B.Sc. in Computer Science and Engineering Thesis

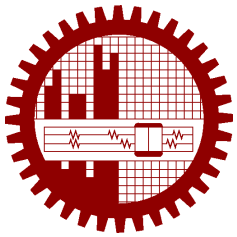
Selection of K in K-Means Algorithm

Submitted by

MD. Nayeem Reza
201205106

Supervised by

Dr. Md. Monirul Islam



Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

Dhaka, Bangladesh

September 2017

CANDIDATES' DECLARATION

This is to certify that the work presented in this thesis, titled, "Selection of K in K-Means Algorithm", is the outcome of the investigation and research carried out by us under the supervision of Dr. Md. Monirul Islam.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

MD. Nayeem Reza
201205106

CERTIFICATION

This thesis titled, **"Selection of K in K-Means Algorithm"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in September 2017.

Group Members:

MD. Nayeem Reza

Supervisor:

Dr. Md. Monirul Islam
Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

ACKNOWLEDGEMENT

First of all I would like to thank my supervisor, Dr. Md. Monirul Islam for introducing me to the amazingly interesting world of Machine Learning and Data Analysis. And he is the very first person who taught me how to perform research work efficiently. Without him and his continuous supervision, guidance and valuable advice, it would have been impossible for me to come at this point and have some output from the thesis. I am specially grateful to him for allowing me greater freedom in choosing the problems to work on, for his encouragement at times of disappointment, and for his patience with my wildly sporadic work habits.

I would also want to thank my fellow friends Md. Thohidul Islam for his encouragement, insightful comments, and hard questions. They gave their best support to complete my thesis timely. I would like to express my gratitude to all our teachers. Their motivation and encouragement in addition to the education they provided meant a lot to me.

Last but not the least, I would like to thank my family, my parents, for giving birth to me at the first place and supporting me spiritually throughout my life.

Contents

CANDIDATES' DECLARATION i

CERTIFICATION ii

ACKNOWLEDGEMENT iii

List of Figures vi

List of Tables vii

List of Algorithms viii

ABSTRACT ix

1 Introduction 1

1.1 Clustering . . . 1

1.1.1 Applications . . . 2

1.1.2 Requirements . . . 2

1.1.3 Problems . . . 3

1.1.4 Classification . . . 3

1.2 Literature Survey . . . 6

1.2.1 Values of K specified within a range or set . . . 6

1.2.2 Values of K specified by the user . . . 7

1.2.3 Values of K determined in a later processing step . . . 7

1.2.4 Values of K equated to the number of generators . . . 7

1.2.5 Values of K determined by statistical measures . . . 7

1.2.6 Values of K determined through visualization . . . 8

1.3 Objective . . . 8

1.4 Thesis Organization . . . 9

2 Background 10

2.1 Introduction . . . 10

2.1.1 Machine Learning . . . 10

2.2 A Deep Dive into Cluster Analysis . . . 12

2.2.1 Centroid Based Cluster . . . 12
2.2.2 K-means . . . 13
2.2.3 The Basic K-means Algorithm . . . 13
2.2.4 Assigning Points to the Closest Centroid . . . 13
2.2.5 Centroids and Objective Functions . . . 14
2.2.6 Data in Euclidean Space . . . 14
2.2.7 Choosing Initial Centroids . . . 15
2.2.8 Updating Centroids Incrementally . . . 15

3 Direct and Statistical Testing Methods for Selecting K 17
3.1 Introduction . . . 17
3.2 Elbow Method . . . 18
3.3 Average Silhouette Method . . . 19
3.4 Gap Statistic Method . . . 21
3.5 Comparison of Three Methods . . . 22

4 Experimental Study 23
4.1 Introduction . . . 23
4.2 Data Sets . . . 23
4.3 Performance Metric . . . 25
4.4 Results . . . 26
4.5 Discussion . . . 38

5 Conclusion and Future Work 39
5.1 Introduction . . . 39
5.2 Conclusion . . . 39
5.3 Future Work . . . 40

References 41

Index 43

A Codes 45
A.1 R-Code of Elbow Method for k-means clustering . . . 45
A.2 R-Code of Average Silhouette Method for k-means clustering . . . 45
A.3 R-Code of Gap Statistic Method for k-means clustering . . . 46

List of Figures

1.1 Clustering Process 1

3.1 Elbow Method Example 19

4.1 Elbow Method on Iris Dataset 27
4.2 Elbow Method on Isotopic Composition Plutonium Batches 27
4.3 Elbow Method on Votes for Republican Candidate in Presidential Elections . . 28
4.4 Elbow Method on Ruspini Dataset 28
4.5 Elbow Method on Violent Crime Rates by US State 29
4.6 Elbow Method on Wine Data Set 29
4.7 Elbow Method on Bivariate Data Set 30
4.8 Average Silhouette Method on Iris Dataset 30
4.9 Average Silhouette Method on Isotopic Composition Plutonium Batches 31
4.10 Average Silhouette Method on Votes for Republican Candidate in Presidential Elections 31
4.11 Average Silhouette Method on Ruspini Data Set 32
4.12 Average Silhouette Method on Violent Crime Rates by US State 32
4.13 Average Silhouette Method on Wine Data Set 33
4.14 Average Silhouette Method on Bivariate Data Set 33
4.15 Gap Statistic Method on Iris Dataset 34
4.16 Gap Statistic Method on Isotopic Composition Plutonium Batches 35
4.17 Gap Statistic Method on Republican Candidate in Presidential Elections 35
4.18 Gap Statistic Method on Ruspini 36
4.19 Gap Statistic Method on Violent Crime Rates by US State 36
4.20 Gap Statistic Method on Wine Data Set 37
4.21 Gap Statistic Method on Bivariate Data Set 37

List of Tables

4.1 Table to compare outputs of different methods 38

List of Algorithms

1 Basic K-means algorithm. 13
2 Elbow Method 18
3 Silhouette Method 21
4 Gap Statistic Method 22

ABSTRACT

The K-means algorithm is a popular data-clustering algorithm. However, one of its drawbacks is the requirement for the number of clusters, K, to be specified before the algorithm is applied. This thesis work reviews three most important existing methods for selecting the number of clusters for the algorithm. Factors that affect this selection are then discussed. The paper concludes with an analysis of the results of using the methods to determine the number of clusters for the K-means algorithm for different data sets.

Chapter 1

Introduction

1.1 Clustering

Clustering can be considered as the most important *unsupervised learning* problem; so, as every other problem of this type, it deals with finding a *structure* in a collection of given unlabeled datasets. A very common informal definition of clustering could be "the process of organizing objects into groups whose members are similar in some features of the given dataset". A *cluster* is therefore a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters.

We can show this with a simple graphical example:

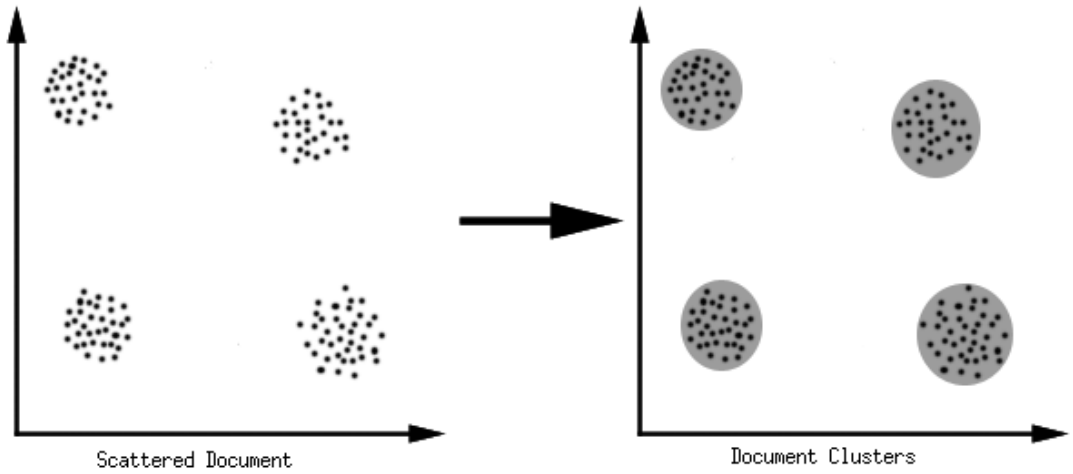


Figure 1.1: Clustering Process

In Figure 1.1 we can easily identify the 4 clusters into which the data can be divided; the similarity criterion is *distance*: two or more objects belong to the same cluster if they are "close" according to a given distance (in this case geometrical distance). This is called *distance-based*

clustering. Another kind of clustering is *conceptual clustering*: two or more objects belong to the same cluster if this one defines a concept common to all that objects. In other words, objects are grouped according to their fit to descriptive concepts, not according to simple similarity measures.

The main goal of clustering is to determine the intrinsic grouping in a given set of unlabeled data. But there is no such factors to decide what constitutes a good clustering. It can be shown that there is no absolute "best" criterion which would be independent of the final aim of the clustering. Consequently, it is the user which must supply this criterion, in such a way that the result of the clustering will suit their needs. For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding "natural clusters" and describe their unknown properties ("natural" data types), in finding useful and suitable groupings ("useful" data classes) or in finding unusual data objects (outlier detection).

1.1.1 Applications

Clustering algorithms can be applied in a very wide range of fields. In *Marketing* we can use clustering to find groups of customers with similar behavior from a large database of customers data containing their properties and past buying records. In *Biology* we can classify plants and animals by their given feature sets. Book ordering and sorting can be a good application of clustering in the *Libraries*. *Insurance* companies use clustering for identifying groups of insurance policy holders with a high average claim cost. Recently urban developers are using clustering methods for identifying groups of houses according to their house type, value and geographical location in *city-planning*. To identify dangerous zones of earthquake we can observe earthquake epicenters by using clustering algorithms on. These days clustering algorithms are mostly used in *WWW* for document classification; clustering weblog data to discover groups of similar access patterns.

1.1.2 Requirements

The main requirements that a clustering algorithm should satisfy are:

- **Scalability** : We need highly scalable clustering algorithms to deal with large databases.
- **Ability to deal with different kinds of attributes** : Algorithms should be capable to be applied on any kind of data such as interval-based (numerical) data, categorical, and binary data.
- **Discovery of clusters with attribute shape** : The clustering algorithm should be capable of detecting clusters of arbitrary shape. They should not be bounded to only distance

measures that tend to find spherical cluster of small sizes.

- **High dimensionality** : The clustering algorithm should not only be able to handle low-dimensional data but also the high dimensional space.
- **Ability to deal with noisy data** : Databases contain noisy, missing or erroneous data. Some algorithms are sensitive to such data and may lead to poor quality clusters.
- **Interpretability** : The clustering results should be interpretable, comprehensible, and usable.

1.1.3 Problems

There are a number of problems with clustering. Among them:

- Current clustering techniques do not address all the requirements adequately (and concurrently);
- Dealing with large number of dimensions and large number of data items can be problematic because of time complexity;
- The effectiveness of the method depends on the definition of distance (for distance-based clustering);
- If an obvious distance measure doesnt exist we must define it, which is not always easy, especially in multi-dimensional spaces;
- The result of the clustering algorithm (that in many cases can be arbitrary itself) can be interpreted in different ways.

1.1.4 Classification

Clustering algorithms may be classified as listed below:

- **Hierarchical Clustering** : Hierarchical clustering, is based on the core idea of objects being more related to nearby objects than to objects farther away. These algorithms connect "objects" to form "clusters" based on their distance. A cluster can be described largely by the maximum distance needed to connect parts of the cluster. At different distances, different clusters will form, which can be represented using a *dendrogram*, which explains where the common name "Hierarchical Clustering" comes from: these algorithms do not provide a single partitioning of the data set, but instead provide an extensive hierarchy of clusters that merge with each other at certain distances. In a dendrogram, the Y-axis

marks the distance at which the clusters merge, while the objects are placed along the X-axis such that the clusters don't mix.

Connectivity based clustering is a whole family of methods that differ by the way distances are computed. Apart from the usual choice of *distance functions*, the user also needs to decide on the linkage criterion (since a cluster consists of multiple objects, there are multiple candidates to compute the distance to) to use. Popular choices are known as *single-linkage clustering* (the minimum of object distances), *complete linkage clustering* (the maximum of object distances) or UPGMA ("Unweighted Pair Group Method with Arithmetic Mean", also known as average linkage clustering). Furthermore, hierarchical clustering can be agglomerative (starting with single elements and aggregating them into clusters) or divisive (starting with the complete data set and dividing it into partitions).

These methods will not produce a unique partitioning of the data set, but a hierarchy from which the user still needs to choose appropriate clusters. They are not very robust towards outliers, which will either show up as additional clusters or even cause other clusters to merge (known as "chaining phenomenon", in particular with single-linkage clustering). In the general case, the complexity is $O(n^3)$ for agglomerative clustering and $O(2^{n-1})$ for *divisive clustering*, which makes them too slow for large data sets. For some special cases, optimal efficient methods (of complexity $O(n^2)$ are known as single-linkage and complete-linkage clustering. In the *data mining* community these methods are recognized as a theoretical foundation of cluster analysis, but often considered obsolete. They did however provide inspiration for many later methods such as density based clustering.

- **Centroid-based Clustering** : In centroid-based clustering, clusters are represented by a central vector, which may not necessarily be a member of the data set. When the number of clusters is fixed to k, k-means clustering gives a formal definition as an optimization problem: find the k cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized.

  The optimization problem itself is known to be NP-hard, and thus the common approach is to search only for approximate solutions. A particularly well known approximative method is Lloyd's algorithm, often actually referred to as "K-means algorithm". It does however only find a local optimum, and is commonly run multiple times with different random initializations. Variations of K-means often include such optimizations as choosing the best of multiple runs, but also restricting the centroids to members of the data set (k-medoids), choosing medians (k-medians clustering), choosing the initial centers less randomly (k-means++) or allowing a fuzzy cluster assignment (fuzzy c-means).

  Most k-means-type algorithms require the number of clusters - k to be specified in advance, which is considered to be one of the biggest drawbacks of these algorithms. Furthermore, the algorithms prefer clusters of approximately similar size, as they will always assign an object to the nearest centroid. This often leads to incorrectly cut borders of

clusters (which is not surprising since the algorithm optimizes cluster centers, not cluster borders).

- **Distribution-based clustering** : The clustering model most closely related to statistics is based on distribution models. Clusters can then easily be defined as objects belonging most likely to the same distribution. A convenient property of this approach is that this closely resembles the way artificial data sets are generated: by sampling random objects from a distribution.

  While the theoretical foundation of these methods is excellent, they suffer from one key problem known as overfitting, unless constraints are put on the model complexity. A more complex model will usually be able to explain the data better, which makes choosing the appropriate model complexity inherently difficult.

  One prominent method is known as Gaussian mixture models (using the expectation-maximization algorithm). Here, the data set is usually modelled with a fixed (to avoid overfitting) number of Gaussian distributions that are initialized randomly and whose parameters are iteratively optimized to better fit the data set. This will converge to a local optimum, so multiple runs may produce different results. In order to obtain a hard clustering, objects are often then assigned to the Gaussian distribution they most likely belong to; for soft clusterings, this is not necessary.

