Clustering and Classification Techniques in the Presence of Outliers: An Application to the Johannesburg Stock Exchange Stocks

By

Retsebile Maphalla

Student number: 1632723

(Orcid:0000-0003-1351-1822)

Supervisor

Dr HW Chipoyera

(Orcid:0000-0003-4133-5459)

A dissertation submitted to the Faculty of Science, University of the Witwatersrand, in partial fulfilment of the requirements for the degree of Master of Science

June 10, 2024

Declaration

I declare that this research work is my own, unaided work. It is being submitted for the degree of Master of Science at the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination at the University of the Witwatersrand or any other university.

RetSebile

Signature
Date: June 10, 2024

Abstract

In this study, the impact of outliers on clustering using the K-means algorithm was explored. It was observed that a high prevalence of outliers can seriously compromise the results of clustering. A novel algorithm called *Clustering-quality-aided outlier detection* (CQAOD) is proposed in this study. The novelty stems from the fact that apart from identifying outliers, good quality clustering is achieved and the "optimal" number of clusters for K-means clustering of multivariate Gaussian data is simultaneously proffered. In the case of the Johannesburg Stock Exchange (JSE) data, an investigation to compare the efficacy of the following clustering techniques: Hierarchical clustering, spectral clustering, Clustering Large Applications (Clara), Density-based spatial clustering of applications with noise (DBSCAN) was done with the aim of constructing a diversified stock portfolio. The study found that the hierarchical clustering algorithm is the best algorithm to cluster the shares on the JSE.

Contents

List of Figures

List of Tables

Dedication

I dedicate this research report to my parents (Elizabeth and Thomas Maphalla), my brothers, sister and friends for their relentless support.

Declaration

I declare that this research project is solely my own unaided work. It has not been submitted before for any degree or examination to any other University.

Acknowledgements

The author wishes to thank his supervisor, Dr Honest Walter Chipoyera for his valuable input, patience and support during his MSc study. He also wishes to express his gratitude to Dr Charles Chimedza and Dr Justine Nasejje who read the initial MSc proposal and suggested insightful pathways for the research.

The author also wishes to thank the anonymous examiners for their invaluable comments and suggestions which enriched the quality of this research work.

List of Acronyms

GLASSO	Graphical Lasso
JSE	Johannesburg Stock Exchange
MCD	Minimum Covariance Determinant
MCN	Multivariate Contaminated Normal
MINLP	Mixed Integer Non-Linear Programming
MLE	Maximum Likelihood Estimate
MPT	Modern Portfolio Theory
RDT	Resource Dependence Theory
SVM	Support Vector Machines
SSE	Sum of Squared Errors
VRC	Variance Ratio Criterion

Chapter 1

Introduction

1.1 Introduction

The world today produces and collects huge amounts of data, with current estimates of the size of data collected daily being being more than 2.5 quintillion bytes (Marr, 2018). One critical question that has frequently arisen is: how can the data be used to ensure that it is possible to make well informed decisions?

Data mining is one tool that can possibly be used to comprehensively address the latter question. Data mining is the process of sifting through large quantities of data in order to identify hidden trends and patterns that will assist decision makers to make informed decisions. Data mining has it's roots dating back as far as the 1930's (Hand, 1998). The term data mining is synonymous with data dredging or fishing and has been used to describe the process of searching thoroughly through data with the hope of identifying patterns. Data mining started being mainstream in the 1990's as organisations were starting to derive value from the ever-increasing amount of data that was being generated.

The advent of powerful computer technology in the last decade coupled with the use of data mining techniques has resulted in organisations making sense of data (which has been growing exponentially) and in the process identifying hidden opportunities. For

instance, a business organisation could finally do any or all of the following: extract non-intuitive valuable insights, forecast likely business outcomes and mitigate risks. Data mining has become increasingly popular in many fields and it now plays a critical role in deciding whether or not a business succeeds.

The development of data mining tools that are capable of analysing huge amounts of data at high speeds is therefore imperative. Classification and clustering (which are essentially grouping of objects) are two of the techniques at the forefront of modern data mining. One aim of data mining is to make search algorithms more efficient. In their research work involving a large database of fingerprint images for example, Nowak-Brzezińska and Gaibei (2022) have mentioned that fingerprint recognition algorithms would be much more efficient if it took attributes of current examined fingerprint images and compared them to a subset (cluster) with the most similar attributes rather than searching through a whole database of fingerprint images.

Similarities of objects in based on the distances between them be based on the distances between them [1], with objects that are closest to each other being assumed to be more similar than objects further apart. According to Nelson (2012), the most commonly used tool to compute similarities in clustering algorithms is the Euclidean distance. The Euclidean distance is the distance between two points in space.

An intriguing use of clustering and classification algorithms is seen in the financial sector. Investors and risk managers employ the strategy of ensuring that their investment portfolio is a mix of assets whose behaviours are unrelated [2]. Such a strategy is known as diversification. In their paper, Chollete et al. (2011) say that diversification is a method of investing in multiple financial assets that have relatively little interdependence such that the individual risk of each asset is negligible to the whole portfolio.

Gomes and Livdan (2004) say that one method ordinarily used in grouping assets (companies, bonds etc.) listed on the stock exchange is by the sector in which they

[1] various distance measures are available:this is discussed in Chapter 2

[2] e.g. stocks whose share prices are uncorrelated

operate/originate. This approach intuitively makes sense (on one hand) since companies offering similar products or services will ordinarily tend to face similar risks as well as opportunities. However, the approach can be criticized as naive (on the other hand). The naivety stems from the fact that many companies from different sectors depend on each other to fully function and the performance of one or more companies in one sector can affect the performance of others belonging to some other sectors. Pfeffer and Salancik (2003) have observed that different companies depend on each other to operate regardless of their diversity. Pfeffer and Salancik (2003) have coined this Resource Dependence Theory (RDT).

In order to do clustering, data on pertinent features for the observations must be available. For instance, a careful selection of financial metrics of companies to be used as features in a clustering algorithm should be done before the actual clustering. After the careful selection of the relevant features, there is a need to appropriately scale the features data. The scaling will address disparities in size of companies. Compared to a startup or a small company, a larger organisation is more likely to produce higher financial numbers such as high revenues and net profits (Marvin, 2015). The worthiness of an investment does not always coincide with the size. The values of financial metrics are often scaled for size, e.g. revenues and net income are divided by value of assets to scale for size. These ratios are weighted averages, and the variation between the weighted averages of several firms is used to determine how comparable the two firms are. The ratios are frequently used to represent the size of a corporation (Marvin, 2015).

Clustering and classification algorithms may be used in the financial sector to separate groups of assets for purposes of diversifying portfolios. It is important for investors to acknowledge that two companies from the same sector have different portfolio risks and therefore when it comes to clustering or classification, such companies may not belong to the same group. Hillman et al. (2009) have reviewed and ratified the RDT by (Pfeffer and Salancik, 2003). The theory acknowledges that organizations are dependent on each other in the environment which they operate.

1.2 Statement of the problem

Clustering is a broad subject encompassing varied techniques. There are two types of clustering: hard clustering [1] and soft clustering. With hard clustering, each data point is uniquely assigned to a cluster whilst in the case of soft clustering each data point has a probability of belonging to each of the clusters.

The choice of a clustering technique to use on a given dataset is a tricky one. Arguably, a clustering technique that results in good quality clusters is deemed the right choice. Another important angle to factor in as far as the choice of a clustering technique to use is concerned is its relative ease to implement. When it comes to evaluating quality of clusters, popular indices like Dunn index, Davies-Bouldin index, Variance Ratio Criterion (VRC) can be utilised. An exhaustive discussion of the latter indices is given in Chapter 2.

The choice of a distance measure to use for clustering has an impact on the clustering process. The Euclidean distance is touted as a good measure of similarities. However, challenges may arise when clustering data in general. Clustering of stocks using the Euclidean distance is one particular example where challenges arise. Nelson (2012) explains that the Euclidean distance does not factor in the variance-covariance structure of variables and shows how other distance measures could be better alternatives when there is a relationship between variables. Typically, because companies operate in the same economic environment, one expects some dependence in share price movements to exist. Mimmack et al. (2001) also acknowledges the shortcomings of Euclidean distance when there are di-similarities in variances of variables.

Another scenario that often creates a challenge in clustering and classifying stocks is that of companies from the same sector experiencing radically different price movements. This can be due to extreme movements in individual share prices. Consequently, data on share prices can have outliers. Nowak-Brzezińska and Gaibei (2022) have investigated the effect of outliers on the resulting clusters obtained using the

[1] this work solely focuses on hard clustering techniques

K-means and agglomerative hierarchical clustering algorithms on three datasets containing work absenteeism records, auction records and hand postures data. In their investigation, they explored how the outliers in the K-means algorithm and agglomerative hierarchical algorithm affected the clusters obtained. The findings from their research have confirmed their hypothesis that outliers negatively affect the quality of the clusters created. This, they have demonstrated by clustering the datasets with outliers included and then clustering the datasets after weeding out outliers. One wonders whether similar findings could be arrived at in the case of stocks on the Johannesburg Stock Exchange and in particular two questions arise:

1. Does the presence of outliers materially impact the number of clusters and the quality of clusters obtained when K-means clustering is employed?

2. Is the answer to Question 1 the same when other clustering algorithms are used on the same dataset?

Clustering essentially is about unsupervised learning techniques, whilst classification algorithms are supervised learning techniques. This means in classification the algorithms use prior information to classify new data objects. Tallón-Ballesteros and Riquelme (2014) have investigated the effects of outliers on classification algorithms. In their investigation, they have explored the effects of outliers on binary and multi-class classification problems. In their study, they have used a two-pronged approach. In the first approach, they have removed all identified outliers and performed their investigations. In their second approach, they have removed some selected outliers and performed their investigations on ten binary and multi-class classification problems. They have found that partial removal of outliers improved one or two performance measures. They have however, found that the outright removal of outliers does not result in substantial improvements compared to the partial removal in the classification algorithms. Their results have also shown that some classification algorithms such as support vector machines (SVM) are more robust and outliers did not affect their performance. Again, one wonders whether similar findings could be arrived at in the case of stocks on the Johannesburg Stock Exchange. Just like in the case of clustering, the question that arises also is: does the presence of outliers materially impact the performance of classification algorithm used?

As alluded to earlier on, different clustering methods have varying degrees of ease of implementation. Often one has to consider different aspects such as the similarity measure to use, the performance of the algorithm in the presence of outliers and other data attributes (qualitative, quantitative, missingness, etc) before deciding on the appropriate clustering algorithm. As part of this research work, the idea of employing non-linear integer programming has been mooted and explored. The goal has been to establish how well this type of clustering algorithm would fare in terms of ease of use and the quality of clusters obtained.

1.3 Aims and objectives

At the outset, the aims of this research have been (1) to compare the efficacy of different clustering algorithms and classification algorithms in the presence of outliers and when outliers are removed and (2) to develop a new clustering algorithm that utilises non-linear mixed integer programming. To realise these aims, the objectives have been:

1. To explore the use of classical methods to do outlier detection.

2. To assess the impact of different levels of outlier contamination in a dataset on clustering algorithms using the indices discussed in Section 2.5.1. The specific details on how the impact of outliers will be assessed are discussed in detail in Chapter 3.

3. To assess the different levels of outlier contamination on the optimal number of clusters to be used in clustering.

4. More importantly, an algorithm that simultaneously and iteratively does outlier identification, weeding and K-means clustering is proposed.

5. To assess the impact of outliers on classification algorithms. The specific details on how the impact of outliers will be assessed are discussed in detail in Chapter 3.

1.4 Significance of study

According to Kose et al. (2020) the world economy has experienced four global recessions in the last seven decades; this does not include the coronavirus disease 19 (COVID-19) period. In each of the recessions, businesses have had to cut costs and laid off workers. Companies have experienced negative returns and some unfortunately even closed down e.g. Lehman Brothers Inc. It is therefore crucial that investors have meticulous risk management strategies in place as non-diversified or poorly diversified investors can lose all of their investments. This study demonstrates the prowess of statistical methods in risk management. One of the major findings of this research is that clustering and classification are indispensable tools when it comes to assembling an arsenal of tools for tackling risk, especially for data bedevilled with outliers.

1.5 Layout of dissertation

The layout of this dissertation is as follows. Chapter 1 gives a comprehensive introduction on how classification and clustering may be harnessed to aid decision making in business. It is in Chapter 1 that the problem of getting good clusters for data bedeviled with outliers is discussed. The discussion culminates in giving a clear account of the statement of the problem accompanied by the aims and objectives. Chapter 2 takes a look at the literature reviewed that is in line with the aims and objectives of this research. Chapter 3 introduces the data made use of in this research and explains the methods used to carry out this research. Chapter 4 gives an in-depth analysis of the results obtained in line with the methodology discussed in Chapter 3. Chapter 5 contains an in-depth discussion of the results obtained in Chapter 4: the latter discussion gives rise to conclusions of the study. A concise account of the limitations of the study is also included in Chapter 5.

1.6 Deliverables from this thesis

A new algorithm for clustering and outlier detection has been developed. A research article has been submitted to the Pattern Recognition journal for possible publication.

Chapter 2

Literature review

2.1 Introduction

People invariably confuse clustering with classification. According to Johnson et al. (2002), clustering is different from classification in that classification pertains to a known number of groups and the objective is to assign new observations to one of those groups while clustering does not make any apriori assumptions concerning the number of groups or group structures.

2.2 Clustering algorithms

There are two types of clustering algorithms, hierarchical and non-hierarchical clustering algorithms. Most clustering algorithms use proximity methods to put objects into different clusters.

2.2.1 Hierarchical clustering approaches

Johnson et al. (2002) describe hierarchical clustering methods as the grouping of objects into clusters without predetermining the number of clusters. There are two ways of performing hierarchical clustering, namely agglomerative and divisive clustering.

Agglomerative clustering uses a bottom-up approach to group observations. This clustering technique treats each individual observation as a cluster at the outset and then successively merges the clusters by their similarities. Divisive clustering differs from agglomerative clustering in that it uses a top-down approach. This clustering technique treats all the observations as initially belonging to one big cluster and then successively and iteratively breaks up the big cluster into smaller clusters on the basis of observations' dissimilarities. This happens until all data objects have been split into an optimal number of clusters. The common ways of joining objects into clusters are single linkage (nearest neighbour), centroid linkage, average linkage and complete linkage (furthest neighbour).

Single linkage

In clustering that utilises the single linkage approach, in an iteration, the minimum distance between points from all pairs of clusters is calculated and the pair with the least such distance are merged (in hierarchical agglomerative clustering) (Mohbey and Thakur, 2013).

Centroid linkage

In clustering that utilises the centroid linkage approach, in an iteration, the algorithm merges pairs of clusters with the minimum distance between their centroids. Each time clusters are joined the algorithm recalculates the centroid of the newly formed cluster. In the case of K-means clustering the process will iterate until the set number of required clusters is obtained (Lattanzi et al., 2020).

Average linkage

Clustering which utilises the average linkage approach involves calculating the average distances of separation for pairs of points belonging to any pair of clusters in an iteration. Those two clusters with the lowest average are merged. Each time clusters are joined the algorithm recalculates the mean distance of the newly formed cluster

and other clusters. (Fernández and Gómez, 2008).

Complete linkage

Clustering which utilises the complete linkage approach involves calculating the maximum distance of separation between any two points which belong to a pair of clusters in an iteration. The algorithm merges the pair of clusters with the minimum of these maximum distances (Sharma et al., 2019).

2.2.2 Non-hierarchical clustering approaches

Non-hierarchical clustering is the grouping of objects into clusters by minimising or maximising a defined criterion (Johnson et al., 2002). Before the algorithm can be applied, one must first choose how many clusters the data should be assigned. The three common ways of joining objects into clusters used in non-hierarchical clustering algorithms are centroid based, density based and distribution based.

Density based clustering algorithms

Density based clustering algorithms assign areas of high densities into clusters. According to Xu and Tian (2015), density based algorithms are not recommended for data with a high degree of variability in density and their performance tends to be diminished with increasing number of dimensions. The objects outside dense areas are not assigned to any clusters and can be treated as outliers.

Dinandra et al. (2019) have applied density based clustering methods on the New York stock market. In their work they have applied a popular density based clustering algorithm known as DBSCAN (Density-Based Spatial Clustering of Application with Noise and Genetic Algorithm) on seven stock features to find an optimal investment portfolio. They have found that the portfolio constructed using DBSCAN outperforms the S&P500 [1] when backtested. Interestingly, their results have shown that optimum

[1] S&P500 is an index constructed using the top 500 companies on the New York Stock Exchange

portfolios with lower variances perform better than the optimum portfolios with high variances.

Distribution based clustering algorithms

Distribution based clustering assumes the data objects come from a mixture of distributions whose forms is known. Tong and Tortora (2022) have explored a distribution based clustering algorithm. In their work, they have assumed the data at hand follows a multivariate contaminated normal (MCN) distribution. The MCN distribution is a heavy tailed generalization of the multivariate normal distribution. The MCN is recommended to cluster data which contains few or no outliers. The MCN can detect outliers and at the same time produces robust estimates of parameters for each cluster. The downside of using MCN is that it only works for a dataset with no missingness.

Tong and Tortora (2022) gives a framework of how a mixture of MCN algorithms can be used to impute missing data and then parameters of each cluster are estimated using the expectation-conditional maximisation algorithm. For the same dataset, they have also explored the student-t distribution which is renowned for outlier detection. They have found that the MCN distribution performs better at detecting outliers than the student-t distribution. Another notable finding from their work has been that as the level of missingness of the data increases the performance of both the MCN distribution and Student-t distribution deteriorates to the extent that results obtained are not dependable.

Centroid based clustering algorithms

A centroid based clustering algorithm groups data with (numerical variables only) by minimising the sum of squares of distances between the data values and their corresponding cluster centroids (Virvou et al., 2012). A centroid based clustering algorithm aims to group the n observations into k clusters in such a way that the separation between centroids of different clusters is maximized. The K-means clustering algorithm is one of the most commonly used centroid based clustering algorithms (MacQueen et al., 1967; Schubert, 2023).

2.3 K-means Algorithm

MacQueen et al. (1967) and Gupta and Panda (2018) have given detailed accounts of the K-means clustering algorithm. Rodriguez et al. (2019) has done an extensive comparison of many clustering methods using both simulated and real life data sets and contends that K-means clustering is very popular with researchers. Schubert (2023) also concurs. The popularity of this algorithm mainly stems from it being a simple algorithm to implement as well as its added advantage of it being a fast algorithm. Saha and Mukherjee (2021) mention that one of the reasons why the K-means algorithm has enjoyed acclaim is that it has clustering stability. What this means is that for different samples generated by the same process, K-means clustering will produce clustering structures that are hardly different.

2.3.1 Overview of K-means algorithm

With K-means clustering, the number of clusters k is pre-specified by the user and each of the k clusters is represented by its centre (i.e. centroid), which corresponds to the mean of points assigned to the cluster.

The basic idea behind K-means clustering is to create k non-overlapping clusters $C_1, \cdots, C_k$ such that the total intra-cluster variation, $W(C_k)$ is minimized. Each observation $(\mathbf{x}_i,\ i = 1, \cdots, n)$ is appropriately assigned to a cluster such that the sum of squares of distances of the observations from their assigned clusters with centres $(\underline{\mu}_j,\ j = 1, \cdots, k)$ is minimized. The total intra-cluster variation denoted by W_k:

$$W_k = \sum_{j=1}^{k} \left(\sum_{\mathbf{x}_{ij} \in C_j} \|\mathbf{x}_{ij} - \underline{\mu}_j\|^2 \right) \tag{2.1}$$

where $\mathbf{x}_{ij}$ is a data point belonging to cluster C_j whose centroid or mean vector is $\underline{\mu}_j$; $\underline{\mu}_j$ is the mean value of all the points assigned to cluster C_j.

Remark 1. *An ideal clustering process results in the formation of k clusters which are internally homogeneous and are such that different clusters are dissimilar.*

The K-means algorithm can be summarized as follows:

1. The first step when using K-means clustering is to choose the number of clusters (k) that data should be partitioned in.
2. The algorithm then starts by randomly selecting k data points from the data set to serve as the initial centres for the clusters. The selected objects are also known as cluster means or centroids.
3. Next, each of the remaining objects are assigned to their closest centroid, where "closest" is defined using the distance (i.e., Euclidean distance) between the object and the cluster mean. This step is called the "cluster assignment step".
4. After the assignment step, the algorithm computes new centroids. All the observations are now assigned to the clusters represented by the new centroids in such a way that each observations is assigned to the nearest centroid.
5. The cluster assignment and centroid update steps are iteratively repeated until the cluster assignments stop changing (i.e. until convergence is achieved). That is, the clusters formed in the current iteration are the same as those obtained in the previous iteration.

This whole process results in the formation of k clusters with the least total within-sum-of-squares. Minimising the total within-sum-of-squares ensures the clusters are internally homogeneous while also maximizing the dissimilarity between the different resulting clusters.

2.3.1.1 Downside of K-means Clustering

The biggest drawbacks of the K-means algorithm cited in literature are

- The first drawback of K-means relate to the fact that the ”true” need to be specified before clustering takes place. This problem is explored in more detail in Section 2.3.2.

- the problem associated with centroid initialisation.
- the problem of an unequivocal optimal number of clusters. The algorithm hinges on randomly choosing k datapoints to use as initial centroids. This creates a situation that it maybe possible to fall into a local optimal solution. When this happens, two scenarios may obtain. The first scenario is that the final solution obtained may not result in a global minimum distance and consequently, falls short in the the determination of optimal clustering. The second scenario is that two sets of initial centroids may yield different clusters assignments.
- K-means clustering using euclidean distance requires that clusters are isotropic [1]; if the clusters are not isotropic, the algorithm does not work well.
- Rodriguez et al. (2019) mention that the k-means clustering algorithm is not particularly recommended in cases where the clusters do not show convex distribution or have very different sizes. Another downside of the algorithm is its sensitivity to the initial seed selection.

2.3.2 The choice of k

The "true" number of clusters M is unknown and when clustering, a plausible number of clusters k may be used. Rodriguez et al. (2019) point out that this is one of the main limitations of the K-means algorithm mainly because the clusters obtained strongly depend on the choice of k. One big set back of the problem of what value of k to use (sometimes referred to as *cluster validation problem*), according to Sugar and James (2003), has been the tendency to work with data using model-based algorithms which require strict adherence to parametric assumptions and/or sometimes resulting in computation-intensive solutions. In their paper, Saha and Mukherjee (2021) have shown that an inappropriate value of k can result in cluster instability.

2.3.2.1 Scree plot cluster validation method and its weaknesses

One popular heuristic procedure used for determining what may be perceived to be an ideal value for k involves the construction of a scree plot and from it visually inspect

[1] Isotropic clusters are spherical clusters with the same radius

and identify an elbow in order to propose the ideal number of clusters. This procedure, called the scree plot elbow method is attributed to Thorndike (1953).

