St. Xavier's College [Autonomous], Kolkata

CUSTOMER SEGMENTATION

By

Meghna Mondal (526),
Tanushree Sarkar (529),
Kwanan Mondal (553)

Under Guidance of
Prof. Debabrata Datta

Submitted to the Department of Computer Science in partial fulfilment of the requirements for the degree of B.Sc. Computer Science

CERTIFICATE OF AUTHENTICATED WORK

This is to certify that the project report entitled " CUSTOMER SEGMENTATION " submitted to Department of Computer Science, ST. XAVIER'S COLLEGE [AUTONOMOUS], KOLKATA, in partial fulfilment of the requirement for the award of the Bachelor of Science(B.Sc.) is an original work carried out by:

Name	Roll No.	Registration No.
TANUSHREE SARKAR	529	A01-2122-0907-20
KWANAN MONDAL	553	A01-2122-0923-20
MEGHNA MONDAL	526	A01-2122-0904-20

The matter embodied in this project is authentic and is genuine work done by the student and has not been submitted whether to this College or to any other Institute for the fulfilment of the requirement of any course of study.

...…… ...……

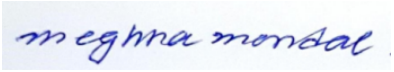
Signature of the Supervisor Signature of the Head of the Department

……..……

Name of the Supervisor

Date: …….. Date:……

Name and Signature of the **Project Team members**:

Name	**Signature**
1.TANUSHREE SARKAR	 ……..
2.KWANAN MONDAL	 ……..
2.MEGHNA MONDAL	 ……..

IMPLEMENTATION RESPOSIBILITIES

ROLL	NAME	RESPONSIBILITY
553	Kwanan Mondal	Result and Analysis
		Workflow diagram and Algorithm Formation
		Documentation
529	Tanushree Sarkar	Coding Details
		Background literature Study
		Documentation
526	Meghna Mondal	Coding
		Initial and K-Means Algorithm Background Study
		Documentation

ABSTRACT

In today's highly competitive business environment, understanding customers' behaviours and preferences has become a critical success factor for businesses. The ability to segment customers into distinct groups with similar characteristics enables businesses to create targeted marketing campaigns and improve customer engagement. However, traditional segmentation methods often rely on simple rules or intuition and may not effectively capture complex customer behaviour patterns.

In recent years, machine learning techniques have emerged as a powerful tool for customer segmentation. Machine learning algorithms can process large amounts of data, identify complex patterns, and provide more accurate and consistent results. Among the various machine learning algorithms, the K-means algorithm is a popular unsupervised clustering technique used for customer segmentation. The elbow method is a technique used to determine the optimal number of clusters in the data and is commonly used with the K-means algorithm.

In this research paper, we explore the application of the K-means algorithm and the elbow method in customer segmentation using machine learning techniques. We apply the algorithm to a customer dataset and evaluate the effectiveness of the resulting customer segments. We compare the performance of K-means with other clustering techniques and analyse the impact of different input parameters on the segmentation results.

The findings of this study can help businesses to improve their marketing strategies by targeting specific customer segments with customized marketing campaigns. This research demonstrates the effectiveness of machine learning techniques in customer segmentation and highlights the advantages of using K-means and elbow method for this task. Overall, this study contributes to the growing body of knowledge on the application of machine learning in marketing and provides valuable insights for businesses seeking to improve customer engagement and increase revenue.

KEYWORDS: Customer Segmentation, Unsupervised Machine learning, Clustering, K-Means Algorithm, Elbow Graph, Silhouette Graph, Behavioral Segmentation, Geographical Segmentation.

ACKNOWLEDGEMENT

We would like to take this occasion to express our heartfelt gratitude to everyone who has helped and guided us throughout the development and completion of this project.

We thank the Head of the Department of Computer Science, for offering us an opportunity to work on such a project and thereby exposing us to the beauty and intricacies of machine learning along with developing knowledge about the latest technologies employed for building recommendation engines.

We thank our project mentor and guide, Prof. Debabrata Datta, for guiding us through the process of the background study, analyses, debugging, coding and implementation and for explaining major concepts in a lucid manner.

We would like to immensely thank all the teaching and non-teaching staff of the Department of Computer Science for their untiring support and help without whom the project would never have been completed.

INDEX

CHAPTER 1: INTRODUCTION

The advent of internet connectivity has brought about a significant transformation in the way businesses operate. Online businesses have grown exponentially, creating a highly competitive market space. As the number of competitors continues to rise, it has become increasingly challenging for businesses to stand out and succeed in the online space. In this highly competitive business landscape, the need to gain strategic insights from large and complex data sets has become paramount. This has led to the widespread adoption of data mining techniques, which are designed to extract valuable information from databases and present it in a format that is easy to understand and use for decision making [2].

Data mining involves the use of advanced statistical, artificial intelligence, machine learning, and database technologies to analyse large data sets and identify patterns[4], relationships, and trends that would be difficult or impossible to detect using traditional data analysis techniques[3]. By leveraging these techniques, organizations can uncover valuable insights about their customers, competitors, and markets, and use this information to gain a competitive edge[6].

Some of the key benefits of data mining include improved decision making, enhanced customer understanding, increased operational efficiency, and better risk management. However, successful data mining requires a combination of advanced technical expertise, strong business acumen, and a deep understanding of the specific industry and market dynamics at play.

Machine learning lets us take advantage of datasets and target our communication better. Save time while being more accurate. Machine learning methods are great for finding patterns and insights in customer data. Artificially intelligent models are powerful tools for decision-makers. They can precisely identify customer segments, which is much harder and takes longer when done manually or with conventional analytical methods.

In today's fast-paced and data-driven world, businesses are constantly seeking innovative ways to understand their customers' behaviour and preferences. One such technique is customer segmentation.

Customer segmentation is the subdivision of a business customer base into groups called customer segments such that each customer segment consists of customers who share similar market characteristics[1]. This segmentation is based on factors that can directly or indirectly influence market or business such as products preferences or expectations, locations, behaviours and so on. The importance of customer segmentation include, inter alia, the ability of a business to customise market programs that will be suitable for each of its customer segments; business decision support in terms of risky situation such as credit relationship with its

customers; identification of products associated with each segments and how to manage the forces of demand and supply; unravelling some latent dependencies and associations amongst customers, amongst products, or between customers and products which the business may not be aware of; ability to predict customer defection, and which customers are most likely to defect; and raising further market research questions as well as providing directions to finding the solutions[12].

Here are some of the different types of customer segmentation[7]:

i. Demographic Segmentation: This type of segmentation is based on customer characteristics such as age, gender, income, education, occupation, and family status. It is commonly used in marketing and advertising to create targeted messaging.

ii. Geographic Segmentation: This type of segmentation is based on geographic location, such as region, city, climate, or even neighbourhood. It is useful for businesses with a physical presence, as they can tailor their offerings and promotions to the needs and preferences of customers in specific areas.

iii. Psychographic Segmentation: This type of segmentation is based on customer personality, values, attitudes, and lifestyles. It is useful for understanding customer behaviours and preferences, as well as developing marketing strategies that resonate with specific customer groups.

iv. Behavioural Segmentation: This type of segmentation is based on customer behaviour, such as buying habits, usage patterns, loyalty, and engagement. It is useful for predicting future behaviour and identifying opportunities to increase customer retention and loyalty.

v. Firmographic Segmentation: This type of segmentation is used in B2B marketing and is based on the characteristics of the customer's business, such as size, industry, revenue, and location. It is useful for tailoring marketing strategies to specific industries or businesses.

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

The key idea behind machine learning is to train a model on a set of input data, which can include text, images, audio, or other types of data, in order to learn patterns or relationships that can be used to make predictions or decisions. The model is typically designed to optimize some objective function, such as accuracy, precision, or recall, based on a set of labelled or unlabelled data.

There are several different types of machine learning algorithms, including supervised learning, unsupervised learning, semi-supervised learning, reinforcement learning, and deep learning. Each of these approaches has its own strengths and weaknesses and can be used for different types of tasks and applications.

i. Supervised Learning: Supervised learning involves training a machine learning model on a labelled dataset, where each data point has an associated label or output value. The goal of supervised learning is to learn a mapping between the input features and the output labels, so that the model can accurately predict the output values for new, unseen data. Examples of supervised learning algorithms include linear regression, decision trees, and neural networks.

ii. Unsupervised Learning: Unsupervised learning involves training a machine learning model on an unlabelled dataset[6], where there are no output labels. The goal of unsupervised learning is to discover patterns or structures in the data, such as clusters or associations. Unsupervised learning is often used for tasks such as anomaly detection, dimensionality reduction, and data visualization. Examples of unsupervised learning algorithms include k-means clustering[2], principal component analysis (PCA), and autoencoders.

iii. Reinforcement Learning: Reinforcement learning is a type of machine learning in which a model learns by interacting with an environment to achieve a goal. The goal of reinforcement learning is to learn a policy that maximizes a cumulative reward signal over time. Reinforcement learning is often used in applications such as robotics, game playing, and autonomous agents. Examples of reinforcement learning algorithms include Q-learning and policy gradients.

The key difference between supervised and unsupervised learning is the presence or absence of labelled data. Supervised learning is used when the goal is to predict a specific output value based on input features, while unsupervised learning is used when the goal is to explore and discover patterns or structures in the data. In some cases, a combination of both supervised and unsupervised learning may be used, such as in semi-supervised learning, where a small amount of labelled data is used to improve the performance of an unsupervised learning model. Reinforcement learning is used when the goal is to learn an optimal policy for interacting with an environment to achieve a desired outcome.

Machine learning, specifically clustering algorithms, has emerged as a powerful tool for customer segmentation, enabling businesses to identify patterns and insights that may have gone unnoticed using traditional methods.

Clustering has been proven effective to implement customer segmentation[8]. It is a task of dividing the data sets into a certain number of clusters in such a manner that the data points belonging to a cluster have similar characteristics. Clusters are nothing but the grouping of data points such that the distance between the data points within the clusters is minimal [8]. The goal of clustering is to identify patterns and structure in the data that may not be apparent by simply examining it. Clustering comes under unsupervised learning, having ability to find clusters over unlabelled dataset.

There are several types of clustering techniques, including:

i. Hierarchical Clustering: In this technique, the data is divided into a tree-like structure, called a dendrogram, which represents the relationships between the data points. The two main types of hierarchical clustering are agglomerative (bottom-up) and divisive (top-down).

ii. K-Means Clustering: K- means algorithm is one of the most popular partitioning clustering algorithm [2]. This technique involves partitioning the data into k clusters, where k is a predefined number. The goal is to minimize the sum of squared distances between each data point and the centroid of its cluster.

iii. Density-Based Clustering: In this technique, clusters are formed based on the density of data points in a given region. Data points that are located in dense areas are grouped together, while those in sparse areas are considered noise or outliers.

iv. Fuzzy Clustering: This technique allows data points to belong to multiple clusters with varying degrees of membership. It is useful when there is overlap between clusters or when data points are ambiguous.

v. Spectral Clustering: This technique uses the eigenvalues and eigenvectors of a similarity matrix to partition the data into clusters. It is particularly useful for non-linearly separable data.