  Distribution-based clustering produces complex models for clusters that can capture correlation and dependence between attributes. However, these algorithms put an extra burden on the user: for many real data sets, there may be no concisely defined mathematical model (e.g. assuming Gaussian distributions is a rather strong assumption on the data).

- **Density-based Clustering** : In density-based clustering, clusters are defined as areas of higher density than the remainder of the data set. Objects in these sparse areas that are required to separate clusters are usually considered to be noise and border points.

  The most popular density based clustering method is DBSCAN. In contrast to many newer methods, it features a well-defined cluster model called “density-reachability”. Similar to linkage based clustering, it is based on connecting points within certain distance thresholds. However, it only connects points that satisfy a density criterion, in the original variant defined as a minimum number of other objects within this radius. A cluster consists of all density connected objects (which can form a cluster of an arbitrary shape, in contrast to many other methods) plus all objects that are within these objects' range. Another interesting property of DBSCAN is that its complexity is fairly low it requires a linear number of range queries on the database - and that it will discover essentially the same results (it is deterministic for core and noise points, but not for border points) in each run, therefore there is no need to run it multiple times. OPTICS is a generalization of DBSCAN that removes the need to choose an appropriate value for the

range parameter ε, and produces a hierarchical result related to that of linkage clustering. DeLi-Clu, Density-Link-Clustering combines ideas from single-linkage clustering and OPTICS, eliminating the ε parameter entirely and offering performance improvements over OPTICS by using an R-tree index.

The key drawback of DBSCAN and OPTICS is that they expect some kind of density drop to detect cluster borders. On data sets with, for example, overlapping Gaussian distributions a common use case in artificial data the cluster borders produced by these algorithms will often look arbitrary, because the cluster density decreases continuously. On a data set consisting of mixtures of Gaussians, these algorithms are nearly always outperformed by methods such as EM clustering that are able to precisely model this kind of data.

Mean-shift is a clustering approach where each object is moved to the densest area in its vicinity, based on kernel density estimation. Eventually, objects converge to local maxima of density. Similar to k-means clustering, these "density attractors" can serve as representatives for the data set, but mean-shift can detect arbitrary-shaped clusters similar to DBSCAN. Due to the expensive iterative procedure and density estimation, mean-shift is usually slower than DBSCAN or k-Means. Besides that, the applicability of the mean-shift algorithm to multidimensional data is hindered by the unsmooth behaviour of the kernel density estimate, which results in over-fragmentation of cluster tails.

1.2 Literature Survey

1.2.1 Values of K specified within a range or set

The performance of a clustering algorithm may be affected by the chosen value of K. Therefore, instead of using a single predefined K, a set of values might be adopted. It is important for the number of values considered to be reasonably large, to reflect the specific characteristics of the data sets. At the same time, the selected values have to be significantly smaller than the number of objects in the data sets, which is the main motivation for performing data clustering.

Reported studies [10–15] on K-means clustering and its applications usually do not contain any explanation or justification for selecting particular values for K. First, a number of researchers [14, 15] used only one or two values for K. Second, several other researchers [10, 12] utilized relatively large K values compared with the number of objects. These two actions contravene the above mentioned guidelines for selecting K. Therefore,the clustering results do not always correctly represent the performance of the tested algorithms.

In general, the performance of any new version of the K-means algorithm could be verified by com- paring it with its predecessors on the same criteria. In particular, the sum of cluster

distortions is usually employed as such a performance indicator. [12, 15] Thus, the comparison is considered fair because the same model and criterion are used for the performance analysis.

1.2.2 Values of K specified by the user

The K-means algorithm implementation in many data-mining or data analysis software packages [16] requires the number of clusters to be specified by the user. To find a satisfactory clustering result, usually, a number of iterations are needed where the user executes the algorithm with different values of K. The validity of the clustering result is assessed only visually without applying any formal performance measures. With this approach, it is difficult for users to evaluate the clustering result for multi-dimensional data sets.

1.2.3 Values of K determined in a later processing step

When K-means clustering is used as a pre-processing tool, the number of clusters is determined by the specific requirements of the main processing algorithm. [17] No attention is paid to the effect of the clustering results on the performance of this algorithm. In such applications, the K-means algorithm is employed just as a 'black box' without validation of the clustering result.

1.2.4 Values of K equated to the number of generators

Synthetic data sets, which are used for testing algorithms, are often created by a set of normal or uniform distribution generators. Then, clustering algorithms are applied to those data sets with the number of clusters equated to the number of generators. It is assumed that any resultant cluster will cover all objects created by a particular generator. Thus, the clustering performance is judged on the basis of the difference between objects covered by a cluster and those created by the corresponding generator. Such a difference can be measured by simply counting objects or calculating the information gain. [18]

Unfortunately, this method of selecting K cannot be applied to practical problems. The data distribution in practical problems is unknown and also the number of generators cannot be specified.

1.2.5 Values of K determined by statistical measures

There are several statistical measures available for selecting K. These measures are often applied in combination with probabilistic clustering approaches. They are calculated with certain assumptions about the underlying distribution of the data. The Bayesian information criterion

or Akeikes information [19] criterion is calculated on data sets which are constructed by a set of Gaussian distributions. The measures applied by Hardy are based on the assumption that the data set fits the Poisson distribution. Monte Carlo techniques, which are associated with the null hypothesis, are used for assessing the clustering results and also for determining the number of clusters.

1.2.6 Values of K determined through visualization

Visual verification is applied widely because of its simplicity and explanation possibilities. Visual examples are often used to illustrate the drawbacks of an algorithm or to present the expected clustering results. [14]

The assessment of a clustering result using visualization techniques depends heavily on their implicit nature. The clustering models utilized by some clustering methods may not be appropriate for particular data sets. The application of visualization techniques implies a data distribution continuity in the expected clusters. If the K-means approach is applied to such data sets, there is not any cluster that satisfies the K-means clustering model and at the same time corresponds to a particular object grouping in the illustrated data sets. Therefore, the K-means algorithm cannot produce the expected clustering results. This suggests that the K-means approach is unsuitable for such data sets.

1.3 Objective

Partitioning methods or Centroid-base Clustering algorithms, such as K-means clustering require the users to specify the number of clusters to be generated. One fundamental question is: If the data is clusterable, then how to choose the right number of expected clusters K? Unfortunately, there is no definitive answer to this question. The optimal clustering is somehow subjective and depend on the method used for measuring similarities and the parameters used for partitioning.

In this work, our prime objectives are:

- Describing different methods of determining the optimal number of clusters for K-means clustering.
- These methods include direct methods and statistical testing methods. Direct methods consists of optimizing a criterion, such as the within cluster sums of squares or the average silhouette. Statistical testing methods consists of comparing evidence against null hypothesis. An example is the gap statistic.

- We'll apply these methods on various data sets and find the optimal number of clusters for K-means clustering.
- Finally we'll compare performance among these methods on the basis of determining number of clusters for different data sets.

1.4 Thesis Organization

In Chapter One, we try to give a general overview about what is machine learning, clustering, why clustering is so important and what are the factors behind determining number of clusters. In Chapter Two, we will discuss the idea behind determining number of clusters of a dataset and how it affects the output of K-means algorithm. In Chapter Three, we will describe different methods for selecting number of clusters and there approach, algorithms. In Chapter Four, we want to show experimental result of our reviewed methods algorithm and output. Then we'll compare the output with the original output. In Chapter Five, we put a discussion about the future of this work, where it can give promising result and how it can be improved further.

Chapter 2

Background

2.1 Introduction

2.1.1 Machine Learning

Machine learning is the subfield of computer science that gives computers the ability to learn without being explicitly programmed. Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data such algorithms overcome following strictly static program instructions by making data driven predictions or decisions, through building a model from sample inputs. Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms is infeasible; example applications include spam filtering, detection of network intruders or malicious insiders working towards a data breach, optical character recognition (OCR), search engines and computer vision.

Tom M. Mitchell [4] provided a widely quoted, more formal definition:

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E."

Machine learning tasks are typically classified into three broad categories, depending on the nature of the learning "signal" or "feedback" available to a learning system. These are:

Supervised Learning
Supervised learning is where you have input variables (x) and an output variable (Y) and you

use an algorithm to learn the mapping function from the input to the output. Exact functon may be as $Y = f(X)$. The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data.

It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers, the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance. Supervised learning problems can be further grouped into regression and classification problems.

- **Classification** : A classification problem is when the output variable is a category, such as "red" or "blue" or "disease" and "no disease".
- **Regression** : A regression problem is when the output variable is a real value, such as "dollars" or "weight".

Unsupervised Learning

No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).Unsupervised learning is where you only have input data (X) and no corresponding output variables.

The goal for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data.These are called unsupervised learning because unlike supervised learning above there is no correct answers and there is no teacher. Algorithms are left to their own devises to discover and present the interesting structure in the data.Unsupervised learning problems can be further grouped into clustering and association problems.

- **Clustering** : A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- **Association** : An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Semi-Supervised Machine Learning

Problems where there is a large amount of input data (X) and only some of the data is labeled (Y) are called semi-supervised learning problems.These problems sit in between both supervised and unsupervised learning. A good example is a photo archive where only some of the images are labeled, (e.g. dog, cat, person) and the majority are unlabeled.

Many real world machine learning problems fall into this area. This is because it can be expensive or time-consuming to label data as it may require access to domain experts. Whereas unlabeled data is cheap and easy to collect and store.

Reinforcement Learning
A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). The program is provided feedback in terms of rewards and punishments as it navigates its problem space.

2.2 A Deep Dive into Cluster Analysis

Cluster analysis groups data objects based only on information found in the data that describes the objects and their relationships. The goal is that the objects within a group be similar (or related) to one another and different from (or unrelated to) the objects in other groups. The greater the similarity (or homogeneity) within a group and the greater the difference between groups, the better or more distinct the clustering.

Cluster analysis is related to other techniques that are used to divide data objects into groups. For instance, clustering can be regarded as a form of classification in that it creates a labeling of objects with class (cluster) labels. However, it derives these labels only from the data. In contrast, classification in the sense of our previous section is **supervised classification**; i.e., new, unlabeled objects are assigned a class label using a model developed from objects with known class labels. For this reason, cluster analysis is sometimes referred to as **unsupervised classification**. When the term classification is used without any qualification within data mining, it typically refers to supervised classification.

2.2.1 Centroid Based Cluster

A cluster is a set of objects in which each object is closer (more similar) to the center that defines the cluster than to the centers of any other cluster. For data with continuous attributes, the prototype of a cluster is often a centroid, i.e., the average (mean) of all the points in the cluster. When a centroid is not meaningful, such as when the data has categorical attributes, the prototype is often a medoid, i.e., the most representative point of a cluster. For many types of data, the centroid can be regarded as the most central point. One of the most popular centroid-based algorithm is K-means. Let's have a look on to the details about K-means algorithm.

2.2.2 K-means

Centroid-based clustering techniques create a one-level partitioning of the data objects. There are a number of such techniques, but two of the most prominent are K-means and K-medoid. K-means defines a prototype in terms of a centroid, which is usually the mean of a group of points, and is typically applied to objects in a continuous n dimensional space. K-medoid defines a prototype in terms of a medoid, which is the most representative point for a group of points, and can be applied to a wide range of data since it requires only a proximity measure for a pair of objects. While a centroid almost never corresponds to an actual data point, a medoid, by its definition, must be an actual data point. In this subsection, we will focus solely on K-means, which is one of the oldest and most widely used clustering algorithms and our objective is based on this algorithm.

2.2.3 The Basic K-means Algorithm

The K-means clustering technique is simple, and we begin with a description of the basic algorithm. We first choose K initial centroids, where K is a user-specified parameter, namely, the number of clusters desired. Each point is then assigned to the closest centroid, and each collection of points assigned to a centroid is a cluster. The centroid of each cluster is then updated based on the points assigned to the cluster. We repeat the assignment and update steps until no point changes clusters, or equivalently, until the centroids remain the same.

K-means is formally described by Algorithm 1

Algorithm 1 Basic K-means algorithm.

Require: Select K points as initial centroids.
  while Centroids do not change **do**
    Form K clusters by assigning each point to its closest centroid.
    Recompute the centroid of each cluster.
  end while

2.2.4 Assigning Points to the Closest Centroid

To assign a point to the closest centroid, we need a proximity measure that quantifies the notion of “closest” for the specific data under consideration. Euclidean (L2) distance is often used for data points in Euclidean space, while cosine similarity is more appropriate for documents. However, there may be several types of proximity measures that are appropriate for a given type of data. For example, Manhattan (L1) distance can be used for Euclidean data, while the Jaccard measure is often employed for documents.

Usually, the similarity measures used for K-means are relatively simple since the algorithm repeatedly calculates the similarity of each point to each centroid. In some cases, however, such as when the data is in low-dimensional Euclidean space, it is possible to avoid computing many of the similarities, thus significantly speeding up the K-means algorithm. Bisecting K-means is another approach that speeds up K-means by reducing the number of similarities computed.

2.2.5 Centroids and Objective Functions

Step 4 of the K-means algorithm was stated rather generally as "recompute the centroid of each cluster," since the centroid can vary, depending on the proximity measure for the data and the goal of the clustering. The goal of the clustering is typically expressed by an objective function that depends on the proximities of the points to one another or to the cluster centroids; e.g., minimize the squared distance of each point to its closest centroid. However, the key point is this: once we have specified a proximity measure and an objective function, the centroid that we should choose can often be determined mathematically.

2.2.6 Data in Euclidean Space

Consider data whose proximity measure is Euclidean distance. For our objective function, which measures the quality of a clustering, we use the sum of the squared error (SSE), which is also known as scatter. In other words, we calculate the error of each data point, i.e., its Euclidean distance to the closest centroid, and then compute the total sum of the squared errors. Given two different sets of clusters that are produced by two different runs of K-means, we prefer the one with the smallest squared error since this means that the prototypes (centroids) of this clustering are a better representation of the points in their cluster.

$SSE = \sum_{i=1}^{K} \sum_{x \epsilon C_i} dist(C_i, x)^2$

Where, x = An object.
C_i = The iήh cluster.
c_i = The centroid of cluster C_i.
c = The centroid of all points.
m_i = The number of objects in the iήh cluster.
m = The number of objects in the dataset.
K = The number of clusters.
$dist$ = Standard Euclidean distance between two objects in Euclidean Space.

2.2.7 Choosing Initial Centroids

When random initialization of centroids is used, different runs of K-means typically produce different total SSEs. Choosing the proper initial centroids is the key step of the basic K-means procedure. A common approach is to choose the initial centroids randomly, but the resulting clusters are often poor.

An optimal clustering will be obtained as long as two initial centroids fall anywhere in a pair of clusters, since the centroids will redistribute themselves, one to each cluster. Unfortunately, as the number of clusters becomes larger, it is increasingly likely that at least one pair of clusters will have only one initial centroid. In this case, because the pairs of clusters are farther apart than clusters within a pair, the K-means algorithm will not redistribute the centroids between pairs of clusters, and thus, only a local minimum will be achieved.