Schubert (2023) has advanced arguments that are critical of the scree plot elbow method and has been vociferously advocating against the use of the scree plot heuristic method as a means for recommending the number of clusters to use in clustering. The problems with the scree plot cited by Schubert (2023) are:

- the axes of the plot have very different meanings which makes it problematic to compare. Schubert (2023) laments the lack of a meaningful measurement of angle to characterize an elbow and that by simply changing the scaling of the axis for k the interpretation of an "elbow" is human subjective.
- the scree plot for any dataset is always a descending curve as k increases and one serious indictment on its part is that it cannot therefore choose $k = 1$ for data that does not apparently contain any clusters.

2.3.2.2 The jump cluster validation method

Sugar and James (2003) have also weighed in in the search for coming up with a credible procedure for determining the actual number of clusters present is concerned. The method for k that has been proposed by Sugar and James (2003) is essentially a non-parametric method which hinges on *distortion* which is essentially an average distance measure between each observation and the cluster center that is closest to it.

The jump method makes use of a *distortion function* d_K and an associated *jump function* to determine a plausible estimate of the "true" number of clusters. The downsides of the jump cluster validation method are the estimation of the distortion function is not straight-forward and it is not always possible to obtain and estimate of the value of M, the true number of clusters using the method.

2.3.2.3 Severity of using a wrong value of k

Milligan and Cooper (1985) have warned that the error of clustering using a fewer number of clusters than the actual number of clusters can be considered as more se-

vere in most applied analyses because merging clusters result in loss of information. Rodriguez et al. (2019) affirm the assertion by Milligan and Cooper (1985) and point out in the concluding remarks of their paper that interestingly, use of a larger number of clusters in clustering gives better results in comparison to when a lower number of clusters than the actual number is used. This is an observation they made for all the clustering algorithms that they explored.

2.3.3 The problem of clustering for data with outliers

Clustering using the K-means algorithm for data bedevilled with outliers is oftenly challenging (Gupta and Panda, 2018). Nowak-Brzezińska and Gaibei (2022) have investigated the effect of outliers on the resulting clusters obtained using the K-means and agglomerative hierarchical clustering algorithms on three datasets containing work absenteeism records, auction records and hand postures data. In their investigation, they have explored how the parameters in the K-means algorithm and agglomerative hierarchical algorithm affected the clusters obtained. The findings from their research have confirmed their hypothesis that outliers negatively affect the quality of the clusters created; the authors actually concluded their paper by saying that their research confirmed the intuitive belief that the more outliers removed from a data set, the better the quality of the clusters become. This, they have demonstrated by separately clustering the datasets with outliers included and then clustering the datasets after weeding out outliers using the Local Outlier Factor and Connectivity-based Outlier Factor algorithms.

2.4 Other non-hierarchical clustering algorithms

Other popular non-hierarchical clustering algorithms include Spectral clustering algorithm, Density-based spatial clustering of applications with noise (DBSCAN) clustering algorithm and Clustering Large Applications (Clara) clustering algorithm. The

DBSCAN is a non-parametric density-based clustering algorithm, The DBSCAN clusters points in a given space that are closely packed together. Points that lie further from areas of high densities are are marked as outliers (Deng, 2020). The spectral clustering algorithm is a non-parametric clustering algorithm that utilizes the eigenstructure of a similarity matrix to cluster data (Ng et al., 2001). The technique makes use of the spectral decomposition of an affinity matrix created using the data to be clustered. Other clustering algorithms can then be employed to cluster the resulting eigenvectors. The Clara algorithm is an adaptation of the Partitioning Around Medoids clustering algorithm that can be applied to large datasets. The algorithm manages to cluster large datasets by clustering samples of datasets.

2.5 Outlier detection

Multivariate datasets usually contain outliers that may not be easily detectable. As alluded to in Section 2.3.3, clustering processes can be negatively affected by outliers. It is therefore important that outlier detection and weeding out of the outliers is done before utilising clustering algorithms. Hubert et al. (2008) have mentioned that if the number of outliers in the data exceeds some threshold, variability of the data is inflated leading to models fitted to the data being unreliable.

2.5.0.1 Outlier detection for Gaussian distributed data

One popular distance based outlier detection tool for datasets that approximately follow a Gaussian distribution is the Mahalanobis distance. According to Ghorbani (2019), one notable reason in favour of the use of the Mahalanobis distance is its prowess in the detection of outliers in multivariate datasets. The Mahalanobis distance is a well-known multivariate statistical distance measure. It is a measure of the distance between $\mathbf{X} \in R^p$ from a p-variate distribution f_x and the mean $\underline{\mu} = E[\mathbf{X}]$ of the distribution taking into account the variability of the data. Given that f_x has finite mean vector and variance-covariance matrix denoted by $\underline{\mu} = E[\mathbf{X}]$ and $\Sigma = E[(\mathbf{X} - \underline{\mu})(\mathbf{X} - \underline{\mu})^T]$,

respectively, the Mahalanobis distance between $\mathbf{X}$ and $E[\mathbf{X}]$, denoted by $D(\mathbf{X}, \underline{\mu})$:

$$D(\mathbf{X}, \underline{\mu}) = \sqrt{(\mathbf{X} - \underline{\mu})^T \Sigma^{-1} (\mathbf{X} - \underline{\mu})} \tag{2.2}$$

Remark 2. *If the variance-covariance matrix of* $\mathbf{X}$ *equals the identity matrix, then the Mahalanobis distance will simply be the Euclidean distance.*

The disadvantage of using the Euclidean distance in K-means clustering when the assumption of isotropic clusters breaks down (discussed in Section 2.3.1.1) can be avoided by using the Mahalanobis distance. This is because the Mahalanobis distance can be adapted to non-spherical clusters. Another advantage of the Mahalanobis distance over the Euclidean distances is that the Mahalanobis distance also scales the weight of individual variables to the distance value according to the variability of each variable, therefore scaling of the data is not required. Similar arguments in support of the use of the Mahalanobis distance are also advanced in the paper by Lapidot (2018).

A major disadvantage of using the Mahalanobis distance in the K-means algorithm is that convergence is not guaranteed. Lapidot (2018) explains that non-convergence of distances could result in the minimum distance change required for the iterative process to stop before the optimal solution. Lapidot (2018) suggests two notable changes to the data to ensure convergence: (1) constrain the variance-covariance matrix such that the determinant equals to one or (2) by maximising the Gaussian log-likelihood.

One criterion explained in Remark 3 oftenly used for outlier detection when the underlying data are from a Gaussian distribution utilizes the fact that the Mahalanobis distances follows a chi-squared distribution.

Remark 3. *Any* p*-variate observation* $\mathbf{x}_i$ *with a Mahalanobis distance* $D(x_i, \underline{\mu}^*; \Sigma^*)$ *such that*

$$D^2(x_i, \underline{\mu}^*; \Sigma^*) > \chi^2_\alpha(p) \tag{2.3}$$

is an outlier, i.e. any observation whose Mahalanobis distance exceeds the critical value $\sqrt{\chi^2_\alpha(p)}$ [1] *is an outlier.*

[1] $\chi^2_\alpha(p)$ is the $100(1-\alpha)$ percentile of a chi-square distribution with p degrees of freedom.

Remark 4. *The choice of the value of α to use in outlier detection has not been explained clearly in literature. Cerioli (2010) mentions that a value of α such that $\alpha \in (0.01, 0.05)$ is acceptable; the rule of thumb used by many researchers is to use $\alpha = 2.5\%$.*

There are two major drawbacks when it comes to the use of the Mahalanobis distance to detect outliers.

Firstly, the assumption that the data at hand is multivariate normal is obviously a major limitation in the event that the data at hand is non-Gaussian.

Secondly, the estimates of the Mahalanobis distance values hinge on the estimates of the mean vector and variance-covariance matrix. The outliers being sought can impact the Mahalanobis distances. Tarr et al. (2016) have explained how the advent of outliers impact the detection of the outliers. Data with many outliers will have Mahalanobis distances $D(\mathbf{X}, \underline{\mu})$ that are small because the Mahalanobis distance utilises the inverse of the variance-covariance matrix.

An additional pitfall mentioned by Cerioli (2010) is that the criterion exhibits an inadequacy when it comes to obtaining a reliable cut-off value when the data contain no outliers.

Remark 5. *For data bedevilled with many outliers, a consequence of the use of the classical method which utilises the Mahalanobis distance for outlier detection as discussed above is under reporting of the number of outliers which translates to a higher incidence of false negatives for higher numbers of outliers in a dataset. The phenomenon of Mahalanobis distances being low because of a high prevalence of outliers in the dataset has been termed "masking".*

Hubert et al. (2008) have proposed the use of the minimum covariance determinant estimator (MCD) introduced by Rousseeuw (1984) to circumvent the problem alluded

to in Remark 5. The algorithm for finding the MCD estimator looks for a subset of size h out of n observations (where $\frac{1}{2}n < h \leq n$) with the smallest variance-covariance determinant. The MCD estimator works on the basis that the dataset contains $n - h$ outliers and thus their influence on the estimates of the mean and variance needs to be avoided. The choice of an appropriate value h can be tricky, thus domain expertise may be needed. This algorithm works well to eliminate outliers but can be time-consuming and computationally intensive. This is more prominent when the dimension p and/or the size of the dataset n increases as the MCD estimator requires the evaluation of all $\binom{n}{h} \approx 2^n(1 - 2^{-\frac{1}{2}n})$ subsets of $h > \frac{1}{2}n$ size. Jobe and Pokojovy (2015) say that for multivariate data $\mathbf{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_n)^T$ whose underlying mean vector and variance-covariance matrix are $\underline{\mu}$ and Σ, the MCD robust estimators of the latter parameters calculated from the identified sample of $\mathbf{x}$ denoted $\mathcal{I}_h$ are:

$$\hat{\mu}_{MCD} = \frac{1}{h} \sum_{i \in \mathcal{I}_h} \mathbf{x}_i \tag{2.4}$$

$$\text{and} \quad \hat{\Sigma}_{MCD} = \frac{k_{MCD}}{h-1} \sum_{i \in \mathcal{I}_h} (\mathbf{x}_i - \hat{\mu}_{MCD})((\mathbf{x}_i - \hat{\mu}_{MCD})^T \tag{2.5}$$

where the constant of proportionality k_{MCD} serves the purpose of ensuring the desirable properties of the latter two estimators being unbiased and consistent estimators.

Rousseeuw and Driessen (1999) have introduced a faster MCD algorithm adaptation which they have called FAST-MCD. To its credit, the FAST-MCD can find the exact MCD. In addition, when applied to large datasets, the FAST-MCD gives more accurate results when compared to other algorithms.

2.5.1 Quality of clusters

When comparing two or more clustering methods, the quality of clusters produced by the clustering methods is key. Higher-quality clusters have low intra-cluster distance and clear separation between clusters. The assessment of how good clustering is is mostly done using popular indices like the Dunn index, the Davies–Bouldin index and the Variance Ratio Criterion (VRC).

2.5.1.1 Dunn Index

Bezdek and Pal (1995) and Rodriguez et al. (2019) give detailed accounts of indices that can be used for measuring quality of clusters. Rodriguez et al. (2019) says that the Dunn's validation index quantifies the degree of separation of clusters as well as the degree of compactness of the clusters. The goal of clustering is to maximize the inter-cluster distance while minimizing the intra-cluster distance.

For multivariate data $\mathbf{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_n)^T$ partitioned into k clusters $C_1, \cdots, C_k$, the *diameter* of cluster $C_j, j = 1, \cdots, k$ denoted by $D^{(j)}$ is the maximum distance between any two points belonging to cluster C_j, i.e.

$$D^{(j)} = \max_{\mathbf{x}_r, \mathbf{x}_s \in C_j} \{||\mathbf{x}_r - \mathbf{x}_s||\}. \tag{2.6}$$

The largest of the diameters of the clusters, denoted by D_{max}:

$$D_{max} = \max\{D^{(j)} | j = 1, \cdots, k\} \tag{2.7}$$

and the distance between two clusters C_i and C_j, $D^{(ij)}$:

$$D^{(ij)} = \min\{||\mathbf{x}_r - \mathbf{x}_s|| \; |\mathbf{x}_r \in C_i, \mathbf{x}_s \in C_j\}. \tag{2.8}$$

so that the distance between the two closest clusters is

$$D_{closest} = \min\left\{\{D^{(rs)}\} | 1 \leq r, s \leq k, r \neq s\right\} \tag{2.9}$$

The Dunn index for the clustering structure is

$$U_{Dunn}(k) = D_{closest} \div D_{max} \tag{2.10}$$

Remark 6. $0 < U_{Dunn}(k) < \infty$ *and higher values of the Dunn index indicate better clustering. Bezdek and Pal (1995) remark that one downside of the Dunn index is that just a few outliers to the clusters may easily impact on the value of the index.*

2.5.1.2 Davies Bouldin Index

The ingredients for calculating the Davies Bouldin index are:

1. The mean of Euclidean distances of the n_i observations in the ith cluster ($i = 1, \cdots, k$) from their cluster centroid $\mathbf{c}_i$, denoted by S_i:

$$S_i = \frac{1}{n_i} \sum_{\mathbf{x}_i \in C_i} ||\mathbf{x}_i - \mathbf{c}_i||. \tag{2.11}$$

2. The inter-cluster distance between two clusters C_i and C_j, M_{ij} is measured using the Euclidean distance between the clusters:

$$M_{ij} = ||\mathbf{c}_i - \mathbf{c}_j||. \tag{2.12}$$

The Davies Bouldin Index (DB) for the clustering structure with the atrributes given in Equation 2.11 and Equation 2.12 is:

$$DB = \frac{1}{k} \sum_{i<j}^{k} \max_{i \neq j} \left(\frac{S_i + S_j}{M_{ij}} \right). \tag{2.13}$$

A lower value of the Davies-Bouldin index is indicative of higher-quality clusters.

2.5.1.3 Variance ratio criterion

Caliński and Harabasz (1974) are credited with developing the variance ratio criterion. To compute VRC, we start by defining the total of within clusters sum of squares for a k-clustering structure which is denoted by $WCSS(k)$ or the sum of squared errors SSE_k:

$$SSE_k = \sum_{i=1}^{k} \sum_{x_j \in C_i} ||x_j - c_i||^2. \tag{2.14}$$

The VRC of a k-clustering structure ($k \geq 2$), $VRC(k)$ is defined:

$$VRC(k) = \left(\frac{SSE_1 - SSE_k}{k-1} \right) \div \left(\frac{SSE_k}{n-k} \right). \tag{2.15}$$

The value of k used in a K-means clustering process for a dataset does impact on the quality of clusters and consequently the values of quality indices like the Dunn

index, the Davies Bouldin index and the VRC. As the value of k increases from 2, the Dunn index and VRC generally increase (whilst the Davies Bouldin index generally decreases). After some threshold value of k, the Dunn index and the VRC generally start decreasing while the Davies Bouldin index is generally increasing. The threshold value is what is proposed to be the ideal number of clusters which in all likelihood will be equal to the true or underlying number of clusters, M.

2.6 Some related works on clustering-aided outlier detection

2.6.1 Clustering aided outlier detection

Cherednichenko (2005) has developed what he has termed as *Clustering Outlier Removal (COR) Algorithm* as part of their Masters degree thesis. The algorithm is an iterative process. At each iteration, k-means clustering is performed followed by outlier identification and weeding of the outliers. Any observations identified to be distant from their cluster centroids are deemed as outliers.

Jobe and Pokojovy (2015) have explored the problem of cluster-based outlier detection for a p-dimensional random sample obtained from a population of a finite mixture of multivariate normal densities. Their work assumes that the bulk of points are from one generating source which gives one cluster and all data points not in the source are potential outliers. The potential outliers may be a set of individual points that are scattered or some of them may form clusters. They have developed what they have called finite sample re-weighted minimum covariance determinant model association clustering (FFSRMCD-MAC) method for outlier detection. The method hinges on the finite sample re-weighted minimum covariance determinant (FSRMCD) estimators of the mean vector μ and variance-covariance matrix Σ which are a direct extension of MCD estimators.

2.6.2 Other related research on clustering

In this section a discussion of research works done which are along the same lines as what has been done in this research work are discussed.

Saha and Mukherjee (2021) have developed a method which they have called Cluster Number Aided K-means (CNAK). The method is unique in that it can detect a single cluster for data that are unclustered. The authors hail their method in that it can additionally be used for a high dimensional dataset. The method, they say, has robustness to cluster imbalance outliers.

In their research, Marvin (2015) have used revenues and net income to cluster companies with the aim of creating a diversified portfolio. The data is divided into groups using a clustering algorithm, which uses the difference between weighted averages as a gauge of similarity. They end up with two financial ratios, revenues-to-assets and net income-to-assets as the foundation for the clustering process. Ideally, the clusters are quite distinct from one another. A portfolio that includes stocks from each cluster will therefore be diversified. Stocks are chosen to maximise the return-to-risk ratio in the pursuit of high returns and minimal hazards. The average return over the risk-free rate per unit, or the Sharpe ratio of volatility, is a measure of calculating risk-adjusted return. The stocks from each cluster with the highest Sharpe ratio are therefore selected to form the portfolio. The portfolio is then diversified and comprised of historically high-performing stocks (Marvin, 2015).

Roy (2019) has done a study on the CNX NIFTY Mid Cap 50 Index from the Indian stock market. The aim is to identify homogeneous clusters of stocks with the purpose of generating optimal portfolio diversification. He has selected 9 financial ratios as vector features for the clustering process whilst in the case of the Johanesburg Stock Exchange (JSE) data, asset turnover, price earnings ratio and return on assets are features used in the clustering processes. Like Lee et al. (2010), Roy (2019) has also used hierarchical agglomerative clustering to determine how many clusters should be produced by the K-means algorithm using the Euclidean distance. The only difference

in their approaches is that Roy (2019) has standardized the raw data using the normalizing technique before the clustering process while Lee et al. (2010) summarised features as 0 or 1. Another step that Roy (2019) took was to remove the outliers in the dataset before clustering; he has not mentioned the method used to identify the outliers.

Both Lee et al. (2010) and Roy (2019) took similar approaches to clustering stocks. Both the hierarchical agglomerative algorithm and K-means algorithm were used to cater for their disadvantages whilst at the same time utilizing the strengths of the algorithms. The K-means algorithm's strength is that it can cluster a large dataset but the disadvantage it has is that it requires the number of clusters to be specified apriori. The hierarchical agglomerative clustering algorithm on the other hand is not ideal for clustering large datasets but has the advantage of clustering without the need of specifying the number of clusters apriori. Then K-means clustering was used to cluster the dataset. In doing so they managed to successfully cluster stocks and constructed a diversified portfolio. In the case of Roy (2019), the optimum number of clusters in the process has been found to be 2. Companies from the following sectors: banking, chemicals, power, iron and steel industries constituted one cluster whilst companies from the sectors: automobiles, information technology, industrial gas and fuels, healthcare, agriculture construction materials belonged to the other cluster.

Although outliers are suspect, care needs to be taken when making a decision to remove them or not. Outliers can be legitimate anomalies that are vital for capturing information on the subject of interest. Moreover, detected outliers can have information content which can help in decision-making process in many settings and applications (Shehadeh et al., 2022). Outliers in the stock market can actually be a positive thing for investors (i.e., higher earnings than normal) and indicate potential market conditions (i.e., recessions). Therefore, the removal of outliers may not be ideal when evaluating the efficacy of K-means clustering when the aim is to construct a diversified portfolio.

2.7 Proposed clustering-aided outlier identification

The problem discussed in this work relates to the detection of data points that are outlying relative to the clusters or local neighborhoods in the datasets.

The proposed clustering-aided outlier identification algorithm in this paper, which the authors have called *Clustering-Quality-Aided Outlier Detection* (CQAOD) is novel in that apart from identifying outliers, good quality clustering is achieved and the "true" number of clusters for K-means clustering of multivariate Gaussian data is simultaneously proffered.

The dataset at hand is assumed to be a p-dimensional random sample obtained from a population comprising of a finite mixture of k multivariate normal densities $N(\underline{\mu}_1, \mathbf{\Sigma}_1),\ N(\underline{\mu}_2, \mathbf{\Sigma}_2), \cdots, N(\underline{\mu}_k, \mathbf{\Sigma}_k)$ (where $k \geq 2$ is unknown). A short description of the algorithm follows:

For each $k,\ k = 2, \cdots, k_{max}$ (where the value of k_{max} needs to be a reasonable maximum number of clusters possible.

Step 1 Perform K-means clustering and compute clustering-quality indices (i.e. the Dunn index (D_2), the Davies-Bouldin (DB_2) index, etc.) Next, for each cluster, compute parameter estimates of μ_i and Σ_i ($i = 1, \cdots, k$) using the classical technique and MCD technique and identify outliers and weed out all the identified outliers to form a new sample.

Step 2 Perform K-means clustering on the new sample and compute clustering-quality indices (just like in a). Next, check for outliers in each cluster. If there are no outliers stop. If there are outliers identified, weed all of them out to form a new sample and go to Step 3.

Step 3 Repeat Step 2 iteratively until no new outliers can be identified.

On the basis of the Davies Bouldin index values (and the Dunn index), it is recommended that the ideal number of clusters k_{ideal} is the one associated with the smallest

Davies Bouldin index, i.e.

$$k_{ideal} = \arg\min_{j}\{DB_j | j = 2, \cdots, k_{max}\}. \tag{2.16}$$

(and simultaneously by the largest value of the Dunn index as well as the largest value of VRC).

2.8 Classification algorithms

Classification algorithms are data mining techniques that try to distinguish data classes. The goal is to predict the class into which data objects with unknown labels will belong to. Classification algorithms can be based on statistical theory or proximity measures to predict the class a data point belongs to. There are two types of classification algorithms: discriminative and generative algorithms.

2.8.1 Discriminative model approaches

A discriminative model classifier is a classification algorithm that predicts the class of a data object by modelling the boundaries between classes. It models the boundaries separating classes by using previously observed data. The performance of an algorithm depends on the quality of the data provided instead of its distribution.

2.8.1.1 Multinomial logistic regression

An example of a discriminative classifier is multinomial logistic regression. Multinomial logistic regression is a classification method that makes use of statistical techniques to predict multiclass problems (i.e. that is where the response variable will fall in one of $k \geq 2$ classes) that do not have a given rank or order. The multinomial logistic regression predicts outcomes by establishing a relationship between the response variable and one or more independent predictor variables. The predictor variables can all be categorical in nature or all continuous or a mix of continuous or categorical variables (Ranganathan et al., 2017).

A logistic regression classifier is a special case of multinomial logistic regression (where $k = 2$) that makes use of statistical techniques to predict a binary outcome. The model is similar to multiple linear regression model, with the difference being that the response variable in logistic regression is binary. The response variable is classified as $Y = 0$ or $Y = 1$. The model is as follows:

$$\theta = P(X) = \frac{e^{\beta_0 + \underline{\beta}^T \mathbf{X}}}{1 + e^{\beta_0 + \underline{\beta}^T \mathbf{X}}} \tag{2.17}$$

where $\mathbf{X}$ is the input data, β_0 and $\underline{\beta}$ are the parameters of the model. The model can equivalently be expressed as:

$$\ln\left(\frac{\theta}{1-\theta}\right) = \beta_0 + \underline{\beta}^T \mathbf{X} \tag{2.18}$$

The function $\ln\left(\frac{\theta}{1-\theta}\right)$ is called a link function. A new object is assigned to class 0 if the probability $\theta < 0.5$ and it is assigned to class 1 if the probability $\theta \geq 0.5$ (James et al., 2013).

