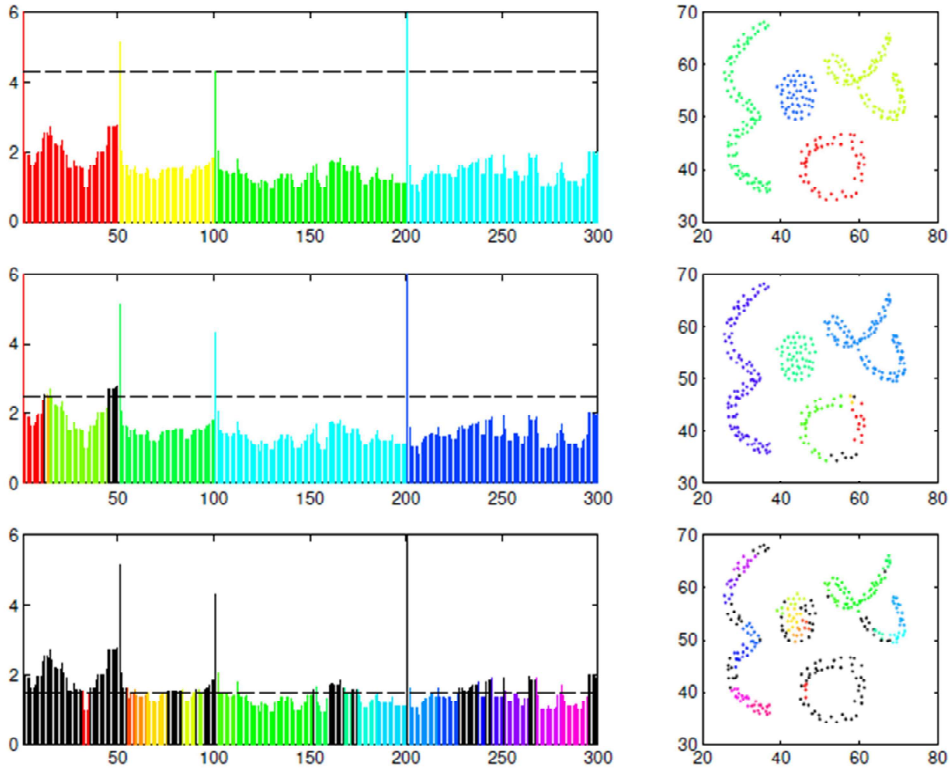

Fig. 2.4 The reach-ability diagrams (left) of the clustering results (right) of OPTICS

The above figure 2.4 represent the reach-ability plots of the clustering results of OPTICS for different values of input parameters. Anomalies are represented in black. The value of

the threshold ϵ^* has been set to 6. From the top to bottom the input parameter ϵ has been set to 3, 2.5 and 1.5. The time complexity of OPTICS algorithm is similar to the DBSCAN algorithm. But as quoted by some researchers the OPTICS algorithm is slower than the DBSCAN algorithm by a factor of 1.6. As already quoted, the usage of an indexing structure for storing the distances and finding the data instances for creating a cluster can reduce the complexity to *O(M logM)*.

Chapter 3

Hadoop and Map Reduce

3.1 Hadoop

It is an open source framework managed by Apache Software foundation, for writing and running programs that process huge data sets. The key points which make Hadoop invincible include:-

1) **Accessibility:** Hadoop can either run as a single node configuration, multi-node configuration where the hardware of the participating machines in the cluster construct a File System commonly known as Hadoop distributed File System (HDFS) or on cloud platform.

2) **Robustness:** Hadoop has been designed to overcome the shortcoming of failure of hardware. So, the data stored on commodity hardware of the participating machines in the cluster are replicated across multiple places to achieve high fault tolerance.

3) **0Scalability:** Hadoop can handle data ranging from linear data to huge categorical data of large data sets by adding multiple nodes in the cluster.

4) **Simplicity:** Hadoop has the taste of simplicity introduced by enabling everything to be done in Map Reduce and use the built in functions in the Hadoop framework.

Hadoop multi-node cluster is nothing but a set of machines combined together such that any large data set to be processed is placed on the combined data capacity of the commodity hardware of the machines in the cluster. Any participating machine can place any job to be worked upon through clients present on the system itself or through any remote machine existing in the cluster itself.

3.1.1 Hadoop Distributed File System

It is a distributed file system which works on commodity hardware of the cluster. By commodity hardware we mean the combined hardware capacity of the participating machines in the cluster. It is used to process very large data sets. It has many advantages over other file system like it is highly fault tolerant, provides high throughput accession and major factor is that it has very low cost.

3.1.1.1 Characteristics of Hadoop Distributed File System

1) **Fault Tolerant:** If any node in the cluster wears out, then any other node can take its place but the operations on the data set will not be stopped. Hence the breaking of one node does not promise the broken status of the entire cluster. Part of this is achieved by making the data set on each node to be copied over multiple stations so that even if one node fails, then the data does not need to be copied back to HDFS.

2) **Streaming Data:** It provides streaming access to the large data sets. It has been designed to work in batch processing fashion rather than with interaction by the users of the system. High importance is given to throughput rather than the low access rate of the huge data set.

3) **Scalability:** HDFS can easily add huge number of nodes as the need arises to the cluster and providing commodity hardware to store the huge data sets making it possible to scale the data for the application.

4) **Large Datasets:** HDFS has been designed essentially on the principle of processing large data sets in order of Gigabytes to Terabytes in a batch processing approach.

5) **Simple Coherency Model:** It creates a write-once-read-many time policy for any file stored in HDFS. A file uploaded into the file system, opened, written and closed does not need to be created again and again. This approach solves the

problem of data coherency and in addition increases the data access capability of the huge data sets.

6) **Transferring Computation is cheaper than Transferring Data:** If we execute the data close to the place where it resides then it can speed up functioning of the application. It holds true in case where data is very large and overcomes the problems like network congestion and provides better throughput access. So, the assumption made is that it is better to transport the execution of the application closer to the place where the data resides rather than vice versa.

7) **Portability across Heterogeneous Software and Hardware Platforms:** It is designed to work across platforms independently i.e. write once run anywhere approach which exists in Programming language Java.