Because of the problems with using randomly selected initial centroids, which even repeated runs may not overcome, other techniques are often employed for initialization. One effective approach is to take a sample of points and cluster them using a hierarchical clustering technique. K clusters are extracted from the hierarchical clustering, and the centroids of those clusters are used as the initial centroids. This approach often works well, but is practical only if the sample is relatively small, e.g., a few hundred to a few thousand (because hierarchical clustering is expensive), and K is relatively small compared to the sample size.

2.2.8 Updating Centroids Incrementally

Instead of updating cluster centroids after all points have been assigned to a cluster, the centroids can be updated incrementally, after each assignment of a point to a cluster. Notice that this requires either zero or two updates to cluster centroids at each step, since a point either moves to a new cluster (two updates) or stays in its current cluster (zero updates). Using an incremental update strategy guarantees that empty clusters are not produced since all clusters start with a single point, and if a cluster ever has only one point, then that point will always be reassigned to the same cluster.

In addition, if incremental updating is used, the relative weight of the point being added may be adjusted; e.g., the weight of points is often decreased as the clustering proceeds. While this can result in better accuracy and faster convergence, it can be difficult to make a good choice for the relative weight, especially in a wide variety of situations. These update issues are similar to those involved in updating weights for artificial neural networks.

Yet another benefit of incremental updates has to do with using objectives other than "minimize SSE." Suppose that we are given an arbitrary objective function to measure the goodness of a set of clusters. When we process an individual point, we can compute the value of the objec-

tive function for each possible cluster assignment, and then choose the one that optimizes the objective.

On the negative side, updating centroids incrementally introduces an order dependency. In other words, the clusters produced may depend on the order in which the points are processed. Although this can be addressed by randomizing the order in which the points are processed, the basic K-means approach of updating the centroids after all points have been assigned to clusters has no order dependency. Also, incremental updates are slightly more expensive. However, K-means converges rather quickly, and therefore, the number of points switching clusters quickly becomes relatively small.

Chapter 3

Direct and Statistical Testing Methods for Selecting K

3.1 Introduction

Determining the number of clusters in a data set, a quantity often labelled K as in the K-means algorithm, is a frequent problem in data clustering, and is a distinct issue from the process of actually solving the clustering problem. Clustering solutions may vary as different numbers of clusters are specified. A clustering technique would most possibly recover the underlying cluster structure given a good estimate of the true number of clusters.

For a certain class of clustering algorithms (in particular k-means, k-medoids and expectation-maximization algorithm), there is a parameter commonly referred to as K that specifies the number of clusters to detect. Other algorithms such as DBSCAN and OPTICS algorithm do not require the specification of this parameter; hierarchical clustering avoids the problem altogether.

The correct choice of K is often ambiguous, with interpretations depending on the shape and scale of the distribution of points in a data set and the desired clustering resolution of the user. In addition, increasing K without penalty will always reduce the amount of error in the resulting clustering, to the extreme case of zero error if each data point is considered its own cluster (i.e., when K equals the number of data points, n). Intuitively then, the optimal choice of K will strike a balance between maximum compression of the data using a single cluster, and maximum accuracy by assigning each data point to its own cluster. If an appropriate value of K is not apparent from prior knowledge of the properties of the data set, it must be chosen somehow. There are several categories of methods for making this decision.

A number of strategies for estimating the optimal number of clusters have been proposed. A very extensive comparative evaluation was conducted by Milligan and Cooper [6], where they compared 30 proposed methods in estimating the true number of clusters when applying hierar-

chical clustering algorithms to simulated data with well-separated clusters. According to their work, Calinski and Harabaszs index [7] is the most effective one, followed by Duda and Harts method [5] and the C-index.

3.2 Elbow Method

The oldest method for determining the true number of clusters in a data set is inelegantly called the elbow method. It's pure simplicity, and for that reason alone has probably been reinvented many times over (ed. note: This is a problem peculiar to clustering; since there are many intuitively plausible ways to cluster data, it's easy to reinvent techniques, and in fact one might argue that there are very few techniques in clustering that are complex enough to be 'owned' by any inventor). The idea is this:

Start with $k = 1$, and keep increasing it, measuring the cost of the optimal quality solution. If at some point the cost of the solution drops dramatically, that's the true k.

The intuitive argument behind the elbow method is this: you're trying to shoehorn k boxes of data into many fewer groups, so by the pigeonhole principle, at least one group will contain data from two different boxes, and the cost of this group will skyrocket. When you finally find the right number of groups, every box fits perfectly, and the cost drops.

Recall that, the basic idea behind partitioning methods, such as k-means clustering, is to define clusters such that the total intra-cluster variation (known as total within-cluster variation or total within-cluster sum of square) is minimized:

$minimize(\sum_{k=1}^{K} WC_k)$;

where C_k is the k^{th} cluster and $W(C_k)$ is the within-cluster variation. That means the total within-cluster sum of square (wss) measures the compactness of the clustering and we want it to be as small as possible.

The optimal number of clusters can be defined as follow:

Algorithm 2 Elbow Method

1. Compute clustering algorithm (e.g., K-means clustering) for different values of K. For instance, by varying K from 1 to 10 clusters.
2. For each K, calculate the total within-cluster sum of square (wss).
3. Plot the curve of wss according to the number of clusters K.
4. The location of a bend (knee) in the plot is generally considered as an indicator of the appropriate number of clusters.

In Figure 3.1 we can see that Total within-clusters sum of squares have been plotted along the Y-axis and Number of clusters K along the X-axis. The bend(knee) in the graph is detected

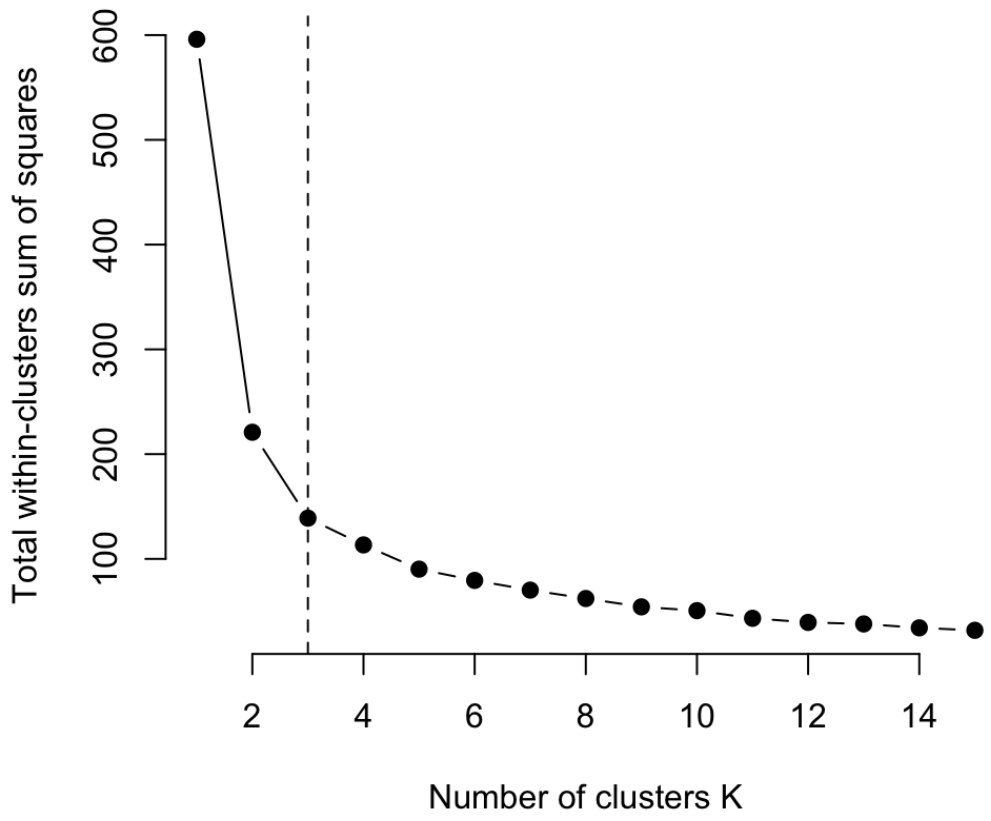


Figure 3.1: Elbow Method Example

where the value of K is three. So, here in the example graph the elbow method suggests 3 cluster solutions.

It looks deceptively simple. But it has that aspect that we've mentioned earlier - it defines the desired outcome as a transition, rather than a state. In practice of course, "optimal quality" becomes "whichever clustering algorithm you like to run", and "drops dramatically" becomes one of those gigantic hacks that make Principle and Rigor run away crying and hide under their bed. So, here the problem with elbow method is: This "elbow" cannot always be unambiguously identified. Sometimes there is no elbow, or several elbows

3.3 Average Silhouette Method

Silhouette refers to a method of interpretation and validation of consistency within clusters of data. The technique provides a succinct graphical representation of how well each object lies within its cluster. It was first described by Peter J. Rousseeuw in 1987. Kaufman and

Rousseeuw [3] proposed the silhouette index as to estimate the optimum number of clusters in the data. The definition of the silhouette index is based on the silhouettes introduced by Rousseeuw [2], which are constructed to show graphically how well each object is classified in a given clustering output.

The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from -1 to 1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

The silhouette can be calculated with any distance metric, such as the Euclidean distance or the Manhattan distance.

- **Euclidean Distance** : The Euclidean distance between points p and q is the length of the line segment connecting them ($\overline{\mathbf{pq}}$). In Cartesian coordinates, if p = (p_1, p_2,..., p_n) and q = (q_1, q_2,..., q_n) are two points in Euclidean n-space, then the distance d from p to q, or from q to p is given by the Pythagorean formula:

  $d(p, q)$ = $\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$

- **Manhattan Distance** : The distance between two points measured along axes at right angles. In a plane with p_1 at (x_1, y_1) and p_2 at (x_2, y_2), it is $|x_1 - x_2| + |y_1 - y_2|$.

Assume the data have been clustered via any technique, such as K-means, into K clusters. For each datum i, let (i) be the average dissimilarity of i with all other data within the same cluster. We can interpret $a(i)$ as how well i is assigned to its cluster (the smaller the value, the better the assignment). We then define the average dissimilarity of point i to a cluster c as the average of the distance from i to all points in c.

Let $b(i)$ be the lowest average dissimilarity of i to any other cluster, of which i is not a member. The cluster with this lowest average dissimilarity is said to be the "neighbouring cluster" of i because it is the next best fit cluster for point i. We now define a silhouette:

$s(i) = \frac{b(i)-a(i)}{max(a(i),b(i))}$; where $-1 \leq s(i) \leq 1$

For $s(i)$ to be close to 1, we require $a(i) \ll b(i)$. As $a(i)$ is a measure of how dissimilar i is to its own cluster, a small value means it is well matched. Furthermore, a large $b(i)$ implies that i is badly matched to its neighbouring cluster. Thus an $s(i)$ close to one means that the data is appropriately clustered. If $s(i)$ is close to negative one, then by the same logic we see that i would be more appropriate if it was clustered in its neighbouring cluster. An $s(i)$ near zero means that the datum is on the border of two natural clusters.

The average $s(i)$ over all data of a cluster is a measure of how tightly grouped all the data in the cluster are. Thus the average $s(i)$ over all data of the entire dataset is a measure of how appropriately the data have been clustered. If there are too many or too few clusters, as may occur when a poor choice of K is used in the clustering algorithm (e.g.: K-means), some of the clusters will typically display much narrower silhouettes than the rest. Thus silhouette plots and averages may be used to determine the natural number of clusters within a dataset. One can also increase the likelihood of the silhouette being maximized at the correct number of clusters by re-scaling the data using feature weights that are cluster specific [1].

The average silhouette approach we'll be described comprehensively in the chapter cluster validation statistics. Briefly, it measures the quality of a clustering. That is, it determines how well each object lies within its cluster. A high average silhouette width indicates a good clustering. The algorithm is similar to the elbow method and can be computed as follow:

Algorithm 3 Silhouette Method

1. Compute clustering algorithm (e.g., K-means clustering) for different values of K. For instance, by varying K from 1 to 10 clusters.
2. For each K, calculate the average silhouette of observations $(avg.sil)$.
3. Plot the curve of $avg.sil$ according to the number of clusters K.
4. The location of the maximum is considered as the appropriate number of clusters.

3.4 Gap Statistic Method

The gap statistic has been published by Tibshirani, Walther and Hastie [8]. The approach can be applied to any clustering method (K-means clustering, Hierarchical clustering, etc.). The gap statistic compares the total within intracluster variation for different values of K with their expected values under null reference distribution of the data, i.e. a distribution with no obvious clustering. We know that the total within intra-cluster variation for a given K clusters is the total within sum of square (W_k). The idea behind their approach was to find a way to standardize the comparison of $\log W_k$ with a null reference distribution of the data, i.e. a distribution with no obvious clustering. Their estimate for the optimal number of clusters K is the value for which $\log W_k$ falls the farthest below this reference curve. This information is contained in the following formula for the gap statistic:

$Gap_n(k) = E_n^*\{\log W_k\} - \log W_k$

The reference datasets are in our case generated by sampling uniformly from the original datasets bounding box. To obtain the estimate $E_n^*\{\log W_k\}$ we compute the average of B copies $\log W_k^*$ for $B = 10$, each of which is generated with a Monte Carlo sample from the reference distribution. Those $\log W_k^*$ from the B Monte Carlo replicates exhibit a standard deviation $\mathrm{sd}(k)$ which, accounting for the simulation error, is turned into the quantity

$s_k = \sqrt{1 + \frac{1}{B}}.\mathrm{sd}(k)$

Finally, the optimal number of clusters is the smallest K such that $\mathrm{Gap}(k) \geq \mathrm{Gap}(k+1) - s_{k+1}$.

The computation of the gap statistic involves the following steps:

Algorithm 4 Gap Statistic Method

1. Cluster the observed data, varying the number of clusters from $k = 1, ..., k_{\max}$, and compute the corresponding W_k.
2. Generate B reference data sets and cluster each of them with varying number of clusters $k = 1, ..., k_{\max}$. Compute the estimated gap statistic $\mathrm{Gap}(k) = (1/B)\sum_{b=1}^{B} \log W_{kb}^* - \log W_k$.
3. With $\bar{w} = (1/B)\sum_b \log W_{kb}^*$, compute the standard deviation $\mathrm{sd}(k) = [(1/B)\sum_b (\log W_{kb}^* - \bar{w})^2]^{1/2}$ and define $s_k = \sqrt{1 + 1/B}\,\mathrm{sd}(k)$.
4. Choose the number of clusters as the smallest K such that $\mathrm{Gap}(k) \geq \mathrm{Gap}(k+1) - s_{k+1}$.

3.5 Comparison of Three Methods

The disadvantage of elbow and average silhouette methods is that, they measure a global clustering characteristic only. A more sophisticated method is to use the gap statistic which provides a statistical procedure to formalize the elbow/silhouette heuristic in order to estimate the optimal number of clusters. The gap statistic, proposed by Tobshirani et al. formalizes this approach and offers an easy-to-implement algorithm that successfully finds the correct K in the case of globular, Gaussian-distributed, mildly disjoint data distributions.