In clustering, distance measures play a crucial role in determining the similarity or dissimilarity between data points. The choice of distance measure depends on the type of data being analysed and the specific goals of the clustering analysis. Here are some commonly used distance measures in clustering:

i. Euclidean distance: It is the most commonly used distance measure and calculates the straight-line distance between two data points in n-

dimensional space[5]. This measure is suitable for continuous data with a Euclidean geometry.

$$deuc(x, y) = \sqrt{\sum i=1n(xi-yi)2} \quad -(i)$$

ii. Manhattan distance: Also known as the city-block distance or L1 distance, it calculates the distance between two data points by summing up the absolute differences between their coordinates. This measure is useful for data with a non-Euclidean geometry.

$$dman(x, y) = \sum i=1n|(xi-yi) \quad -(ii)$$

Where, x and y are two vectors of length n.

iii. Cosine similarity: It measures the angle between two vectors in n-dimensional space, which indicates the degree of similarity between them. This measure is commonly used for text and image data analysis.

$$deisen(x, y) = 1-(/\sum i=1nxiyi/) / (\sqrt{\sum i=1nx2i\sum i=1ny2i}) \quad -(iii)$$

The choice of distance measure should be based on the type of data being analysed, the clustering algorithm being used, and the specific goals of the analysis. It is important to choose a distance measure that is appropriate for the data to ensure that the clustering results are meaningful and useful.

The elbow method and silhouette method are both used to evaluate the quality of clustering results.

- The elbow method [2]is a technique used to determine the optimal number of clusters in a dataset. It involves plotting the within-cluster sum of squares (WCSS) against the number of clusters. The WCSS measures the sum of the squared distances between each data point and its closest centroid (the centre of the cluster it belongs to). The idea is that as the number of clusters increases, the WCSS will decrease, but at some point, the rate of decrease will start to level off, forming an elbow shape in the graph. The number of clusters corresponding to the elbow point is considered the optimal number of clusters for the dataset.

- The silhouette method is another technique used to evaluate the quality of clustering results. It involves computing the silhouette coefficient for each data point, which measures how similar it is to its own cluster compared to other clusters. The silhouette coefficient ranges from -1 to 1, where a value close to 1 indicates that the point is well-clustered, a value close to 0

indicates that the point is close to the decision boundary between clusters, and a value close to -1 indicates that the point may be assigned to the wrong cluster. The average silhouette coefficient across all data points is used as a measure of the overall quality of the clustering. A higher average silhouette coefficient indicates better clustering results.

Both the elbow method and the silhouette method can be used in combination to determine the optimal number of clusters for a dataset.

CHAPTER 2: BACKGROUND LITERATURE SURVEY

Several studies have explored the use of machine learning algorithms, including K-means clustering, for customer segmentation. Machine learning algorithms can identify patterns and relationships in customer data that traditional statistical methods may not detect, making them a powerful tool for customer segmentation. Customer segmentation is a crucial aspect of marketing and business strategy, as it helps companies understand their customers' behaviour, preferences, and needs. Machine learning techniques, particularly K-means clustering, have emerged as popular approaches to segmenting customers based on their characteristics and behaviour.

Jain, Murty, and Flynn's (1999) paper[16], "Data Clustering: A Review," provides a comprehensive survey of clustering algorithms, which are used for grouping data points into clusters or subgroups based on their similarity. The paper reviews over 400 research articles and categorizes clustering algorithms based on their underlying assumptions, types of data, and clustering criteria. The authors begin by defining clustering and explaining its importance in various domains, including pattern recognition, image analysis, data mining, and information retrieval. They then provide a taxonomy of clustering algorithms based on their characteristics and features. The taxonomy includes partitioning algorithms, hierarchical algorithms, density-based algorithms, grid-based algorithms, model-based algorithms, and others. The authors explain each algorithm's working principle, strengths, and weaknesses. The paper also highlights some common clustering criteria used to evaluate the quality of the clustering solutions, including the sum of squared errors, the silhouette coefficient, the purity measure, and the entropy measure. The authors also provide some guidelines for choosing the appropriate clustering algorithm and parameters for a given dataset, such as the size of the dataset, the dimensionality of the data, and the desired number of clusters. The authors then present a detailed discussion of some specific clustering algorithms, such as the k-means algorithm, the single-link and complete-link hierarchical clustering algorithms, and the density-based spatial clustering of applications with noise (DBSCAN) algorithm. They compare these algorithms in terms of their computational complexity, memory usage, and performance on different types of data. Overall, the paper provides a comprehensive review of the clustering algorithms and criteria and serves as an essential resource for researchers and practitioners in the field of clustering and data mining. The authors' clear and concise presentation of the material and their insightful analysis make this paper an excellent starting point for anyone interested in clustering algorithms.

Clustering in Data Mining: A Survey," by Xu and Wunsch (2005)[13] is a comprehensive review paper that provides an overview of various clustering algorithms in data mining. The authors discuss the different types of clustering algorithms, their advantages and disadvantages, and their applications in real-world scenarios. The paper begins by defining the concept of clustering and its significance in data mining. The authors explain that clustering is the process of grouping similar objects together based on their attributes, with the goal of finding meaningful patterns and relationships within the data. The authors then discuss the different types of clustering algorithms, including hierarchical, partitioning, density-based, and grid-based methods. They provide a detailed description of each algorithm, including its underlying principles and how it operates. They also highlight the strengths and weaknesses of each algorithm, along with their applications in different fields such as biology, medicine, and finance. In addition, the paper discusses the issue of choosing the appropriate distance metric or similarity measure for clustering, as well as the problem of determining the optimal number of clusters in a given dataset. The authors provide a thorough explanation of various validation techniques such as silhouette score, Calinski-Harabasz index, and Davies-Bouldin index that can be used to assess the quality of clustering results.The authors also examine recent advancements in clustering algorithms, such as the use of ensemble clustering, subspace clustering, and clustering with constraints. They highlight the potential benefits and challenges of these emerging techniques and discuss their future implications in data mining. Overall, "Clustering in Data Mining: A Survey" provides an excellent overview of various clustering algorithms and their applications. It is a valuable resource for researchers and practitioners in the field of data mining who want to gain a deeper understanding of clustering techniques and their strengths and limitations.

MacQueen's 1967 [14]paper titled "Some Methods for Classification and Analysis of Multivariate Observations" is considered a seminal work in the field of clustering algorithms. In the paper, MacQueen proposed the k-means clustering algorithm, which has since become one of the most widely used clustering methods. The k-means algorithm aims to partition a set of n data points into k clusters, such that each data point belongs to the cluster with the nearest mean. The algorithm starts by randomly selecting k points as initial cluster centers, and then iteratively assigns each data point to the cluster with the nearest center and updates the centers to the mean of the points in the cluster. The algorithm terminates when the cluster assignments no longer change. MacQueen's paper also introduced the concept of within-cluster variance as a measure of the goodness of the clustering. He suggested using the ratio of the between-cluster variance to the within-cluster variance, known as the F-statistic, to determine the

optimal number of clusters. The k-means algorithm has since become a widely used method for clustering data in various applications, such as image segmentation, document clustering, and customer segmentation. One of the strengths of k-means is its simplicity and efficiency, making it suitable for large datasets. Despite its popularity, k-means does have some limitations. One major drawback is its sensitivity to the initial cluster centers, which can result in suboptimal clustering solutions. Various modifications and extensions of the k-means algorithm have been proposed to address these limitations, such as hierarchical clustering and fuzzy clustering. MacQueen's 1967 paper proposed a simple yet powerful algorithm for clustering data that has since become a fundamental tool in the field of data mining and machine learning. The k-means algorithm continues to be widely used and has inspired numerous extensions and modifications, making it a cornerstone of clustering methods.

"Introduction to Data Mining" by Tan, Steinbach, and Kumar[15] is a widely used textbook that provides an in-depth overview of data mining techniques, including clustering methods such as k-means. The authors introduce the concept of clustering as a process of grouping data objects into clusters such that objects in the same cluster are more similar to each other than to those in other clusters. They explain that clustering is an unsupervised learning technique, meaning that the algorithm attempts to find patterns in the data without being given prior knowledge or labelled examples. The book discusses the k-means algorithm as a popular and simple clustering method. The authors explain that the algorithm partitions a set of data objects into k clusters by iteratively optimizing the sum of squared distances between each data point and its assigned cluster center. They provide step-by-step instructions for implementing the k-means algorithm and emphasize the importance of selecting the appropriate number of clusters (k) for a given dataset. In addition to describing the k-means algorithm, the book also covers variations of the algorithm, such as k-medoids and fuzzy k-means, as well as techniques for evaluating clustering results. The authors provide examples and case studies throughout the book to illustrate the application of clustering in various fields. Overall, "Introduction to Data Mining" is a comprehensive and accessible resource for learning about clustering and other data mining techniques, including the popular k-means algorithm.

CHAPTER 3: SURVEY OF TECHNOLOGY

3.1 Role of K-Means Algorithm:

In the context of customer segmentation, K-means can be used to divide a customer base into smaller, more homogeneous groups based on their demographic, psychographic, and behavioral characteristics. To do this, the algorithm requires a set of numerical variables that describe the customers, such as age, income, education, occupation, and buying behavior.
The K-means algorithm works by iteratively assigning each customer to the closest cluster centroid, and then updating the centroid location based on the mean of the customer data points in each cluster. This process is repeated until the cluster centroids no longer change or a predetermined stopping criterion is met[10][11].

3.1.1 Working of K-Means algorithm :

i. Choose the number of clusters K that you want to create.
ii. Initialize the centroids (or means) for each cluster by randomly selecting K observations from the dataset.
iii. Assign each observation to the cluster with the nearest mean. The distance between an observation and a mean can be calculated using various distance metrics, but the most commonly used is Euclidean distance.
iv. Calculate the within-cluster sum of squares (WCSS) for each cluster. WCSS is a measure of how spread out the observations are within each cluster, and it is calculated as the sum of squared distances between each observation and its cluster mean.
v. Use the WCSS values to create an elbow plot, which shows the relationship between the number of clusters and the WCSS. The elbow plot is used to determine the optimal number of clusters, where the WCSS starts to decrease at a slower rate. This point is called the "elbow" of the plot.
vi. Repeat steps 3 to 5 until the optimal number of clusters is found or a maximum number of iterations is reached.
vii. Once the optimal number of clusters is determined, reassign the observations to their respective clusters based on the updated means.
viii. Repeat step 4 to calculate the final WCSS for each cluster.
ix. The k-means algorithm is complete, and the resulting clusters can be analyzed.

The WCSS formula is defined as:
WCSS = sum(d^2)/n

where d is the distance between an observation and its cluster mean, n is the

number of observations in the cluster, and sum(d^2) is the sum of the squared distances between each observation and its mean.
The elbow graph method is used to determine the optimal number of clusters by plotting the WCSS values against the number of clusters. The "elbow" of the plot represents the point where the rate of decrease in WCSS slows down, and this is usually the optimal number of clusters to use.