The model parameters $\beta_0, \beta_1, \cdots, \beta_p$ are estimated using the training data available. Estimators of the parameters are usually found using maximum likelihood procedure. The likelihood function for logistic regression is given by:

$$\mathcal{L}(\beta_0, \beta_1, ..., \beta_p) = \prod_{i=1}^{n} [p(x_i)]^{y_i} \cdot [1 - p(x_i)]^{1-y_i}$$

where:

- y_i represents the observed binary outcome for the ith observation (0 or 1).
- $p(x_i)$ represents the predicted probability of the ith observation belonging to the positive class (1).

The maximum likelihood estimates of the parameters β_0 and $\underline{\beta}$ of the logistic regression are found using the Newton-Raphson iterative method (Febrianti et al., 2021). Features with very small coefficients have very little importance in predicting the class in which the response variable will fall.

One notable application of logistic regression in the analysis of stock market data was done by James et al. (2013) to predict the direction of share price. Daily closing prices from the $S\&P500$ from 2003 to 2005 were used. The lagged daily closing for previous 5 days (Lag1, Lag2, Lag3, Lag 4 and Lag5), Volume (the number of shares traded on the previous day, in billions), Today (the percentage return on the date in question) and Direction (whether the market was Up or Down on this date) are the features used in the classification problems.

2.8.1.2 Support Vector Machines

Another popular classification algorithm is support vector machines (SVM). SVM are powerful and versatile supervised learning models used for classification and regression tasks. SVMs are widely used in fields such as finance, text categorization and image classification. The basic idea behind SVM is to find an optimal hyperplane that best separates different classes in the feature space while maximizing the margin between them (Tsyurmasto et al., 2014).

According to Auria and Moro (2008), advantages of SVM over other classification techniques such as logistic regression and discriminant analysis are:

- SVM can be robust, even when the training sample has some bias.
- Ability to handle non-linear decision boundaries using kernels.
- Flexibility in choosing different kernel functions to customize the decision boundary.

One can deduce from the above reasons that SVM are very effective, especially in high-dimensional spaces. This undoubtedly makes them a popular choice in many applications.

A notable application of the SVM algorithm was done on the $S\&P$ BSE TECK index from the Indian stock market data, Tripathy (2019) used the SVM moel to predict the closing price of the index. The study uses the performance metric *Hit ratio* to determine accuracy of the SVM model. The results of the study show that after the financial

crisis of 2008, the average forecast accuracy is 60.2 %. The study also finds that the direction of market movement is positive if the closing price is higher than the previous day's closing price.

In binary classification, a SVM constructs a hyperplane or a set of hyperplanes in a high-dimensional space, which can be used for classification. Given a set of training data points, each belonging to one of two classes, an SVM training algorithm builds a model that assigns new data points to one of two classes. The hyperplane is chosen such that it maximizes the margin between the two classes. Data points that lie closest to the hyperplane are called support vectors, as they determine the position and orientation of the hyperplane.
The decision function for SVMs can be expressed as:

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b\right)$$

where:

- $f(x)$ is the decision function that predicts the class label of a new data point x.
- α_i are the Lagrange multipliers obtained during training.
- y_i are the class labels of the training data points.
- $K(x_i, x)$ is the kernel function that computes the similarity between two data points x_i and x.
- b is the bias term.

The choice of a kernel function $K(x_i, x)$ is critical in SVMs, as it determines the mapping of the input data into a higher-dimensional space. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid kernels.
The optimization problem for SVMs can be formulated as:

$$\text{minimize}\left(\frac{1}{2}\|w\|^2\right)$$

subject to the constraints:

$$y_i(w^T x_i + b) \geq 1, \quad \forall i$$

where w is the weight vector, and b is the bias term.

2.8.1.3 Limitations of SVM and proposed solutions

According to Tsyurmasto et al. (2014), SVM can be sensitive to outliers and overfitting because the support vectors (separation boundaries) are based on a small portion of the training data. Several approaches to circumvent the problem of outliers on SVM have been proposed as discussed in the next paragraph. Baldomero-Naranjo et al. (2021) also mentions that the other drawback of the classical SVM models aside from non-robustness to outliers, is that their accuracy depreciates when irrelevant features are used in the model. Baldomero-Naranjo et al. (2021) have introduced a modified SVM model which deals with both feature selection and outliers.

Song et al. (2002) and Zhang (1999) have proposed constructing a classification boundary using support vectors as well as class centers. They have found that the incorporation of the centres to the decision function makes it less detoured by outliers and improved the performance compared to when a standard SVM training is used. In an attempt to achieve a robust SVM, Wang (2020) proposes the use of concave and convex (CC) estimators.

Yang et al. (2007) have presented an alternative method to deal with outliers to improve outlier sensitivity to two-class data classification. Their approach is to assign different weights to different data points such that the Weighted Support Vector Machine (WSVM) training algorithm learns the decision surface according to the relative importance of data points in the training data set. The weights used in WSVM are generated by kernel-based possibilistic c-means clustering algorithm, whose partitioning generates relative high values for important data points but low values for outliers. Experimental results indicate that the proposed method reduces the affect of outliers and yields higher classification rate than standard SVM does when outliers exist in the training data set.

Mohammadi and Sarmad (2019) propose the use of the robust (MCD) Mahalanobis distance prior to employing the SVM algorithm: they called their algorithm MCD-SVM. In this algorithm, the robust Mahalanobis distance is used to remove outliers in each class before training the data. The logic behind the method is that support

vectors will constitute the outlying observations and hence the outliers will affect the classification accuracy of the SVM model. The results from their research confirmed the superiority of MCD-SVM over the traditional SVM algorithm.

Chapter 3

Methodology

This chapter gives summary information of the datasets used in this research work and the statistical procedures used to analyse the data in line with the aims and objectives of this study which are discussed in Section 1.3.

3.1 Data and computer software used

The analysis in this research was done using the R statistical software. The datasets analysed were

- A 2-features dataset of 1000 observations simulated using a Gaussian distribution. The dataset was mainly used to establish how well the CQAOD algorithm performs in detecting that the dataset comes from a single cluster. The purpose of simulating a dataset with one cluster is to examine the values of the cluster quality indices when varying number of clusters are used to cluster the data. The minimum value of k for which the indices can be calculated is 2. One would hope that the Dunn index and the VRC index would peak at $k = 2$ while the Davies-Bouldin index deeps at $k = 2$ for data with one cluster.

  The 2-variate Gaussian dataset with a total of 1000 observations and $p = 2$ artificial features X_1 and X_2 was simulated using R. The dataset (with Seed 1123) was generated using parameters $\mu = (100, 50)^T$ and variance-covariance

matrix $\mathbf{V}$:

$$\mathbf{V} = \begin{pmatrix} 10 & 5 \\ 5 & 10 \end{pmatrix}.$$

- The Iris data which is a well-known and publicly available dataset on three flower species: setosa, versicolor and virginica. The dataset consists of 150 observations on the following features: Sepal width, sepal length, petal width and petal length. The CQAOD algorithm has been applied on the dataset to check the quality of clusters obtained and to recommend the ideal number of clusters.

- Datasets with 4 features each were simulated from a mixture of $M = 5$ multivariate normal distributions. The datasets had varying number of outliers. The numbers of outliers f were 0, 40, 80, 120, 160 and 200 resulting in the number of observations in the datasets being 1000, 1040, 1080, 1120, 1160 and 1200, respectively. The objective was to explore the impact of outliers on clustering. The CQAOD algorithm was also applied on these datasets to check the efficacy of the algorithm in detection of outliers and in the process determining the optimal number of clusters. The datasets with $n = 1000$ observations each having $p = 4$ features/variables was simulated using the Mixsim R package. The functions *Mixsim* and *simdataset* from the package Mixsim were used. To ensure that only the effects of outliers on the K-means clustering algorithm are studied, the datasets were simulated in such a way that the assumptions of the K-means algorithm are satisfied. The simulation of data was done as follows:

  1. Set a seed of 1123 (for purposes of data reproducibility).
  2. Simulate a total of $n = 1000$ observations from $M = 5$ Gaussian mixture models. The R code used for this purpose is:

     MixSim(MaxOmega = 0.01,K = 5,p = 4,hom = TRUE,sph = TRUE,ecc = 1,int = c(0,50)).

$MaxOmega = 0.01$ in the code ensures a maximum overlap of 1% and this guarantees clear a distinction between features.

3. Simulate 200 outliers. From the 200 outliers, select the first 40 outliers and append them to the original dataset to create Dataset_{40}. From 200 outliers, select the first 80 outliers and append them to the original dataset to create Dataset_{80}. Continue this way to create Dataset_{120}, Dataset_{160} and Dataset_{200}. The maximum number of trials used to simulate the outliers has been set to 1000.

- Datasets with 4 features each were simulated from a mixture of $M = 10$ multivariate normal distributions (just like in the case of datasets where $M = 5$). The datasets had varying numbers of outliers. The numbers of outliers f were 0, 40, 80, 120, 160 and 200 resulting in the number of observations in the datasets being 1000, 1040, 1080, 1120, 1160 and 1200, respectively. The objective was to explore the impact of outliers on clustering when there is a higher number of clusters in the data (i.e. $M = 10$ instead of $M = 5$). The CQAOD algorithm was also applied on these datasets to check the efficacy of the algorithm in detection of outliers and in the process determining the optimal number of clusters.

- JSE shares data (See Section 3.1.1).

3.1.1 JSE data

The JSE data used in this project consists of 157 JSE listed companies and one JSE listed index, which is the JSETop40. JSE data on companies *price-earnings ratio* (P/E), *debt-to-equity ratio* (D/E), *return-on-assets ratio* (R/A), lagged daily closing price for previous 5 days (Lag1, Lag2, Lag3, Lag4 and Lag5), *volume* (the number of shares traded on the previous day, in billions) and *direction* (whether the index was Up or Down on this date) for the year 2020 to 2023 has been used. These quantitative measures are used by investors and investment professionals to evaluate companies' performance. An advantage of using financial ratios is that they enable one to compare companies of different sizes since they are not influenced by the size of a company.

Appendix C is a screenshot of the JSE data used.

The P/E ratio is a measure of a company's share price over earnings per share (EPS). This ratio is frequently employed as a valuation indicator to assess how different organisations compare in terms of value. The R/A is a ratio of a company net income and total assets. The R/A ratio measures the efficiency with which a company generates profit from the total assets listed on its balance sheet. R/A is expressed as a percentage; the higher the figure, the more effectively the management of a company manages its balance sheet to produce profits. D/E ratio shows how much debt a company has relative to its assets. A company's D/E ratio is found by dividing the amount of debt the company has by its shareholder's equity (assests less liabilities). A higher D/E ratio is generally indicative of the fact that it might be more difficult for the business to pay its debts.

3.2 Efficacy of the proposed CQAOD algorithm

In this project the efficacy of the proposed CQAOD algorithm is assessed using the simulated datasets and Iris dataset. As explained in Section 2.7, the algorithm is an iterative algorithm which makes use of the K-means algorithm in tandem with outlier detection. For outlier detection, the classical and robust Mahalanobis distance are used. The robust outlier detection makes use of the MCD algorithm to estimate the parameters used in the Mahalanobis distance as discussed in Section 2.5. The CQAOD makes use of clustering quality indices to determine an optimal number of clusters. The indices which have been used in this research work are the Dunn index, Davies-Bouldin index and the VRC.

3.3 Analysis of JSE data: Clustering and classification

In this study, some clustering algorithms have been applied on the JSE data using *price-earnings ratio* (P/E), *debt-to-equity ratio* (D/E) and *return-on-assets ratio* (R/A)

from the year 2020. Classification algorithms have also been applied to the JSE data using lagged daily closing prices (Lag1, Lag2, Lag3 and Lag5), volume and direction for the years 2020 to 2023.

3.3.1 Clustering the JSE data

The spectral clustering algorithm, DBSCAN clustering algorithm, CLARA clustering algorithm and hierarchical clustering algorithm have been applied on the the JSE dataset. As discussed in Section 2.6.2, the presence of outliers in the stock market can signify investing opportunities (i.e., higher returns than normal), since some of the best stock might actually be detected as outliers. In such a case, it is not prudent to exclude such stocks in the analysis of the JSE data. By doing this, we avoid excluding stocks that are removed as outliers due to them performing exceptionally well, therefore, in the analysis of JSE data, clustering is performed without the removal of outliers.

3.3.2 Formulating a diversified portfolio using K-means clustering algorithm

In the formulation of a diversified portfolio, there is a need to select an equal number of companies (roughly) with the highest Sharpe ratio from each cluster in a similar approach to the one used by Marvin (2015). A standard practice in South Africa is to construct a portfolio consisting of 40 companies. The performance of the constructed portfolio is compared against four popular well divesified exchange traded funds (ETFs) listed on the JSE. The ETFs to be compared against are 1nvest Top 40 ETF (ETFT40J), CoreShares S&P SA Top 50 Exchange Traded Fund (CTOP50J), Satrix 40 ETF(STX40J) and Sygnia Itrix Top 40 ETF (SYGT40J). The construction of the cluster based portfolio is as follows:

- Construct a portfolio from the resulting clusters.
- In the interest of having high returns and low risks, stocks are picked to maximize the return-to-risk ratio. Therefore, using the stock prices from the JSE, compute

the Sharpe ratio of each stock. The Treasury's 10-year bond yield for the year 2020 was $8,26\%$. This is used as the risk-free rate.

- From the clusters: select equal numbers (roughly) of stocks from each of the clusters. The stocks selected are the ones with the highest Sharpe ratio from each cluster. The clusters are ideally very different from each other. Therefore, a portfolio containing a mixture of stocks from each cluster will be diversified.
- For expedience of computations, assume an investor buys R100 worth of shares of each stock in the constructed portfolio at the beginning of the year 2021. This creates an equally weighted investment portfolio where each stock has equal weighting on the portfolio and the value of the portfolio is worth R4000 (40 stocks x R100).
- Track the monthly value of the portfolio from January 2021 to December 2022.
- Compute the Sharpe ratio of the cluster-based portfolio against the other ETFs after year 1 (2021) and again at the end of year 2 (2022).

3.3.3 Classification algorithms on the JSE data

In the analysis of outliers on classification algorithms, the logistic regression and support vector machines are fitted to the JSE data to predict direction using all of the lagged prices (Lag1 through Lag5) and Volume. To investigate the impact of outliers on logistic regression and SVM algorithm, a similar approach to that of Mohammadi and Sarmad (2019) is used. The investigation into the effects of outliers on the logistic regression and SVM algorithm proceeds as follows:

- Divide the JSE data into training and testing data. Data from 2021 to 2022 is used as training data and data from the year 2023 is used as testing data.
- Fit the logistic regression model and SVM model to the training data. Use the trained models to predict share price direction on the testing data.
- Construct classification tables and classification accuracy of the models.

- Segment the training data by classes, remove outliers in each class using the robust Mahalanobis distance.
- Combine the two training datasets after removing outliers. Train the logistic regression and SVM model on the new dataset with outliers removed.
- Predict the share price direction for both classification algorithms and compute the accuracy.
- Compare the accuracy of the two models before outlier removal and after outlier removal.

Chapter 4

Results

This chapter contains the results obtained in the analyses of the datasets discussed in Chapter 3.

4.1 Efficacy of CQAOD algorithm on simulated datasets

In this section, the CQAOD algorithm is applied on the simulated datasets in order to evaluate its efficacy. The results obtained from the use of CQAOD are presented.

4.1.1 Gaussian dataset with $M = 1$ cluster

The results presented in this section are for the 2-variate Gaussian dataset with a total of 1000 observations and $p = 2$ artificial features X_1 and X_2 was simulated using R. The dataset (with Seed 1123) was generated using parameters $\mu = (100, 50)^T$ and variance-covariance matrix $\mathbf{V}$:

$$\mathbf{V} = \begin{pmatrix} 10 & 5 \\ 5 & 10 \end{pmatrix}.$$

Figure 4.1 is a scatterplot of the features (X_2 versus X_1) and the boxplots of X_1 and X_2, respectively. From the boxplots, it can be seen that there are 7 outlier X_1 values and 1 outlier X_2 value. Interestingly, the number of observations retained after applying the CQAOD algorithm is not affected by the number of clusters used (see Figure 4.2 and 4.3).

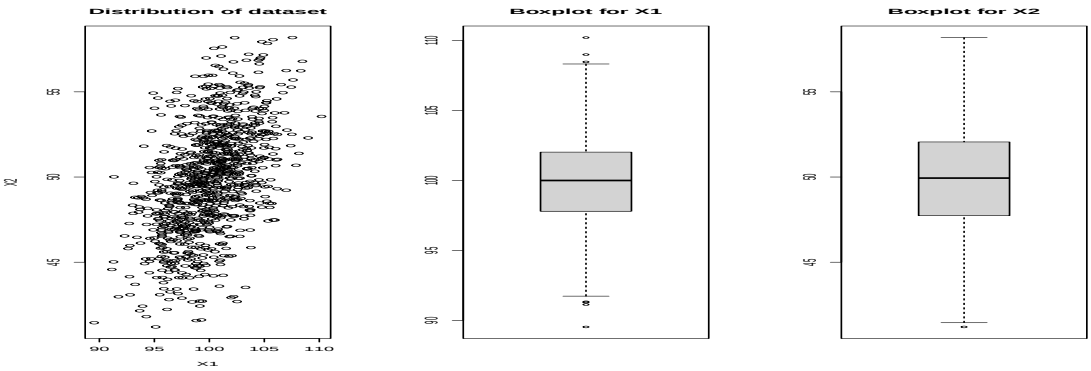

Figure 4.1: Plots for bivariate Gaussian dataset

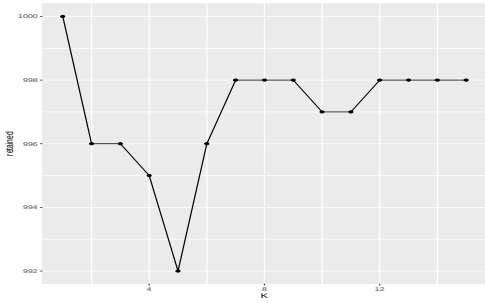

Figure 4.2: Number of observations retained vs k (classical)

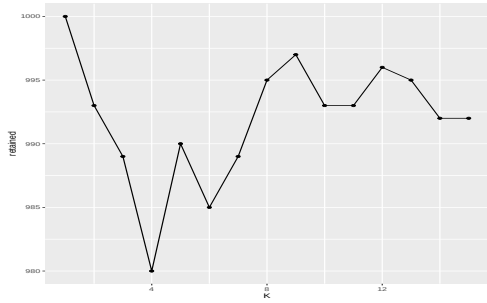

Figure 4.3: Number of observations retained observations vs k (MCD)

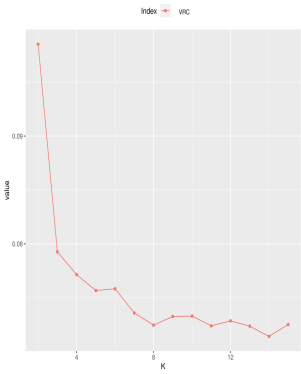

Figure 4.4: VRC index vs k (classical)

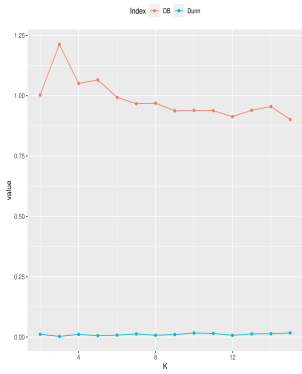

Figure 4.5: Dunn and DB indices vs k (classical)

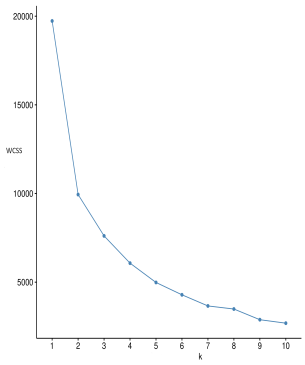

Figure 4.6: Screeplot (Classical)

For unclustered data (i.e. data from $M = 1$ cluster), the indices plot does not unequivocally tell us that the ideal number is indeed 1 because the plot starts at $k = 2$.

However, the VRC plots peak at $k = 2$ (for both classical and MCD) and from that one can infer that the ideal number of clusters may be 1 or 2. The scree plot does not show a distinctly clear elbow.

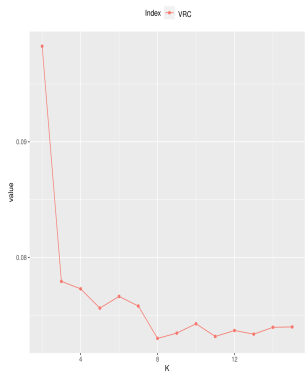

Figure 4.7: Line plot of VRC index vs k (MCD)

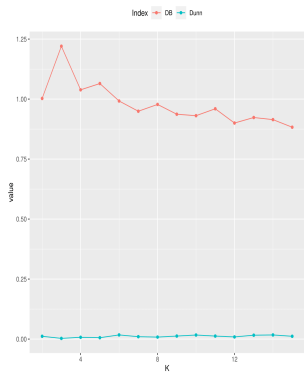

Figure 4.8: Line plots of Dunn and DB indices vs k (MCD)

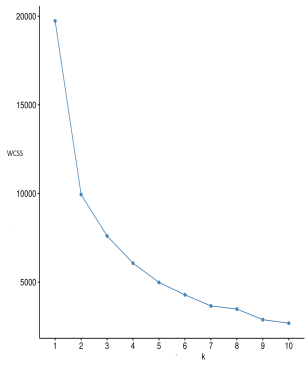

Figure 4.9: Scree-plot for bi-variate data (MCD)

4.1.2 Gaussian Datasets with more clusters and $p = 4$ features

This section gives the results obtained from the application of CQAOD on the two datasets with $n = 1000$ observations each having $p = 4$ features/variables which were simulated using the R statistical software; the first dataset was simulated using $M = 5$ clusters and the second dataset was simulated using $M = 10$ clusters.

4.1.2.1 Parameters for the clusters

The cluster means for the dataset with no outliers are:

$$\mu_1 = (37.98, 43.05, 43.29, 27.59)^T, \quad \mu_2 = (41.32, 8.10, 9.06, 35.67)^T,$$
$$\mu_3 = (45.83, 17.35, 28.85, 31.73)^T, \quad \mu_4 = (24.21, 23.31, 41.39, 21.22)^T,$$
$$\mu_5 = (32.25, 17.93, 6.32, 26.45)^T$$

$$V = \begin{bmatrix} 1.25 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.25 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.25 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.25 \end{bmatrix}$$

and the cluster centroids after the application of CQAOD are presented in Table 4.1 where the symbol p means convergence could not be achieved.