Chapter 4

Experimental Study

4.1 Introduction

An experiment is a procedure carried out to support, refute, or validate a hypothesis. Experiments provide insight into cause-and-effect by demonstrating what outcome occurs when a particular factor is manipulated. Experiments vary greatly in goal and scale, but always rely on repeatable procedure and logical analysis of the results. In engineering and the physical sciences, experiments are a primary component of the scientific method. They are used to test theories and hypotheses about how physical processes work under particular conditions. Typically, experiments in these fields focus on replication of identical procedures in hopes of producing identical results in each replication. No theory can have a real significance without an experimentation on that theory. To establish some kind of proposition we must have conducted some experiments and anlysis its output results. It is true for our three methods also. Thats why we have arranged some experiments to test our studied methods.

In this chapter, at first, we have introuduced the data sets on which experiments are conducted. Then we have discussed about the parameters which are used to determine cluster characteristic. Then we discussed about different kinds of outcome of the experiments.

4.2 Data Sets

We've used different datasets and applied three methods i.e. *Elbow Method*, *Average Silhouette Method*, *Gap Statistic Method* for finding optimal number of clusters in the dataset. Attribute informations of different datasets are described below:

1. **Iris Data Set** : This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this

day. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other. This is an exceedingly simple domain.

- sepal length in cm
- sepal width in cm
- petal length in cm
- petal width in cm
- class:
  - Iris Setosa
  - Iris Versicolour
  - Iris Virginica

2. **Violent Crime Rates by US State** : This data set contains statistics, in arrests per 100,000 residents for assault, murder, and rape in each of the 50 US states in 1973. Also given is the percent of the population living in urban areas. A data frame with 50 observations on 4 variables.

   - *Murder* - Numeric Murder arrests (per 100,000)
   - *Assault* - Numeric Assault arrests (per 100,000)
   - *UrbanPop* - Numeric Percent urban population
   - *Rape* - Numeric Rape arrests (per 100,000)

3. **Bivariate Data Set** : An artificial data set consisting of 3000 points in 3 well-separated clusters of size 1000 each. A data frame with 3000 observations on 2 numeric variables giving the x and y coordinates of the points, respectively.

4. **Isotopic Composition Plutonium Batches** : The pluton data frame has 45 rows and 4 columns, containing percentages of isotopic composition of 45 Plutonium batches. This data frame contains the following columns:

   - Pu_{238} - Percentages of the Pu_{238} isotope, always less than 2 percent.
   - Pu_{239} - Percentages of the Pu_{239} isotope, typically between 60 and 80 percent (from neutron capture of Uranium, U_{238}).
   - Pu_{240} Percentage of the Pu_{240} isotope.
   - Pu_{241} Percentage of the Pu_{241} isotope.

5. **Ruspini Data Set** : The Ruspini data set, consisting of 75 points in four groups that is popular for illustrating clustering techniques. A data frame with 75 observations on 2 variables giving the x and y coordinates of the points, respectively.

6. **Wine Data Set** : These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivators. The analysis determined the quantities of 13 constituents found in each of the three types of wines. All attributes are continuous.

7. **Votes for Republican Candidate in Presidential Elections** : A data frame with the percents of votes given to the republican candidate in presidential elections from 1856 to 1976. Rows represent the 50 states, and columns the 31 elections.

4.3 Performance Metric

Different clustering algorithms usually lead to different clusters of data; even for the same algorithm, the selection of different parameters or the presentation order of data objects may greatly affect the final clustering partitions. Thus, effective evaluation standards and criteria are critically important to give users confidence regarding the clustering results. At the same time, these assessments also provide some meaningful insights on how many clusters are hidden in the data.

In fact, in most real life clustering situations, the user faces the dilemma of selecting the number of clusters or partitions in the underlying data. As such, numerous indices for determining the number of clusters in a data set have been proposed.

All these clustering validity indices combine information about intracluster compactness and intercluster isolation, as well as other factors, such as geometric or statistical properties of the data, the number of data objects and dissimilarity or similarity measurements.

- **Elbow Index** : The basic idea behind partitioning methods, such as k-means clustering, is to define clusters such that the total intra-cluster variation (known as total within-cluster variation or total within-cluster sum of square) is minimized:

  $minimize(\sum_{k=1}^{K} WC_k)$;

  where C_k is the k^{th} cluster and $W(C_k)$ is the within-cluster variation. That means the total within-cluster sum of square (wss) measures the compactness of the clustering and we want it to be as small as possible.

- **Silhouette index** : Rousseeuw [2] introduced the silhouette index computed using the following equation:

  Silhouette = $\frac{\sum_{i=1}^{n} S(i)}{n}$; Silhouette $\epsilon[-1, 1]$

  where,

- $S(i) = \frac{b(i)a(i)}{max\{a(i),b(i)\}}$
- $a(i) = \frac{\sum_{j\epsilon\{C_r,i\}} d_{ij}}{n_r - 1}$ is the average dissimilarity of the ith object to all other objects of cluster C_r
- $b(i) = \min(d_{iC_s})$; $s \neq r$
- $d_{iC_s} = \frac{\sum_{j\epsilon C_s} d_{ij}}{n_s}$

The maximum value of the index is used to determine the optimal number of clusters in the data [3]. $S(i)$ is not defined for $k = 1$ (only one cluster).

- **Gap Index** : The estimated Gap statistic proposed by Tibshirani [8] is computed using the following equation:

$$Gap(q) = \frac{1}{B}\sum_{b=1}^{B} \log W_{qb} - \log W_q$$

where B is the number of reference data sets generated using uniform prescription and W_{qb} is the within-dispersion matrix. The optimal number of clusters is chosen via finding the smallest q such that:

$$Gap(q) \geq Gap(q+1) - s_{q+1}, (q = 1, ..., n-2)$$

where,

- $s_q = sd_q\sqrt{1 + \frac{1}{B}}$
- sd_q is the standard deviation of $\log W_{qb}, b = (1, ..., B) : sd_q = \sqrt{\frac{1}{B}\sum_{b=1}^{B}(\log W_{qb} - \bar{l})^2}$
- $\bar{l} = \frac{1}{B}\sum_{b=1}^{B}(\log W_{qb})$

4.4 Results

We have applied three different methods i.e. elbow method, silhoutte method and gap statistic method on various datasets and output the optimal number of clusters the method suggests. Now we will see the outputs generated for three methods in the following section:

- **Elbow Method** : Here we plot Total within sum of square (wss) along the Y-axis and Number of clusters K along the X-axis. The total within-cluster sum of square (wss) measures the compactness of the clustering and we want it to be as small as possible. And from the algorithm we know that The location of a bend (knee) in the plot is generally considered as an indicator of the appropriate number of clusters.

In Figure 4.1 we can see that three clusters have been suggested for *iris* dataset which has originally three classes.

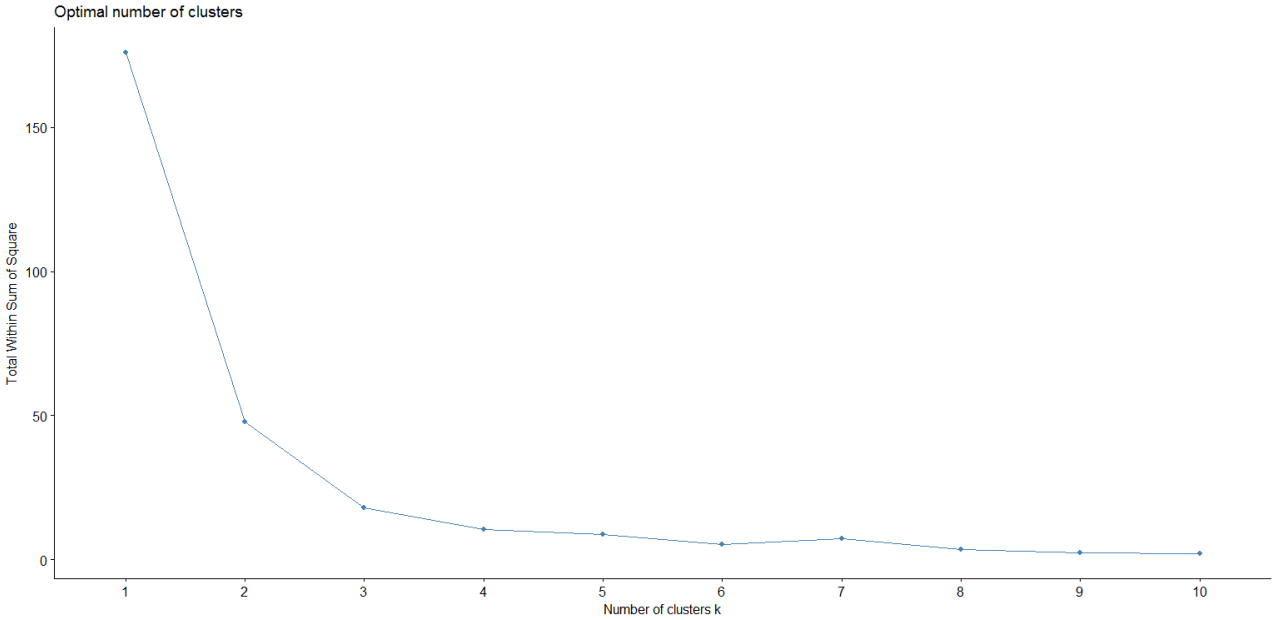


Figure 4.1: Elbow Method on Iris Dataset

In Figure 4.2 we can see that three clusters have been suggested for *Isotopic Composition Plutonium Batches* dataset.

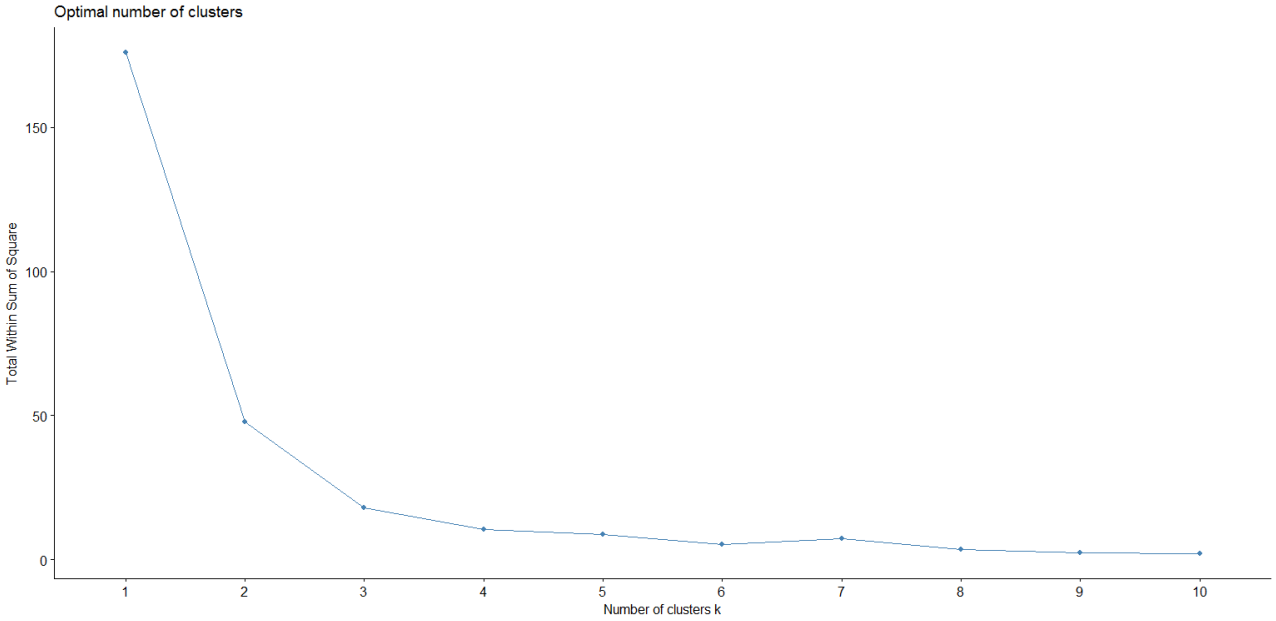


Figure 4.2: Elbow Method on Isotopic Composition Plutonium Batches

In Figure 4.3 we can see that four clusters have been suggested for *Votes for Republican Candidate in Presidential Elections* dataset.

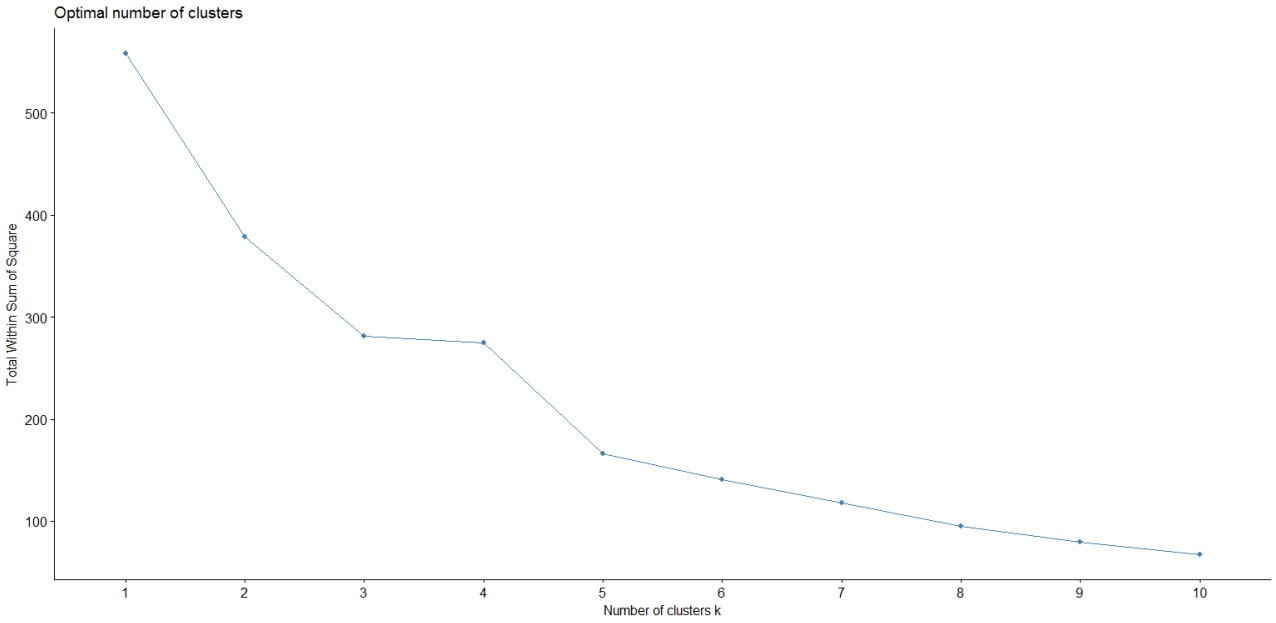


Figure 4.3: Elbow Method on Votes for Republican Candidate in Presidential Elections

In Figure 4.4 we can see that four clusters have been suggested for *Ruspini* dataset.