3.1.1.2 HDFS Architecture

HDFS cluster broadly contains NameNode and DataNode. NameNode is responsible for managing the metadata for the respective cluster while DataNode is responsible for storing the actual data. NameNode is usually located on the master machine while the DataNode functions in the individual slave nodes. Files and Directories are represented in the NameNode as *inodes* as existing in traditional Linux systems. Inodes keep track of information like permissions allowed, accessing and modification times of the files, the owner permissions for the files, the Space quotas for the disks as well as namespace for the disks. The file contents are typically broken down in sizes of blocks each of size of **128MB** in general where the block size data is copied independently at multiple DataNodes. The blocks are kept on the local file system used to create the HDFS and where the DataNodes run on the individual slave machines. The NameNode present on the master machine continuously keeps the track and health of the data replicated on the individual DataNodes of a block respectively. If the DataNode containing a block goes down then NameNode creates another copy of the block on another DataNode. The NameNode takes the responsibility of maintaining the Namespace tree along with

matching of the data blocks to appropriate DataNodes which hold them, holding the entire information regarding the namespace on the RAM.

The DataNodes at regular intervals send the health report of the data blocks residing on them to the NameNode. The NameNode in turn sends instructions back to the DataNodes like copying data blocks to other DataNodes, removing the local block replications, registering the DataNode again and sending the block report or just power off the DataNode. The HDFS architecture containing DataNodes, NameNode are represented in Fig. 3.1.

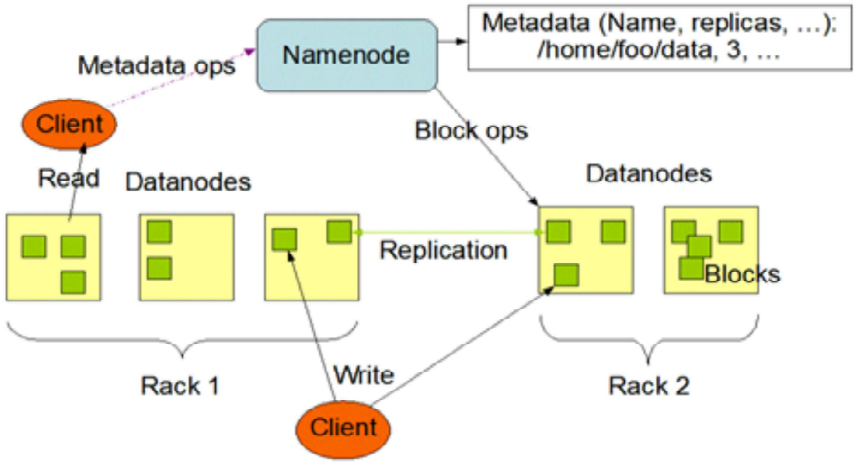

Fig. 3.1 HDFS Architecture

1) **Data Replication:** HDFS makes the data to be copied across multiple DataNodes to achieve fault tolerance. An application can control the number of places where the data will be copied by setting the replication factor, which can be changes later as the need arises. The guardian who takes all decisions regarding the amount of copying to how many nodes is given to NameNode. High reliability and throughput is achieved using a good approach by the HDFS. Having a check on the number of copying places for the data blocks makes HDFS different from other distributed file systems in the market, which owes much of its ability due to the special rack-aware copy management facility which uses the network

bandwidth very wisely and properly. Large HDFS usually require many computers to be part of a cluster. So, the NameNode rules the optimum communication between the DataNodes present on the individual slave nodes. The NameNode can individually identify the DataNodes by their special identity numbers called rack-IDs.

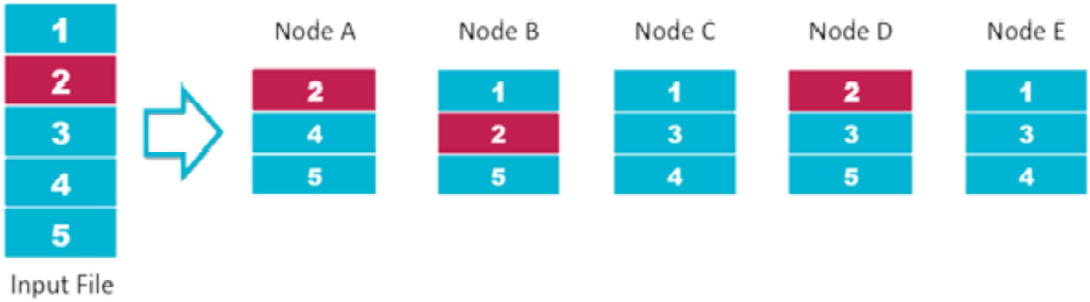

Fig. 3.2 Data Replication

In the figure 3.2 above the input split (original data) is divided into five splits to be copied across the individual DataNodes. Note one more important thing that each part in each smaller split is replicated across three separate Nodes. In case of any one DataNode failure the copies from other DataNodes can be used to carry on the computation which shows why HDFS is termed as **fault-tolerant** in nature. If all the replicas of the each part of the split are present on the same DataNode then if the DataNode fails then all the copies would have been lost.

2) **Robustness:** The main goal of HDFS is to maintain the multiple copies of the same data and store them effectively even if there is failure of any DataNode. The common types of failures that can occur most frequently are NameNode malfunctions, DataNode malfunctions and network partitions.

   a) **Data Disk Malfunction, Heartbeats and Re-Replication:** The DataNodes at regular intervals send health report which can have its similarity to humans as pulses to the NameNode stating that it is healthy and alive. When the NameNode is not able get the pulse from the

DataNode after the fixed time interval then the NameNode blocks the DataNode and does not send any Input-Output requests to be operated on the DataNode. The data block which was copied on the DataNode which failed is now unavailable and HDFS takes on the task to use another copy of the lost data block from any other DataNode or copy the data block back into HDFS if all the DataNodes that contained the copies of the unavailable data block failed. The reasons for copying the data once again may arise due to many factors such as a DataNode may fail, a copy of the data block may be corrupted or replication factor may be increased such that the required file size after copying data to many DataNodes may cross the available size of the file system.