Table 4.1: Cluster centroids obtained from CQAOD

Method	Dataset$_0$, $f = 0$ outliers				Dataset$_{40}$, $f = 40$ outliers				...	Dataset$_{200}$, $f = 200$ outliers			
Classical	24.19	23.36	41.44	21.28	24.19	23.36	41.44	21.28	...	24.19	23.36	41.44	21.28
	37.92	42.88	43.36	27.53	37.92	42.88	43.36	27.53	...	37.92	42.88	43.36	27.53
	45.87	17.34	28.85	31.64	45.87	17.34	28.85	31.64	...	45.87	17.34	28.85	31.64
	41.39	8.14	8.85	35.68	41.39	8.14	8.85	35.68	...	41.39	8.14	8.85	35.68
	32.30	17.94	6.40	26.53	32.30	17.94	6.40	26.53	...	32.30	17.94	6.40	26.53
MCD	24.19	23.36	41.44	21.28	24.19	23.36	41.44	21.28	...	24.19	23.36	41.44	21.28
	37.92	42.88	43.36	27.53	37.92	42.88	43.36	27.53	...	37.92	42.88	43.36	27.53
	45.87	17.34	28.85	31.64	45.87	17.34	28.85	31.64	...	45.87	17.34	28.85	31.64
	41.39	8.14	8.85	35.68	41.39	8.14	8.85	35.68	...	41.39	8.14	8.85	35.68
	32.30	17.97	6.40	26.52	32.30	17.97	6.40	26.52	...	32.30	17.97	6.40	26.52

An examination of the variance-covariance matrices for the clustered datasets reveals that the absolute value of every covariance element in each case is approximately 0 while the variances are close to the values of the variances used to simulate the data. This is confirmation that the CQAOD algorithm does a good job in outlier detection and clustering.

The results in Table 4.1 shows that by applying the CQAOD algorithm, the centroids obtained after clustering are much closer to the actual centroids used to simulate the original clusters which are free of outliers.

4.1.2.2 Numbers of iterations and observations retained as non-outliers

Table 4.2 gives the approximate numbers of iterations done up to the point when no new outliers may be detected as well as the numbers of observations retained for the two datasets with different levels of outlier prevalence and different values of k. The

number of iterations for given values of k, outlier prevalence and the parameter estimation method (i.e. classical or MCD) are not always the same as they depend on the initial centroids selected when the K-means algorithm commences.

The pitfalls of the CQAOD algorithm that have been identified are:

- Sometimes the iterations do not converge fast enough and they go on and on with signs of potential convergence being there though
- At some iteration some clusters generated are too small in number to enable computation of the estimate of the variance-covariance matrix.

Table 4.2: Number of iterations and number of observations retained

Number of iterations

			Number of outliers, f					
M	Method	k	0	40	80	120	160	200
5	Classic	2	2	4	5	4	5	5
		5	2	3	3	3	3	4
		8	6	4	4	3	3	3
		11	18	p	p	3	3	3
	MCD	2	1	2	2	2	2	2
		5	2	3	3	3	3	3
		8	14	p	6	6	6	3
		11	p	p	p	p	7	3
10	Classic	2	2	3	2	5	5	5
		5	3	4	4	5	5	6
		10	3	4	4	5	5	5
		11	9	4	4	5	5	5
	MCD	2	1	2	2	2	4	3
		5	2	4	4	3	3	2
		10	2	4	4	3	3	2
		11	5	p	p	p	3	2

Number of observations retained

	Number of outliers, f					
k	0	40	80	120	160	200
2	983	983	983	983	984	986
5	978	978	978	978	978	978
8	963	999	1026	1050	1075	1100
11	936	p	p	1071	1097	1124
2	986	986	986	986	987	988
5	976	976	976	976	976	976
8	963	p	1014	1041	1056	1098
11	p	p	p	p	1071	1124
2	998	1004	1012	1015	1023	1033
5	985	986	988	989	989	989
10	978	978	978	978	978	978
11	973	983	988	991	1000	1008
2	998	1005	1014	1018	1026	1035
5	977	976	978	981	982	982
10	963	960	961	963	963	963
11	947	p	p	p	985	993

4.1.2.3 Ideal number of clusters and parameter estimates

As discussed in Section 2.5.1, the ideal number of clusters to use when doing K-means clustering (or clustering in general) is critically important. In this work the use of clus-

tering quality indices as a means of getting the ideal number of clusters is discussed. The argument here is that the value of k which yields the smallest value of the Davies Bouldin index and almost certainly concurrently results in the maximization of the Dunn index as well as the maximization of VRC is advanced. Figure 4.10 to Figure 4.33 give line plots of the Dunn index, the Davies Bouldin index and the Variance Ratio Criterion (VRC) as well as the corresponding scree-plots obtained from CQAOD using the classical method for different values of k for data simulated using $M = 5$ clusters. Figure 4.10 and Figure D.3 are apparently giving conflicting results for the ideal number of clusters for the dataset without outliers. The indices plots are correctly suggesting the ideal number as 5 whilst the scree-plot does not give a clear-cut answer - arguably, from the scree-plot, one can deduce that the number is either 4 or 6. For the CQAOD using the MCD method, similar conflicting results are given in that the indices plots once again agree on the optimum number of clusters being 5 while the scree-plot suggest 4. The poor performance of the scree-plot compared to the CQAOD procedure as means of determining the ideal number of k is exhibited throughout from Figure 4.10 to Figure 4.33.

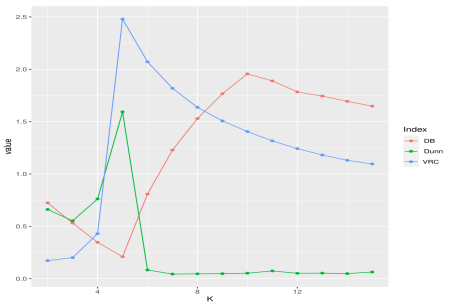

Figure 4.10: Classic (M=5, f=0)

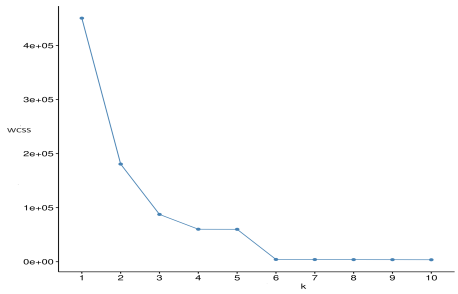

Figure 4.11: Screeplot (M=5, f=0)

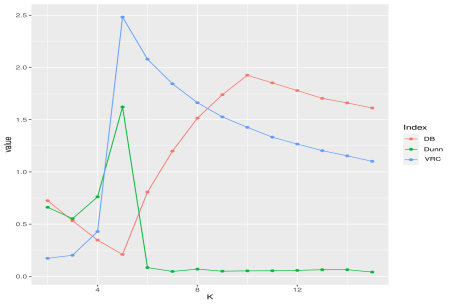

Figure 4.12: MCD (M=5, f=0)

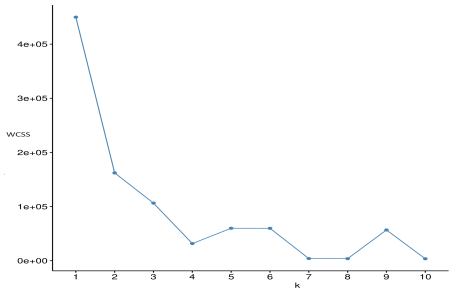

Figure 4.13: MCD (M=5, f=0)

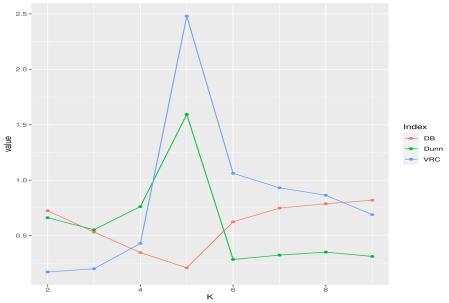

Figure 4.14: Classic (M=5, f=40)

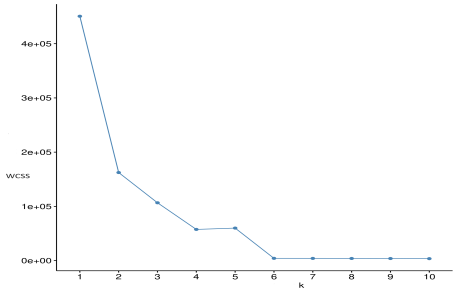

Figure 4.15: Screeplot (M=5, f=40)

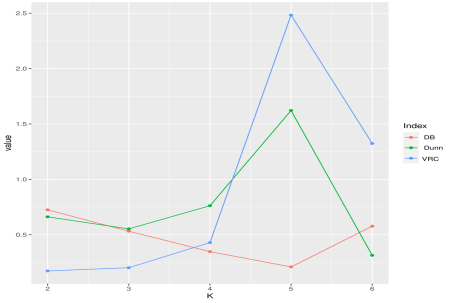

Figure 4.16: MCD (M=5, f=40)

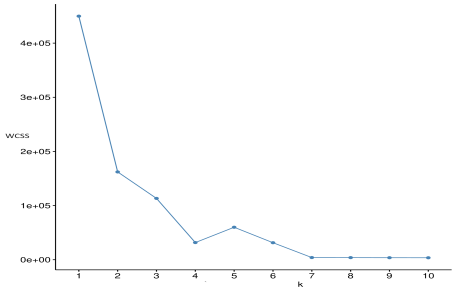

Figure 4.17: Screeplot (M=5, f=40)

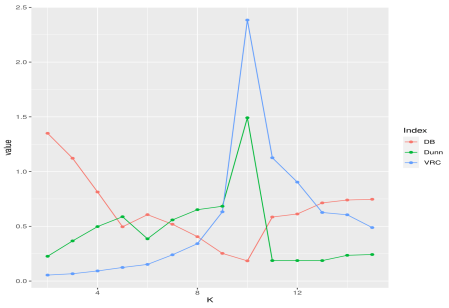

Figure 4.18: Classic (M=5, f=80)

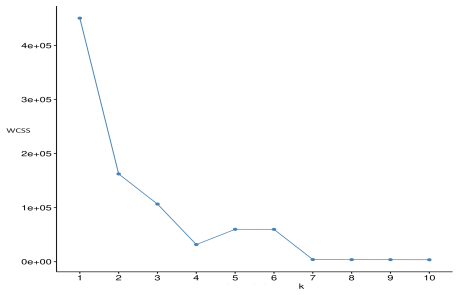

Figure 4.19: Screeplot (M=5, f=80)

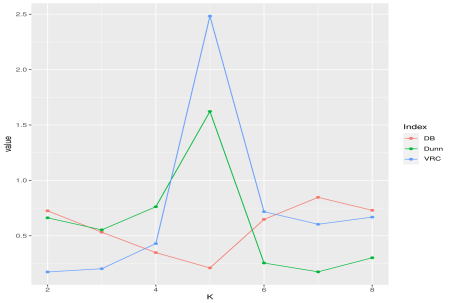

Figure 4.20: MCD (M=5, f=80)

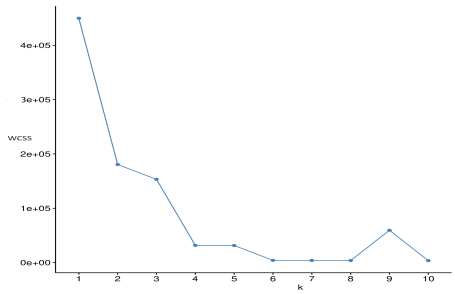

Figure 4.21: Screeplot (M=5, f=80)

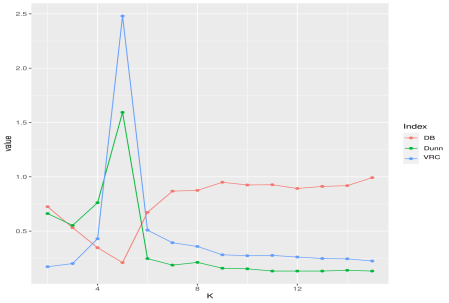

Figure 4.22: Classic (M=5, f=120)

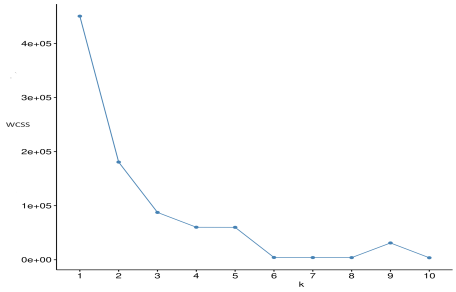

Figure 4.23: Screeplot (M=5, f=120)

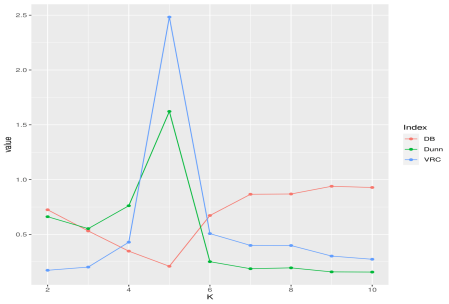

Figure 4.24: MCD (M=5, f=120)

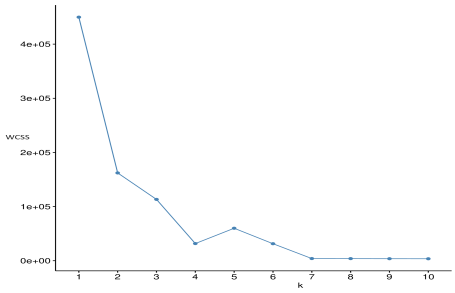

Figure 4.25: MCD (M=5, f=120)

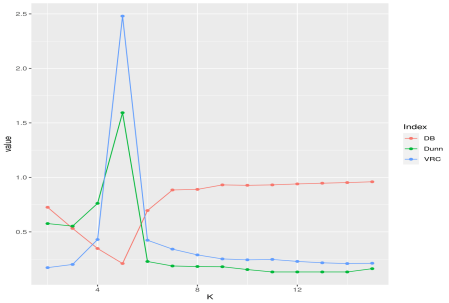

Figure 4.26: Classic (M=5, f=160)

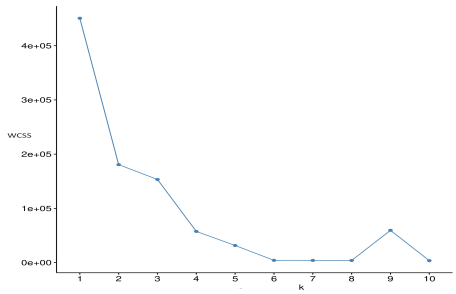

Figure 4.27: Screeplot (M=5, f=160)

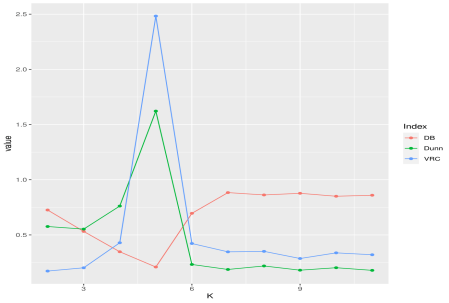

Figure 4.28: MCD (M=5, f=160)

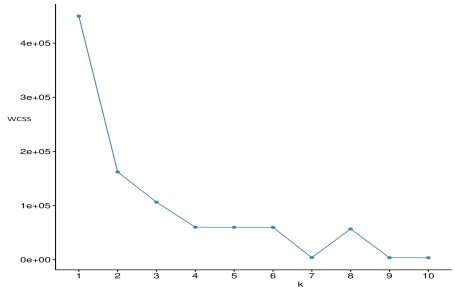

Figure 4.29: Screeplot (M=5, f=160)

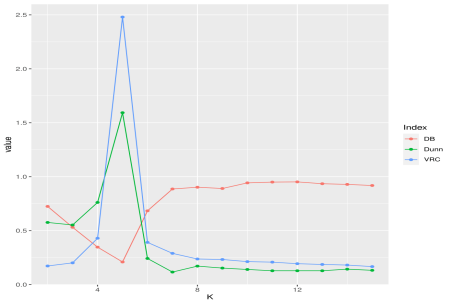

Figure 4.30: Classic (M=5, f=200)

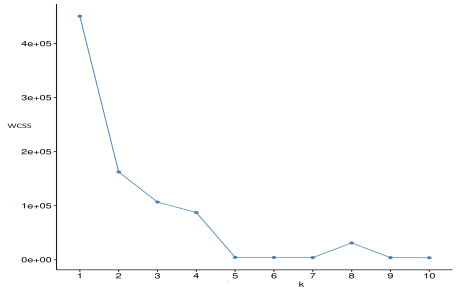

Figure 4.31: Screeplot (M=5, f=200)

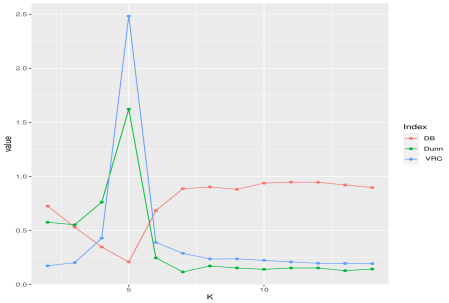

Figure 4.32: MCD (M=5, f=200)

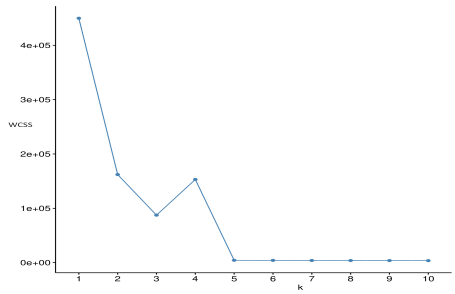

Figure 4.33: Screeplot (M=5, f=200)

The impact of increasing outlier incidence (as the number of outliers increases from $f = 0$ to $f = 200$) when $M = 5$ is illustrated in Figures A.1 to Figure A.6. Similarly, the impact of increasing outlier incidence when $M = 10$ is illustrated Figures A.7 to Figure A.12. Table 4.3 gives the summary of recommended optimal numbers of clusters to use for clustering for different levels of outlier adulteration or contamination. With $M = 5$ clusters, its observed that there is an increasing level of discord in as far as the indices plot recommendations of the optimal number of clusters with the Davies Bouldin index exhibiting robustness. When $M = 10$, the Davies Bouldin again exhibits robustness for moderate levels of contamination and is seen to give unreliable values for high levels of outlier contamination.

Table 4.3: Numbers of recommended clusters

	$M = 5$						$M = 10$					
f	0	40	80	120	160	200	0	40	80	120	160	200
Davies-Boudin	5	5	5	5	5	5	10	10	10	10	5	5
Dunn	5	7	15	9	15	15	10	18	15	15	15	18
VRC	5	6	6	6	6	6	10	15	11	15	12	12

Figures 4.34 to Figure 4.57 give line plots of the Dunn index, the Davies Bouldin index and the Variance Ratio Criterion (VRC) as well as the corresponding scree-

plots obtained from CQAOD using the robust MCD and classical methods for different values of k for data simulated using M = 10 clusters. For the dataset with 80 outliers, when $k > 10$, at some iteration some clusters generated are too small in number to enable computation of the estimate of the variance-covariance matrix for the MCD method. Therefore the indices method may not be relied on to give a credible result in such a case. For all other situations, the scree-plots are suggesting the ideal number of clusters to be between 6 and 7. The scree-plots in this case seriously and consistently under-estimate the true number of clusters.

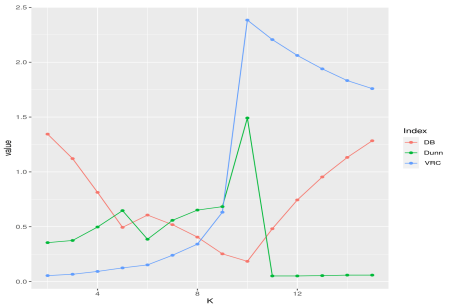

Figure 4.34: Classic (M=10, f=0)

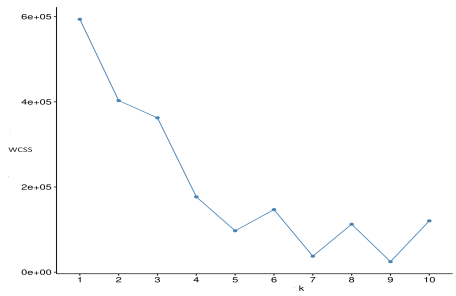

Figure 4.35: Screeplot (M=10, f=0)

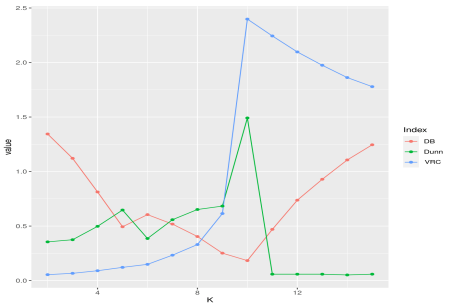

Figure 4.36: MCD (M=10, f=0)

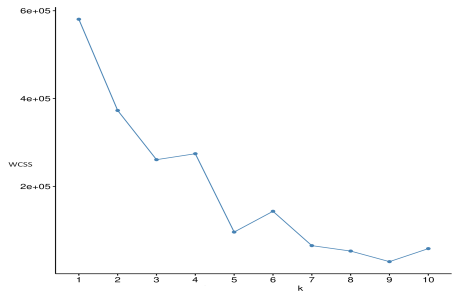

Figure 4.37: Screeplot (M=10, f=0)

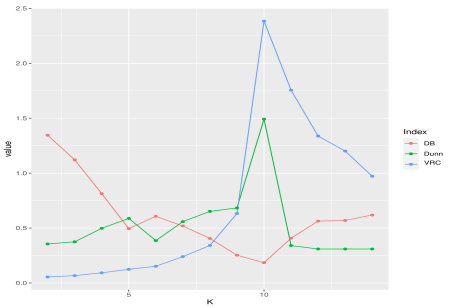

Figure 4.38: Classic (M=10, 5,f=40)

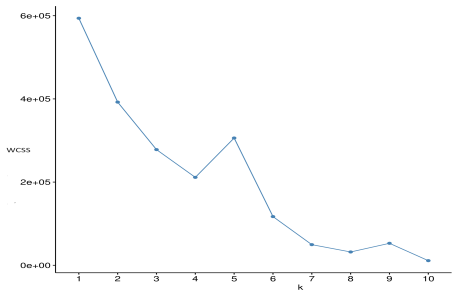

Figure 4.39: Screeplot (M=10, f=40)

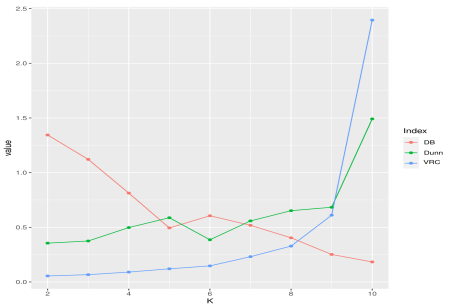

Figure 4.40: MCD (M=10, f=40)