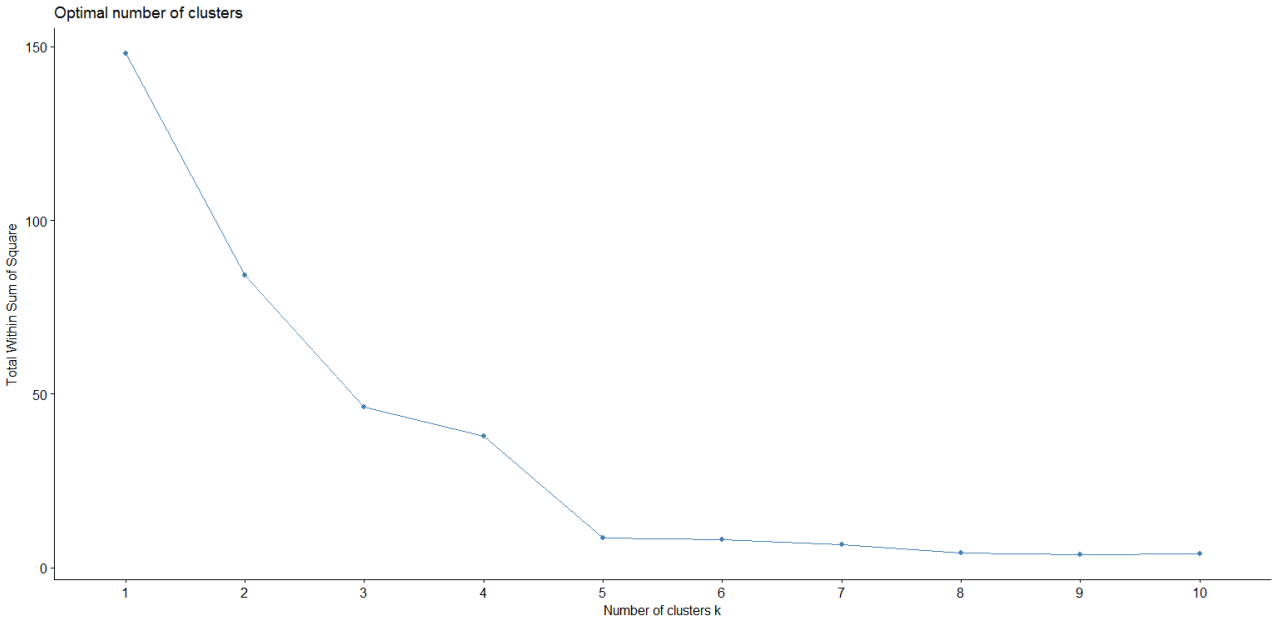


Figure 4.4: Elbow Method on Ruspini Dataset

In Figure 4.5 we can see that four clusters have been suggested for *Violent Crime Rates by US State* dataset.

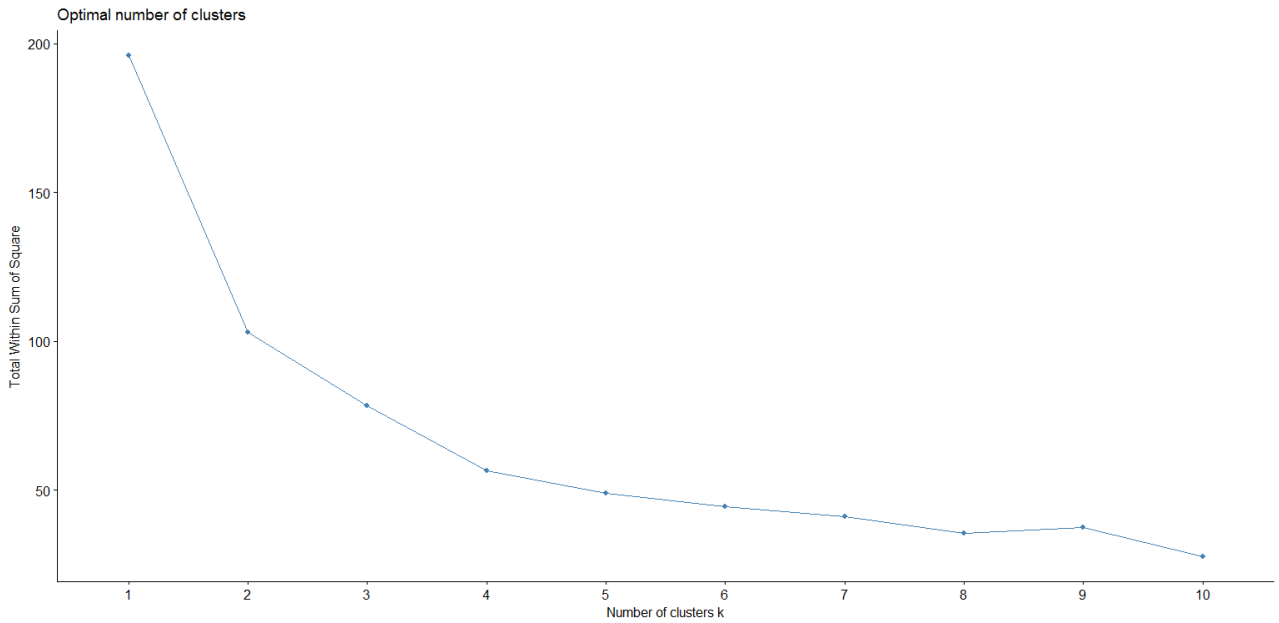


Figure 4.5: Elbow Method on Violent Crime Rates by US State

In Figure 4.6 we can see that three clusters have been suggested for *Wine* dataset.

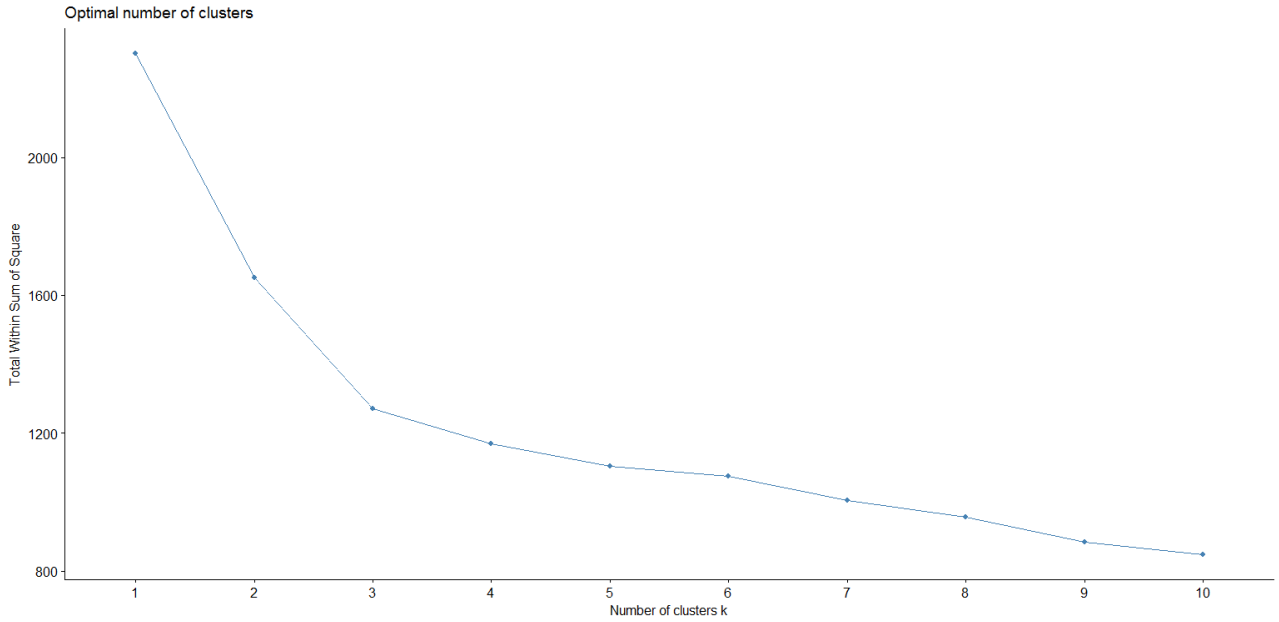


Figure 4.6: Elbow Method on Wine Data Set

In Figure 4.7 we can see that three clusters have been suggested for *Bivariate* dataset.

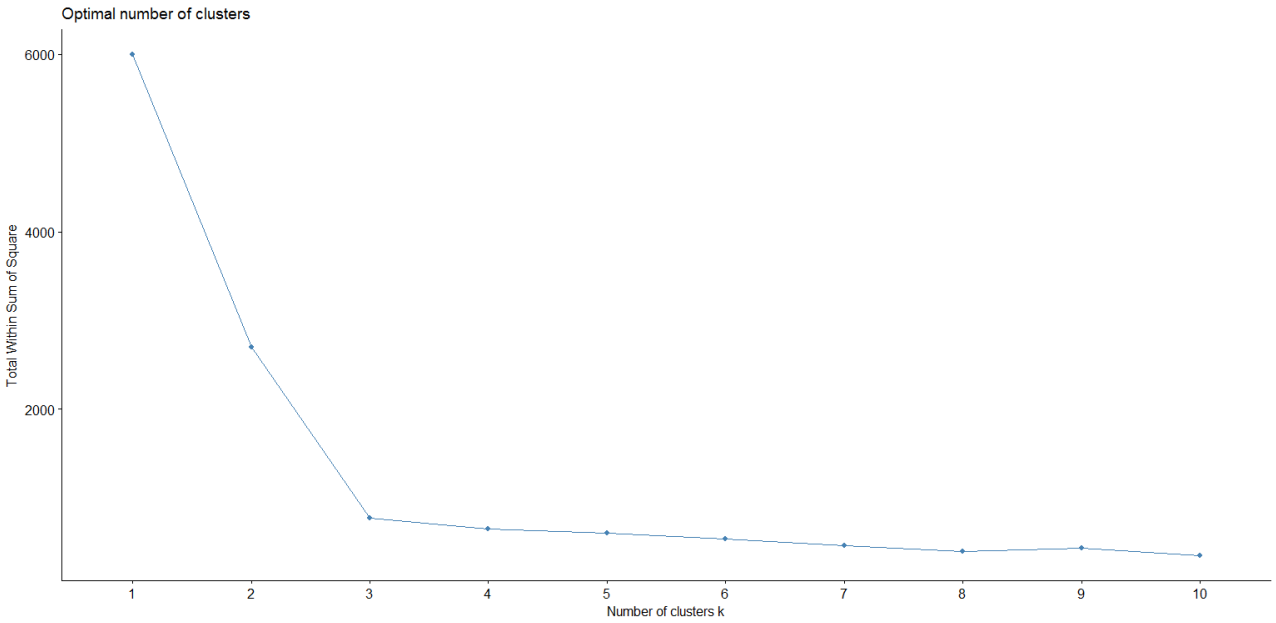


Figure 4.7: Elbow Method on Bivariate Data Set

- **Average Silhoutte Method** : Average silhouette method computes the average silhouette of observations for different values of K. The optimal number of clusters K is the one that maximize the average silhouette over a range of possible values for K. For each K, we calculate the average silhouette of all observations (avg.sil) and the we plot the curve of $avg.sil$ according to the number of clusters K. Finally, the location of the maximum is considered as the appropriate number of clusters.

  In Figure 4.8 we can see that two clusters have been suggested for *Iris* Data Set.

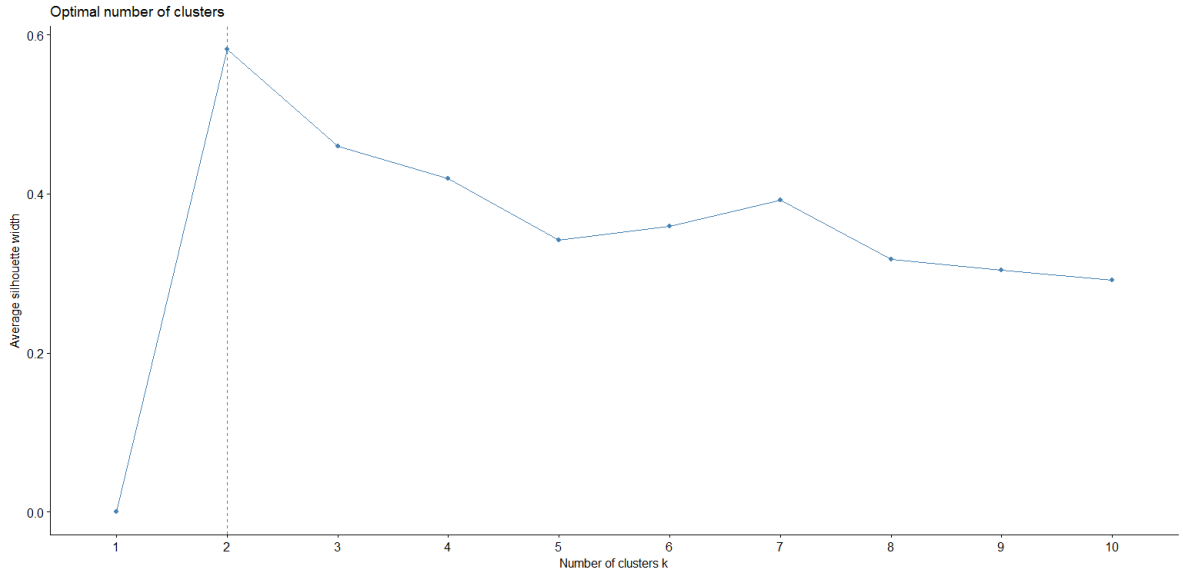


Figure 4.8: Average Silhouette Method on Iris Dataset

In Figure 4.9 we can see that three clusters have been suggested for *Isotopic Composition Plutonium Batches* Data Set.

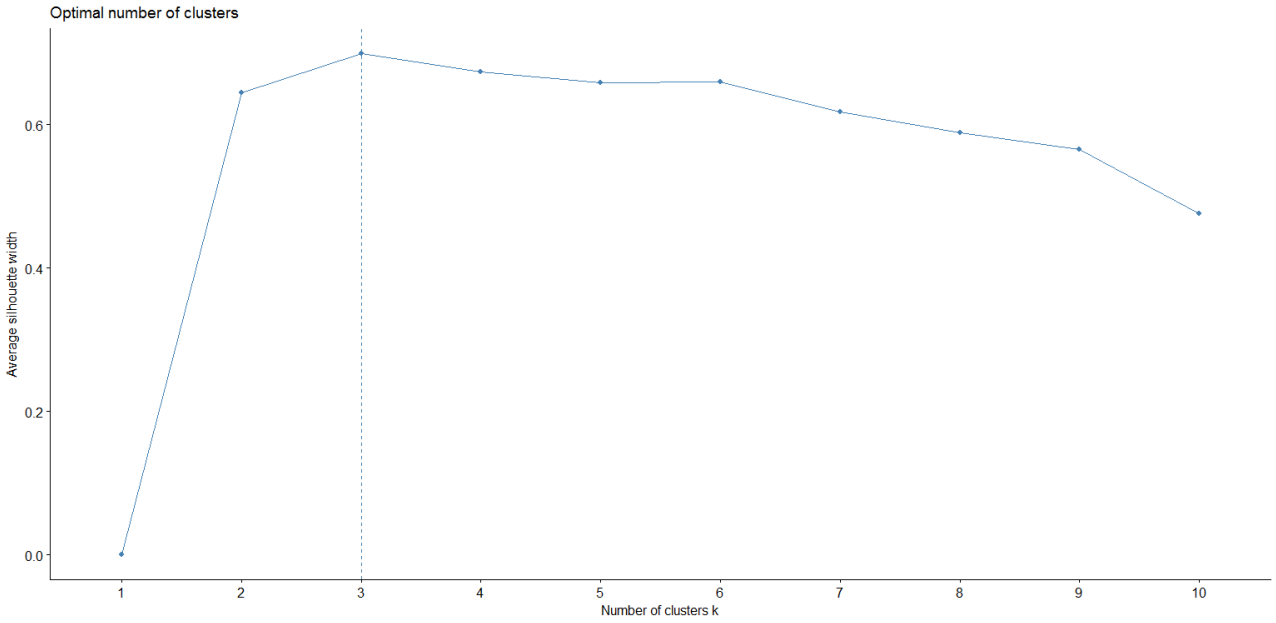


Figure 4.9: Average Silhouette Method on Isotopic Composition Plutonium Batches

In Figure 4.10 we can see that four clusters have been suggested for *Votes for Republican Candidate in Presidential Elections* Data Set.