Once the K-means algorithm has been run, the resulting clusters can be interpreted as customer segments. By analyzing the characteristics of each cluster, businesses can gain valuable insights into the unique needs and preferences of each segment, which can inform targeted marketing efforts and improve customer satisfaction.
One advantage of the K-means algorithm is its simplicity and ease of implementation, as it requires only a few parameters to be set, such as the number of clusters, K, and the distance metric used for measuring similarity between customers. Additionally, the K-means algorithm is highly scalable, and can handle large datasets, making it a popular choice for customer segmentation in a big data context.
The K-means algorithm is a powerful tool for customer segmentation as it can handle large datasets and identify complex patterns in customer behaviour and location. By applying the K-means algorithm to our customer data, we can cluster customers into groups based on their similarities and use these groups to create targeted marketing campaigns and promotions.
In this project, we will use the K-means algorithm to segment customers into groups based on their purchasing behaviour and geographic location. We will analyse the resulting clusters to gain insights into customer behaviour and preferences and use these insights to develop targeted marketing strategies.

The objective function of k-means is to minimize the sum of squared distances between each data point and its assigned centroid:

$J = \sum i=1 \text{ to } n \, \|xi - \mu ci\|2$

where xi is the ith data point, μci is the centroid of the cluster to which xi is assigned, and n is the total number of data points.

Numerical Example:

Suppose we have a dataset of 6 two-dimensional points:

(2, 10), (2, 5), (8, 4), (5, 8), (7, 5), (6, 4)

We want to partition this dataset into 3 clusters using the k-means algorithm. We can follow the steps below:

Step 1: Initialize centroids randomly

Let us randomly initialize the centroids as:

c1 = (2, 10), c2 = (8, 4), c3 = (5, 8)

Step 2: Assign each data point to the closest centroid

We calculate the Euclidean distance between each data point and each centroid, and assign each data point to the closest centroid:

Cluster 1: (2, 10), (2, 5)
Cluster 2: (8, 4), (7, 5), (6, 4)
Cluster 3: (5, 8)

Step 3: Recalculate the centroid for each cluster

We calculate the mean of all data points assigned to each cluster, and update the centroid:

c1 = (2, 7.5), c2 = (7, 4.33), c3 = (5, 8)

Step 4: Repeat steps 2 and 3 until convergence

We repeat steps 2 and 3 until no data points are reassigned. After one iteration, the clusters become:

Cluster 1: (2, 10), (2, 5)
Cluster 2: (8, 4), (7, 5), (6, 4), (5, 8)

After another iteration, the clusters become:

Cluster 1: (2, 10), (2, 5), (5, 8)
Cluster 2: (8, 4), (7, 5), (6, 4)

After the third iteration, the clusters do not change anymore, and we obtain the final clusters:

Cluster 1: (2, 10), (2, 5), (5, 8)
Cluster 2: (8, 4), (7, 5), (6, 4)
Cluster 3:

The final centroids are:

c1 = (3, 7.67), c2 = (7, 4.33), c3 = (5, 8)

3.1.2 Importance:

While there are several clustering algorithms available, the choice of algorithm depends on the nature of the data and the problem at hand. K-means is a popular clustering algorithm due to its simplicity, efficiency, and effectiveness in identifying clusters in data.
K-means is a popular and widely used algorithm for clustering data for several reasons:

- Simplicity: K-means is relatively simple and easy to implement compared to some other clustering algorithms. The algorithm only requires specifying the number of clusters to be formed, and then it iteratively assigns each data point to the nearest centroid, updates the centroid location, and repeats the process until the centroids converge.

- Speed: K-means is a computationally efficient algorithm and can quickly cluster large datasets, making it suitable for many applications.

- Clustering: K-Means algorithm is primarily used for clustering data points into groups based on their similarity. Clustering is an essential technique in data science because it can help identify patterns and relationships in large datasets that may be difficult to detect through other methods.

- Scalability: K-Means is a scalable algorithm, which means it can handle large datasets with millions of data points efficiently. This is particularly important for big data applications, where traditional statistical methods may not be feasible.

- Flexibility: K-Means is a flexible algorithm that can be used for a wide range of applications, such as customer segmentation, image segmentation, and anomaly detection. Its flexibility makes it a valuable tool in many different industries.

- Interpretability: K-Means produces interpretable results, which means the clusters it creates can be easily understood and analyzed by domain experts. This is important for many applications, such as market segmentation, where businesses need to understand the characteristics of each cluster to develop targeted marketing strategies.

- Performance: K-Means is a fast algorithm that can converge to a solution quickly, even for large datasets. This makes it an efficient algorithm for many applications, particularly those that require real-time processing.

Compared to other clustering algorithms, k-means is computationally efficient

and works well with large datasets. Additionally, k-means can handle non-linearly separable data by projecting the data onto a higher-dimensional space. However, k-means assumes that the data is evenly distributed and isotropic (same shape and size), which may not always be the case in real-world datasets. In such cases, other clustering algorithms such as DBSCAN or hierarchical clustering may be more appropriate.

Overall, the choice of clustering algorithm should be based on the characteristics of the data, the goals of the analysis, and the computational resources available. K-means is a popular choice due to its simplicity, efficiency, and effectiveness.

3.1.3 Applicability:

- Customer segmentation: Imagine a retail company wants to segment its customers into different groups based on their purchasing behavior. The company can use K-Means algorithm to cluster customers based on their spending habits, frequency of purchases, and other relevant features. The algorithm can group similar customers into different clusters, such as frequent buyers, occasional buyers, and bargain hunters. The company can then tailor its marketing strategies to each cluster, such as sending targeted promotions or discounts to frequent buyers, offering deals to bargain hunters, and so on.

- Image segmentation: In computer vision, image segmentation is a process of partitioning an image into multiple segments, each of which corresponds to a different object or region in the image. K-Means algorithm can be used for image segmentation by clustering pixels based on their color values. The algorithm can group similar pixels into different clusters, which can then be used to identify different objects or regions in the image.

- Anomaly detection: Anomaly detection is a process of identifying unusual or unexpected data points in a dataset. K-Means algorithm can be used for anomaly detection by clustering data points and identifying data points that do not belong to any cluster. These data points can then be flagged as potential anomalies.

- Document clustering: Document clustering is a process of grouping similar documents together based on their content. K-Means algorithm can be used for document clustering by representing each document as a vector of word frequencies or other relevant features. The algorithm can then cluster similar documents into different groups, which can be used to organize and analyze large collections of documents.

- Network analysis: In network analysis, K-Means algorithm can be used to identify clusters of nodes in a network based on their connectivity. For example, the algorithm can cluster social media users based on their connections or cluster genes based on their interactions.

These examples demonstrate the versatility of K-Means algorithm, which can be used in a wide range of applications for clustering data points based on their similarity.

3.2 Role of Elbow Graph:

The elbow graph is a visual tool used to determine the optimal number of clusters to use in a K-means clustering algorithm. In the context of customer segmentation, the elbow graph can help identify the optimal number of customer segments or clusters based on the available customer data.

The elbow graph is plotted with the number of clusters on the x-axis and the Within-Cluster-Sum-of-Squares (WCSS) on the y-axis. The WCSS measures the sum of distances between the data points and their assigned cluster centroid. As the number of clusters increases, the WCSS decreases, since each data point has a closer centroid. However, beyond a certain number of clusters, the improvement in WCSS begins to decrease at a slower rate, resulting in an elbow-shaped curve.

The "elbow point" on the curve represents the optimal number of clusters to use. In the context of customer segmentation, this optimal number of clusters is the one that produces a reasonable balance between the granularity of the segments and the size of each segment.

For example,

Suppose we have a dataset of customer purchases and we want to segment them into different groups based on their buying patterns. We can use the elbow method to determine the optimal number of clusters for our dataset.
We first apply K-means clustering to our data for different values of k (number of clusters). We calculate the WCSS for each value of k and plot it on a graph. We can see that initially, the WCSS decreases rapidly as we increase the number of clusters. However, after a certain point, the rate of decrease slows down significantly, indicating the optimal number of clusters.
Suppose we observe an elbow point at k=3. This means that the optimal number of clusters for our dataset is 3, and we can use this information to segment our customers into 3 groups based on their buying patterns.

Overall, the elbow graph is an important tool for optimizing customer

segmentation using the K-means clustering algorithm.

3.2.1 Need of Elbow Graph :

- Retail: A retail company may want to segment their customers based on their purchase behavior. The company collects data on the amount of money each customer spends per month and the frequency of their purchases. They want to group customers based on their similarity to target marketing campaigns and promotions.
  By using the elbow method, the retail company can determine the optimal number of clusters to use for customer segmentation. If the elbow point occurs at 4, then the retail company would divide customers into four groups based on their purchase behavior. These groups could be "high spenders," "frequent buyers," "low spenders," and "infrequent buyers."

- Healthcare: In healthcare, patient clustering can help identify groups of patients who may benefit from targeted interventions or treatments. Using patient data such as age, gender, medical history, and medication use, healthcare providers can use the elbow method to determine the optimal number of clusters for patient segmentation. This segmentation can help healthcare providers identify high-risk patients who require more intensive interventions or targeted treatment plans.

- E-commerce: In e-commerce, clustering algorithms are used to segment customers based on their purchase history, search history, and other factors. The elbow method can help e-commerce companies determine the optimal number of clusters for personalized recommendations and targeted advertising.

- Social Media Marketing: In social media marketing, companies use clustering analysis to segment their customers based on demographics, interests, behavior, and other factors. The elbow method can be used to determine the optimal number of clusters for effective customer segmentation.

Overall, the elbow method is a useful tool for identifying the optimal number of clusters in a data set, and it can be applied in a variety of real-life scenarios to improve customer segmentation, marketing, and other business operations.

3.3 Role of Silhouette Graph:

Silhouette analysis is a technique used to determine the quality of a clustering algorithm's output. It evaluates the quality of the clustering by measuring the distance between each data point and other data points within the same cluster compared to the distance between that data point and data points in other clusters. The silhouette score ranges from -1 to 1, where a score closer to 1 indicates that the object is well-matched to its own cluster and poorly matched to neighboring clusters. On the other hand, a score closer to -1 indicates that the object may be better matched to a neighboring cluster. A score of 0 means that the object is on the boundary between two clusters.

The silhouette score is calculated using two metrics:

- The mean distance between a sample and all other points in the same class.
- The mean distance between a sample and all other points in the next nearest cluster.

The silhouette score is calculated using the following formula:

$S= (b-a) / \max(a,b)$

where ‘a’ is the average distance between a data point and other data points in the same cluster, and ‘b’ is the average distance between a data point and data points in the nearest other cluster. The ‘S’ value ranges between -1 and 1, where a value of 1 indicates that the data point is very similar to other data points in its cluster, and very dissimilar to data points in other clusters. A value of -1 indicates the opposite, with values around 0 indicating that the data point is equally similar to points in its own cluster and other clusters.

The silhouette score can be plotted for each value of ‘k’ (number of clusters), and the cluster with the highest average silhouette score is considered to be the optimal number of clusters for the data set.
The Silhouette graph plots the silhouette score for each cluster against the number of clusters. The graph is used to find the optimal number of clusters, which is the number of clusters with the highest average silhouette score.