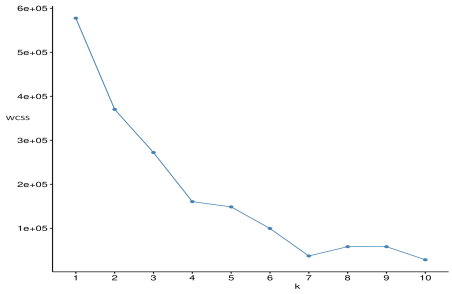

Figure 4.41: Screeplot (M=10, f=40)

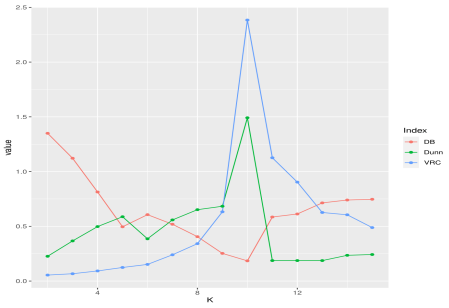

Figure 4.42: Classic (M=10, f=80)

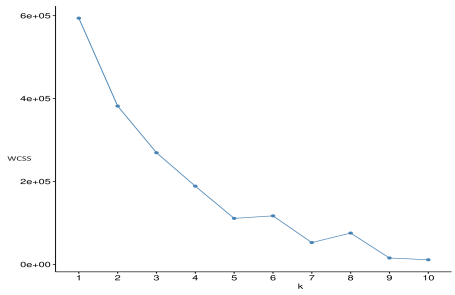

Figure 4.43: Screeplot (M=10, f=80)

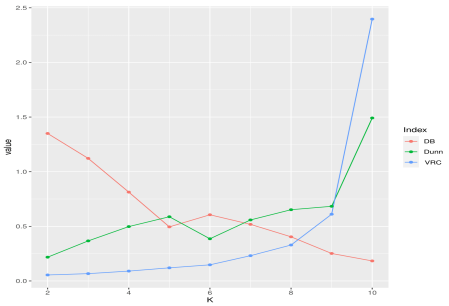

Figure 4.44: MCD (M=10, f=80)

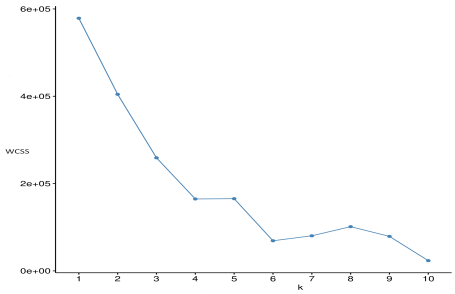

Figure 4.45: Screeplot (M=10, f=80)

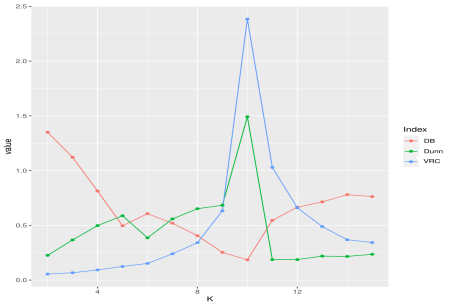

Figure 4.46: Classic (M=10, f=120)

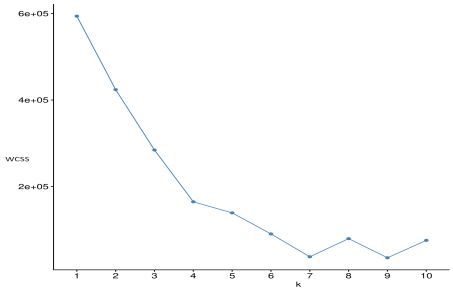

Figure 4.47: Screeplot (M=10, f=120)

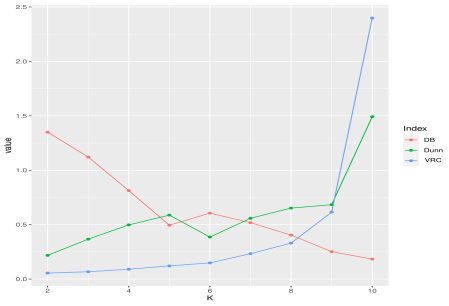

Figure 4.48: MCD (M=10, f=120)

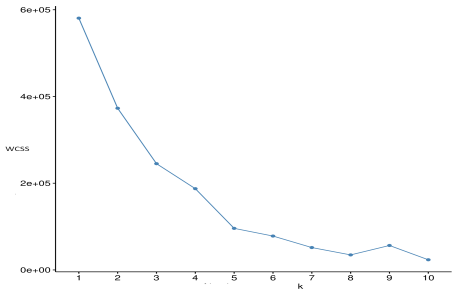

Figure 4.49: Screeplot (M=10, f=120)

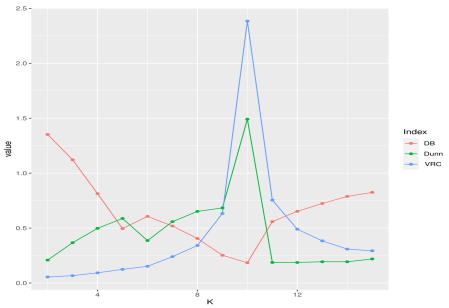

Figure 4.50: Classic (M=10, f=160)

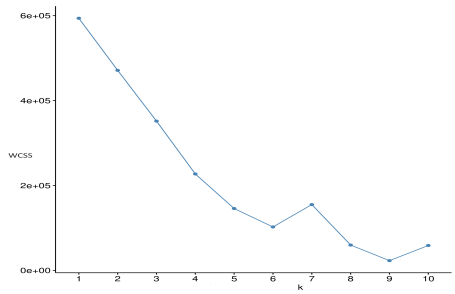

Figure 4.51: Screeplot (M=10, f=160)

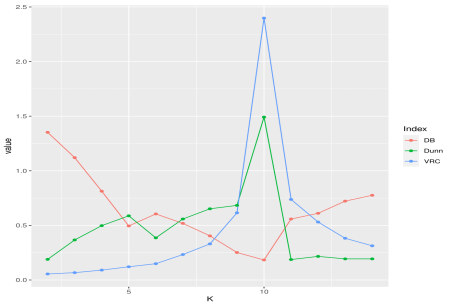

Figure 4.52: MCD (M=10, f=160)

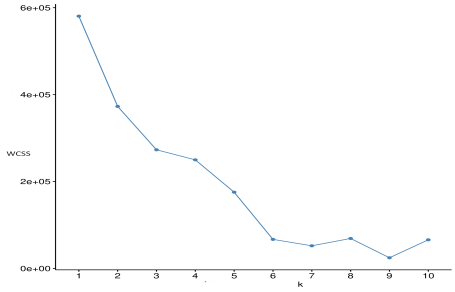

Figure 4.53: Screeplot (M=10, f=160)

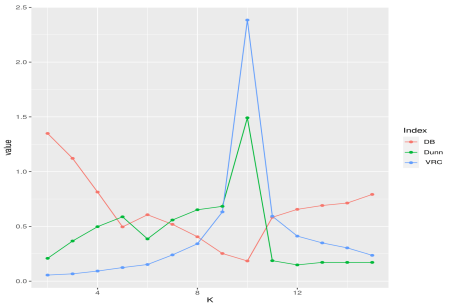

Figure 4.54: Classic (M=10, f=200)

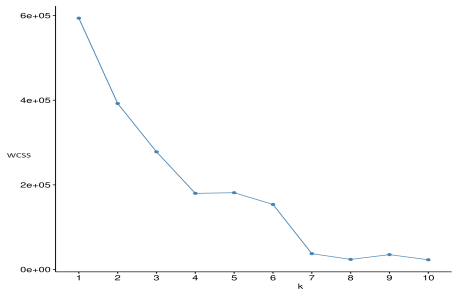

Figure 4.55: Screeplot (M=10, f=200)

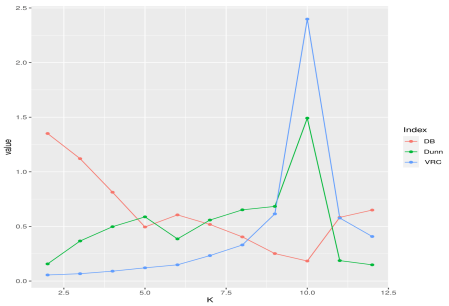

Figure 4.56: MCD (M=10, f=200)

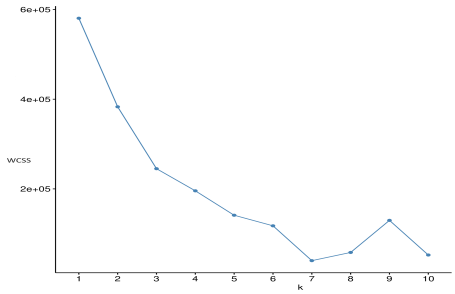

Figure 4.57: Screeplot (M=10, f=200)

4.2 The Iris dataset

Gupta and Panda (2018) have compared K-Means clustering of Iris data and Clustering Large Applications (Clara) algorithms for the same dataset. The dataset relates to three species of flowers which are setosa, versicolor and virginica. On each of a random sample of the flowers, measurements on $p = 4$ features have been recorded: Sepal length, sepal width, petal length, and petal width. The data for the Setosa species were found not follow a multivariate normal distribution while those of the other species do follow normal distributions (Korkmaz et al., 2014).

Sugar and James (2003) have applied the jump method on the Iris dataset. In their paper they report that a reasonable choice of the underlying number of clusters is either 2 or 3. Saha and Mukherjee (2021) have applied CNAK algorithm and predicted the number of clusters for the Iris dataset to be 3.

The plots in Figures 4.58 and 4.59 are both suggesting that the ideal number of clusters is 2. Interestingly, when $k = 2$ is used for clustering, all the flowers that are of the Setosa species fall in one cluster while the Versicolor and Virgica fall in the other cluster. The boxplots for all the four feature variables, the three flower species and 2 clusters (call them C_1 and C_2) when clustering is done using $k = 2$ in Appendix B are very revealing.

The boxplots in Appendix B are for the 4 features and $k = 3$ clusters. When $k = 3$ is used for clustering, all the flowers of the Setosa species fall in one cluster (call it C_1). A majority of the versicolor flower species fall in Cluster C_2 and a majority of the virginica flower species fall in cluster C_3. Cluster C_2 is adulterated with a few Virginica flowers and Cluster C_3 is adulterated with a few Versicolor flowers. The critical question which then arises at this juncture is: are cluster centers of C_2 and C_3 materially different? If not, then an argument can be made that by using $k = 3$, the clustering is superfluous. Mann-Whitney tests were conducted individually on the features to test whether or not there is significant differences between Virginica and Versicolor. [1]

[1] Petal width (p-value $p = 2.2 \times 10^{-16}$), Petal length ($p = 2.2 \times 10^{-16}$), Sepal width($p = 0.004572$) and Sepal length ($p = 5.869 \times 10^{-7}$)

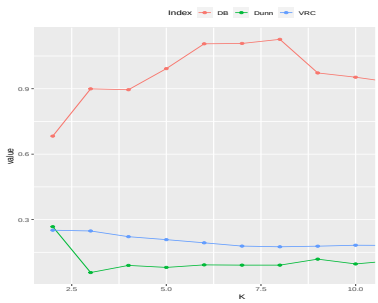

Figure 4.58: Classical

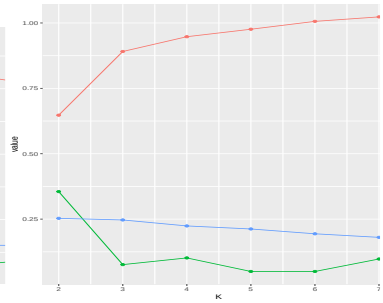

Figure 4.59: MCD

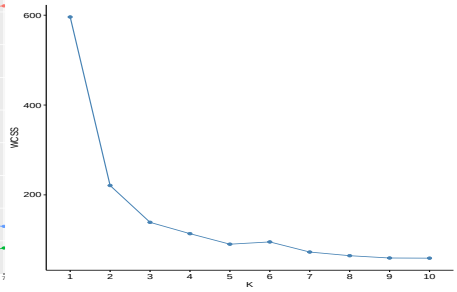

Figure 4.60: Scree plot

4.3 Results of the JSE data

This section gives the results obtained from the analysis of JSE data.

4.3.1 Preliminary analysis of features

The histograms in Figure 4.61 depict the distributions of the features while Figure 4.62 contains the boxplots of the three features. The P/E ratio dataset is skewed to the left while D/E and R/A datasets are skewed to the right; the coefficients of skewness are -3.87, 3.13 and 2.72, respectively. .

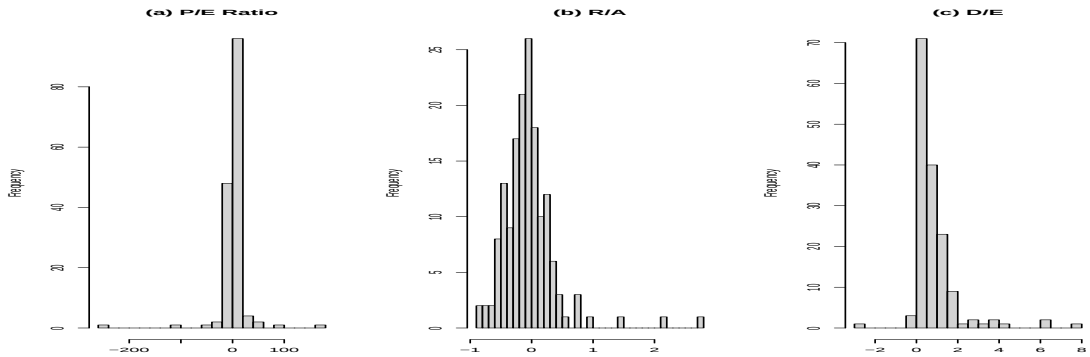

Figure 4.61: Histograms of JSE data

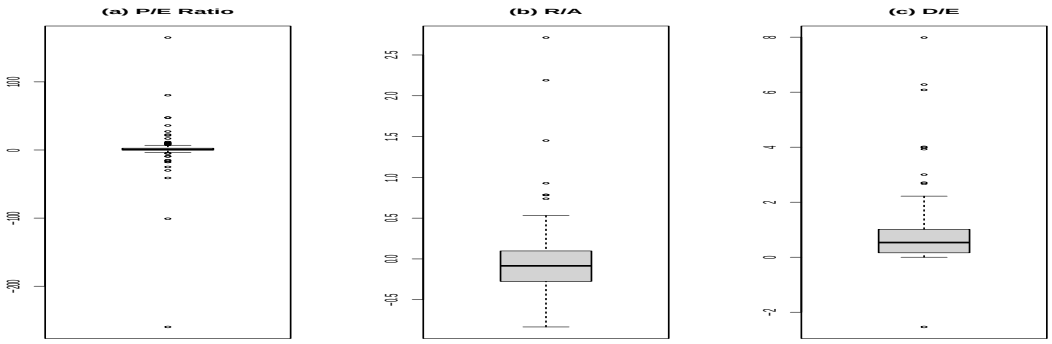

Figure 4.62: Box plots of JSE data

The multivariate normality tests comprising Doornik-Hansen, Royston and Henze-Zirkler's multivariate normality tests were conducted on the JSE dataset. The MVN package in R was used for the purpose. The tests gave p-values of 0.00, 2.03×10^{-49} and 9.76×10^{-71}, respectively. This indicates that all three tests say that the JSE dataset does not follow a multivariate normal distribution.

4.3.2 Clustering results

Dubes and Jain (1976), Jain and Law (2005) and Kapil et al. (2016) mention that one famous problem when it comes to clustering is called *the clustering dilemma*. The problem revolves around trying to answer the question: what is the best clustering algorithm for the given dataset?

An application of the CQAOD algorithm on the JSE data has been done for purpose of checking if the data possibly has outliers. When $k = 2$ clusters are used in the CQAOD, the number of stocks retained as non-outliers for the two clusters are 77 and 44, respectively. This means 36 stocks were detected as outliers. Tests for multivariate normality for the two clusters data were conducted using Doornik-Hansen, Royston and Henze-Zirkler's multivariate normality tests and for all the three tests and both clusters, the hypotheses of the data following multivariate normal distributions were rejected. This then put paid to the idea of clustering the data using model based clustering assuming a mixture of two multivariate normal distributions.

When $k = 3$ clusters are used in the CQAOD, the number of stocks retained as non-outliers for the two clusters are 77 and 44 and 7, respectively. This means 29 stocks were detected as outliers. Tests for multivariate normality for the two of the larger clusters data were conducted again using Doornik-Hansen, Royston and Henze-Zirkler's multivariate normality tests and for all the three tests and both clusters, the hypotheses of the data following multivariate normal distributions were rejected. This then put paid to the idea of clustering the data using model based clustering assuming a mixture of three multivariate normal distributions. Application of CQAOD using $K \geq 4$ met with the challenge of small clusters.

Other candidate clustering algorithms that were then applied on the JSE data without outlier weeding are the K-means clustering algorithm, spectral clustering algorithm, CLARA clustering algorithm and hierarchical clustering algorithm. Figures 4.63, 4.66 and 4.67 are the line plots of the three indices versus k when the K-means clustering algorithm, spectral clustering algorithm and Clara clustering algorithms are applied, respectively.

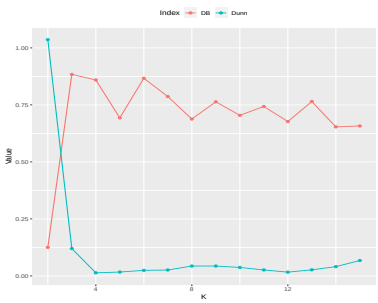

Figure 4.63: Dunn & DB

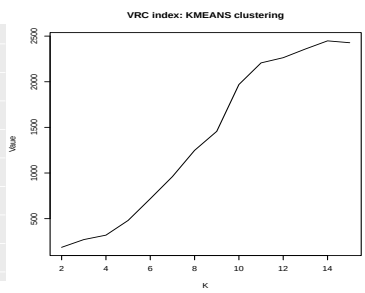

Figure 4.64: VRC index

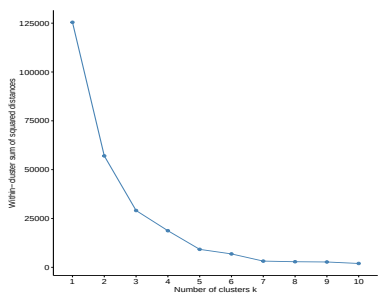

Figure 4.65: Screeplot

Table 4.4 gives a summary of the optimal numbers that the line plots of indices recommend when each of the three clustering algorithms is used to cluster the JSE data. The apparent discord or disharmony in the number of the three indices may be attributed to the higher number of outlier stocks in the data or possibly clusters which may not have clear separation. The line plot of the VRC versus k appears to be increasing with

increasing values of k and hence does not give a conclusive optimal number of k.

Table 4.4: Numbers of recommended clusters

	Dunn	Davies-Boudin	VRC
K-means	2	2	inconclusive
Spectral	10	2	2
CLARA	14	15	20

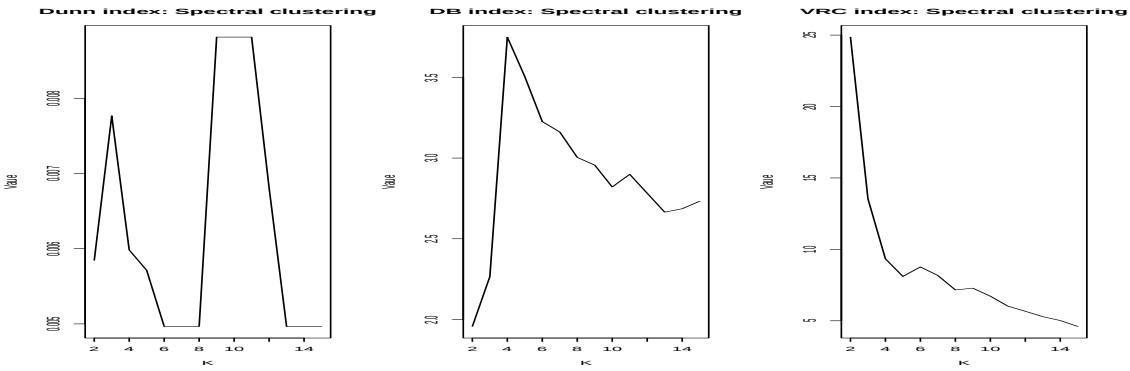

Figure 4.66: Spectral clustering

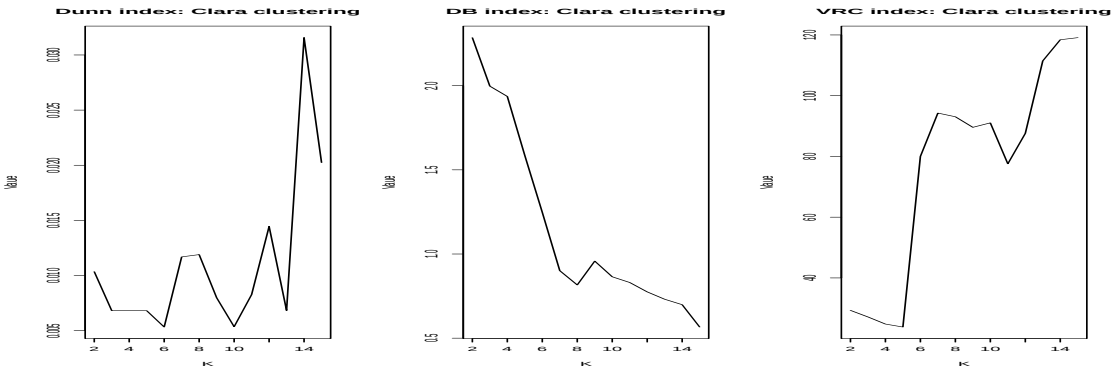

Figure 4.67: CLARA clustering

Figure 4.68 is the dendrogram obtained after employing hierarchical clustering on the JSE data (for all 157 stocks). The four cuts defined by lines cutting across the den-

drogram (the blue (Line 1), purple (Line 2), green (Line 3) and red (Line 4)) help to decide on the ideal number of clusters. Line 1, which is suggestive of 2 clusters with 156 observations in one cluster and 1 observation in the other cluster. This maybe tantamount to saying that there is 1 cluster with 156 observations and the odd 157th observation is an outlier. Line 2 is suggestive of three clusters (with the clusters having 152, 4 and 1, respectively). It can be argued that the use $k = 3$, results in 1 clusters and 2 small clusters that can be viewed as outlier clusters. Line 3 is suggestive of four clusters with the clusters having 152, 3, 1 and 1 observations, respectively. Again this be viewed as the case of one cluster and three outlier clusters. Line 4 partitions the observations into 5 clusters (the numbers being 58, 94, 3, 1 and 1, respectively) which translates to 2 genuine clusters and 3 outlier clusters.