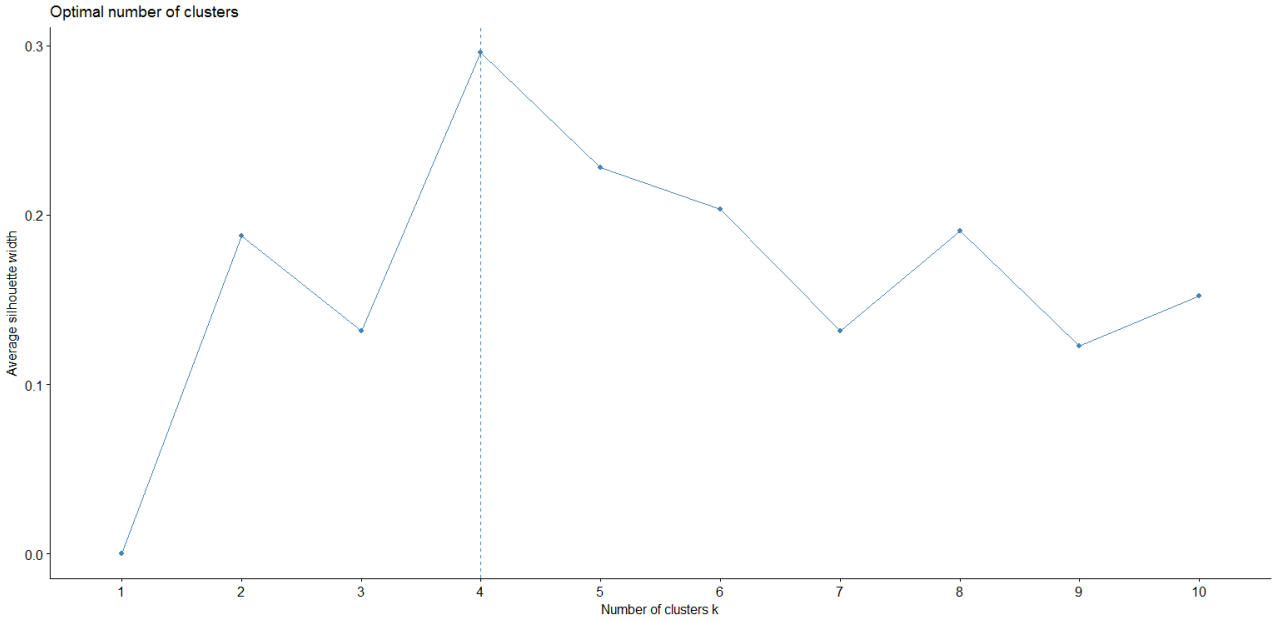


Figure 4.10: Average Silhouette Method on Votes for Republican Candidate in Presidential Elections

In Figure 4.11 we can see that six clusters have been suggested for *Ruspini* Data Set.

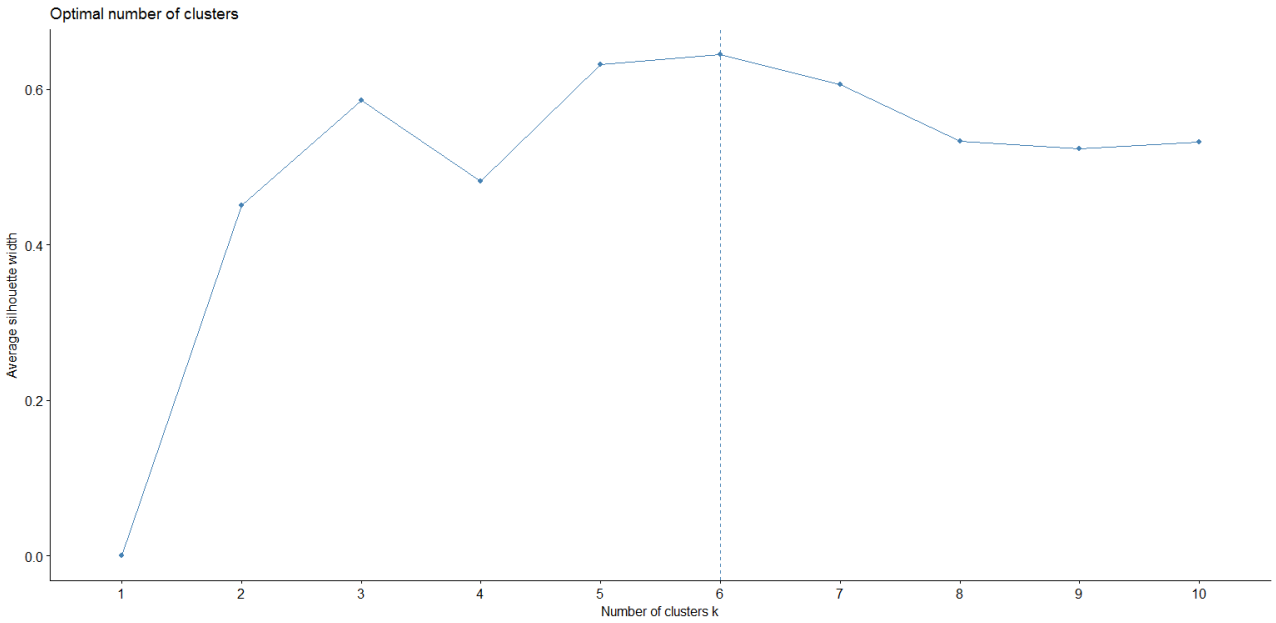


Figure 4.11: Average Silhouette Method on Ruspini Data Set

In Figure 4.12 we can see that two clusters have been suggested for *Violent Crime Rates by US State* Data Set.

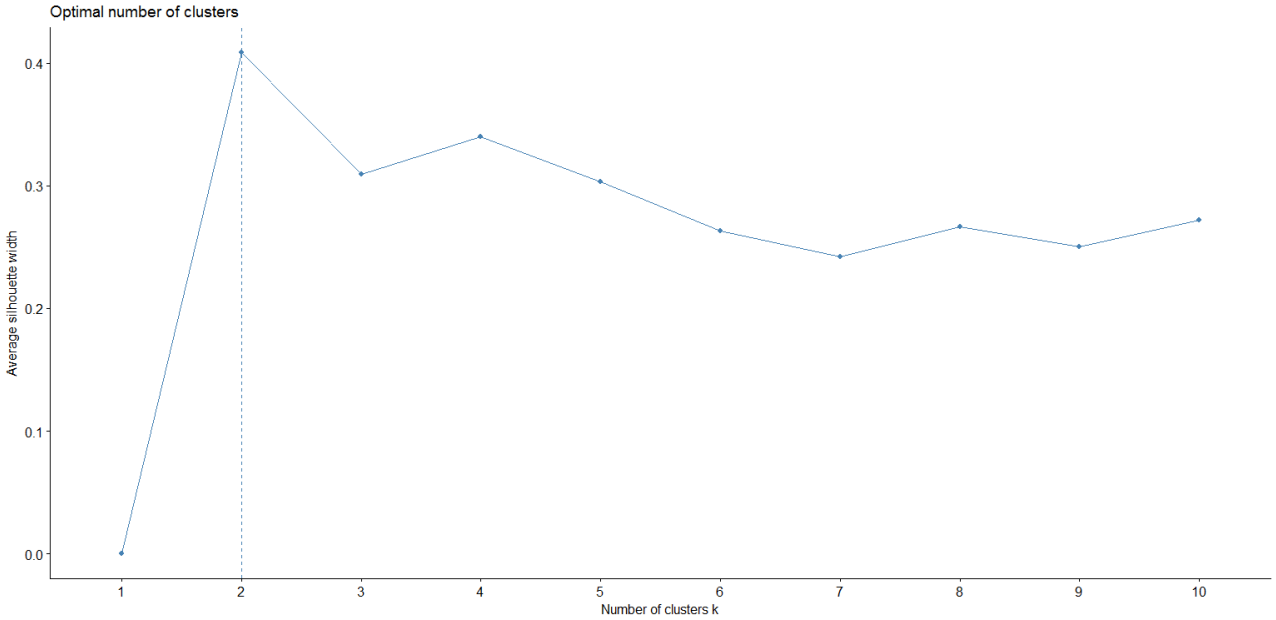


Figure 4.12: Average Silhouette Method on Violent Crime Rates by US State

In Figure 4.13 we can see that three clusters have been suggested for *Wine* Data Set.

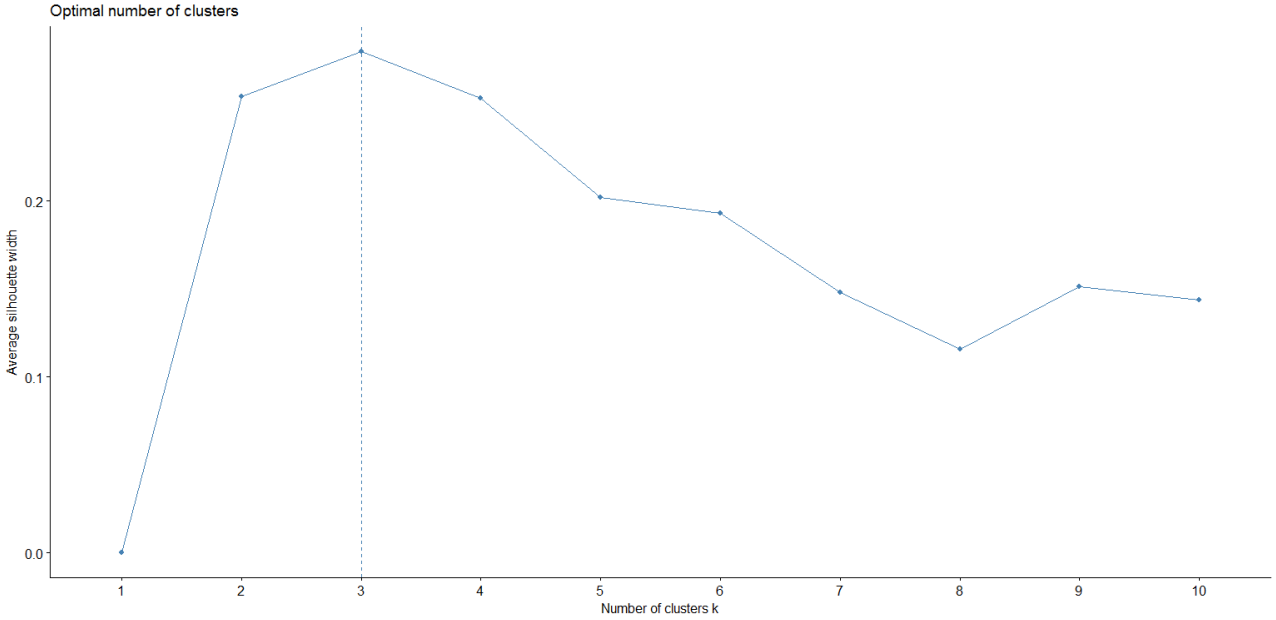


Figure 4.13: Average Silhouette Method on Wine Data Set

In Figure 4.14 we can see that three clusters have been suggested for *Bivariate* Data Set.

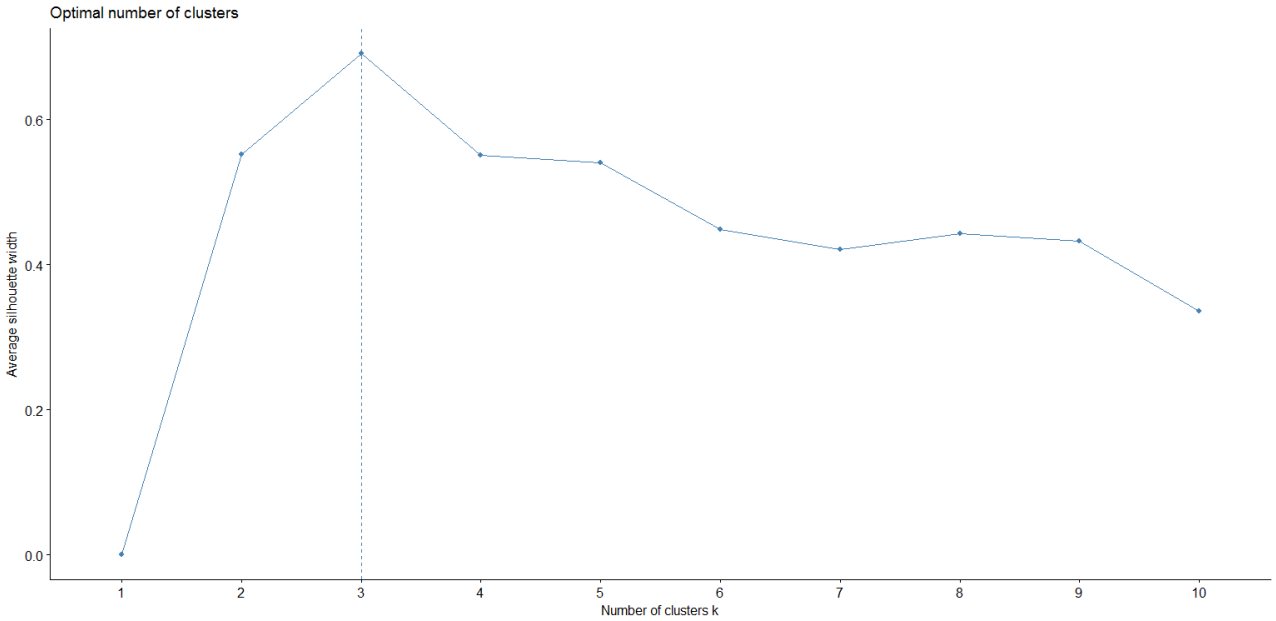


Figure 4.14: Average Silhouette Method on Bivariate Data Set

- **Gap Statistic Method** : The gap statistic compares the total within intracluster variation for different values of k with their expected values under null reference distribution of the data, i.e. a distribution with no obvious clustering. The reference dataset is generated using *Monte Carlo* Simulations of the sampling process. That is, for each variable (x_i) in the data set we compute its range $[min(x_i), max(x_j)]$ and generate values for the n points uniformly from the interval min to max. For the observed data and the the reference data, the total intracluster variation is computed using different values of k. The gap statistic for a given k is defined as follow:

  $Gap_n(k) = E_n^*\{\log W_k\} - \log W_k$

  In Figure 4.15 we can see that three clusters have been suggested for *Iris* Data Set.