In practice, a silhouette score of 0.5 or greater is considered a strong separation between clusters, while a score between 0.25 and 0.5 indicates a reasonable separation, and a score less than 0.25 indicates that the clustering may not be meaningful. The thickness of the silhouette plot representing each cluster is also a deciding point . Plots with moderately similar thickness are better choice as the optimal cluster value .

In the context of customer segmentation, the silhouette analysis can be used to

determine the optimal number of customer segments that provide meaningful insights into customer behavior. By analyzing the silhouette scores for different values of k, businesses can gain insights into the number of distinct customer segments in their data set, and the quality of the clustering for each segment.

3.3.1 Need of Silhouette Graph :

- Image Segmentation: Silhouette analysis can be used to determine the optimal number of clusters for image segmentation tasks. In image segmentation, the goal is to partition an image into meaningful regions, where each region corresponds to a separate object or part of an object. Silhouette analysis can help determine the optimal number of clusters for this task by measuring how well the pixels within each cluster fit together, and how distinct they are from pixels in other clusters.

- Customer Segmentation: Silhouette analysis can also be used to determine the optimal number of clusters for customer segmentation. In customer segmentation, the goal is to group customers based on similar characteristics, such as demographics, buying behavior, or interests. Silhouette analysis can help determine the optimal number of clusters for this task by measuring how well the customers within each cluster fit together, and how distinct they are from customers in other clusters.

- Gene Expression Analysis: Silhouette analysis can also be used in gene expression analysis to identify clusters of genes that exhibit similar patterns of expression across different conditions or samples. In this context, each gene is treated as a data point, and the goal is to group genes based on similar expression patterns. Silhouette analysis can help determine the optimal number of clusters for this task by measuring how well the genes within each cluster fit together, and how distinct they are from genes in other clusters.

In all of these examples, the silhouette score provides a quantitative measure of how well the data points fit into their assigned clusters, and how well separated the clusters are from each other. By examining the silhouette scores for different numbers of clusters, we can identify the optimal number of clusters that results in well-defined, well-separated clusters. This can help improve the accuracy and interpretability of our clustering results and can also help guide downstream analyses and applications

CHAPTER 4: SYSTEM DESIGN

4.1 Conceptual Model

4.1.1 Work-Flow Diagram:

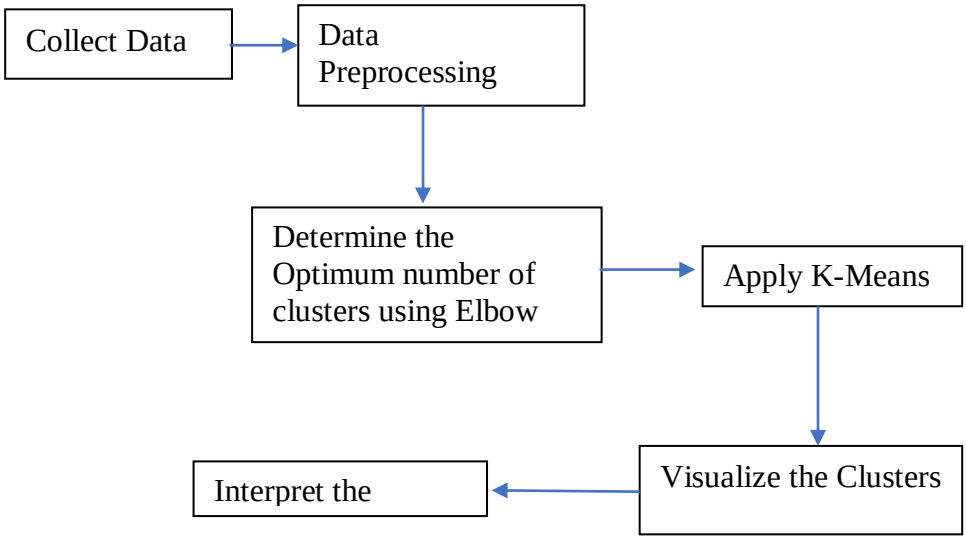


Figure 4.1 Work-Flow Diagram

4.2 Procedural Design

4.2.1 Algorithm for Behavioural Customer Segmentation

In this segment we are trying to visualize the clusters formed by customers based on their

- Spending Score and Annual Income
- Spending Score and Age

i. Define the variables: Start by defining the variables that will be used in the analysis. These variables should describe the behaviour of your customers, such as their spending nature.

ii. Pre-process the data: Pre-process the data to make sure it is ready for clustering. This includes dealing with missing values, outliers, and scaling the data to make sure all variables have the same impact on the analysis.

iii. Determine the number of clusters: Use elbow method, silhouette analysis or any other relevant method to determine the optimal number of clusters.

iv. Initialize the centroids: Randomly select the initial centroids for each cluster.

v. Assign data points to clusters: Assign each data point to the nearest centroid.

vi. Recalculate centroids: Calculate the new centroids for each cluster by taking the mean of all the data points assigned to it.

vii. Repeat steps 5 and 6: Repeat steps 5 and 6 until the centroids no longer move or until a certain number of iterations have been reached.

viii. Evaluate the results: Evaluate the results of the clustering by analysing the characteristics of each cluster, such as the average spending and purchase frequency of each cluster. Determine whether the clusters make sense and whether they provide useful insights for your business.

ix. Use the results: Use the results of the clustering to tailor your marketing and sales strategies to each cluster. For example, you may want to offer promotions or discounts to customers in a particular cluster to encourage them to make more purchases.

4.2.2 Algorithm for Geographical Customer Segmentation

In this segment we are trying to show the clusters of customers with respect to their spending and their geographical location which is taken in the form of their pin code. This will show us how the spending nature of customers vary from one given area to another.

i. Load the data: Load the customer data, including their geographical locations (latitude and longitude).

ii. Pre-processing: Perform any necessary data pre-processing steps, such as removing missing or duplicate data.

iii. Feature scaling: Scale the Spending Score and the Pin code values so that they have the same range of values. This is important because k-means uses the Euclidean distance to determine the similarity between observations,

and different features with different ranges will cause some features to dominate others.

iv. Determine the number of clusters: Determine the number of clusters you want to create. This can be done through trial and error or using techniques such as the elbow method or silhouette score.

v. Fit the k-means model: Fit the k-means model to the scaled data, using the determined number of clusters.

vi. Assign each customer to a cluster: Each customer is assigned to the cluster with the closest centroid, based on their geographical location.

vii. Analyse the clusters: Analyse the characteristics of each cluster to understand the different segments of customers. This can include visualizing the geographical distribution of customers in each cluster, and examining other features of the customers (e.g., age, income, purchase history) to understand the common characteristics of customers in each cluster.

viii. Validate the results: Validate the results using techniques such as cross-validation or by comparing the clusters with external data sources.

ix. Implement the results: Use the insights gained from the customer segmentation to inform marketing and sales strategies. This could include targeted promotions, personalized recommendations, and custom pricing strategies.

x. Repeat the process: The customer segments may change over time, so it is important to periodically repeat the customer segmentation process to ensure that the segments remain relevant and accurate.

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1 Coding Details

5.1.1 Importing Dependencies

```
import numpy as np  # for numpy arrays
import pandas as pd  #for dataframes/data manupulation and rep
resentation
import matplotlib.pyplot as plt  # for data visualization
import seaborn as sns
from sklearn.cluster import KMeans  # k means library
from sklearn.metrics import silhouette_score #Imports the silh
ouette_samples function from the Scikit-
Learn library, which will be used to compute the silhouette sc
ores for individual samples.
from sklearn.metrics import silhouette_samples
import matplotlib.cm as cm #Imports the cm module from the Mat
plotlib library and sets an alias of "cm" for convenience when
 using color maps. This will be used to assign different color
s to each cluster in the visualization.
```

numpy: It is a Python library that is used for scientific computing and numerical operations on arrays and matrices. It is particularly useful when working with large datasets that require fast and efficient numerical computations. Numpy provides various functions for mathematical operations, statistical analysis, and linear algebra, making it a versatile library for data analysis. It is imported with the alias np to enable numerical computations on arrays and matrices efficiently.

pandas: It is a Python library that is used for data manipulation, analysis, and representation. Pandas provides various data structures like Series and DataFrame, which make it easy to perform data manipulation and analysis tasks such as filtering, merging, grouping, and reshaping data. Pandas also provides functions for data visualization and reading and writing data to different file formats. It is imported with the alias pd to enable data manipulation and analysis using data structures such as DataFrames.

matplotlib.pyplot: It is a Python library for creating visualizations such as line plots, scatter plots, and histograms. Matplotlib provides a simple interface for

creating plots and charts and customizing them with labels, titles, and other features. It is a popular choice for creating data visualizations in Python. It is imported with the alias plt to enable data visualization and plotting.

seaborn: It is a Python library for data visualization built on top of matplotlib. Seaborn provides additional functionality and customization options that are not available in matplotlib. It makes it easy to create complex and informative statistical visualizations such as heat maps, cluster maps, and regression plots. It is imported to provide additional data visualization functionality.

sklearn.cluster.KMeans: It is a Python library that provides an implementation of the k-means clustering algorithm, which is a widely used unsupervised learning algorithm for clustering similar data points together. It works by dividing the data into k clusters, where each data point belongs to the cluster with the closest centroid. The k-means algorithm is commonly used for customer segmentation, anomaly detection, and image compression, among other applications. Scikit-learn provides a robust implementation of k-means clustering that can handle large datasets efficiently. It is imported from the Scikit-learn library to perform k-means clustering analysis on data.

sklearn.metrics.silhouette_samples: Silhouette coefficient is a measure of how well each sample in a dataset is assigned to its assigned cluster compared to other clusters. This library computes the silhouette coefficient for each sample in a dataset, which measures how well each sample is assigned to its assigned cluster compared to other clusters.

sklearn.metrics.silhouette_score: Silhouette score is an average silhouette coefficient over all samples in a dataset, which is used as a measure of the quality of clustering. This library computes the average silhouette score over all samples in a dataset, which can be used to compare different clustering algorithms or parameter settings.

matplotlib.cm: This library provides a collection of color maps that can be used to colorize plots and charts. A color map is a mapping between values and colors, which can be used to create more meaningful visualizations of data.

Overall, these libraries provide a wide range of tools and techniques for working with data and extracting insights from it. They are widely used in data science and machine learning applications, and can be a powerful asset for anyone working with data. Once these libraries are imported, they can be used in the script to perform various data analysis tasks, such as clustering analysis.

5.1.2 Data Collection and Analysis

```
#loading the data from csv file to pandas dataframe
customer_data = pd.read_csv('/content/exp.csv' ,names=['Custom
erID','Gender','Age','Annual_Income','Spending_Score','Ware_Pi
n','Customer_Pin','Zone'], header=0, encoding='utf-8')
```

This line of code reads in a CSV file using the pandas read_csv() function and creates a pandas DataFrame called customer_data. The column names are specified using the names argument, and the first row of the CSV file is used as the header row to specify the column names in the DataFrame. The encoding of the file is specified as 'utf-8'. Once the CSV file is read in and the DataFrame is created, the data can be further analyzed, manipulated, and visualized using pandas and other Python libraries.

customer_data is the name of the pandas DataFrame that will be created when the CSV file is read in. This name can be any valid Python variable name, but it should be descriptive of the data that it will contain.

pd.read_csv() is a function from the pandas library that is used to read in a CSV (Comma Separated Values) file. It takes several arguments that specify how the data should be read.