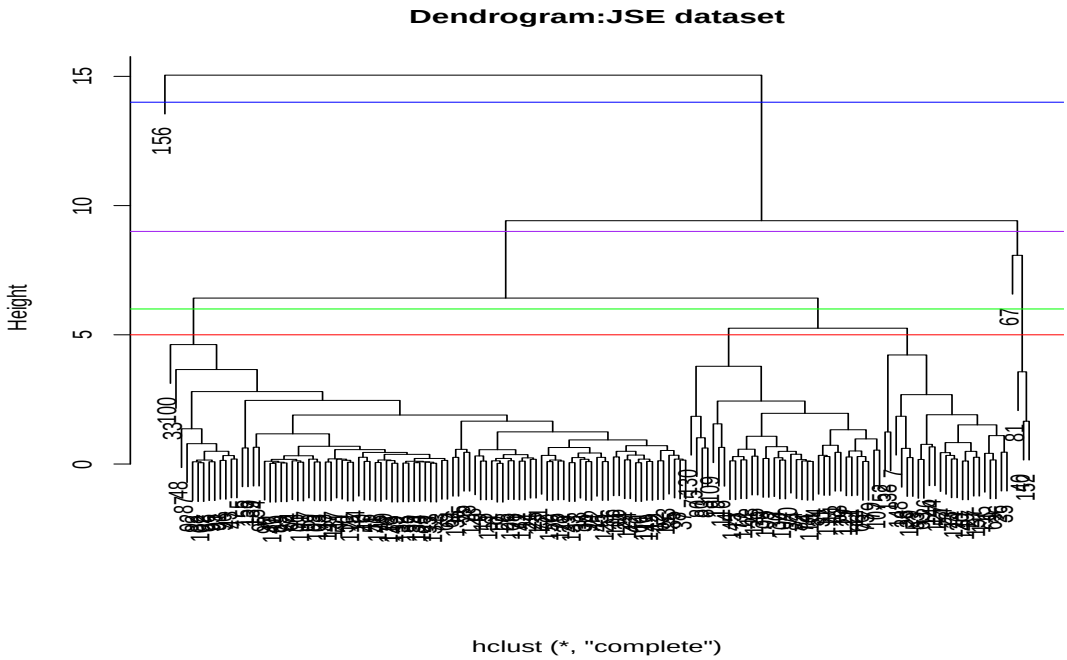

Figure 4.68: Dendrogram from hierarchical clustering of JSE data

Proceeding with this line of approach, the following are found:

- For $k = 6$, the sizes of the clusters are 94, 35, 23, 3, 1 and 1, respectively, indicating that the number of genuine clusters is 3.

- For $k = 7$, the sizes of the clusters are 93, 35, 23, 3, 1, 1 and 1, respectively indicating that the number of genuine clusters is again 3.
- For $k = 8$, the sizes of the clusters are 93, 35, 21, 3, 2, 1, 1 and 1, respectively indicating yet again that the number of genuine clusters is 3.
- For $k = 9$, the sizes of the clusters are 93, 31, 21, 3, 2, 1, 4, 1 and 1, respectively, indicating yet again that the number of genuine clusters is 3.
- For $k = 10$, the sizes of the clusters are 92, 31, 21, 4, 3, 2, 1, 1, 1 and 1, respectively, indicating once more that the number of genuine clusters is 3.

From the foregoing discussion, it is observed that the number of big clusters is steadfast at 3. From this observation, one can be confidently infer that the dendrogram indicates that the ideal number of clusters in the JSE dataset is 3. In the next section, thus, the number of clusters used to construct a diversified portfolio is 3.

4.4 Portfolio creation

In this section, the analysis of the portfolio is done. Forty stocks are selected from the three clusters based on their Sharpe ratio from 2020. Table 4.5 shows the number of stocks picked from each cluster.

Table 4.5: Stock selected from each cluster

Cluster	Number of stocks in cluster	Number of stocks selected
1	94	25
2	35	9
3	23	6

Figure 4.69 is a boxplot of the monthly simple returns for each index from the year 2021 to 2022. The index created is named "Cluster based stock" and two things stand

out when looking at its boxplot. Firstly the median monthly simple return is higher than the other indices and secondly, the monthly simple returns also seem to have the most variation. It is desirable that the variation in returns should be low since it forms part of the Sharpe ratio calculation used to determine the "risk-to-reward" of the returns. The median of the cluster-based monthly simple returns is the highest, and the monthly simple returns are negatively skewed.

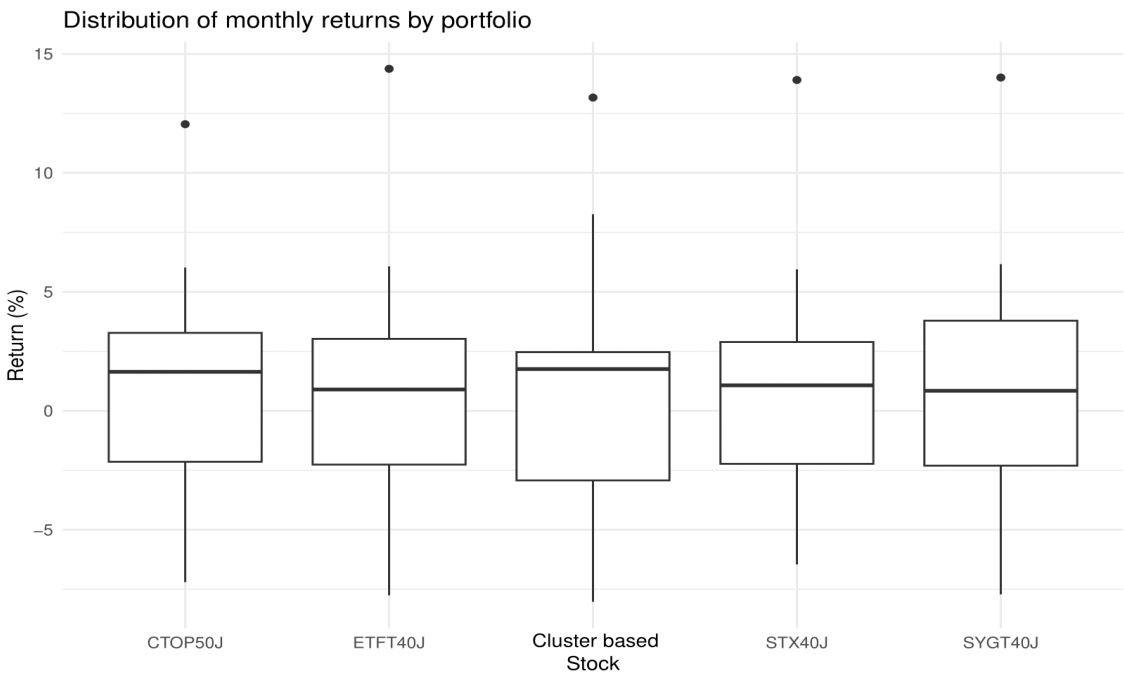

Figure 4.69: Monthly simple returns

A visual inspection of the cumulative returns (the monthly cumulative returns measure the overall gain/loss made on the investment) for the first twelve months are presented in Figure 4.70.

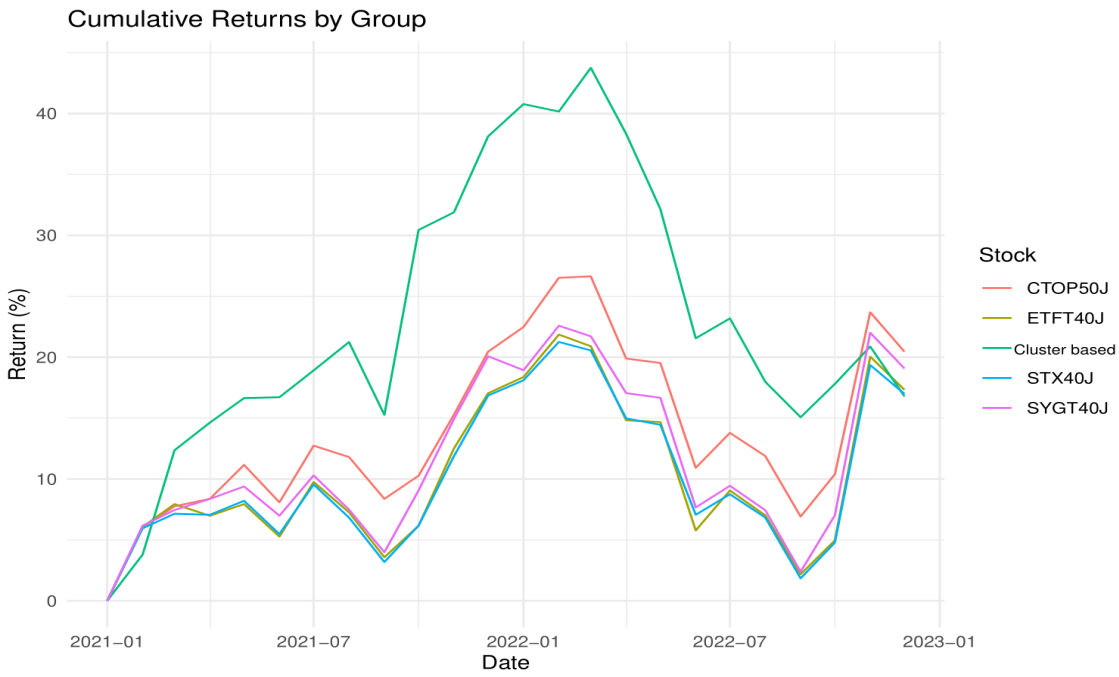

Figure 4.70: Monthly cumulative returns

From Table 4.6, it can be seen that the cluster-based index, with a return of 20.62% outperformed all the other indices when it comes to cumulative returns. The index with the least volatility was CTOP50J, with a standard deviation of 3.06. When looking at the Sharpe ratio, the performance of the indices from best to worst are: CTOP50J, cluster-based index, SYGT40J, ETFT40J and STX40J.

Table 4.6: Comparisons of cluster index against popular indices after year 1

	ETFT40J	CTOP50J	STX40J	SYGT40J	ClusterIndex
Return	17.04	20.45	16.85	20.08	20.62
Standard deviation	3.49	3.06	3.34	3.40	3.63
Sharpe Ratio	1,64	2,12	1,64	2,02	2,06

From Table 4.7, it can be seen that the CTOP50J index, with a return of 20.45% outperformed all the other indices when it comes to cumulative returns after 24 months. It

is also the index with the least volatility. With a Sharpe ratio of 1.73, the cluster-based index is the third best risk-adjusted investment following CTOP50J (Sharpe ratio of 1.96) and SYGT40J (Sharpe ratio of 1.75). In terms of cumulative returns aspect, the cluster-based index has the second-highest return of 19.65% only beaten by CTOP50J index.

Table 4.7: Comparisons of cluster-based portfolio against popular indices after year 2

	ETFT40J	CTOP50J	STX40J	SYGT40J	cluster-based
Return	17.32	20.45	16.99	19.07	19.65
Standard deviations	4.65	4.23	4.40	4.59	5.36
Sharpe Ratio	1.53	1.96	1.52	1.75	1.73

From the foregoing results, the inference made is that the cluster-based index performs well on the basis of the cumulative returns aspect. Its downside is that its returns are more volatile compared to other indices as reflected by the fact that it has the highest standard deviation. On a risk-adjusted returns basis, measured using the Sharpe ratio, the cluster-based index appears to be an average performer.

4.5 Classification models

In this section, the results obtained from the prediction of share price after using the logistic regression model (LR) and SVM model on the JSE dataset are presented.

4.5.1 The logistic regression model

In this section, the results obtained from the prediction of share price after employing logistic regression (LR) on the JSE dataset are presented. The variables Direction, Lag1, Lag2, Lag3, Lag4, Lag5 and Volume are denoted by y, x_1, x_2, x_3, x_4, x_5 and x_6,

respectively. The correlation matrix for the variables was computed using R and found to be:

	Lag5	Lag4	Lag3	Lag2	Lag1	Volume
Lag5	1.00	0.99	0.99	0.98	0.97	−0.46
Lag4	0.99	1.00	0.99	0.99	0.98	−0.47
Lag3	0.99	0.99	1.00	0.99	0.99	−0.48
Lag2	0.98	0.99	0.99	1.00	0.99	−0.48
Lag1	0.97	0.98	0.99	0.99	1.00	−0.48
Volume	−0.46	−0.47	−0.48	−0.48	−0.48	1.00

A plot of the variance inflation factor confirms the presence of multicollinearity.

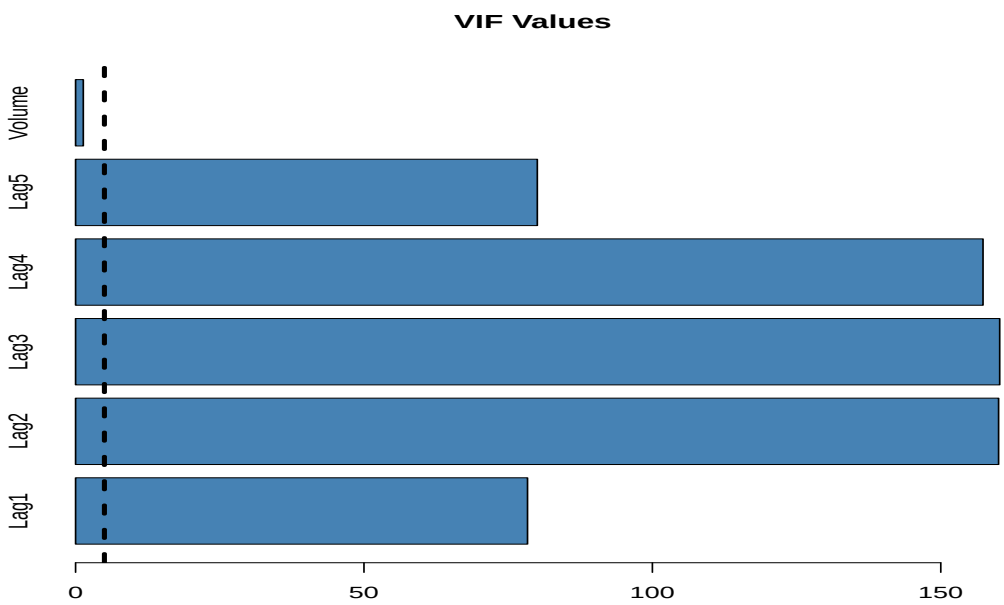

Figure 4.71: Variance inflation factor

The logistic regression models before (Model 1) and after removing outliers (Model 2) that were fitted are given in Equation 4.1 and Equation 4.2:
Model 1: Before outlier removal

$$\begin{aligned} y &= 2.405 - 3.561 \times 10^{-5}x_1 - 1.593 \times 10^{-4}x_2 + 9.353 \times 10^{-5}x_3 - 1.023 \times 10^{-4}x_4 \\ &\quad +1.71 \times 10^{-4}x_5 - 3.072 \times 10^{-3}x_6. \end{aligned} \tag{4.1}$$

Model 2: After outlier removal using $\alpha = 0.025$

$$\begin{aligned} y &= 4.204 - 6.380 \times 10^{-5}x_1 - 1.888 \times 10^{-4}x_2 + 5.980 \times 10^{-5}x_3 - 8.625 \times 10^{-5}x_4 \\ &\quad +2.256 \times 10^{-4}x_5 - 8.009 \times 10^{-3}x_6. \end{aligned} \tag{4.2}$$

The explanatory variables are clearly dogged by the problem of multicollinearity.

In Model 3 (Equation 4.3), only one variable, which is x_5 is significant (p-value of 0.00828) in the prediction of share price direction. After the removal of outliers (Model 4.4), two variables, x_5 and x_6 (with p-values of 0.04610 and 0.00027, respectively) are seen to be significant. Apparently, the presence of outliers clouds or masks the importance of volume (x_6) in predicting share price direction. One additional clouding factor is the multicollinearity.

Backward variable selection was next done and the variables x_5 and x_6 were retained. The logistic regression models (Model 3) before and after removing outliers (Model 4) that were fitted using the retained predictor variables are given below.
Model 3: Before outlier removal

$$y = 2.405 - 2.641 \times 10^{-5}x_5 - 2.468 \times 10^{-3}x_6. \tag{4.3}$$

Model 4: After outlier removal using $\alpha = 0.025$

$$y = 3.713 - 4.654 \times 10^{-5}x_5 - 7.561 \times 10^{-3}x_6. \tag{4.4}$$

For both models, both the retained variables were found to be significant.

4.5.2 SVM model

In this section, the impact of outliers on the fitted SVM models is discussed. The SVM model obtained after the SVM on the JSE dataset without weeding outliers contains 689 support vectors. After weeding outliers, the fitted SVM model had 594 support vectors. Weeding of outliers resulted in the model needing fewer data observations to form boundaries that separate the two classes.

4.5.3 Impact of outliers on the classification algorithms

Table 4.8 shows the accuracy of the logistic regression model and SVM model on the JSE share price prediction. When the logistic regression and SVM model are fitted to the JSE data without removing outliers, the accuracy rate is 49.10% for SVM and 49.54 for logistic regression. This is slightly better than the 48% accuracy achieved by James et al. (2013) when logistic regression was used to predict the share price direction on the Standard and Poors index (S&P500) [1].

After the removal of outliers using the robust Mahalanobis distance at $\alpha = 2.5$, no improvement in accuracy for the SVM model is observed. The accuracy of the logistic regression did improve to 50.45% after weeding outliers at $\alpha = 2.5\%$. An interesting observation is that the accuracy of the SVM model decreased to 48.65% when outliers were removed using $\alpha = 5.0\%$ while the accuracy of the logistic regression remained at 50.45%. It is only when outliers are removed at $\alpha = 7.5\%$ that an increase in accuracy (49.55%) is observed for the SVM model. When outliers are removed using $\alpha = 10.0\%$, the accuracy of the logistic regression model yet against remains 50.45% while the accuracy for the SVM model also remained the same as at 49.55%.

Table 4.8: Accuracy of classification algorithms

α	LR accuracy	SVM Accuracy
0.0 %	49.54%	49.01%
2.5 %	50.45%	49.01%
5.0 %	50.45%	48.65%
7.5 %	50.45%	49.55%
10.0 %	50.45%	49.55%

[1] S& P500 is an index containing the top 500 most valuable companies listed on the United States exchanges

Outlier removal using different values of α for the two classification models, it appears, is beneficial (in terms of improving accuracy) only up to some threshold value.

Chapter 5

Conclusions and Recommendations

This chapter gives the conclusions and recommendations of the study taking into account the results that were obtained in Chapter 4.

5.1 Conclusions

In this thesis, an iterative outlier detection procedure called CQAOD has been proposed for data that follow a mixture of multivariate Gaussian distributions. In the initial iterations of the procedure, higher numbers of outliers are detected and the numbers of outliers detected per iteration progressively decreases to zero. In most cases, this happens within a few iterations. It has been demonstrated that K-means clustering along with the use of outlier detection procedures like the classical method and the MCD method can result in clusters whose parameters closely resemble those of the original clusters when the correct number of clusters is employed. The question of how to determine the true or actual number of clusters present in a dataset of course invariably arises and this work has sought to provide a convincing solution.

More importantly, this work has demonstrated that the quality indices (such as the Davies Bouldin index, Dunn index and VAR) for assessing clustering can be harnessed as tools for determining a plausible estimate of the actual number of clusters present. The use of the quality indices put paid to the idea of the scree plot as a tool or guide for

determining the unknown value of k which is of course a big boost for the argument advanced by Schubert (2023) for researchers doing clustering to stop their reliance on the scree plot. In any case, it has been observed that for data generated where the true number of clusters is large, the screeplot method under-estimates the actual number of clusters.

One downside or pitfall of the proposed CQAOD algorithm is that in some instances, some of the clusters generated at some point in the iterations may contain too few observations (in relation to the number of features p) leading to some of the computed variance-covariance matrices being singular. This phenomenon is associated with high values of k which then hampers the determination of an ideal value of k if the plot of the Dunn index for instance will not have reached its peak when this happens. Also when this happens, the iterations prematurely stop before achieving convergence. A solution to this problem may be to use boot-strapping.

The other pitfall (which is of a lesser gravity), is that of the iterations not resulting in fast convergence. One way of circumventing this problem and ensure convergence is to simply allow a higher number of iterations.

This work has clearly demonstrated that use of indices in prescribing an ideal number of clusters for data bedevilled by outliers is untenable as there is a tendency of disharmony between the indices. However, if CQAOD is applied, there is harmony with the indices suggesting the same optimal number of clusters.

The other limitation stems from determining the ideal number of clusters for unclustered data. For unclustered data (i.e. data from $M = 1$ cluster), the indices plot does not unequivocally tell us that the ideal number is indeed 1 because the plot starts at $k = 2$. The three indices plots fail to either peak (in the case of the Dunn and the VRC) or deep). However, the plot has been seen to peak at $k = 2$ and from that we can infer that the ideal number of clusters maybe 1 or 2.

The main contribution of CQAOD is the laying out of a clear roadmap for a process that results in an efficient outlier detection in that it utilises cluster quality indices to determine a plausible number of the underlying number of clusters.
The analyses of JSE data has involved testing for multivariate normality, clustering of the data and application of classification algorithms encompassing logistic regression and SVM. K-means clustering has been separately used for $k = 2$ and $k = 3$. The data from the resulting clusters has been tested for multivariate normality. None of the clusters has exhibited multivariate normality and thus CQAOD is not appropriate for clustering JSE data. Also, a model-based clustering algorithm is not appropriate. Other clustering algorithms that have been used to cluster the JSE data are spectral clustering algorithm, CLARA clustering algorithm and hierarchical clustering algorithm. For CLARA and spectral clustering algorithms, there is discord in terms of optimal number of clusters to use. This discord apparently stems from the outlier contamination of the dataset.

After applying the hierarchical clustering algorithm to the JSE dataset, the number of optimal clusters is found to be three. Using this number, a diversified portfolio is constructed. The performance of the portfolio when compared to four other popular share indices on the JSE gives positive results. On a risk-adjusted return basis, the cluster-based portfolio outperforms two of the four indices it was compared against. On the cumulative return basis, the cluster-based portfolio outperforms three out of the four indices it was compared against.

However, the performance of both the LR model and SVM model in terms of accuracy, hovering at around 50%, is not that pleasing.

5.2 Recommendations

In this section, leads or pointers for possible related research in the future are discussed.

An adaptation of the CQAOD algorithm to do outlier detection where K-medoids or such other clustering algorithm is applied is envisaged. This research has fallen short

on investigating the issue of detection of false negative observations (observations that are outliers and have been detected to be non-outliers) and false positive observations (i.e. observations that are not outliers and have been detected to be outliers).

The proposed use of bootstrapping to overcome the problem of clusters with two few observations need to be explored further.

Starczewski (2017) have recently developed the STR index which can be used to determine changes in compactness and separability of clusters during a clustering process. Its use in clustering and outlier detection may be worthwhile to explore.

Additional features may be helpful in future analysis using classification tools for modelling the share price of JSE data. In the JSE data on the prediction of share price direction, the inclusion of additional features is recommended. This potentially may result in improved accuracy of the classification algorithms.