b) **Cluster Rebalancing:** A rebalancing approach may appropriately copy the data block from one DataNode to another if the available space of the DataNode falls below a marked threshold. In situations where there is an increased requirement of the space, the scheme might increase the copies of the data blocks to other nodes which can be added dynamically to the cluster.

c) **Data Integrity:** Situations can happen where the intended data block received from the DataNode can be faulty. The above mentioned fault can be due to one of the following reasons like storage issues, network issues or due to the fault of the software. So, to overcome the problem HDFS computes the checksum on the data blocks. When a HDFS file is created, HDFS computes the checksum of each of the blocks in the file and stores these checksums in one separate file which is hidden called the HDFS namespace. When the client approaches for a file from the DataNode, it performs checks to see that the data blocks it got matches the checksum from the associated namespace file. If the checksums of both the files do not match then the client can take another option of copying the data block from another DataNode which has copied the data block.

d) **Metadata Disk Failure:** The NameNode manages two data structures which if failed can cause massive failure to the entire installations are called *FsImage* and *EditLog*. Any update to either of the data structures can lead to updating the data structures synchronously. The upgrading of the data structures synchronously on multiple nodes can lead to degrading of the namespace transactions per unit time that the namespace supports. This feature can be blessing in disguise as despite being data intensive in nature HDFS is not metadata intensive in nature. When the NameNode is shut down and restarted, the NameNode takes the most consistent latest EditLog or FsImage into consideration. For a HDFS cluster, the master machine which has the NameNode is the single point of failure. In case the master machine fails then it can only be corrected by manual intervention and no automated tool can start the system again.

3.1.2 Map Reduce Programming Model

It is a programming model which provides assistance in implementation for working and creating huge data sets. The model is influenced by the Google File System, which also happens to be a distributed file system for handling and executing large data sets. Programs written on Map Reduce model format, automatically partitions the large input split into block sizes of 128 MB separately on individual DataNodes separately which exist on different slave machines existing in the cluster. The input which is fed to the Map phase is a set <Key, Value> pairs which in turn generates another set of intermediate <Key, Value> pairs which are fed to Shuffle phase before handling to the Reducer phase. During the shuffle phase, all values which map to the same key are passed on to the same reducer. The Reducer then works on all the values which match a corresponding key and then generates another set of <Key, Value> pairs. Let us take some examples to demonstrate how the Map Reduce model helps in working on problems in a parallel fashion when the dataset is large.

1) **Example 1:** Let there be a huge collection of documents and we are given the task of finding the count and occurrence of each distinct word that occurs in all of the documents. The input file which is given as input will produce a set of <Key, Value> pairs. As the general format of giving the TextInputFormat as the input format for Hadoop Map Reduce programming model, an unique offset is generated which behaves as key while the remaining data on the line acts as the value assigned to that particular key. These <Key, Value> pairs form the input part of the Map-Reduce phase. The Map part generates each word as key and then sets a value '1' against each of these words which is designated as value for that word i.e. key. The Shuffle part will collect all the values i.e. 1's for a particular key. In the final run the reducer will sum all the 1's for a particular key and produces the sum as value for the particular key. In addition we can have another phase called the Combiner phase which exists between the Shuffle and the Reducer phase.

   The combiner worker on the key, value generated from the map phase and collection of the intermediate 1's for a single map output for a single key are sent to the reducer part to do its work. The Combiner facilitates the lower overhead of the network and resources to be used for computation function as it functions on the output from the Map phase and the reducer part has to do with less work as compared to work it has to do in absence of the Combiner part.

2) **Example 2:** Let us have a collection of tables. Our task is to perform the Join of the tables. But the need of the task is with what efficiency we can achieve the task?

For instance we have the Student table containing column names as depicted in table 3.1 and the Branch Table depicted in table 3.2 while the join relation Student ⋈ Branch is shown in table 3.3.

TABLE 3.1 Student Table

Name	RollNo
Sourajit	801431029
Rajeev	801331021

TABLE 3.2 Branch Table

Branch_RollNo	BranchName
801431029	Software
801331021	InformationSecurity
801331021	ComputerScience

TABLE 3.3 Student ⋈ Branch Table

Name	RollNo	Branch_RollNo	BranchName
Sourajit	801431029	801431029	Software
Rajeev	801331021	801331021	InformationSecurity
Rajeev	801331021	801331021	ComputerScience

Above data seems to be very small in size but if we consider the dataset to be large then we can solve the problem using Map Reduce approach with the following steps.

Step 1: Bring together all the data from the rows along with the respective tables in a single dataset.

Student, Sourajit, 801431029

Student, Rajeev, 801331021

Branch, 801431029, Software

Branch, 801331021, InformationSecurity

Branch, 801331021, ComputerScience

Step 2: Consider the common value upon which the join is made between the tables, its values are used as the key and the row is then considered as the value. It is done so that when the data is fed to the shuffle phase then all the keys being shuffled then all the rows having common value of the attribute on which join is performed will be fed to the same reducer.

Key: 801431029, Value: (Student, Sourajit, 801431029)

Key: 801331021, Value: (Student, Rajeev, 801331021)

Key: 801431029, Value: (Branch, 801431029, Software)

Key: 801331021, Value: (Branch, 801331021, InformationSecurity)

Key: 801331021, Value: (Branch, 801331021, ComputerScience)

Step 3: Individual reducers will then do the join of the rows sent to them as input.

The reducer having the following input row:

Key: 801431029, Values: [(Student, Sourajit, 801431029), (Branch, 801431029, Software)] gives the following output: Sourajit, 801431029, 801431029, Software

Similarly for the input record:Key: 801331021, Values: [(Student, Rajeev, 801331021), (Branch, 801331021,InformationSecurity), (Branch, 801331021, ComputerScience)]

gives the following output:

Rajeev, 801331021, 801331021, InformationSecurity

Rajeev, 801331021, 801331021, ComputerScience

3.1.2.1 How Map Reduce Works?

When a job configuration is submitted to JobTracker by client thus beginning of program execution, the job configuration prepares the consequent information starting from specifying the map, the combine and the reduce function along with the path directories for the input and output separately. The JobTracker will try to find out the number of input splits which is configurable in the size of 16-64 MB from the directory of the input path.