'/content/exp.csv' is the path to the CSV file that will be read. It is specified as a string and should be the full or relative path to the file.

names=['CustomerID','Gender','Age','Annual_Income','Spending_Score','Ware_Pi
n','Customer_Pin','Zone']: This is an optional argument that specifies the names of the columns in the DataFrame that will be created. If the CSV file contains a header row, this argument can be omitted.

header=0: This argument specifies that the first row of the CSV file contains the column names. If the CSV file does not contain a header row, this argument can be omitted.

encoding='utf-8': This argument specifies the encoding of the CSV file. 'utf-8' is a common encoding used for text data. If the CSV file uses a different encoding, this argument can be set to the appropriate value.

```
# first five rows in the data frame
customer_data.head()
34
```

	CustomerID	Gender	Age	Annual_Income	Spending_Score	Ware_Pin	Customer_Pin	Zone
0	1	Male	19	15	39	121003	507101	d
1	2	Male	21	15	81	121003	486886	d
2	3	Female	20	16	6	121003	532484	d
3	4	Female	23	16	77	121003	143001	b
4	5	Female	31	17	40	121003	515591	d

customer_data.head() is a pandas DataFrame method that returns the first n rows of the DataFrame. By default, it returns the first 5 rows, but you can pass a different number of rows as an argument. This information can be useful for understanding the structure and contents of the DataFrame, and for planning further analysis or visualization.

customer_data.shape is a Pandas DataFrame attribute that returns a tuple representing the dimensions of the DataFrame. The tuple contains two values: the number of rows and the number of columns, respectively.

When customer_data.shape is executed, it returns the number of rows and columns in the DataFrame customer_data. The output will be in the form (n_rows, n_cols), where n_rows is the number of rows in the DataFrame and n_cols is the number of columns in the DataFrame. This attribute is useful to quickly check the size of the DataFrame, which is important for performing data manipulation and analysis tasks.

```
1 #getting some information about the dataset
2 customer_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 124 entries, 0 to 123
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   CustomerID      124 non-null    int64
 1   Gender          124 non-null    object
 2   Age             124 non-null    int64
 3   Annual_Income   124 non-null    int64
 4   Spending_Score  124 non-null    int64
 5   Ware_Pin        124 non-null    int64
 6   Customer_Pin    124 non-null    int64
 7   Zone            124 non-null    object
dtypes: int64(6), object(2)
memory usage: 7.9+ KB
```

customer_data.info() is a Pandas DataFrame method that provides a summary of the DataFrame, including the data types, the number of non-null values, and memory usage. It is used to get an overview of the data types and missing values in the DataFrame.

When customer_data.info() is executed, it prints a summary of the DataFrame, including the column names, the number of non-null values, and the data types of each column. It also prints the memory usage of the DataFrame. The output of customer_data.info() can be used to identify missing values and potential data type issues, which are important for data cleaning and preparation. This method is especially useful when working with large datasets, as it provides a quick overview of the data without having to load the entire dataset into memory.

Our code shows that customer_data has 124 rows and 8 columns, with 6 columns of integer data type and 2 columns of object data type. All columns have 124 non-null values, indicating that there are no missing values in the DataFrame. The amount of memory used by the DataFrame is 7.9 KB.

```
#checking for missing values
customer_data.isnull().sum()
```

```
CustomerID        0
Gender            0
Age               0
Annual_Income     0
Spending_Score    0
Ware_Pin          0
Customer_Pin      0
Zone              0
dtype: int64
```

customer_data.isnull().sum() is a Pandas DataFrame method that returns the number of missing values in each column of the DataFrame. When executed, it returns a Pandas Series that contains the number of missing values in each column of the DataFrame customer_data. The index of the Series is the column names, and the values of the Series are the number of missing values in each column.

It is useful for identifying missing values in the dataset, which is an important step in data cleaning and preparation. By identifying the columns with missing values, we can decide how to handle them, such as imputing the missing values or removing the rows or columns with missing values depending on the nature of the problem and the amount of missing data.

5.1.3 Choosing the Required Columns

```
X = customer_data.iloc[:,[4,6]].values #splicing the data , ta
king only the 3rd and 4th column
```

X is a numpy array that contains a subset of the columns from the customer_data DataFrame. Specifically, it contains columns 4 and 6, which correspond to the Spending_Score (column 5) and Customer Pin (column 7) variables in the original DataFrame.

The iloc method is used to select rows and columns of the DataFrame by their integer location. In this case, iloc[:,[4,6]] selects all rows (denoted by the colon :) and the columns at integer locations 4 and 6 (denoted by the list [4,6]). The resulting object is a DataFrame that contains only the Spending_Score and customer Pin columns.

The values attribute is then used to extract the values of this DataFrame as a numpy array. The resulting numpy array, X, has two columns and as many rows as the original DataFrame, with each row representing a customer's spending score and customer pin.

This technique of selecting a subset of columns from a DataFrame is often used when preparing data for machine learning or data analysis tasks, where only certain variables are relevant to the task at hand.

The print(X) statement will print the values of the numpy array X on the console.

```
[[    39 507101]
 [    81 486886]
 [     6 532484]
 [    77 143001]
 [    40 515591]
 [    76 326502]
 [     6 208019]
 [    94 140301]…]
```

X is a 2-dimensional numpy array that contains the values of the Spending_Score and Customer_Pin columns of the customer_data DataFrame. Specifically, X is a subset of the original DataFrame that contains all rows and only columns 4 and 6.

Printing the array is useful to visually inspect the values and ensure that the data was correctly extracted and formatted for further analysis or machine learning.

5.1.4 Calculating and visualizing the silhouette value for each possible number of clusters

```
# set the number of clusters to test
range_n_clusters = [2, 3, 4, 5, 6]

# perform silhouette analysis for each cluster value
for n_clusters in range_n_clusters:
    # create subplots for silhouette plot and cluster visualiz
ation
    fig, (ax1, ax2) = plt.subplots(1, 2)
    fig.set_size_inches(18, 7)

    # set the range for the silhouette plot
    ax1.set_xlim([-0.1, 1])
    ax1.set_ylim([0, len(X) + (n_clusters + 1) * 10])

    # initialize the KMeans clusterer with the current number
of clusters
    clusterer = KMeans(n_clusters=n_clusters,n_init="auto",ran
dom_state=10)
    cluster_labels = clusterer.fit_predict(X)

    # compute the silhouette score for the current clustering
    silhouette_avg = silhouette_score(X, cluster_labels)
    print("For n_clusters =", n_clusters, "The average silhoue
tte_score is :", silhouette_avg)

    # compute the silhouette values for each sample
    sample_silhouette_values = silhouette_samples(X, cluster_l
abels)

    y_lower
 = 10
    for i in range(n_clusters):
```

```
        # aggregate the silhouette values for samples belongin
g to the current cluster
        ith_cluster_silhouette_values = sample_silhouette_valu
es[cluster_labels == i]

        ith_cluster_silhouette_values.sort()

        size_cluster_i = ith_cluster_silhouette_values.shape[0
]
        y_upper = y_lower + size_cluster_i

        # choose a color for the current cluster
        color = cm.nipy_spectral(float(i) / n_clusters)
        ax1.fill_betweenx(
            np.arange(y_lower, y_upper),
            0,
            ith_cluster_silhouette_values,
            facecolor=color,
            edgecolor=color,
            alpha=0.7,
        )

        # label the silhouette plots with their cluster number
s at the middle
        ax1.text(-
0.05, y_lower + 0.5 * size_cluster_i, str(i))

        # update the y_lower value for the next cluster
        y_lower = y_upper + 10  # 10 for the 0 samples

    # add labels and title for the silhouette plot
    ax1.set_title("The silhouette plot for the various cluster
s.")
    ax1.set_xlabel("The silhouette coefficient values")
    ax1.set_ylabel("Cluster label")

    # add a vertical line for the average silhouette score
    ax1.axvline(x=silhouette_avg, color="red", linestyle="--")

    # clear yaxis labels/ticks for better visualization
    ax1.set_yticks([])

    # set the xaxis ticks for the silhouette plot
    ax1.set_xticks([-0.1, 0, 0.2, 0.4, 0.6, 0.8, 1])
```

```
    # 2nd Plot showing the actual clusters formed
    colors = cm.nipy_spectral(cluster_labels.astype(float) / n
_clusters)
    ax2.scatter(
        X[:, 0], X[:, 1], marker=".", s=30, lw=0, alpha=0.7, c
=colors, edgecolor="k"
    )

    # Labelling the clusters
    centers = clusterer.cluster_centers_
    # Draw white circles at cluster centers
    ax2.scatter(
        centers[:, 0],
        centers[:, 1],
        marker="o",
        c="white",
        alpha=1,
        s=200,
        edgecolor="k",
    )

    for i, c in enumerate(centers):
        ax2.scatter(c[0], c[1], marker="$%d$" % i, alpha=1, s=
50, edgecolor="k")

    ax2.set_title("The visualization of the clustered data.")
    ax2.set_xlabel("Feature space for the 1st feature")
    ax2.set_ylabel("Feature space for the 2nd feature")

    plt.suptitle(
        "Silhouette analysis for KMeans clustering on sample d
ata with n_clusters = %d"
        % n_clusters,
        fontsize=14,
        fontweight="bold",
    )
plt.show()
```

This above code initializes a list of range_n_clusters with values 2, 3, 4, 5, and 6. It then iterates over this list using a for loop, and for each value of n_clusters, it creates a figure with two subplots: ax1 for the silhouette plot and ax2 for the cluster visualization. The fig.set_size_inches() function sets the size of the figure.

The silhouette plot shows the silhouette coefficient for each sample, which measures how similar a sample is to its own cluster compared to other clusters. The silhouette coefficient ranges from -1 to 1, with 1 indicating that the sample is

well-matched to its own cluster and poorly matched to neighboring clusters. The plot also shows a vertical line indicating the average silhouette coefficient for all samples, and horizontal bars for each cluster indicating the range of silhouette coefficients for that cluster.

The cluster visualization plot shows the clusters as determined by the KMeans algorithm, with each cluster represented by a different color. The plot also shows the cluster centers as white circles with black edges, and each point is labeled with its cluster number.

By performing silhouette analysis for each cluster value, we can determine the optimal number of clusters for our data based on the highest average silhouette coefficient.

Next, the x-axis and y-axis limits of the first subplot ax1 are set using ax1.set_xlim([-0.1, 1]) and ax1.set_ylim([0, len(X) + (n_clusters + 1) * 10]), respectively. The KMeans clustering model is then initialized with the current number of clusters n_clusters using KMeans(n_clusters=n_clusters,n_init="auto",random_state=10).

Then, the fit_predict method is called on the clusterer object to obtain the cluster labels for each sample in the dataset. The average silhouette score for the current clustering is then computed using silhouette_score(X, cluster_labels) and printed to the console.