Bibliography

Auria, L. and Moro, R. A. (2008). Support vector machines (svm) as a technique for solvency analysis. 29

Baldomero-Naranjo, M., Martinez-Merino, L. I., and Rodriguez-Chia, A. M. (2021). A robust svm-based approach with feature selection and outliers detection for classification problems. *Expert Systems with Applications*, 178:115017. 31

Bezdek, J. C. and Pal, N. R. (1995). Cluster validation with generalized dunn's indices. In *Proceedings 1995 second New Zealand international two-stream conference on artificial neural networks and expert systems*, pages 190–193. IEEE. 21

Caliński, T. and Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27. 22

Cerioli, A. (2010). Multivariate outlier detection with high-breakdown estimators. *Journal of the American Statistical Association*, 105(489):147–156. 19

Cherednichenko, S. (2005). Outlier detection in clustering. *Master's Thesis, University of Joensuu, Department of Computer Science*. 23

Chollete, L., de la Pena, V., and Lu, C.-C. (2011). International diversification: A copula approach. *Journal of banking & finance*, 35(2):403–417. 2

Deng, D. (2020). Dbscan clustering algorithm based on density. In *2020 7th international forum on electrical engineering and automation (IFEEA)*, pages 949–953. IEEE. 17

Dinandra, R., Hertono, G., and Handari, B. (2019). Implementation of density-based spatial clustering of application with noise and genetic algorithm in portfolio optimization with constraint. In *AIP Conference Proceedings*, volume 2168, page 020026. AIP Publishing LLC. 10

Dubes, R. and Jain, A. K. (1976). Clustering techniques: the user's dilemma. *Pattern Recognition*, 8(4):247–260. 58

Febrianti, R., Widyaningsih, Y., and Soemartojo, S. (2021). The parameter estimation of logistic regression with maximum likelihood method and score function modification. In *Journal of physics: Conference series*, volume 1725, page 012014. IOP Publishing. 28

Fernández, A. and Gómez, S. (2008). Solving non-uniqueness in agglomerative hierarchical clustering using multidendrograms. *Journal of Classification*, 25(1):43–65. 10

Ghorbani, H. (2019). Mahalanobis distance and its application for detecting multivariate outliers. *Facta Universitatis, Series: Mathematics and Informatics*, pages 583–595. 17

Gomes, J. and Livdan, D. (2004). Optimal diversification: Reconciling theory and evidence. *The Journal of Finance*, 59(2):507–535. 2

Gupta, T. and Panda, S. P. (2018). A comparison of k-means clustering algorithm and clara clustering algorithm on iris dataset. *International Journal of Engineering & Technology*, 7(4):4766–4768. 12, 16, 55

Hand, D. J. (1998). Data mining: statistics and more? *The American Statistician*, 52(2):112–118. 1

Hillman, A. J., Withers, M. C., and Collins, B. J. (2009). Resource dependence theory: A review. *Journal of management*, 35(6):1404–1427. 3

Hubert, M., Rousseeuw, P. J., and Van Aelst, S. (2008). High-breakdown robust multivariate methods. 17, 19

Jain, A. K. and Law, M. H. (2005). Data clustering: A user's dilemma. In *Pattern Recognition and Machine Intelligence: First International Conference, PReMI 2005, Kolkata, India, December 20-22, 2005. Proceedings 1*, pages 1–10. Springer. 58

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer. 28, 29, 68

Jobe, J. M. and Pokojovy, M. (2015). A cluster-based outlier detection scheme for multivariate data. *Journal of the American Statistical Association*, 110(512):1543–1551. 20, 23

Johnson, R. A., Wichern, D. W., et al. (2002). Applied multivariate statistical analysis. 8, 10

Kapil, S., Chawla, M., and Ansari, M. D. (2016). On k-means data clustering algorithm with genetic algorithm. In *2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pages 202–206. IEEE. 58

Korkmaz, S., Göksülük, D., and Zararsiz, G. (2014). Mvn: An r package for assessing multivariate normality. *R JOURNAL*, 6(2). 55

Kose, M. A., Sugawara, N., and Terrones, M. E. (2020). Global recessions. 7

Lapidot, I. (2018). Convergence problems of mahalanobis distance-based k-means clustering. In *2018 IEEE International Conference on the Science of Electrical Engineering in Israel (ICSEE)*, pages 1–5. IEEE. 18

Lattanzi, S., Lavastida, T., Lu, K., and Moseley, B. (2020). A framework for parallelizing hierarchical clustering methods. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I*, pages 73–89. Springer. 9

Lee, A. J., Lin, M.-C., Kao, R.-T., and Chen, K.-T. (2010). An effective clustering approach to stock market prediction. 24, 25

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA. 11, 12

Marr, B. (2018). How much data do we create every day? the mind-blowing stats everyone should read. *Forbes*, 21:1–5. 1

Marvin, K. (2015). Creating diversified portfolios using cluster analysis. *Princeton University*. 3, 24, 37

Milligan, G. W. and Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50:159–179. 15, 16

Mimmack, G. M., Mason, S. J., and Galpin, J. S. (2001). Choice of distance matrices in cluster analysis: Defining regions. *Journal of climate*, 14(12):2790–2797. 4

Mohammadi, M. and Sarmad, M. (2019). Outlier detection for support vector machine using minimum covariance determinant estimator. *Journal of AI and Data Mining*, 7(2):299–309. 31, 38

Mohbey, K. K. and Thakur, G. (2013). An experimental survey on single linkage clustering. *International Journal of Computer Applications*, 76(17):6–11. 9

Nelson, J. D. (2012). *On K-Means clustering using Mahalanobis distance*. PhD thesis, North Dakota State University. 2, 4

Ng, A., Jordan, M., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14. 17

Nowak-Brzezińska, A. and Gaibei, I. (2022). How the outliers influence the quality of clustering? *Entropy*, 24(7):917. 2, 4, 16

Pfeffer, J. and Salancik, G. R. (2003). *The external control of organizations: A resource dependence perspective*. Stanford University Press. 3

Ranganathan, P., Pramesh, C., and Aggarwal, R. (2017). Common pitfalls in statistical analysis: logistic regression. *Perspectives in clinical research*, 8(3):148. 27

Rodriguez, M. Z., Comin, C. H., Casanova, D., Bruno, O. M., Amancio, D. R., Costa, L. d. F., and Rodrigues, F. A. (2019). Clustering algorithms: A comparative approach. *PloS one*, 14(1):e0210236. 12, 14, 16, 21

Rousseeuw, P. J. (1984). Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880. 19

Rousseeuw, P. J. and Driessen, K. V. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223. 20

Roy, S. (2019). Clustering mid-cap stocks in indian market using multi-variate data analysis technique. *Indian Journal of Economics and Development*, 7(6):1–10. 24, 25

Saha, J. and Mukherjee, J. (2021). Cnak: Cluster number assisted k-means. *Pattern Recognition*, 110:107625. 12, 14, 24, 55

Schubert, E. (2023). Stop using the elbow criterion for k-means and how to choose the number of clusters instead. *ACM SIGKDD Explorations Newsletter*, 25(1):36–42. 11, 12, 15, 71

Sharma, S., Batra, N., et al. (2019). Comparative study of single linkage, complete linkage, and ward method of agglomerative clustering. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pages 568–573. IEEE. 10

Shehadeh, A. A., Alwadi, S. M., and Almaharmeh, M. I. (2022). Detecting and analysing possible outliers in global stock market returns. *Cogent Economics & Finance*, 10(1):2066762. 25

Song, Q., Hu, W., and Xie, W. (2002). Robust support vector machine with bullet hole image classification. *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)*, 32(4):440–448. 31

Starczewski, A. (2017). A new validity index for crisp clusters. *Pattern Analysis and Applications*, 20:687–700. 73

Sugar, C. A. and James, G. M. (2003). Finding the number of clusters in a dataset: An information-theoretic approach. *Journal of the American Statistical Association*, 98(463):750–763. 14, 15, 55

Tallón-Ballesteros, A. J. and Riquelme, J. C. (2014). Deleting or keeping outliers for classifier training? In *2014 sixth world congress on Nature and Biologically Inspired Computing (NaBIC 2014)*, pages 281–286. IEEE. 5

Tarr, G., Müller, S., and Weber, N. C. (2016). Robust estimation of precision matrices under cellwise contamination. *Computational Statistics & Data Analysis*, 93:404–420. 19

Thorndike, R. L. (1953). Who belongs in the family? *Psychometrika*, 18(4):267–276. 15

Tong, H. and Tortora, C. (2022). Model-based clustering and outlier detection with missing data. *Advances in Data Analysis and Classification*, 16(1):5–30. 11

Tripathy, N. (2019). Stock price prediction using support vector machine approach. In *International Academic Conference on Management & Economics*, pages 44–59. 29

Tsyurmasto, P., Zabarankin, M., and Uryasev, S. (2014). Value-at-risk support vector machine: stability to outliers. *Journal of Combinatorial Optimization*, 28(1):218–232. 29, 31

Virvou, M., Alepis, E., and Troussas, C. (2012). Centroid-based clustering for student models in computer-based multiple language tutoring. In *SIGMAP*, pages 198–203. 11

Wang, Z. (2020). Unified robust estimation. *Australian & New Zealand Journal of Statistics*. 31

Xu, D. and Tian, Y. (2015). A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2:165–193. 10

Yang, X., Song, Q., and Wang, Y. (2007). A weighted support vector machine for data classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(05):961–976. 31

Zhang, X. (1999). Using class-center vectors to build support vector machines. In *Neural networks for signal processing IX: proceedings of the 1999 IEEE signal processing society workshop (cat. no. 98TH8468)*, pages 3–11. IEEE. 31

Appendix A

Line plots of indices for different outlier contamination levels

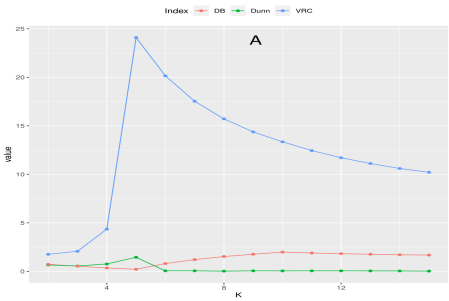

Figure A.1: M=5, f=0)

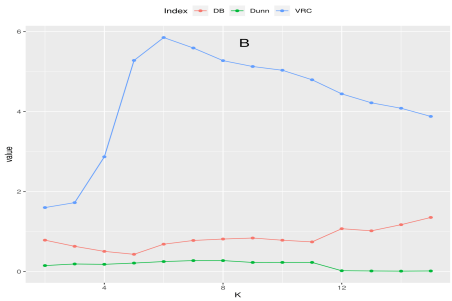

Figure A.2: M=5, f=40)

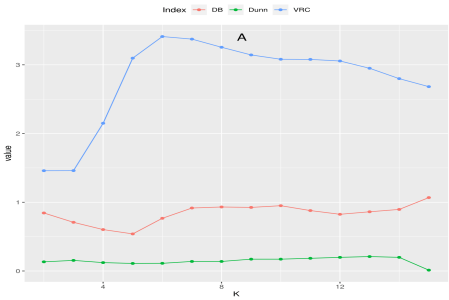

Figure A.3: M=5, f=80)

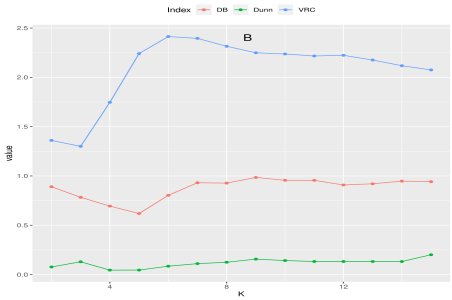

Figure A.4: M=5, f=120

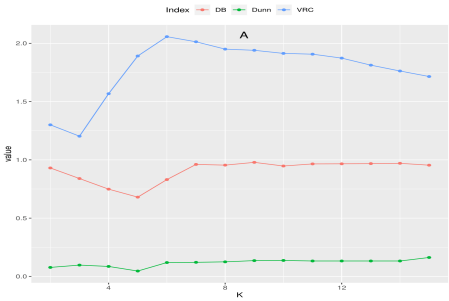

Figure A.5: M=5, f=160)

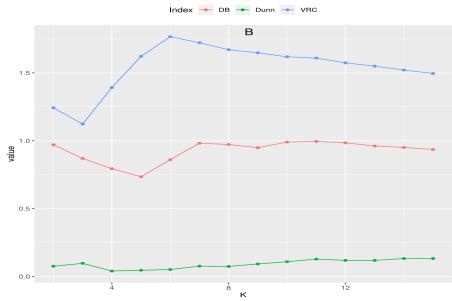

Figure A.6: (M=5, f=200)

For M=10

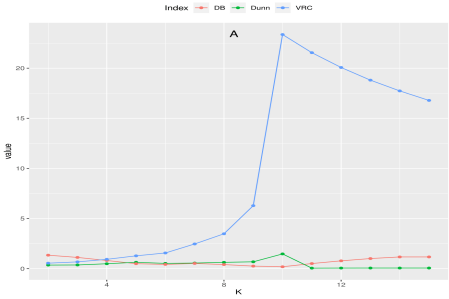

Figure A.7: M=10, f=0

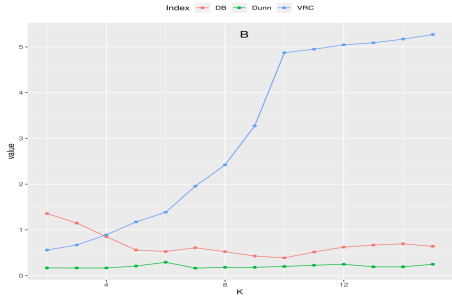

Figure A.8: M=10, f=40

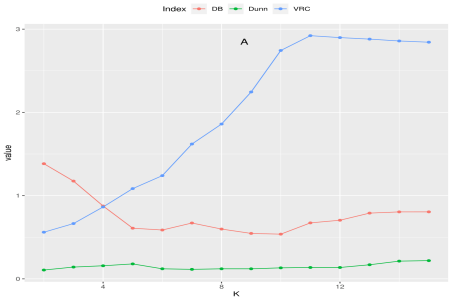

Figure A.9: M=10, f=80

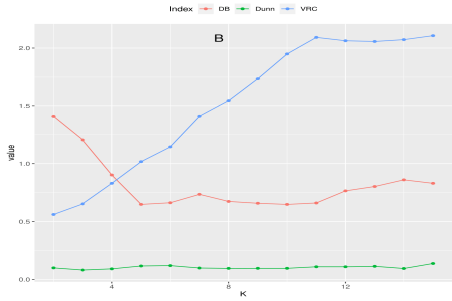

Figure A.10: M=10, f=120

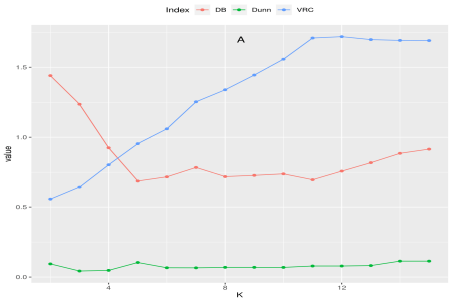

Figure A.11: M=10, f=160

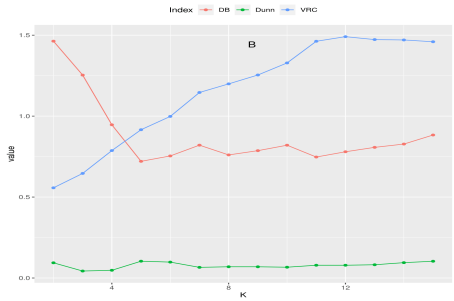

Figure A.12: M=10, f=200

Appendix B

Boxplots for Iris dataset

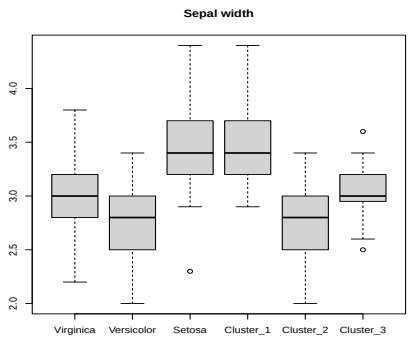

Figure B.1: Sepal width

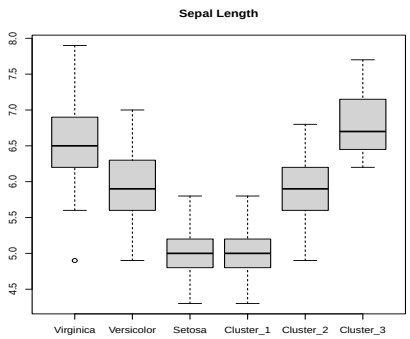

Figure B.2: Sepal length

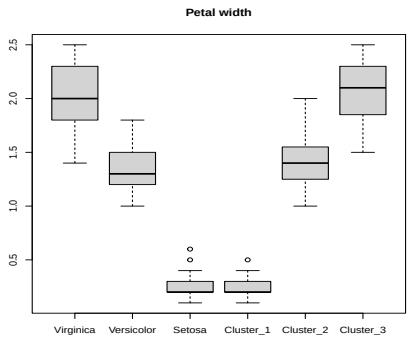

Figure B.3: Petal width

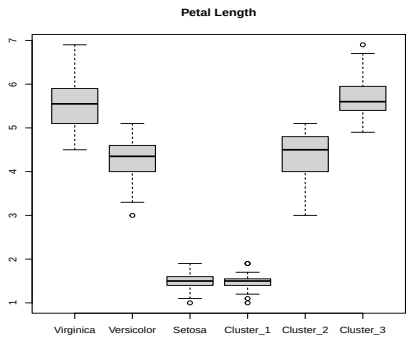

Figure B.4: Petal length

Appendix C

Screenshot of JSE features data

JSE data can be sourced here:

https://www.investing.com/stock-screener/?sp=country::110|sector::a|industry::a|equityType::a%3Ceq_market_cap;1

	Company	PE_Ratio	to-Equity ratio	ROA
1	4Sight	3.20000000	0.603339107	0.333333333
2	Absa Group Limited	2.09691011	1.473195529	-0.192055847
3	Accelerate	-0.20114943	0.910685332	-0.554707379
4	Acsion Limited	0.52845528	0.117915716	-0.409090909
5	Adcock Ingram Holdings Limited	1.35125628	0.068407388	-0.152937470
6	ADvTECH Limited	1.27058824	0.731028734	-0.275167785
7	AECI Ltd	8.42519685	0.567427873	0.202247191
8	African Equity Empowerment Investments Limited	1.32649000	0.040166643	-0.750000000
9	African Rainbow	1.36383333	0.000000000	0.104840343
10	Afrimat Limited	0.97046784	0.175916203	0.198194946