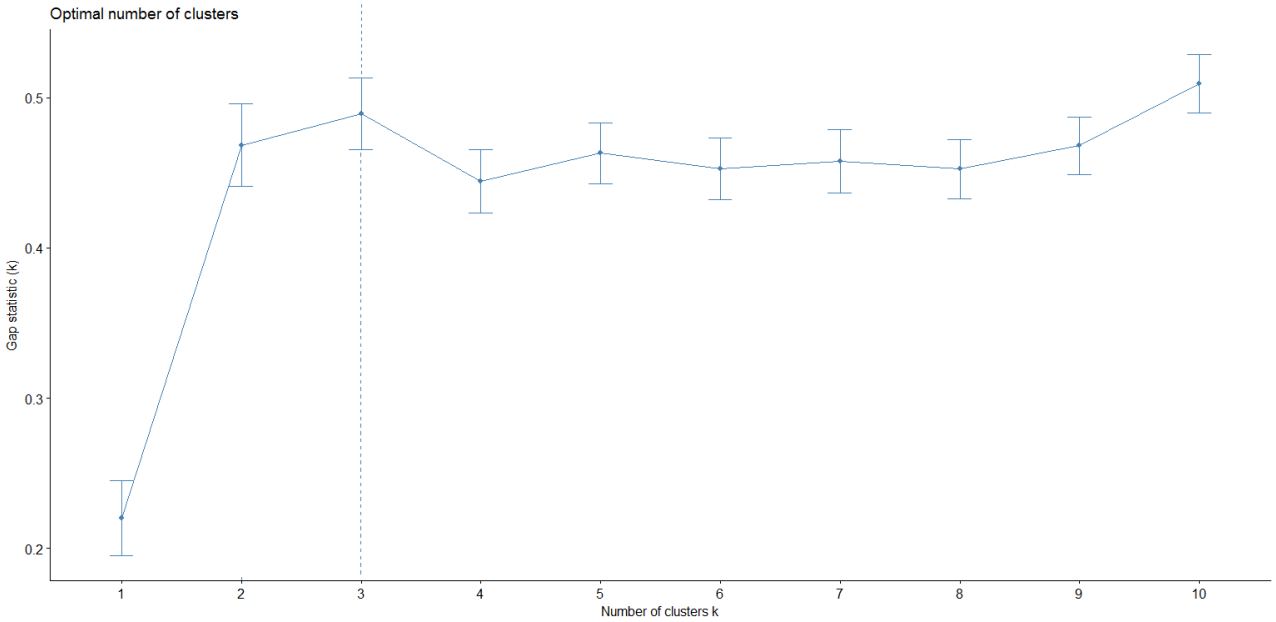


Figure 4.15: Gap Statistic Method on Iris Dataset

In Figure 4.16 we can see that seven clusters have been suggested for *Isotopic Composition Plutonium Batches* Data Set.

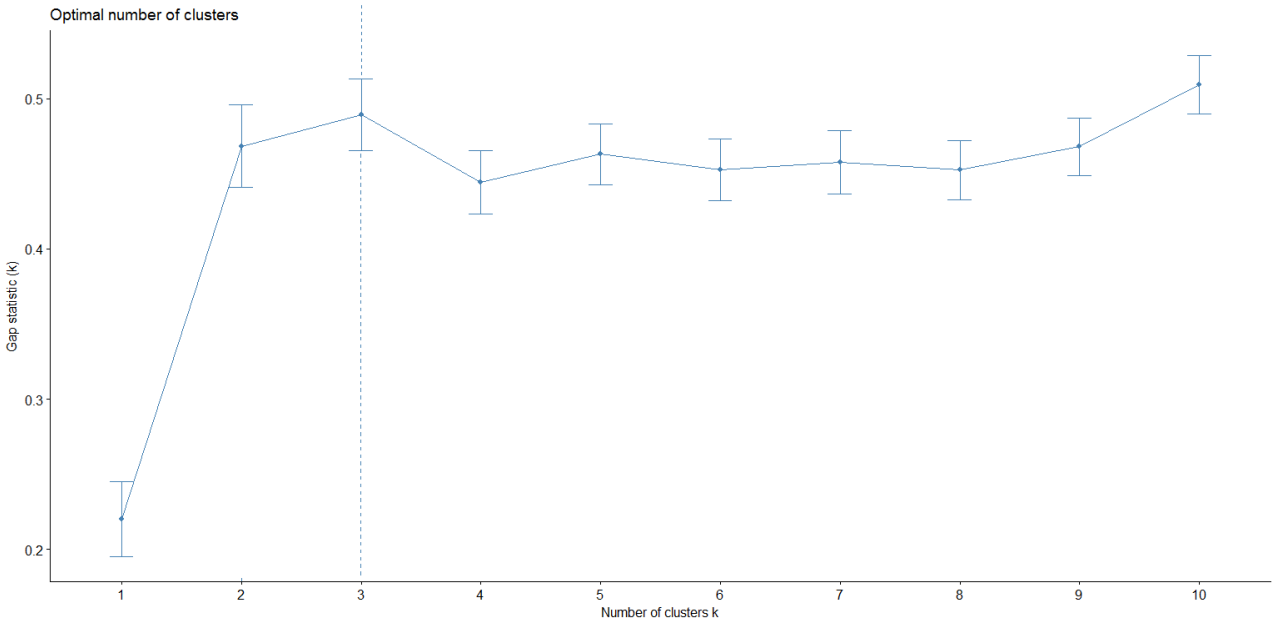


Figure 4.16: Gap Statistic Method on Isotopic Composition Plutonium Batches

In Figure 4.17 we can see that one cluster have been suggested for *Republican Candidate in Presidential Elections* Data Set.

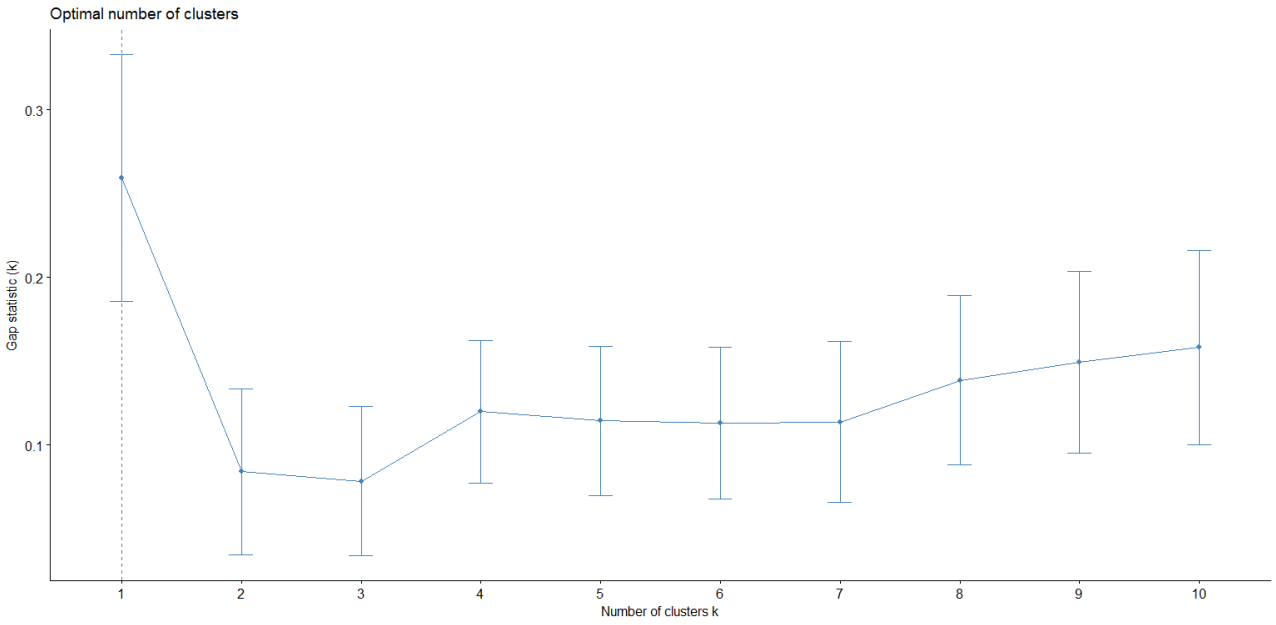


Figure 4.17: Gap Statistic Method on Republican Candidate in Presidential Elections

In Figure 4.18 we can see that four clusters have been suggested for *Ruspini* Data Set.

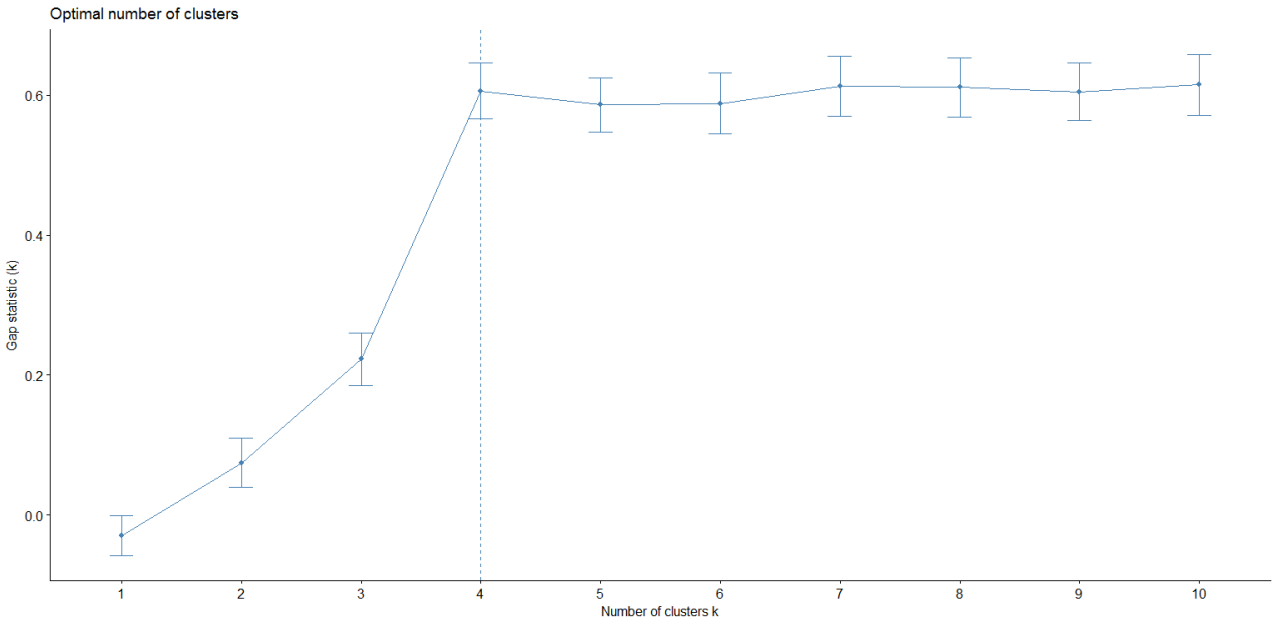


Figure 4.18: Gap Statistic Method on Ruspini

In Figure 4.19 we can see that four clusters have been suggested for *Violent Crime Rates by US State* Data Set.

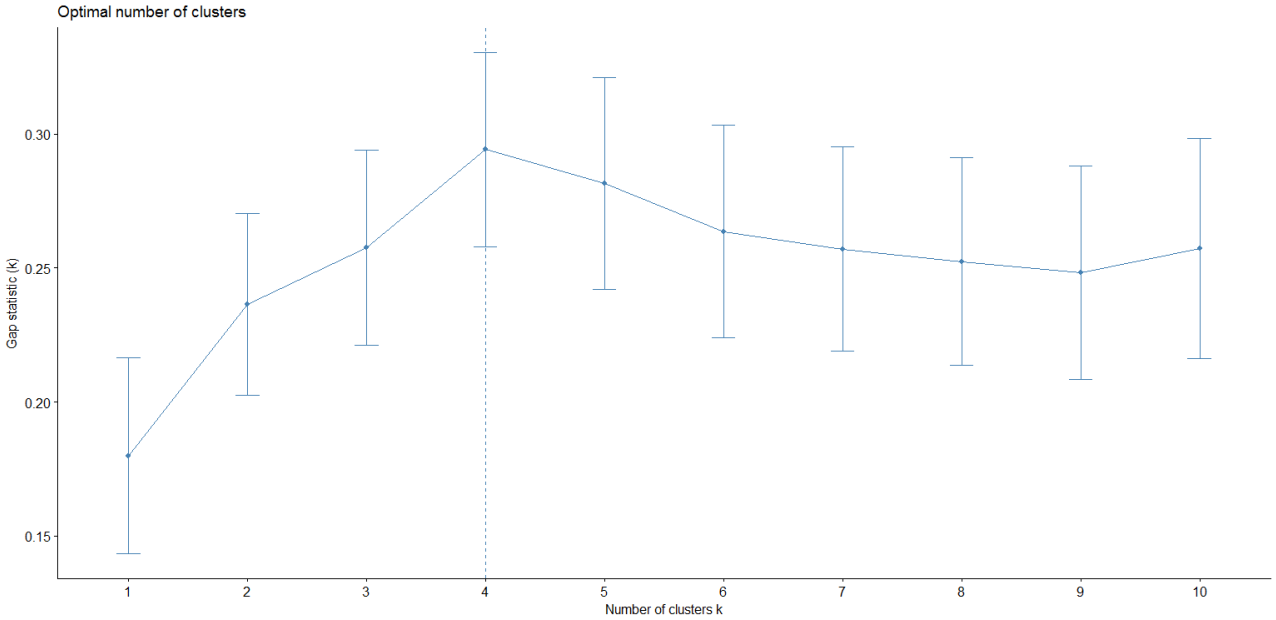


Figure 4.19: Gap Statistic Method on Violent Crime Rates by US State

In Figure 4.20 we can see that three clusters have been suggested for *Wine* Data Set.