The silhouette score is a metric used to evaluate the quality of clustering results. It measures how similar a sample is to its own cluster compared to other clusters, and ranges from -1 to 1, with higher values indicating better clustering results. Therefore, a high average silhouette score is desirable for a good clustering solution.

Then the code computes and plots the silhouette values for each sample in the dataset. The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). For each cluster, the code sorts the silhouette values in ascending order and then fills the space between the minimum value and maximum value with a color representing the cluster. The height of each filled space is proportional to the number of samples in the corresponding cluster. The result is a visualization of the silhouette values for each sample in the dataset and the clusters to which they belong.

Next we are visualizing the silhouette plot and the clusters formed by KMeans clustering in two subplots side by side. In the silhouette plot, we are computing and displaying the silhouette score for each cluster and the silhouette coefficients

for each sample in each cluster. The silhouette score ranges from -1 to 1, where a score closer to 1 indicates a better clustering. The silhouette coefficient for a sample measures how similar it is to its own cluster compared to other clusters. In the second plot, we are showing the actual clusters formed and their respective cluster centers. You are also coloring the data points based on their cluster labels using the cm.nipy_spectral colormap, which is a perceptually uniform colormap designed for visualizing high-dimensional data.

Then the code generates a plot of silhouette scores for different values of the number of clusters in KMeans clustering algorithm, along with a scatter plot of the data points colored by their cluster assignments.

First, a list of possible numbers of clusters is defined (range_n_clusters = [2, 3, 4, 5, 6]).

Then, a loop iterates over each number of clusters and performs the following steps:

- Create a 1x2 subplot figure, where the left subplot will show the silhouette scores and the right subplot will show the scatter plot.
- Set the limits of the left subplot to show the silhouette score between -0.1 and 1, and adjust the y-limit based on the number of clusters and the number of data points.
- Initialize a KMeans clusterer with the current number of clusters, fit it to the data, and obtain the cluster assignments for each data point.
- Compute the average silhouette score for the current clustering using the silhouette_score function.
- Compute the silhouette scores for each data point using the silhouette samples function.
- For each cluster, plot a shaded area in the left subplot between the lower y-limit and the upper y-limit of the cluster, with the x-axis being the silhouette score of the data points in that cluster. The color of the shaded area is chosen based on the cluster number and the number of clusters. The text label of the cluster is also added to the left subplot.
- Add a vertical dashed line at the average silhouette score in the left subplot.
- Remove the y-axis labels/ticks for the left subplot.
- Set the x-axis ticks for the left subplot.
- In the right subplot, scatter plot the data points, with the color of each point representing its cluster assignment. Draw white circles at the centers of the clusters and label each circle with its cluster number.

- Add axis labels and a title to both subplots and add a main title to the figure.

Finally, the plot is displayed using plt.show().

5.1.5 Choosing the Correct Number of Clusters

```
#elbow method for finding wcss value for different number of c
lusters

wcss = []
silhouette_scores = []

for i in range(2,11):
  kmeans = KMeans(n_clusters=i, init='k-
means++',random_state=42)
  kmeans.fit(X)

  wcss.append(kmeans.inertia_) #inertia function lies within t
he kmeans algorith and gives us the wcss values
  silhouette_scores.append(silhouette_score(X, kmeans.labels_)
)# Calculate the silhouette score for the current number of cl
usters
```

We initialize two empty lists, wcss and silhouette_scores, to store the within-cluster sum of squares (WCSS) and silhouette scores for each value of n_clusters. We loop through values of n_clusters from 2 to 10 (inclusive). For each value of n_clusters, we create a KMeans object with the n_clusters parameter set to the current value of i. We fit the KMeans object to our data X. Then we compute and store the WCSS and silhouette score for this value of n_clusters. At the end of the loop, we will have a list of WCSS and silhouette scores for each value of n_clusters we tested.

The WCSS measures the sum of squared distances between each data point and its assigned centroid. Lower values of WCSS indicate better clustering results, as it means that the points are closer to their assigned centroids. The silhouette score, on the other hand, measures how well each data point fits within its assigned cluster compared to other clusters. Higher silhouette scores indicate better clustering results.

Both WCSS and silhouette score are commonly used to evaluate the quality of clustering results. We can plot the WCSS and silhouette scores against the number of clusters to visualize the tradeoff between quality of clustering and number of clusters.

5.1.6 Plotting an elbow and silhouette score graph

```
sns.set()
fig, ax = plt.subplots(1, 2, figsize=(15,5))
ax[0].plot(range(2,11),wcss)
ax[0].set_title('Elbow Graph')
ax[0].set_xlabel('Number of Clusters')
ax[0].set_ylabel('WCSS')

ax[1].plot(range(2,11), silhouette_scores)
ax[1].set_title('Silhouette Score')
ax[1].set_xlabel('Number of Clusters')
ax[1].set_ylabel('Score')

plt.show()
```

This part of the code creates a figure with two subplots using plt.subplots(), and then plots the within-cluster sum of squares (WCSS) and silhouette score for a range of cluster numbers (2 to 10).

The WCSS is plotted on the left subplot, with the x-axis representing the number of clusters and the y-axis representing the WCSS value. The code loops through a range of cluster numbers, and for each number, it creates a KMeans object with n_clusters equal to that number and fits it to the data using the fit() method. The WCSS value for that clustering is then appended to a list called wcss. The resulting list is plotted using ax[0].plot().

The silhouette score is plotted on the right subplot, with the x-axis representing the number of clusters and the y-axis representing the silhouette score. The code loops through the same range of cluster numbers, and for each number, it calculates the silhouette score using the silhouette_score() function and appends it to a list called silhouette_scores. The resulting list is plotted using ax[1].plot().

The sns.set() function is used to set the seaborn style for the plot. The resulting figure allows us to visually identify the optimal number of clusters based on both the elbow point of the WCSS curve and the highest silhouette score.

5.1.7 Training the k-means model

```
kmeans = KMeans(n_clusters=6, init='k-means++',random_state=0)
#return a label for each data point based on their cluster
Y = kmeans.fit_predict(X)    #returns a cluster number for each
 of the data points
print(Y)
```

```
[4 4 4 0 4 3 5 0 1 2 3 1 5 4 2 1 0 4 2 5 1 5 1 5 1 1 2 4 5 4 4 4 2 0 2 1 5
 2 1 1 2 3 1 5 0 1 1 2 2 5 5 1 4 2 1 5 5 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 3 3 3 3 3 3 3 3 3 3 3 3 3 0 0 0 3 3 3 3 3 3 3 3 3 3 3 0 3 3 3 3 3 3 3 3
 3 3 3 3 3 3 3 3 3 3 3 3 0 3]
```

This part of the code performs k-means clustering with 6 clusters on the data in X using the KMeans class from scikit-learn. The fit_predict method of the KMeans class is then called on the data X to assign each data point to one of the 6 clusters based on the k-means algorithm. The resulting cluster assignments are stored in the Y variable, which is a one-dimensional numpy array of length equal to the number of data points in X. Each element of Y corresponds to a data point in X and contains an integer value indicating the cluster assignment of that data point (which of the 6 clusters it belongs to). The cluster assignments are based on the centroid positions of each cluster, which are updated during the iterative process of the k-means algorithm until convergence.

5.1.8 Visualizing the clusters

```
#plotting all the clusters and their centroids

plt.figure(figsize=(10,10))
plt.scatter(X[Y==0,0], X[Y==0,1], s=50, c='green', label='Clus
ter 1')
plt.scatter(X[Y==1,0], X[Y==1,1], s=50, c='red', label='Cluste
r 2')
plt.scatter(X[Y==2,0], X[Y==2,1], s=50, c='yellow', label='Clu
ster 3')
plt.scatter(X[Y==3,0], X[Y==3,1], s=50, c='violet', label='Clu
ster 4')
plt.scatter(X[Y==4,0], X[Y==4,1], s=50, c='blue', label='Clust
er 5')
```

```
plt.scatter(X[Y==5,0], X[Y==5,1], s=50, c='cyan', label='Clust
er 6')

#plot the centroids
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_center
s_[:,1], s=200, c='black', label='Centroids')

plt.title('Customer Groups')
plt.xlabel('Spending Score')
plt.ylabel('Pin Code')
plt.show()
```

Next we create a scatter plot of the customer data points with six different colored clusters based on the KMeans algorithm. The color of each point is determined by the cluster it belongs to (specified by the Y array returned by fit_predict). The x-axis represents the spending score of each customer, while the y-axis represents the pin code. The black points represent the centroids of each cluster. The title of the plot is "Customer Groups", and the x-axis and y-axis are labeled "Spending Score" and "Pin Code", respectively.

5.1.9 Adding a new column called clusters to the dataframe

```
# Create data arrays
cluster_centers = kmeans.cluster_centers_
data = np.expm1(cluster_centers)  # NumPy expm1 function retur
ns the exponential value of minus one for each element given i
nside a NumPy array as output. Therefore, the np.expm1 method
accepts arr_name and out arguments and then returns the array
as outputs.
points = np.append(data, cluster_centers, axis=1)
points
```

```
array([[1.70367483e+26,  inf,  6.04000000e+01,  1.57704700e+05],
[1.00984275e+22,    inf,    5.06666667e+01,    4.13777200e+05],
[4.18441030e+16,    inf,    3.82727273e+01,    7.56027818e+05],
[9.99378148e+21,    inf,    5.06562500e+01,    3.15766625e+05],
[8.81505705e+22,    inf,    5.28333333e+01,    5.14762000e+05],
[1.70170633e+20, inf, 4.65833333e+01, 2.36191833e+05]])
```

Here, we first assign the cluster centers obtained by the KMeans algorithm to a variable cluster_centers.

Then the np.expm1() function is applied to cluster_centers, which is a NumPy function that computes exp(x) - 1 elementwise for any input x. Here, it is used to reverse the natural logarithmic transformation that was applied to the original data before clustering.

Finally, the two arrays, data and cluster_centers, are horizontally stacked using the np.append() function with the axis parameter set to 1, resulting in a new array points that contains the coordinates of the cluster centers in their original scale as well as their logarithmic scale.

```
# Add "clusters" to customers data
points = np.append(points, [[0], [1], [2],[3],[4],[5]], axis=1
)
customer_data["clusters"] = kmeans.labels_
```

The points variable created is a NumPy array that contains the cluster center coordinates and the corresponding feature values. The lines of code append a new column to points containing the cluster labels (0 to 5). Customer_data is a pandas DataFrame that contains the original data used in the clustering analysis. Here, kmeans.labels_ returns the cluster labels assigned to each data point by the KMeans algorithm. By assigning these labels to a new column in the customer_data DataFrame, we can easily perform further analysis or visualization of the clustered data. Finally, the customer_data["clusters"] line assigns the cluster labels to a new column in the customer_data DataFrame.

```
1 customer_data.head()
```

	CustomerID	Gender	Age	Annual_Income	Spending_Score	Ware_Pin	Customer_Pin	Zone	clusters
0	1	Male	19	15	39	121003	507101	d	4
1	2	Male	21	15	81	121003	486886	d	4
2	3	Female	20	16	6	121003	532484	d	4
3	4	Female	23	16	77	121003	143001	b	0
4	5	Female	31	17	40	121003	515591	d	4

The customer_data.head() method by default displays the first 5 rows of the dataframe, as explained earlier.