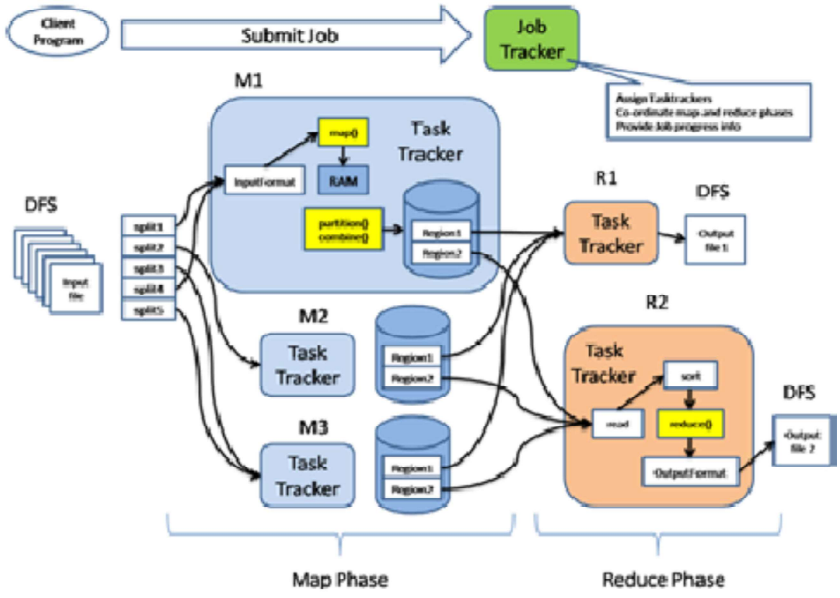

Fig. 3.3 Working of Map Reduce

Figure. 3.3 represent the working of Map Reduce programming model starting from the submission of job from client system, the usage of HDFS, the computations in Mapper and Reducer phases. JobTracker then finds TaskTracker based on their distance from the

data sources following which the JobTracker sends the requests for tasks to some selected TaskTrackers. Then the data is extracted from the input splits which will be fed to the map by the TaskTracker. For each record which is read by InputFormat, it calls the user defined map function which generates a set of <key, value> pairs which is to be stored in the memory buffer.

A timely wake up process will help in sorting the memory buffer into the different reducer nodes by invoking the combine function on the key, value pairs which have been generated from the map function. The <key, value> pairs generated are stored, sorted into one of X local files, where X reports the number of reducer nodes. When the processing has been done on the input splits i.e. map task is completed the TaskTracker will send a message to the JobTracker. When all TaskTrackers are finished, JobTracker takes upon the task of notifying the necessary selected TaskTrackers for the reducer phase. Every TaskTracker will have a scan of the input region files remotely. It will perform the action of sorting the key, value pairs for each of the keys followed by calling the reduce function which collects the key-assembled values into the appropriate file which is created one for each reducer node.

The Map Reduce programming model is fail-safe to failures of any of its components. The prime component is the JobTracker checks at regular intervals the status report of the individual TaskTracker(s). In case any of the TaskTracker crashes, the JobTracker takes the work to re-assign the map task to any other running TaskTracker and re-run all the work on the splits. If the TaskTracker present at the reducer part crashes then the JobTracker assigns the responsibility to run the reducer task at any other living TaskTracker. After both the map and reducer phases are completed the client program which submits the job for work is released by the JobTracker.

3.2 Open research in Field of Big Data

Some of the research areas of interest in field of Big Data are as follows:

1) **To find similarity among items:** Elements can be combined to behave as a set and finding similarity among elements is like finding sets which have a large fraction of commonality of elements. Minhashing and regional sensitive hashing plays an important role in finding the similarity among sets in many applications which otherwise pose a big problem when handled with large data sets.

2) **To Mine Streams of data:** Data present in the form of stream can be a challenging task of performing operations on them. Differentiating between data obtained as a stream and a database is that, if not stored as it is received then we won't be able to do anything for data and it gets lost. Some basic examples include the search query done in surfing many search engines [39] or the page forward requests obtained [41] when we click a link on a web page. New data is collected at regular intervals and the data processing tool needs to work on this updated data as well [42].

3) **Performing PageRank Analysis:** Idea is based on what are the web pages a surfer will tend to visit the most and is a part of an important idea for many search engines. This particular idea has made Google stand out from other search engines available in the market. Fighting battle against classification of e-mails as spam is also an important research issue which is an extension to this PageRank problem.

4) **Finding Frequent Itemset Mining:** It is based on the market basket model. In this approach data is considered to have large number of baskets which in turn contain a small set of items each. Analyzing and choosing the best algorithm to find all possible frequent pairs of items which appear across multiple baskets is a very tedious task to complete. The prime factor to be considered while choosing the appropriate algorithm will be the performance factor.

5) **Anomaly Detection:** Anomaly is defined as a point in a dataset which do not confine to the notion of a normal data point. It is all about finding points in the dataset which reside far away from cluster of points i.e. in simple terms finding points which reside in areas of low neighboring point density. Finding out

algorithms that can detect the anomalies very effectively is a big problem when using large datasets for the same.

6) **Clustering:** Assuming a set of points with a distance measuring scale classifying the closeness of a point to other points. Primary importance is to examine large sets of data and partition them into clusters. Each cluster containing points which are closer to each other than points which are very far apart in other clusters. When working with data sets which are huge, studying them and finding out their characteristics, generalizing them is a very tedious task to achieve.

7) **Online Advertising:** Providing good algorithms such that advertisers pose their product on the internet and fight for the display in response to the requirements searched by a surfer online in a search engine is a very big problem since huge amount of search queries are being searched across many search engines per minute worldwide.

8) **Recommender Systems:** Many applications on the internet expertise to suggest the surfer what to search next [40] based upon their search pattern. One such example is usage of Netflix website where it has to predict what movies will be liked by the user next depending upon the viewership analyzed by the website or in case of Amazon show a pop up or a page on a website about what the user has already searched on its website and showing related results so that the user can buy the relevant items.

Among the various open issues in the area of Big Data, with health care being an emerging field [43], the thesis is based upon the comparison of the results of two density based clustering algorithms when run on a multi node cluster setup of Hadoop.