Appendix D

Line plots of indices for spectral and CLARA clustering

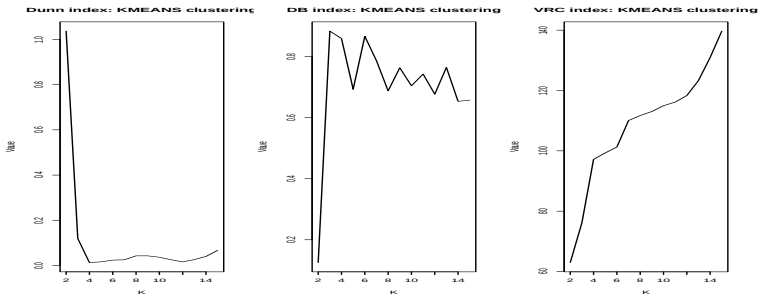

Figure D.1: K-means clustering

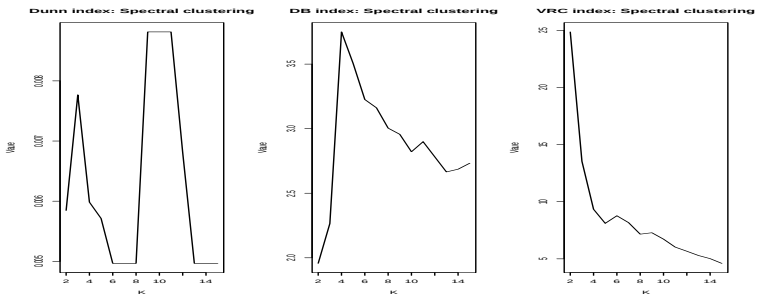

Figure D.2: Spectral clustering

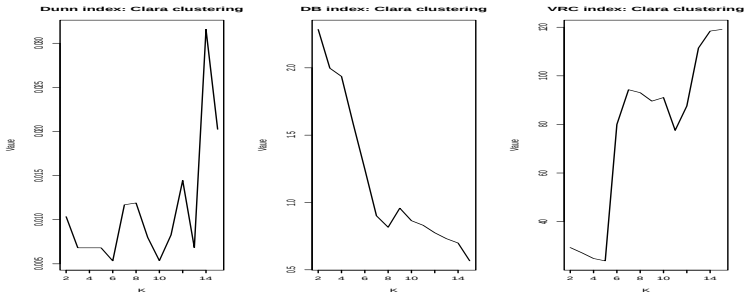

Figure D.3: CLARA clustering

Appendix E

R Code

E.1 R Code

The following functions are written as part of the research project.

```
 library(cluster)
library(factoextra)
library(clValid) #For Dunn index
library(clusterSim) #For Davis-Bouldin Index
library(Hmisc) #Correlations test
library(outliers)
library(ggplot2)
library(MixSim)
library(cluster)
library(MASS)
library(fpc) #VRC index
library(outliers)
library(xtable)
library(MVN)
```

```
library(readxl)
library(dplyr)
library(fpc)
library(moments)
library(tidyr)

#Simulating one cluster data

set.seed(1123)
sample_size <- 1000
sample_meanvector <- c(100,50)
sample_covariance_matrix <- matrix(c(10, 5, 5, 10), 2,2)

# create multivariate normal distribution
sample <- mvrnorm(n = sample_size,
                  mu = sample_meanvector,
                  Sigma = sample_covariance_matrix)



# Plot the first plot
plot(sample, ylab = "X2", xlab = "X1",main="Distribution of dataset")

# Plot the second plot (boxplot for X1)
boxplot(dataset[, 1], main = "Boxplot for X1")

# Plot the third plot (boxplot for X2)
boxplot(dataset[, 2], main = "Boxplot for X2")
```

```
(d1=sort(dataset[, 1]))
(d2=sort(dataset[, 2]))


#Simulating data with M clusters

set.seed(1123)
M=5 # Number of clusters to simulate
out = MixSim(MaxOmega  = 0,K = M,p = 4,hom = TRUE,sph = TRUE,ecc = 1,int = c(0,5
n=1000;


SData=simdataset(n = n,Pi = out$Pi,Mu = out$Mu,S = out$S,n.out =200,max.out = 10
SimulatedData=SData$X



#Cluster simulated data with removal of outliers

# Create an empty dataframe to store results for each k
df_dunn <- data.frame(K = numeric(), I = numeric(), Dunn = numeric())
df_DB <- data.frame(K = numeric(), I = numeric(), DB = numeric())
df_VCR <- data.frame(K = numeric(), I = numeric(), VCR = numeric())
```

```
for (k in 2:20) {

  dataset=SimulatedData[1:1040,]  #Dataset to cluster



  # Perform clustering
  cluster_result <- kmeans(dataset, centers = k, nstart = 1000)

  # Calculate cluster quality indices
  dunn_index <- calculate_dunn_index(dataset, cluster_result$cluster)
  new_row <- data.frame(K = k,  value = dunn_index*10,Index="Dunn")
  df_dunn <- rbind(df_dunn, new_row)

  db_index <- calculate_davies_bouldin_index(dataset, cluster_result$cluster)
  new_row2 <- data.frame(K = k, value = db_index,Index="DB")
  df_DB <- rbind(df_DB, new_row2)

  vcr_index<-calinhara(dataset, cluster_result$cluster, cn=max(cluster_result$cl
  new_row3 <- data.frame(K = k, value = (vcr_index/1000) ,Index="VRC")
  df_VCR <- rbind(df_VCR, new_row3)

}



join=rbind(df_dunn,df_DB ,df_VCR)

ggplot(join, aes(x = K, y = value, colour = Index)) + geom_line() + geom_point()
```

```
#QCAOD code

# Function to calculate Mahalanobis distance and remove outliers using classical
remove_outliers <- function(dataset) {
  mahal_dist <- mahalanobis(dataset, colMeans(dataset), cov(dataset),tol=1e-20)
  New_Data=cbind(dataset,mahal_dist)
  datas <- subset(New_Data[,-ncol(New_Data)],mahal_dist < threshold)
  return(datas)
}

# Function to calculate MCD Mahalanobis distance and remove outliers using class
remove_outliers_mcd <- function(datasetMCD) {
  CovMCD1=covMcd(datasetMCD,nsamp = "deterministic",alpha = alpha)
  mahal_distMcd <- mahalanobis(x = datasetMCD,center = CovMCD1$center,cov = CovM
  # Add Mahalanobis distances as a new column in the data
  New_DataMCd=cbind(datasetMCD,mahal_distMcd)
  dataMCD <- subset(New_DataMCd[,-ncol(New_DataMCd)],mahal_distMcd < threshold,t
  return(dataMCD)
}

# Function to calculate Dunn index
calculate_dunn_index <- function(data, clusters) {
```

```
  dunn(Data = data, clusters = clusters, method = "euclidean")
}

# Function to calculate Davies-Bouldin index
calculate_davies_bouldin_index <- function(data, clusters) {
  index.DB(data, cl = clusters, centrotypes = "centroids")$DB
}




threshold=qchisq(p = 0.975, df = ncol(dataset))
alpha=0.975



#CQAOD Classical
all=data.frame()
all2=data.frame()
all3=data.frame()

# Initialize an empty list to store datasets for each value of 'k'
Classic5_0 <- list()
k=2
for (k in 2:15) {

  dataset=Stocks #assign the dataset to apply CQAOD as 'dataset

  # Create an empty list for the current value of 'k'
```

```
Classic5_0[[as.character(k)]] <- list(dataset)

# Create an empty dataframe to store results for each k
df_dunn <- data.frame(K = numeric(), I = numeric(), Dunn = numeric())
df_DB <- data.frame(K = numeric(), I = numeric(), DB = numeric())
df_VCR <- data.frame(K = numeric(), I = numeric(), VCR = numeric())

for (i in 2:30) {

  # Perform clustering
  cluster_result <- kmeans(dataset, centers = k, nstart = 1000)

  # Calculate cluster quality indices
  dunn_index <- calculate_dunn_index(dataset, cluster_result$cluster)
  new_row <- data.frame(K = k, I = i, value = dunn_index,Index="Dunn")
  df_dunn <- rbind(df_dunn, new_row)

  db_index <- calculate_davies_bouldin_index(dataset, cluster_result$cluster)
  new_row2 <- data.frame(K = k, I = i, value = db_index,Index="DB")
  df_DB <- rbind(df_DB, new_row2)

  vcr_index<-calinhara(dataset, cluster_result$cluster, cn=max(cluster_result$
  new_row3 <- data.frame(K = k, I = i, value = (vcr_index/1000) ,Index="VRC")
  df_VCR <- rbind(df_VCR, new_row3)


  # Separate data by cluster
```

```
    cluster_data <- lapply(1:k, function(i) dataset[cluster_result$cluster == i,

    # Remove outliers in each cluster
    clean_data <- lapply(cluster_data, remove_outliers)

    # Combine cleaned dataframes
    dataset <- do.call(rbind, clean_data)

    Classic5_0[[as.character(k)]] <- append(Classic5_0[[as.character(k)]],list(d


  }



  # Store results for each k
  all <- rbind(all, df_DB)
  all2 <- rbind(all2, df_dunn)
  all3 <- rbind(all3, df_VCR)

}



dt=subset(all,I==20)
dt2=subset(all2,I==20)
dt3=subset(all3,I==20)
join=dt3[,c(1,3,4)]
join=rbind(dt[,c(1,3,4)],dt2[,c(1,3,4)],dt3[,c(1,3,4)])
```

```
ggplot(join, aes(x = K, y = value, colour = Index)) + geom_line() + geom_point()+

fviz_nbclust(x = dataset, FUNcluster = kmeans, method = "wss",k.max = 10)+ylab("W



#MCD QCAOD
allMCD=data.frame()
allMCD2=data.frame()
allMCD3=data.frame()

###For MCD MAHALANOBIS###

# Initialize an empty list to store datasets for each value of 'k'
dataset_lists5MCD40 <- list()

for (k in 2:15) {

  dataset=data #assign the dataset to apply CQAOD as 'dataset

  # Create an empty list for the current value of 'k'
  dataset_lists5MCD40[[as.character(k)]] <- list(dataset)

  # Create an empty dataframe to store results for each k
  df_dunnMCD <- data.frame(K = numeric(), I = numeric(), Dunn = numeric())
```

```
df_DBMCD <- data.frame(K = numeric(), I = numeric(), DB = numeric())
df_VCRMCD <- data.frame(K = numeric(), I = numeric(), DB = numeric())

for (i in 1:30) {
  # Perform clustering
  cluster_resultMCD <- kmeans(dataset, centers = k, nstart = 1000)

  # Calculate cluster quality indices
  dunn_indexMCD <- calculate_dunn_index(dataset, cluster_resultMCD$cluster)
  new_rowMCD <- data.frame(K = k, I = i, value = dunn_indexMCD,Index="Dunn")
  df_dunnMCD <- rbind(df_dunnMCD, new_rowMCD)

  db_indexMCD <- calculate_davies_bouldin_index(dataset, cluster_resultMCD$clu
  new_rowMCD2 <- data.frame(K = k, I = i, value = db_indexMCD,Index="DB")
  df_DBMCD <- rbind(df_DBMCD, new_rowMCD2)

  vcr_indexMCD<-calinhara(dataset, cluster_resultMCD$cluster, cn=max(cluster_r
  new_rowMCD3 <- data.frame(K = k, I = i, value = (vcr_indexMCD/1000) ,Index="
  df_VCRMCD <- rbind(df_VCRMCD, new_rowMCD3)


  # Separate data by cluster
  cluster_dataMCD <- lapply(1:k, function(i) dataset[cluster_resultMCD$cluster

  # Remove outliers in each cluster
  clean_dataMCD <- lapply(cluster_dataMCD, remove_outliers_mcd)
```

```
    # Combine cleaned dataframes
    dataset <- do.call(rbind, clean_dataMCD)

    dataset_lists5MCD40[[as.character(k)]] <- append(dataset_lists5MCD40[[as.cha
  }

  # Store results for each k
  allMCD <- rbind(allMCD, df_DBMCD)
  allMCD2 <- rbind(allMCD2, df_dunnMCD)
  allMCD3 <- rbind(allMCD3, df_VCRMCD)
}


dtMCD=subset(allMCD,I==20)
dtMCD2=subset(allMCD2,I==20)
dtMCD3=subset(allMCD3,I==20)


joinMCD=rbind(dtMCD[,c(1,3,4)],dtMCD2[,c(1,3,4)],dtMCD3[,c(1,3,4)])

ggplot(joinMCD, aes(x = K, y = value, colour = Index)) + geom_line() + geom_poin

fviz_nbclust(x = dataset, FUNcluster = kmeans, method = "wss",k.max = 10)+ylab("


#Iris dataset
```

```
#Assign iris dataset to dataset as follows
dataset=iris[-5] # remove last column

#apply dataset to the classical QCAOD  and MCD QCAOD

fviz_nbclust(x = dataset, FUNcluster = kmeans, method = "wss",k.max = 10)+ylab("

#JSE data

#Preliminary analysis code
#Importing JSE dataset

Data2=read_excel("C:/Users/retse/OneDrive - University of Witwatersrand/Masters/
New_Data=na.omit(Data2[,c(2,3,4)]) #Omit entries with missing data and

#Descriptive statistics of graphs
boxplot(Data2$PE_Ratio,breaks = 30, main = "(a) P/E Ratio",xlab = "")
boxplot(Data2$ROA, breaks = 30, main = "(b) R/A",xlab = "")
boxplot(Data2$`Debt-to-Equity ratio`, breaks = 30, main = "(c) D/E",xlab = "")
```

```
#Histograms

hist(Data2$PE_Ratio,breaks = 30, main = "(a) P/E Ratio",xlab = "")
hist(Data2$ROA, breaks = 30, main = "(b) R/A",xlab = "")
hist(Data2$`Debt-to-Equity ratio`, breaks = 30, main = "(c) D/E",xlab = "")


#Calculate skewness
skewness(Data2$PE_Ratio)
skewness(Data2$`Debt-to-Equity ratio`)
skewness(Data2$ROA)




#Clustering code

data=New_Data

# Compute the similarity matrix
similarity_matrix <- exp(-dist(data)^2)

# Perform spectral decomposition
eigen_result <- eigen(similarity_matrix)


# Create empty data frames to store results for each k and each index
```

```
df_dunn <- data.frame(K = numeric(), Value = numeric(), Index = character())
df_DB <- data.frame(K = numeric(), Value = numeric(), Index = character())
df_VCR <- data.frame(K = numeric(), Value = numeric(), Index = character())
k=2
# Loop through different values of k
for (k in 2:20) {

  # Extract the top-k eigenvectors
  k_eigenvectors <- eigen_result$vectors[, 1:k]

  # Perform clustering
  cluster_result <- kmeans(k_eigenvectors, centers = k, nstart = 1000)

  # Calculate cluster quality indices
  dunn_index <- calculate_dunn_index(data, cluster_result$cluster)
  db_index <- calculate_davies_bouldin_index(data, cluster_result$cluster)
  vcr_index <- calinhara(data, cluster_result$cluster, cn = max(cluster_result$c

  # Append results to respective data frames
  df_dunn <- rbind(df_dunn, data.frame(K = k, Value = dunn_index, Index = "Dunn"
  df_DB <- rbind(df_DB, data.frame(K = k, Value = db_index, Index = "DB",Algorit
  df_VCR <- rbind(df_VCR, data.frame(K = k, Value = vcr_index , Index = "VCR",Al

  #Kmeans
  # Perform clustering
```

```
cluster_result <- kmeans(data, centers = k, nstart = 10000)

# # Calculate cluster quality indices
dunn_index <- calculate_dunn_index(data, cluster_result$cluster)
db_index <- calculate_davies_bouldin_index(data, cluster_result$cluster)
vcr_index <- calinhara(data, cluster_result$cluster, cn = max(cluster_result$c
#
# # Append results to respective data frames
df_dunn <- rbind(df_dunn, data.frame(K = k, Value = dunn_index, Index = "Dunn"
df_DB <- rbind(df_DB, data.frame(K = k, Value = db_index, Index = "DB",Algorit
df_VCR <- rbind(df_VCR, data.frame(K = k, Value = vcr_index , Index = "VCR",Al

#
# #CLARA
# # Perform clustering
cluster_result <- clara(data, k, metric = "euclidean", stand = FALSE,samples =
#
# # Calculate cluster quality indices
dunn_index <- calculate_dunn_index(data, cluster_result$cluster)
db_index <- calculate_davies_bouldin_index(data, cluster_result$cluster)
vcr_index <- calinhara(data, cluster_result$cluster, cn = max(cluster_result$c
#
# # Append results to respective data frames
df_dunn <- rbind(df_dunn, data.frame(K = k, Value = dunn_index, Index = "Dunn"
df_DB <- rbind(df_DB, data.frame(K = k, Value = db_index, Index = "DB",Algorit
df_VCR <- rbind(df_VCR, data.frame(K = k, Value = vcr_index , Index = "VCR",Al
```

```
}

all=rbind(df_dunn,df_DB,df_VCR)
spec=subset(all,Algorith == "Spectral")

#Dunn index
#Spectral clustering
dunnSpec=subset(df_dunn, Algorith == "Spectral")
DBSpec=subset(df_DB, Algorith == "Spectral")
VRCSpec=subset(df_VCR, Algorith == "Spectral")

plot(dunnSpec$K,dunnSpec$Value,type='l',main="Dunn index: Spectral clustering",x

plot(DBSpec$K,DBSpec$Value,type='l',main="DB index: Spectral clustering",xlab="K

plot(VRCSpec$K,VRCSpec$Value,type='l',main="VRC index: Spectral clustering",xlab=

# KMEANS
#KMEANS clustering
dunnKMEANS=subset(df_dunn, Algorith == "KMeans")
DBKMEANS=subset(df_DB, Algorith == "KMeans")
VRCKMEANS=subset(df_VCR, Algorith == "KMeans")
```

```
plot(dunnKMEANS$K,dunnKMEANS$Value,type='l',main="Dunn index: KMEANS clustering"
plot(DBKMEANS$K,DBKMEANS$Value,type='l',main="DB index: KMEANS clustering",xlab=
plot(VRCKMEANS$K,VRCKMEANS$Value,type='l',main="VRC index: KMEANS clustering",xl


join=rbind(dunnKMEANS,DBKMEANS)


ggplot(join, aes(x = K, y = Value, colour = Index)) + geom_line() + geom_point()

fviz_nbclust(x =data, FUNcluster = kmeans, method = "wss",k.max = 10)+labs(title=




#Clara clustering
VRCClara=subset(df_VCR, Algorith == "Clara")
DBClara=subset(df_DB, Algorith == "Clara")
DunnClara=subset(df_dunn, Algorith == "Clara")

plot(DunnClara$K,DunnClara$Value,type='l',main="Dunn index: Clara clustering",xl

plot(DBClara$K,DBClara$Value,type='l',main="DB index: Clara clustering",xlab="K"

plot(VRCClara$K,VRCClara$Value,type='l',main="VRC index: Clara clustering",xlab=
```

```
# Perform DBSCAN clustering
cluster_result <- dbscan(data, eps = 0.45, MinPts = 5)
cluster_result$cluster
```

```
#Hierachical clustering

clusters <- hclust(dist(Stocks))

#Dendrogram
plot(clusters,main="Dendrogram:JSE dataset",xlab="")
abline(h = 14, col = 'blue')
abline(h = 5, col = 'red')
abline(h = 6, col = 'green')
abline(h = 9, col = 'purple')
```

```
(clusterCut <- cutree(clusters,6))
table(clusterCut)
```

```
#DBSCAN clustering
dunnDBSCAN=subset(df_dunn, Algorith == "DBSCAN")
DBDBSCAN=subset(df_DB, Algorith == "DBSCAN")
VRCDBSCAN=subset(df_VCR, Algorith == "DBSCAN")

plot(dunnDBSCAN$K,dunnDBSCAN$Value,type='l',main="Dunn index: DBSCAN clustering"

plot(DBDBSCAN$K,DBDBSCAN$Value,type='l',main="DB index: DBSCAN clustering",xlab=

plot(VRCDBSCAN$K,VRCDBSCAN$Value,type='l',main="VRC index: DBSCAN clustering",xl

#COnstructing a portfolio

#Get stocks to be used in the index
Stocks2=cbind(Data2[,c(1,6)],clusterCut)

# Subsetting Stocks2 based on cluster values 1, 2, and 3
Stocks3 <- subset(Stocks2, clusterCut %in% c(1, 2, 3))

#Select top 27% stocks with highest sharpe ratio
Top40 <- Stocks3 %>%
  arrange(desc(Sharpe_Ratio)) %>%
  group_by(clusterCut) %>%
  top_n(n = round(n() * 0.27), wt = Sharpe_Ratio)

#Stocks to get share price of
Companies=Top40[,1]
```

```
Prices=read_excel("C:/Users/retse/OneDrive/Masters/Official/StockData.xlsx",shee

Prices$Date=as.Date(Prices$Date,origin="1899-12-30")
Prices <- Prices[Prices$Date > '2020-12-31', ]


SelctedStocks=Prices[Top40$Company]
ClusterIndex=round(rowSums(SelctedStocks[,2:40]/130),digits = 2) #For price grap

data$sum_of_rows <- rowSums(data)

AllIndices=read_excel("C:/Users/retse/OneDrive/Masters/Official/StockData.xlsx",
AllIndices$Month=as.Date(AllIndices$Month,origin="1899-12-30")
AllIndices=cbind(AllIndices,ClusterIndex)



# Create a new DataFrame for daily returns
return_df <- AllIndices %>%
  arrange(Month) %>%  # Ensure the DataFrame is sorted by date
  select(Month, everything()) %>%  # Reorder columns with Date as the first colu
  mutate(across(-Month, ~ c(NA, diff(.))/lag(.)*100))


#Plot the volatility
```

```
volatility=return_df %>%
  gather(key = "Stock", value = "Return", -Month) %>%
  ggplot(aes(x = Stock, y = Return)) +
  geom_boxplot() +
  labs(title = "Distribution of monthly returns by portfolio",
       x = "Stock",
       y = "Return (%)") +
  theme_minimal()
ggsave("returns variation.pdf", plot = volatility)



# Calculate standard deviation of returns by group after year 2
standard_deviation_df1 <- return_df %>%
  gather(key = "Stock", value = "Return", -Month) %>%
  group_by(Stock) %>%
  summarise(Standard_Deviation1 = sd(Return, na.rm = TRUE))

# Print or view the resulting DataFrame
print(standard_deviation_df1)





# Calculate standard deviation of returns by group after year 2
standard_deviation_df <- return_df %>%
  gather(key = "Stock", value = "Return", -Month) %>%
```

```
  group_by(Stock) %>%
  summarise(Standard_Deviation = sd(Return, na.rm = TRUE))

# Print or view the resulting DataFrame
print(standard_deviation_df)




# Calculate cumulative returns for all stocks
cumulative_returns_df <- AllIndices %>%
  gather(Stock, StockPrice, -Month) %>%
  group_by(Stock) %>%
  arrange(Month) %>%
  mutate(CumulativeReturn = ((StockPrice / first(StockPrice)) - 1) * 100) %>%
  select(Month, Stock, CumulativeReturn)%>%
  pivot_wider(names_from = Stock, values_from = CumulativeReturn)




# Assuming you have a DataFrame return_df with columns "Date" and "ReturnPercent

# Create a line chart for the distribution of returns by group
cum_returns=cumulative_returns_df %>%
  gather(key = "Stock", value = "Return", -Month) %>%
  ggplot(aes(x = Month, y = Return, color = Stock)) +
  geom_line() +
  labs(title = "Cumulative Returns by Group",
```

```
       x = "Date",
       y = "Return (%)") +
  theme_minimal()

ggsave("cum_returns.pdf", plot = cum_returns)
```

```
#Classification code

#SVM
library(e1071)
Shares=read_excel("C:/Users/retse/OneDrive/Masters/Official/StocksPrices.xlsx",sl
Shares=na.omit(Shares) #Omit entries with missing data and

# Splitting Data
Smarket_train <- subset(Shares, Year < 2023)
Smarket_test <- subset(Shares, Year > 2022)

# Fitting Logistic Regression Model
glm.fit <- glm(factor(Direction) ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
               data = Smarket_train,
               family = binomial)
```

```
summary(glm.fit) # Model Summary
glm.probs <- predict(glm.fit, newdata = Smarket_test, type = "response") # Predi
glm.pred <- rep("Down", length(glm.probs)) # Thresholding Probabilities
glm.pred[glm.probs > 0.5] <- "Up"
table(glm.pred, Smarket_test$Direction) # Confusion Matrix
(accuracy <- mean(glm.pred == Smarket_test$Direction)) # Accuracy Calculation


# Fit the SVM model
svm_model <- svm(factor(Direction) ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, 


# Predict classes on the test data
svm_pred <- predict(svm_model, newdata = Smarket_test)

# Calculate accuracy and error rate
(accuracy_svm <- sum(svm_pred == Smarket_test$Direction) / nrow(Smarket_test))
(error_rate_svm <- 1 - accuracy_svm)


#Multivariate Normality in R
result <- mvn(data = Shares[,c(2:8)], mvnTest = "hz")
result$multivariateNormality

result <- mvn(data = Shares[,c(2:8)], mvnTest = "dh")
result$multivariateNormality
```

```
result <- mvn(data = Shares[,c(2:8)], mvnTest = "royston")
result$multivariateNormality



threshold=qchisq(p = 0.975, df = ncol(Clean_Up))
alpha=0.975
library(dplyr)

#Remove outliers in each class
Clean_Up = subset(Smarket_train,Direction =="Up")
Clean_Up2=remove_outliers_mcd(Clean_Up[,c(2:8)])
Clean_Up3=left_join(Clean_Up2, Clean_Up, by='Today')
Clean_Up=Clean_Up3[,-c(8:14)]

Clean_Down = subset(Smarket_train,Direction =="Down")
Clean_Down2=remove_outliers_mcd(Clean_Down[,c(2:8)])
Clean_Down3=left_join(Clean_Down2, Clean_Down, by='Today')
Clean_Down=Clean_Down3[,-c(8:14)]

Smarket_train_clean=rbind(Clean_Up,Clean_Down)
colnames(Smarket_train_clean)=c("Lag5","Lag4" ,"Lag3","Lag2","Lag1","Today","Vol

# Fit the SVM model
svm_model2 <- svm(factor(Direction) ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,

# Predict classes on the test data
svm_pred2 <- predict(svm_model2, newdata = Smarket_test)
```

```
# Calculate accuracy and error rate
(accuracy_svm2 <- sum(svm_pred2 == Smarket_test$Direction) / nrow(Smarket_test))
#(error_rate_svm2 <- 1 - accuracy_svm2)
svm_model2


#Logistic regression

glm.fit2 <- glm(factor(Direction) ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
                data = Smarket_train_clean,
                family = binomial)
summary(glm.fit2) # Model Summary
glm.probs2 <- predict(glm.fit2, newdata = Smarket_test, type = "response") # Pre
glm.pred <- rep("Down", length(glm.probs2)) # Thresholding Probabilities
glm.pred[glm.probs2 > 0.5] <- "Up"
table(glm.pred, Smarket_test$Direction) # Confusion Matrix
(accuracy <- mean(glm.pred == Smarket_test$Direction)) # Accuracy Calculation
```