5.1.10 Bar graph for number of elements in each cluster

```
valcount = customer_data.clusters.value_counts()
plt.figure(figsize=(8,6))
plt.bar(valcount.index,valcount.values)
176
plt.title('Elements Per Cluster')
plt.show()
```

First, it calls the value_counts() method on the clusters column of the customer_data DataFrame, which returns a pandas Series object with the count of unique values in the column. The resulting Series object is then stored in the variable valcount.

Then, a new figure is created with a size of 8 by 6 inches, using the plt.figure() function.

Next, the plt.bar() function is called with valcount.index as the x-axis values and valcount.values as the height of the bars. This creates a bar plot with the number of data points in each cluster on the y-axis and the cluster number on the x-axis.

Finally, the plot title is set to "Elements Per Cluster" using plt.title() and the plot is displayed using plt.show().

CHAPTER 6: RESULT AND ANALYSIS

6.1 DATASET 1:

https://drive.google.com/file/d/1svkok9BxE1KToHJ93dkQDE5ZwWCDLv6c/view?usp=sharing

The given dataset has 8 columns – Customer ID, Gender, Age, Annual Income, Spending Score, Warehouse Pin-code, Customer Pin-code and Zone and data for 124 customers. The cluster value is appended to the dataset during the coding process.

	CustomerID	Gender	Age	Annual_Income	Spending_Score	Ware_Pin	Customer_Pin	Zone	clusters
0	1	Male	19	15	39	121003	507101	d	4
1	2	Male	21	15	81	121003	486886	d	4
2	3	Female	20	16	6	121003	532484	d	4
3	4	Female	23	16	77	121003	143001	b	0
4	5	Female	31	17	40	121003	515591	d	4

Table.1 : First five rows of the dataset 1

6.1.1 Analyzing the Results :

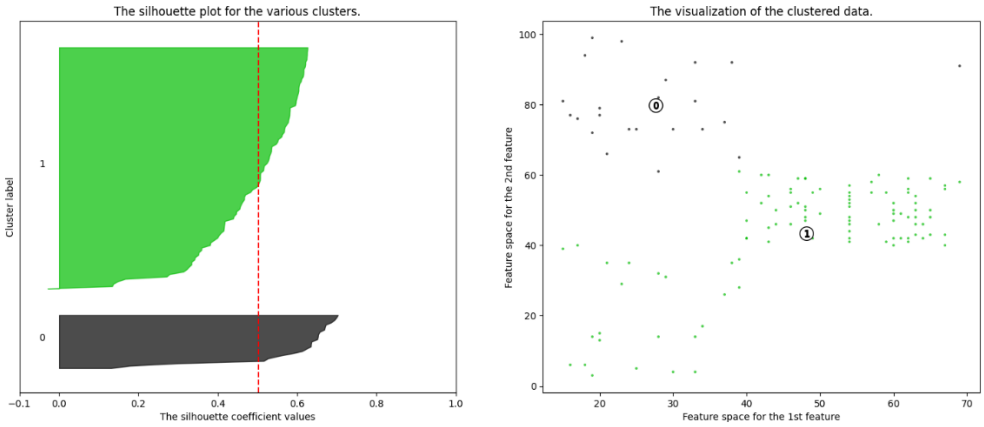
CASE 1 : In the first case we applied K-Means Clustering on the parameters –

Annual Income vs Spending Score and the results looked like:

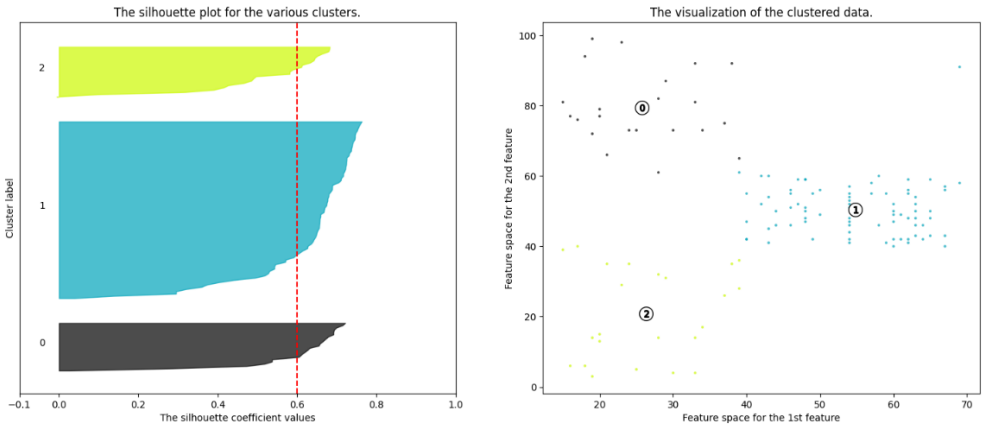
Figure 6.1 :proves how 3 is the optimal number of clusters.

```
For n_clusters = 2 The average silhouette_score is :
0.7051708416017349
For n_clusters = 3 The average silhouette_score is :
0.6595557835052082
For n_clusters = 4 The average silhouette_score is :
0.6918601183884051
For n_clusters = 5 The average silhouette_score is :
0.6936287837961752
For n_clusters = 6 The average silhouette_score is :
0.6922766282170691
```

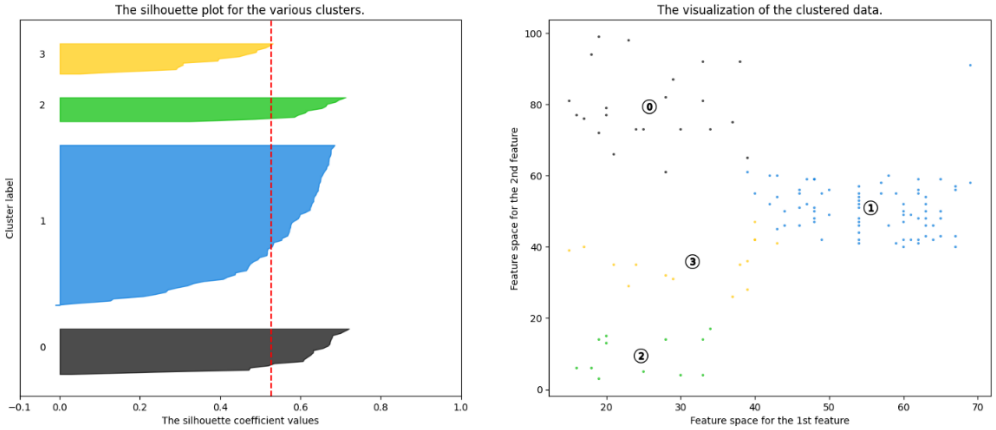
Silhouette analysis for KMeans clustering on sample data with n_clusters = 2



Silhouette analysis for KMeans clustering on sample data with n_clusters = 3



Silhouette analysis for KMeans clustering on sample data with n_clusters = 4



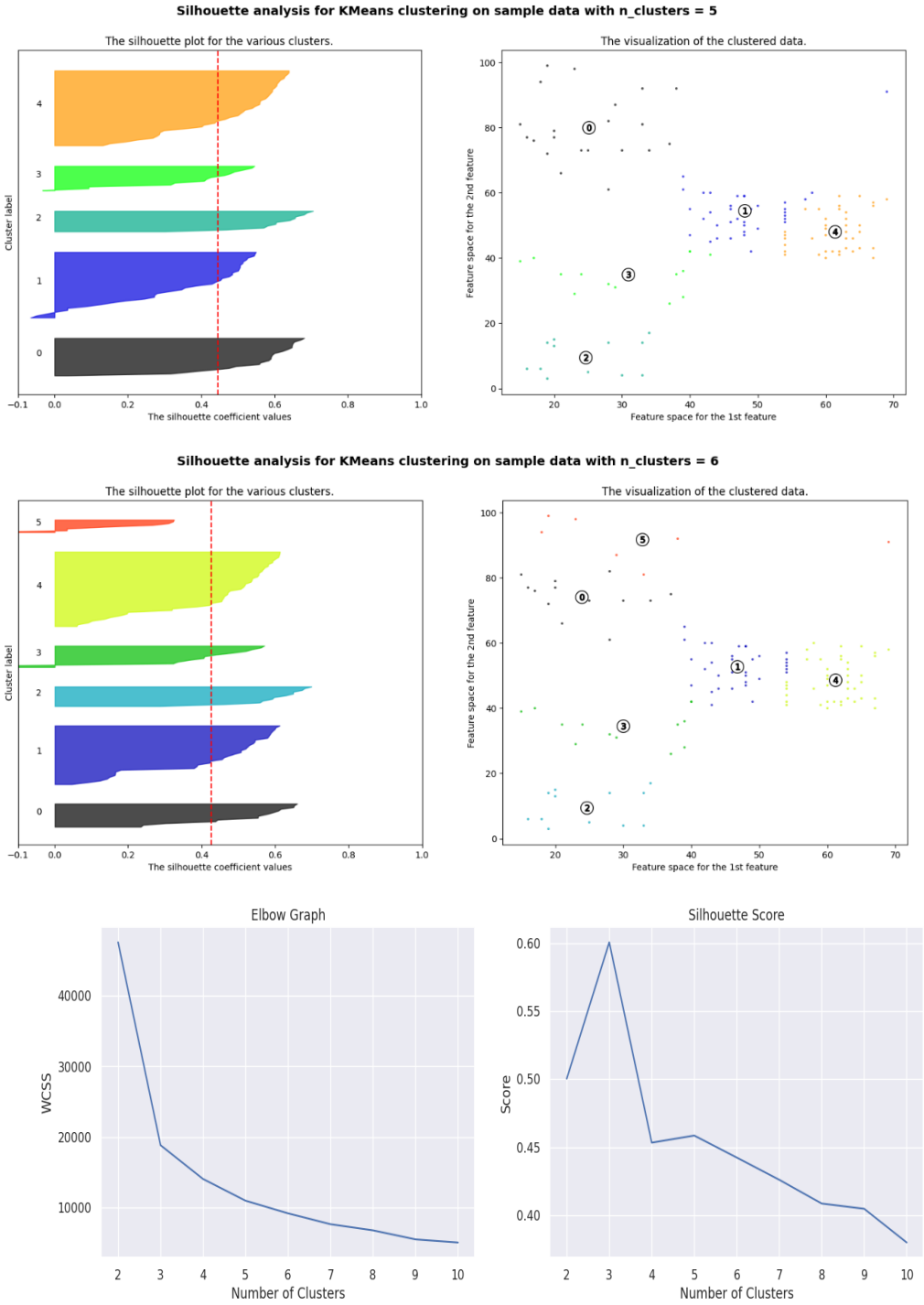


Figure 6.2: Elbow Method and Silhouette Score for optimal number of Clusters

From this elbow graph and silhouette score we can say that the optimum number of clusters are 3.