Chapter 4

Research Problem

4.1 Problem Statement

Here we throw some light on gaps encountered during the research by taking into account the comparison of the algorithms already present in literature in the area of performance comparison of two density based clustering approaches namely DBSCAN and OPTICS for anomaly detection when simulating them on multi node hadoop clusters.

Density based clustering has been the most researched topic in last few decades. It is used for detecting anomalies i.e. instances in data set which do not confer to some well defined behavior finds usage in variety of cases like credit card fraud, system failure, intrusion detection etc. Clustering algorithms suffer from one major problem that they cannot do their computations until the entire data set is available to them.

When working with huge data sets, working on a single node is practically not advantageous as it is very time consuming. Similarly doing all the computations of identifying the outliers in mapper or reducer function in entirety also suffers from the same problem which single machine hadoop implementation suffers. With the Map Reduce programming approach which reads one record at a time, passes it to mapper does computations and passes the intermediate output to reducer fails big time when working with huge data. Moreover most of the proposed algorithm(s) for running DBSCAN and OPTICS on the Hadoop framework lag in terms of having large false alarm rate of identifying the anomalies.

In order to overcome above mentioned problems, we implement an approach to solve the data-set to be available in entirety problem along with a two-step approach for doing the computations taking the advantage of two-step approach of Hadoop framework. This approach has been implemented using Java programming language.

4.2 Objectives

After going through the gaps in research and the problem which is to be solved, following outlined are the objectives of the thesis:

1) To study and explore the concepts about Big Data, Hadoop, Anomaly Detection and related terms.

2) To propose an approach for running DBSCAN and OPTICS on Big Data.

3) To implement the proposed approach for DBSCAN as well as OPTICS on Hadoop framework in Map Reduce programming model.

4) To test the results after running DBSCAN and OPTICS on real data set.

4.3 Methodology

In this chapter, we will have a look at the approaches that have been devised to get the benefits of usage of hadoop framework by implementing algorithms on multi node cluster. We will be discussing about the approaches specifically employed in Map and Reduce phase separately rather than talking about the general working of DBSCAN or OPTICS. The problem of having the entire data for clustering at one place has been solved by using the Hadoop distributed File System (HDFS). Under this approach the entire input file is copied to a temporary file(s) generated according to the multiple nodes in HDFS and then operations are performed on the data there in. The algorithm can be applied on multiple machines with the Mapper code run on all the slave machines and the entire merging of the intermediate results will be done on the Reducer phase working on master machine.

The general steps followed in the approach are:-

1) Provide the input file to the program
2) Copy the input file to machine centric temporary file on HDFS & count records
3) Read the temporary file in the Mapper phase.
4) Apply the required algorithm on the data set in Mapper phase.
5) Generate the required intermediate <key, value> pairs from Mapper phase.

6) Gather the relevant intermediate <key, value> pairs in Reducer phase.
7) Re-apply the original algorithm on the gained <key, value> pairs.
8) Emit the required <key, value> pairs to the output file.

All the above steps mentioned represented in form of a flowchart 4.1 as follows

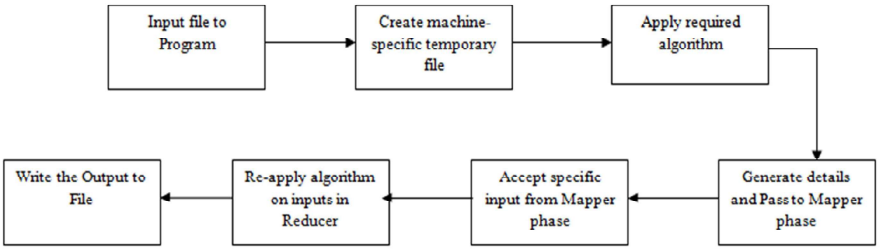

Fig. 4.1 Flowchart of Methodology followed

Now analyzing the working performed in Mapper phase where we use a static variable and array lists explicitly is represented in pseudo code depicted in figure 4.2 as follows:

```
Function Cluster-Mapper (key, value, context)
  I = Static-variable
  A, C, B, O =  Array-List
  L =  Line-Count
  for all input records R_i do
                    if  I equals 0 then
                        Open machine specific temporary file F in HDFS
                        Store the record in A
                        Write record to temporary file F
                        Increment I
                    endif
                    else
                        for all remaining records R_{i+1}....R_m do
                            Write record to temporary file F
                            Store the record in A
```

```
                    Increment I
                  if I equals L then
                      Apply algorithm on input file F
                      Store core, boundary and Outliers in C, B, O
                      Write size of B + size of O to context
                      for all records in C do
                          Append record and string Normal
                          Write to context key and value
                      endfor
                      for all records in B do
                          Append record and string Boundary
                          Write to context key and value
                      endfor
                      for all records in O do
                          Append record and string Noise
                          Write to context key and value
                      endfor
                  endif
              endfor
          endelse
      endfor
```

Fig. 4.2 Pseudo code for Mapper Phase

Now let us look at the pseudo code of approach in Reducer phase which is depicted in figure 4.3 where we get the intermediate <key, value> pairs from the mapper, but only accept those values which have been declared as *Noise* from application on respective algorithm in Mapper phase.

```
Function Cluster-Reducer (key, value, context)
  Input, C,B, O =  Array-List
  I = Static variable
  for all input records R_i from Mapper do
```

```
            if key contains "Boundary" or key contains "Noise" then
                Store the record in input
                Increment I
            endif
        else
            if I equals size of input then
                Apply algorithm on records in input
                Store the core, boundary and Outliers in C, B and O
                for all records in C do
                    Append record and string Normal
                    Write to context key and value
                endfor
                for all records in B do
                    Append record and string Boundary
                    Write to context key and value
                endfor
                for all records in O do
                    Append record and string Noise
                    Write to context key and value
                endfor
            endif
    endelse
endfor
```

Fig. 4.3 Pseudo code for Reducer phase

Getting an insight on the pseudo code will set on the implementation approach that we are following in this thesis. Kindly note that the application of algorithm may it be DBSCAN or OPTICS in Mapper or Reducer phase is the choice of the concerned person but the approach will remain same for both the algorithms.