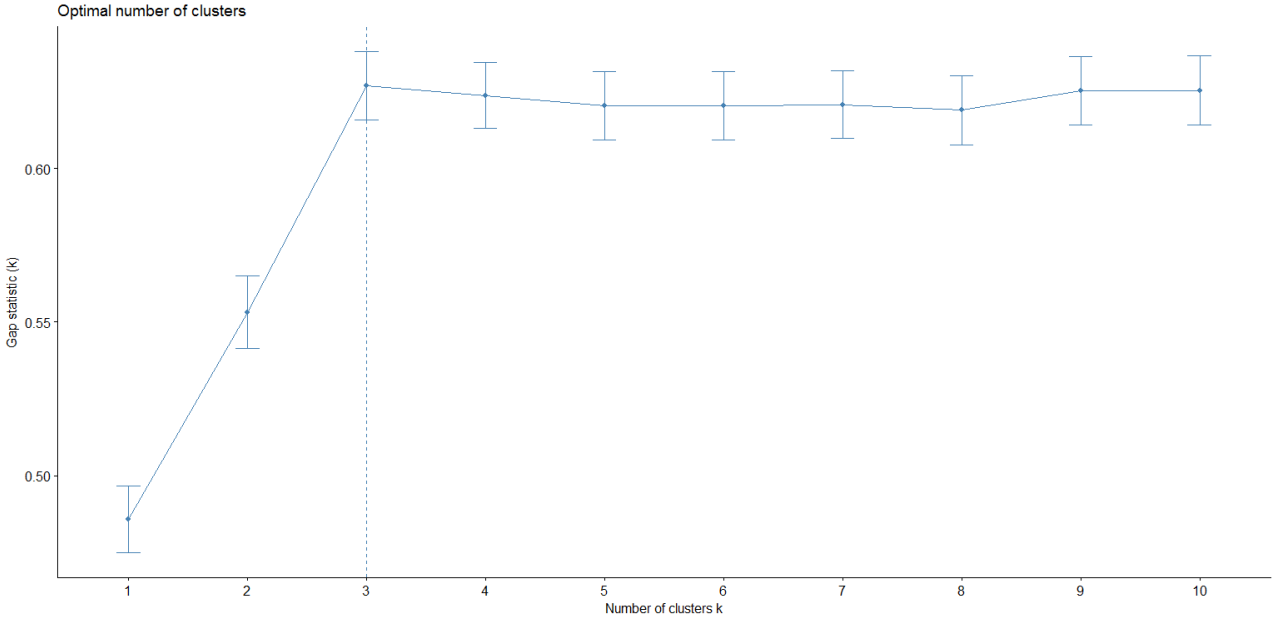


Figure 4.20: Gap Statistic Method on Wine Data Set

In Figure 4.21 we can see that three clusters have been suggested for *Bivariate* Data Set.

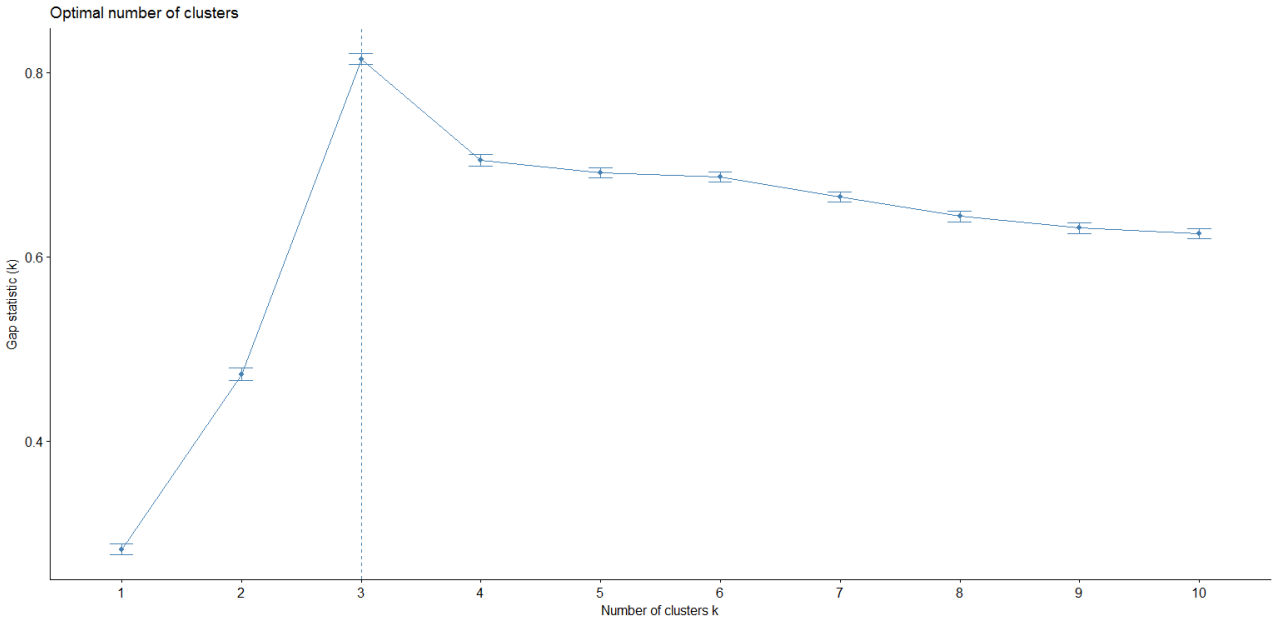


Figure 4.21: Gap Statistic Method on Bivariate Data Set

Now we can make a comparison table to see how different methods are working on different datasets. In Table 4.1 we can see original cluster number and output of three methods accordingly.

Table 4.1: Table to compare outputs of different methods

Data Set	Original Clusters	Elbow Method	Average Silhouette Method	Gap Statistic Method
Iris Data Set	3	3	2	3
Bivariate Data Set	3	3	3	3
Violent Crime Rates by US State	4	4	2	4
Ruspini Data Set	4	4	6	4
Isotopic Composition Plutonium Batches	3	3	3	7
Republican Candidate in Presidential Elections	4	4	4	1
Wine Data Set	3	3	3	3

4.5 Discussion

The problem of estimating the number of clusters in a data set is difficult, underlined by the fact that there is no clear definition of a 'cluster'. Hence, in data that are not clearly separated into groups, different people might have different opinions about the number of distinct clusters. In this paper, we have focused on well-separated clusters and have proposed the gap statistic for estimating the number of groups. When used with a uniform reference distribution in the principal component orientation, it outperforms other proposed methods from the literature in our simulations. The simpler uniform reference (over the range of the data) works well except when the data lie near a subspace.

In this chapter we have discussed about our different methods of determining the number of optimal clusters in a dataset. And we have compared the results of different methods with the original result. In **Elbow Method** the problem is: This "elbow" cannot always be unambiguously identified. Sometimes there is no elbow, or several elbows can be detected. So, then we'll need to run **Average Silhouette Method**. The average $s(i)$ over all data of a cluster is a measure of how tightly grouped all the data in the cluster are. Thus the average $s(i)$ over all data of the entire dataset is a measure of how appropriately the data have been clustered. If there are too many or too few clusters, as may occur when a poor choice of K is used in the clustering algorithm (e.g.: K-means), some of the clusters will typically display much narrower silhouettes than the rest. The simulation studies suggest that the gap estimate is good at identifying well-separated clusters. When data are not well separated, the notion of a cluster is not any more well defined in the literature.

Chapter 5

Conclusion and Future Work

5.1 Introduction

In this beautiful and mysterious world there is nothing as to be final and accurate or flawless. There must be some pitfalls or crevices to everything. This is the ultimate rule and this is the inherent daintiness. There was a time when everyone believes that mass and time are real constant and cannot have different value at different circumstances in the era of Newton. Then there comes Einstein with his great mind and changed the way of thinking of whole world. He had shown how these very obvious things can be realative in different context of speed. Once upon a time the shape of the mother earth is considered as a flat land and center of our solar system. But now what we have learnt by the great minds of our history is the rounding shape of the mother earth and nothing but a tiny planet of the galaxy. We believe all these things without seeing the fact with our own eyes because of mathmetical proof. Once Albert Einstein said that Pure mathematics is, in its way, the poetry of logical ideas.

It is also true about our improvement of the determining the number of clusters. There have so many different methods exist. Different minds have different directions of view. We have tried to highlight the three major methods of finding K for K-means algorithm.

5.2 Conclusion

Existing methods of selecting the number of clusters for K-means clustering have a number of drawbacks. Also, current methods for assessing the clustering results do not provide much information on the performance of the clustering algorithm. Three methods to select the number of clusters for the K-means algorithm have been reviewed in this work. But we did not propose any new method or improvement of existing methods.

5.3 Future Work

In future we like to add some more datasets in our simulation process and compare them with more different methods of selecting number of clusters for K-means algorithm. And thus we hope that we can propose a new way of finding number of clusters efficiently and optimally for any given dataset.

References

[1] R.C. de Amorim, C. Hennig., "Recovering the number of clusters in data sets with noise features using feature rescaling factors," *Information Sciences*, 2015.

[2] P. J. Rousseeuw., "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, 1987.

[3] L. Kaufman and P. J. Rousseeuw., "Finding Groups in Data. An Introduction to Cluster Analysis," *Wiley-Interscience, New York*, 1990.

[4] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, "Machine learning: An artificial intelligence approach," *Springer Science and Business Media*, 2013.

[5] R. O. Duda and P. E. Hart., "Pattern Classification and Scene Analysis," *Communications in Statistics*, 1974.

[6] G. W. Milligan and M. C. Cooper. A, "Examination of procedures for determining the number of clusters in a data set.," *Wiley, New York*, 1973.

[7] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics*, 1974.

[8] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a dataset via the gap statistic.," *Standford University*, 2001.

[9] D T Pham, S S Dimov, and C D Nguyen, "Selection of K in K-means clustering.," *Manufacturing Engineering Centre, Cardiff University*, 2004.

[10] Han, J. and Kamber, M., "Data Mining: Concepts and Techniques.," *Morgan Kaufmann, San Francisco, California*, 2000.

[11] Al-Daoud, M. B., Venkateswarlu, N. B., and Roberts, S. A., "Fast K-means clustering algorithms.," *School of Computer Studies, University of Leeds*, 1995.

[12] Al-Daoud, M. B., Venkateswarlu, N. B., and Roberts, S. A., "New methods for the initialisation of clusters.," *Pattern Recognition Lett.*, 1996.

[13] Alsabti, K., Ranka, S., and Singh, V., "An efficient K-means clustering algorithm.," *In Proceedings of the First Workshop on High-Performance Data Mining, Orlando, Florida*, 1998.

[14] Bilmes, J., Vahdat, A., Hsu, W., and Im, E. J. "Empirical observations of probabilistic heuristics for the clustering problem.," *International Computer Science Institute, Berkeley, California.*, 1997.

[15] Bottou, L. and Bengio, Y. "Convergence properties of the K-means algorithm.," *Adv. Neural Infn Processing Systems.*, 1995.

[16] Kerr, A., Hall, H. K., and Kozub, S. "Doing Statistics with SPSS.," *Sage, London.*, 2002.

[17] Hansen, L. K. and Larsen, J. "Unsupervised learning and generalisation.," *In Proceedings of the IEEE International Conference on Neural Networks*, 1996.

[18] Bradley, S. and Fayyad, U. M. "Refining initial points for K-means clustering.," *In Proceedings of the Fifteenth International Conference on Machine Learning, ICML'98*, 1998.

[19] Ishioka, T. Extended "K-means with an efficient estimation of the number of clusters.," *In Proceedings of the Second International Conference on Intelligent Data Engineering and Automated Learning.*, 2000.

Index

Association Problem, 11
Average Linkage Clustering, 4
Average Silhouette Method, 19

Bayesian Information Criterion, 7
Biology, 2

Centroid Based Clustering, 4, 12
Classification Problem, 11
Cluster, 1
Cluster Analysis, 12
Clustering, 1
Clustering Problem, 11
Complete Linkage Clustering, 4
Computer Vision, 10
Connectivity Based Clustering, 4

Data Mining, 4
DBSCAN, 6
Dendrogram, 3
Density Attractors, 6
Density Based Clustering, 5
Density Link Clustering, 6
Distance Functions, 4
Distribution Based Clustering, 5
Divisive Clustering, 4

Elbow Index, 25
Elbow Method, 18
Euclidean Distance, 13, 20

fuzzy C-means, 4

Gap Index, 26
Gap Statistic Method, 21
Gaussian Mixture Models, 5

Hierarchical Clustering, 3

Insurance, 2

K-means, 4
K-medoids, 4

Libraries, 2
Local Maxima, 6

Machine Learning, 10
Manhattan Distance, 13, 20
Marketing, 2
Mean Shift, 6
Monte Carlo, 8

Natural Clusters, 2
Null Hypothesis, 8

OCR, 10
OPTICS, 6
Optimization Problem, 4
Overfitting, 5

Regression Problem, 11
Reinforcement Learning, 12

Semi-supervised Learning, 11
Silhouette, 20
Silhouette Index, 25
Single Linkage Clustering, 4
SSE, 15
Supervised Classification, 12

Supervised Learning, 10

Unsupervised Classification, 12
Unsupervised Learning, 11
UPGMA, 4

Uraban Planning, 2

World Wide Web, 2
WSS, 21, 26

Appendix A

Codes

A.1 R-Code of Elbow Method for k-means clustering

```
set.seed(123)
# Compute and plot wss for k = 2 to k = 15
k.max <- 15 # Maximal number of clusters
data <- iris.scaled
wss <- sapply(1:k.max,
         function(k){kmeans(data, k, nstart=10 )$tot.withinss})
plot(1:k.max, wss,
       type="b", pch = 19, frame = FALSE,
       xlab="Number of clusters K",
       ylab="Total within-clusters sum of squares")
abline(v = 3, lty =2)

fviz_nbclust(x, FUNcluster, method = c("silhouette", "wss"))

fviz_nbclust(iris.scaled, kmeans, method = "wss") +
    geom_vline(xintercept = 3, linetype = 2)
```

A.2 R-Code of Average Silhouette Method for k-means clustering

```
library(cluster)
k.max <- 15
data <- iris.scaled
```

```
sil <- rep(0, k.max)
# Compute the average silhouette width for
# k = 2 to k = 15
for(i in 2:k.max){
  km.res <- kmeans(data, centers = i, nstart = 25)
  ss <- silhouette(km.res$cluster, dist(data))
  sil[i] <- mean(ss[, 3])
}
# Plot the  average silhouette width
plot(1:k.max, sil, type = "b", pch = 19,
     frame = FALSE, xlab = "Number␣of␣clusters␣k")
abline(v = which.max(sil), lty = 2)

require(cluster)
fviz_nbclust(iris.scaled, kmeans, method = "silhouette")
```

A.3 R-Code of Gap Statistic Method for k-means clustering

```
# Compute gap statistic
library(cluster)
set.seed(123)
gap_stat <- clusGap(iris.scaled, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
# Print the result
print(gap_stat, method = "firstmax")

# Base plot of gap statistic
plot(gap_stat, frame = FALSE, xlab = "Number␣of␣clusters␣k")
abline(v = 3, lty = 2)

# Use factoextra
fviz_gap_stat(gap_stat)

# Print
print(gap_stat, method = "Tibs2001SEmax")
# Plot
fviz_gap_stat(gap_stat,
              maxSE = list(method = "Tibs2001SEmax"))
# Relaxed the gap test to be within two standard deviations
```

```
22 fviz_gap_stat(gap_stat,
23            maxSE = list(method = "Tibs2001SEmax", SE.factor = 2))
```

Generated using Undegraduate Thesis LaTeX Template, Version 1.3. Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh.

This thesis was generated on Thursday 7th September, 2017 at 11:55am.