4.4 Problems faced during Implementation

1) The distance field used to calculate the density of a data instance in its neighborhood required to create a large 2Dimensional array of size of millions by millions which led to **OutOfSpaceError** and **StackOverFlow** Error. So, to best of the knowledge we had to discontinue the usage of the 2Dimensional array.
2) Clustering required the entirety of the data set should be available, but the Map Reduce paradigm was not helpful in this regard. So, a machine specific temporary file has been generated to store the input data as they are received from data partitioned by hadoop for each node.
3) The amounts of time taken for doing computation(s) in the mapper and reducer phases were huge. So, we used the best of our programming efforts to reduce the same.

Chapter 5

Implementation and Results

Both DBSCAN and OPTICS are implemented in Map Reduce programming approach and run on real data sets of 15k, 20k, 25k and 30k data instances each having 10 attributes and their efficiency in finding the outliers has been compared in the next section along with the time taken to complete the task(s) on multi node hadoop cluster testing environment. The version of Hadoop used in 2.6.0. The hadoop cluster consists of 4 nodes and each node has the following configuration Intel® Core(TM) i5-5200 2.20 GHZ having 8GB RAM and Ubuntu 15.10 as operating system running. The choice of Input parameters has been taken as ϵ − 4 units and Minimum points as- 8 units.

Table 5.1 Result set for 15000 Points

Name of Algo	Anomalies after Mapper phase	Anomalies after Reducer phase	Execution Time (in seconds)
DBSCAN-MR	4155	2774	24
OPTICS-MR	5197	3514	41

ϵ − 4 units, MinPts- 8 units, Data set- 15000

Table 5.2 Result set for 20000 Points

Name of Algo	Anomalies after Mapper phase	Anomalies after Reducer phase	Execution Time (in seconds)
DBSCAN-MR	2430	2168	27
OPTICS-MR	3097	2761	47

ϵ − 4 units, MinPts- 8 units, Data set- 20000

Table 5.3 Result set for 25000 Points

Name of Algo	Anomalies after Mapper phase	Anomalies after Reducer phase	Execution Time (in seconds)
DBSCAN-MR	2084	1799	38
OPTICS-MR	3371	2787	61

ϵ – 4 units, MinPts- 8 units, Data set- 25000

Table 5.4 Result set for 30000 Points

Name of Algo	Anomalies after Mapper phase	Anomalies after Reducer phase	Execution Time (in seconds)
DBSCAN-MR	2389	1907	53
OPTICS-MR	2988	2461	83

ϵ – 4 units, MinPts- 8 units, Data set- 30000

The result(s) from the implementation of the two algorithms has been summarized in the above table 5.1, 5.2, 5.3, 5.4 with the same set of input parameters ϵ – 4 units, MinPts- 8 units but different data sets of 15000, 2000, 25000 and 30000 instances in the real data sets. From the table(s) it is evident that, both the algorithms are able to find the outliers in any given data set, but the key factor is the execution time which includes summation of both the time taken in Mapper as well as reducer phase. It is evident that the execution time of OPTICS-MR is around 1.5-1.6 times that of execution time of DBSCAN-MR. The graph of the execution times of the algorithms is represented in figure. 5.1.

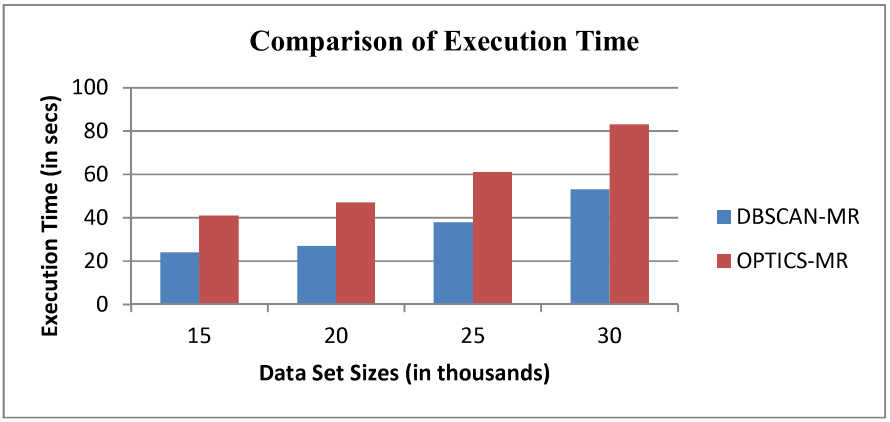

Fig. 5.1 Comparison of Execution Time of Clustering algorithms

Plotted above which clearly shows relevance to the data in the table and the fact about more running time of OPTICS-MR over DBSCAN-MR when both are run on the same specifications in a multi node hadoop environment.

The time consumed by the algorithms to run the same set of points on increasing number of nodes present in the cluster are represented in figure. 5.2.

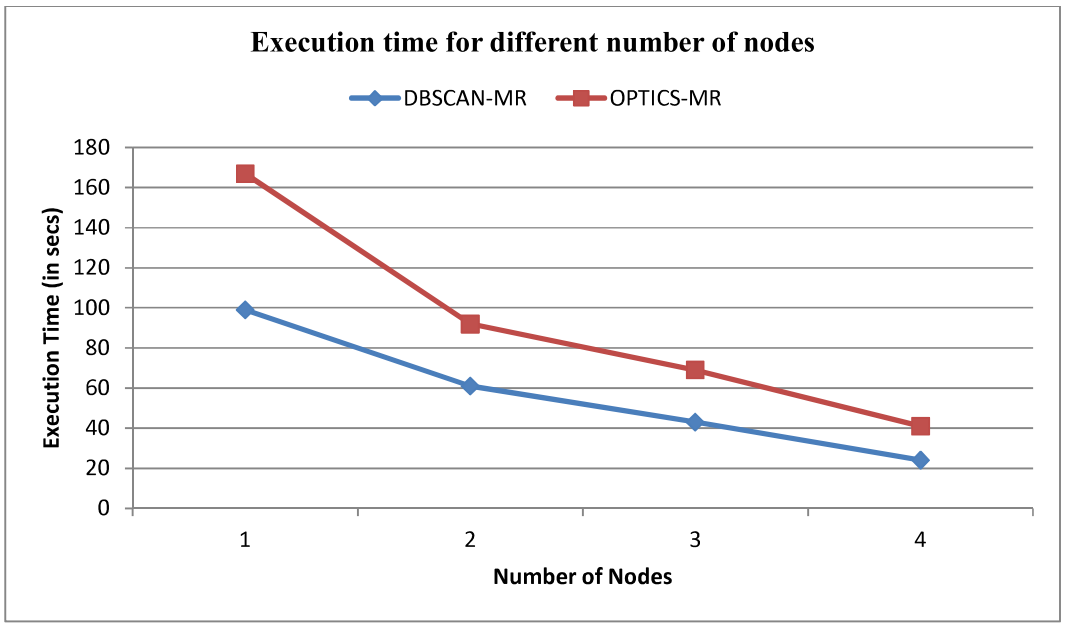

Fig. 5.2 Execution times on increasing nodes

From the Fig. 5.2 it is observed that, with the increase in the number of nodes in the cluster, the execution time of both DBSCAN and OPTICS decreases rapidly. This helps in identifying the fact that the number of independent systems in the cluster helps to do the bulk work partitioned among slaves in cluster thereby reducing the pressure on master node. It is noteworthy that, OPTICS is still taking considerable time over DBSCAN, even if we have an increase in number of nodes, and the factor is about 1.5-1.6.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we presented the two density based clustering algorithms DBSCAN and OPTICS following Map Reduce paradigm, implemented them on multi node clusters on different sizes of data set and are able to conclude that the execution time of OPTICS is around 1.5 times the running times of DBSCAN.

With the increase in the number of nodes it can be seen that for both the algorithms there is gradual decrease in the execution time for the different data sets. So, with an increase in the number of nodes or systems in the cluster we can achieve better execution time for processing the data set.

It is inferred that, even with increasing the number of nodes in a step wise manner in the cluster, the execution time of OPTICS for the same data set is greater than the running time of DBSCAN by a factor of 1.5.

6.2 Future Work

Future work can be done on making the implementation of DBSCAN and OPTICS more efficient by using data partitioning approach as well as using efficient data structure to store and find the distances between data instances so that density can be calculated of instances easily.

References

[1] M. Ester, H. P. Kriegel, J. Sander and X. Xu, "A Density- Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.", in Proceedings of *2nd International Conference on Knowledge Discovery and Data Mining*, vol. 96, no. 34, pp 226-231, 1996.

[2] C. Aggarwal, *Data Clustering*. Hoboken: CRC Press, 2013.

[3] J. Han and M. Kamber, *Data mining*. Amsterdam: Elsevier, 2006.

[4] A. K. Jain, "Data clustering: 50 years beyond k-means", *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651-666, 2010.

[5] B. J. Frey and D. Dueck, "Clustering by passing messages between data points", *Science*, vol. 315, no. 5814, pp. 972-976, 2007.

[6] S. Das, A. Abraham and A. Konar, "Metaheuristic Clustering", *Studies in Computational Intelligence*, vol. 178, 2009.

[7] E. R. Hruschka, R. J. G. B. Campello, A. A. Freitas and A. C. P. L. F. de Carvalho, "A survey of evolutionary algorithms for clustering.", *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 39, no. 2, pp. 133-155, 2009.

[8] H. P. Kriegel, P. Kröger and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering.", in Proceedings of *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 1, pp. 1, 2009.

[9] L. Parsons, E. Haque and H. Liu, "Subspace clustering for high dimensional data: a review.", in Proceedings of *ACM Special Interest Group on Knowledge Discovery and Data Mining Explorations*, vol. 6, no. 1, pp. 90-105, 2004.

[10] G. Moise, A. Zimek, P. Kröger, H. P. Kriegel and J. Sander, "Subspace and projected clustering: experimental evaluation and analysis.", *Knowledge and Information System*, vol. 21, no. 3, pp. 299-326, 2009.

[11] U. von Luxburg, "A tutorial on spectral clustering", *Statistics and Computing*, vol. 17, no. 4, pp. 395-416, 2007

[12] M. Filippone, F. Camastra, F. Masulli and S. Rovetta, "A survey of kernel and spectral methods for clustering", *Pattern Recognition*, vol. 41, no. 1, pp. 176-190, 2008.

[13] H. P. Kriegel, P. Kröger, J. Sander and A. Zimek, "Density-based clustering", *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 231-240, 2011.

[14] E. Muller, S. Gunnemann, I. Farber and T. Seidl, "Discovering Multiple Clustering Solutions: Grouping Objects in different Views of the Data", in Proceedings of *IEEE 28th International Conference in Data Engineering (ICDE)*, pp. 1207-1210, 2012.

[15] I. Davidson and S. Basu, "A Survey of Clustering with Instance Level Constraints", in Proceedings of *ACM Transactions on Knowledge Discovery and Data*, vol. 1, no.1, pp. 1-41, 2007.

[16] A. A. Abbasi and M. F. Younis, "A survey on clustering algorithms for wireless sensor networks", *Computer Communications*, vol. 30, no. 14, pp. 2826-2841, 2007.

[17] J. Y. Yu and P. H. J. Chong, "A survey of clustering schemes for mobile ad hoc networks", *IEEE Communications Surveys and Tutorials*, vol. 7, no. 1, pp. 32-48, 2005.

[18] T. W. Liao, "Clustering of time series data-a survey", *Pattern Recognition*, vol. 38, no. 11, pp. 1857-1874, 2005.

[19] S. Zhong and J. Ghosh, "Generative model-based document clustering: a comparative study", *Knowledge and Information Systems*, vol. 8, no. 3, pp. 374-384, 2005.

[20] P. Berkhin, "A Survey of Clustering Data Mining Techniques", 2001, [Online] Available: online [http://link.springer.com/chapter/10.1007%2F3-540-28349-8_2]

[21] A. Hinneburg and D. A. Keim, "An Efficient Approach to Clustering in Large Multimedia Databases with Noise", in Proceedings of *4th International Conference on Knowledge Discovery and Data Mining,* vol. 98, pp. 58-65, 1998.

[22] M. Ester, H. P. Kriegel, J. Sander, M. Wimmer and X. Xu, "Incremental Clustering for Mining in a Data Warehousing Environment", in Proceedings of *24th Very Large Data Base Endowment Inc.*, vol. 98, pp. 323-333, 1998.

[23] P. Kröger, H. P. Kriegel and K. Kailing, “Density-Connected Subspace Clustering for High-Dimensional Data”, in Proceedings of *Siam International Conference on Data Mining*, vol. 4, pp. 246-256, 2004.

[24] N. Beckmann, H. P. Kriegel, R. Schneider and B. Seeger, “The R*-tree: An efficient and robust access method for points and rectangles”, in Proceedings of *ACM SIGMOD International Conference on Management of Data*, vol. 19, no. 2, pp. 322-331, 1990.

[25] M. Ankerst, M. M. Breunig, H. P. Kriegel and J. Sander, “OPTICS: Ordering Points to Identify the Clustering Structure”, in Proceedings of *ACM SIGMOD International Conference on Management of Data*, vol. 28, no. 2, pp. 49-60, 1999.

[26] E. Chandra and V. P. Anuradha, “A Survey on Clustering Algorithms for Data in Spatial Database Management Systems”, *International Journal of Computer Applications*, vol. 24, no. 9, pp. 19-26, 2011.

[27] H. Teng, K. Chen and S. Lu, “Adaptive real-time anomaly detection using inductively generated sequential patterns”, in Proceedings of *IEEE Computer Society Symposium on Re-search in Security and Privacy*. IEEE Computer Society Press, pp. 278-284, 1990.

[28] P. Rousseeuw and A. Leroy, *Robust regression and outlier detection*. New York: Wiley, 1987.

[29] V. Hodge and J. Austin, “A Survey of Outlier Detection Methodologies”, *Artificial Intelligence Review*, vol. 22, pp. 85-126, 2003.

[30] M. Agyemang, K. Barker and R. Alhajj, “A comprehensive survey of numeric and symbolic outlier mining techniques”, *Intelligent Data Analysis*, vol. 10, no. 6, pp. 521-538, 2006.

[31] M. Markos and S. Singh, “Novelty Detection: A Review-Part 1: Statistical Approaches”, *Journal of Signal Processing*, vol. 83, no. 12, pp. 2481-2497, 2003.

[32] M. Markou and S. Singh, “Novelty detection: A review-Part 2: neural network based approaches”, *Journal of Signal Processing*, vol. 83, no. 12, pp. 2499-2521, 2003.

[33] A. Patcha and J. M. Park, “An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends”, *Computer Networks,* vol. 51, no. 12, pp. 3448-3470, 2007.

[34] P. Tan, M. Steinbach and V. Kumar, *Introduction to data mining*. Boston: Pearson Addison Wesley, 2005.

[35] C. D. Spence, L. Parra, P. Sajda and M. Staib, "Detection, synthesis and compression in mammographic image analysis using a hierarchical image probability model”, in Proceedings of the *IEEE Workshop on Mathematical Methods in Biomedical Image Analysis.* pp. 3-10, 2001.

[36] E. Aleskerov, B. Freisleben and B. Rao 1997, “Cardwatch: A neural network based database mining system for credit card fraud detection”, in *Proceedings of IEEE Computational Intelligence for Financial Engineering*, pp. 220-226, 1997

[37] R. Fujimaki, T. Yairi and K. Machida, “An approach to spacecraft anomaly detection problem using kernel feature space”, in Proceedings of *11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pp. 401-410, 2005.

[38] A. S. Weigend, M. Mangeas and A. N. Srivastava, “Nonlinear gated experts for time-series - discovering regimes and avoiding overfitting”, *International Journal of Neural Systems,* vol. 6*,* no. 4, pp. 373-399*,* 1995

[39] B. Kaur and R. Rani, “Designing URL Access Counter using MapReduce with HBase”, in *Proceedings of Elsevier Second International Conference on Emerging Research In Computing Information, Communication and Applications* - (ERCICA-14), Bangalore, 01-02 Aug. 2014.

[40] P. Garg and R. Rani, “ Analysis and Visualization of Professional’s LinkedIn Data”, in proceedings of *Springer International Conference on Emerging Research in Computing, Information, Communication and Applications,* pp. 1-9, 2015

[41] A. Sharma and R. Rani, “IoT solutions for 3-D visualization of Twitter data”, in proceedings of *IEEE In AdvanceComputing Conference*, pp. 839-843, 2015.

[42] A. and R. Rani, “Analysis and Visualization of Twitter Data using R”, in Proceedings of the *1st International Conference ‘INBUSH ERA 2015’ Theme: Futuristic trends in computational analysis and knowledge management*, 2015.

[43] K. Kaur and R. Rani, "Managing Data in Healthcare Information Systems: Many Models, One Solution", IEEE Computer, vol. 48, no. 3, pp. 52-59, 2015.

List of Publications

1) S. Behera and R. Rani, “Comparative Analysis of Density based Outlier Detection techniques on Breast Cancer data Using Hadoop and Map Reduce”, *IEEE International Conference on Inventive Computation Technologies,* on Aug 26-27, 2016 [Accepted] .
2) Video Link: http://youtu.be/hMtwwD2QZ-c

Plagiarism Certificate

spaper

ORIGINALITY REPORT

3%	0%	3%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	CHANDOLA, VARUN; BANERJEE, ARINDAM and KUMAR, VIPIN. "Anomaly Detection: A Survey", ACM Computing Surveys, 2009. Publication	1%
2	Chumphol Bunkhumpornpat. "DBSMOTE: Density-Based Synthetic Minority Over-sampling TEchnique", Applied Intelligence, 04/14/2011 Publication	<1%
3	Wei Jiang. "A Map-Reduce System with an Alternate API for Multi-core Environments", 2010 10th IEEE/ACM International Conference on Cluster Cloud and Grid Computing, 05/2010 Publication	<1%
4	research.ac.upc.es Internet Source	<1%
5	Jiang, Miao, and Ye Wang. "Optimization of multi-join query processing within MapReduce", 2010 4th International Universal Communication Symposium, 2010. Publication	<1%