The Final clusters of the scatter plot look like –

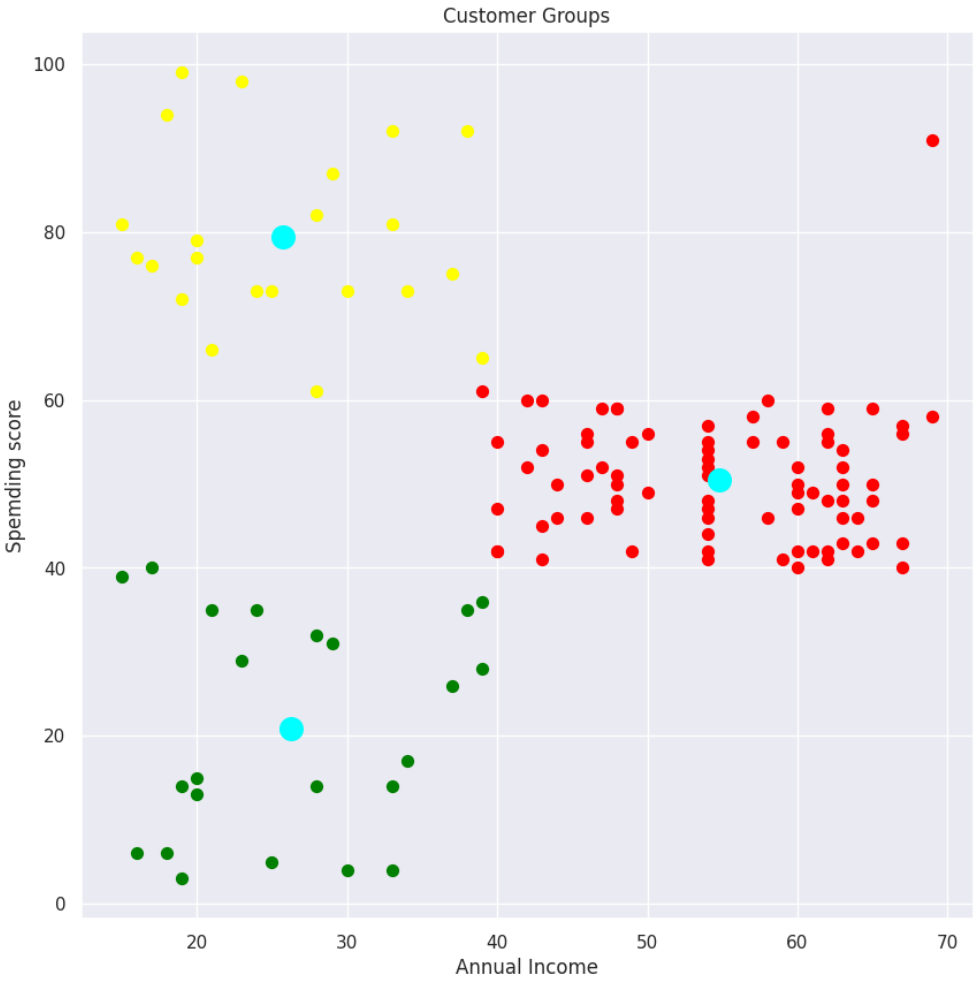


Figure 6.3: Final scatter plot

Bar graph representation of the Clusters :

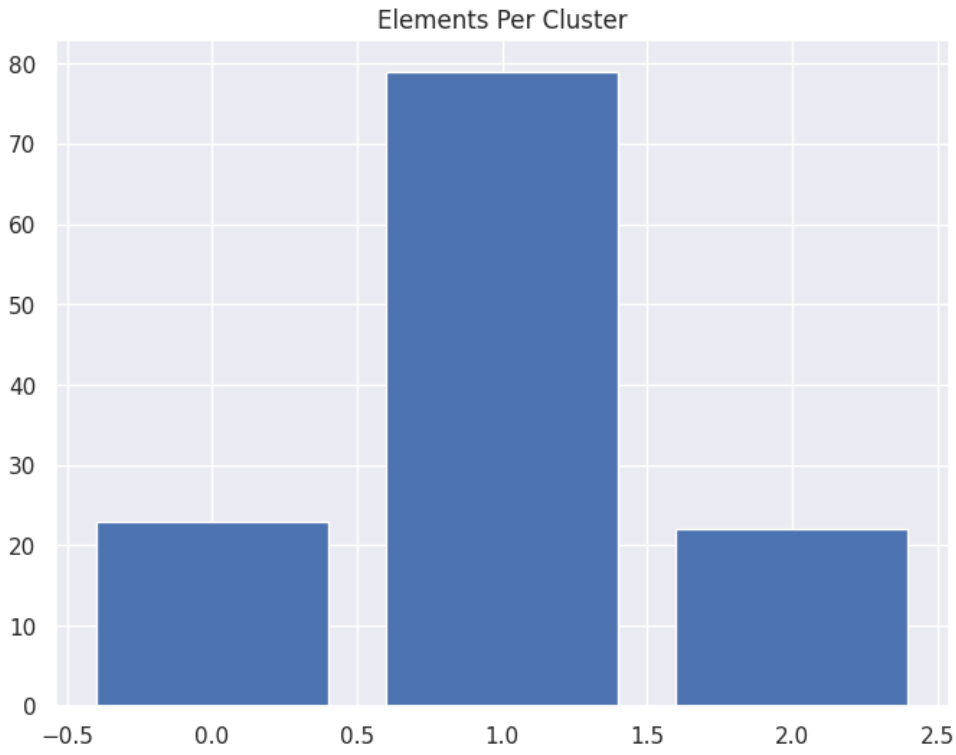


FIGURE 6.4: Elements per cluster

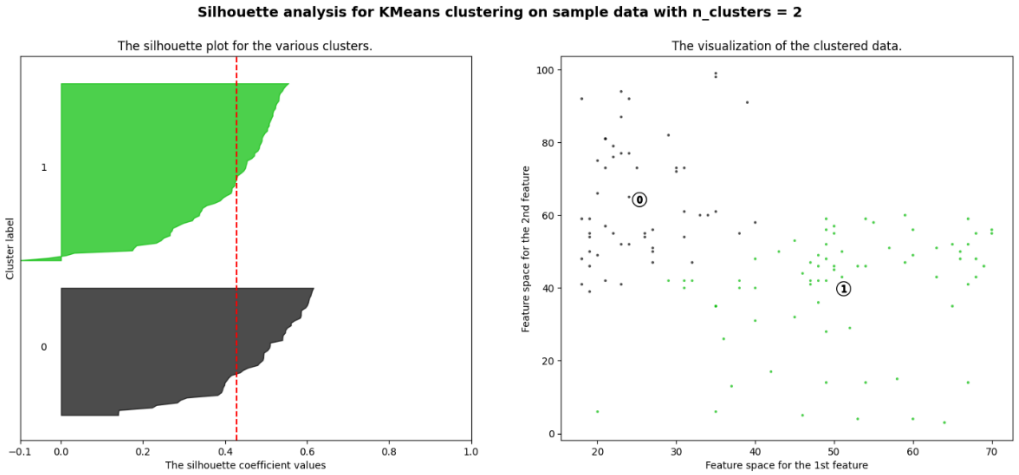
From this output we can say that:

- 0.0 – Green Cluster - Customers with Annual Income in the range <=400k with Spending Score in the range <=40 form a scattered cluster which suggests that there are less customers who are interested in the product of this particular business, hence the business can implement different methods like discounts or loyalty benefits to attract customers of this group.
- 1.0 – Red Cluster – Customers with Annual Income in the range 400k to 700k and a spending score in the range 40-60 form a concentrated cluster hence suggesting that more customers of this group are interested in the products of this particular business. Hence, the business can introduce memberships and give certain benefits to keep them invested.
- 2.0 – Yellow Cluster – Customers with Annual Income in the range <=400k with Spending Score in the range 60-100 form a scattered cluster which suggests that there are less customers who are interested in the product of this particular business, hence the business can implement different methods like better advertising, flashy offers and allocating things like loyalty points to attract customers of this group.

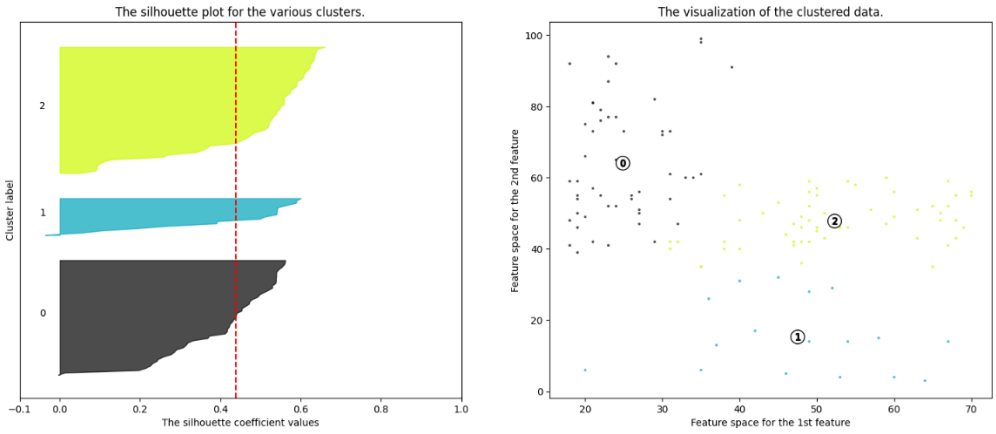
CASE 2 : In the second case we applied K-Means Clustering on the parameters – Spending Score vs Age and the results looked like :-

Figure 6.5 :proves how 4 is the optimal number of clusters.

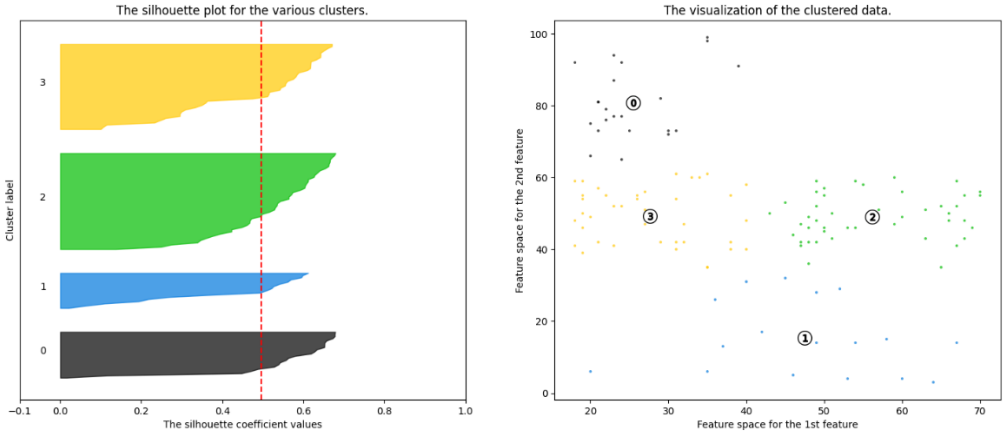
```
For n_clusters = 2 The average silhouette_score is:
0.4273768903935447

For n_clusters = 3 The average silhouette_score is:
0.4392875019111278

For n_clusters = 4 The average silhouette_score is:
0.4953949368236104

For n_clusters = 5 The average silhouette_score is :
0.45172627429691875

For n_clusters = 6 The average silhouette_score is :
0.4500112897884164
```



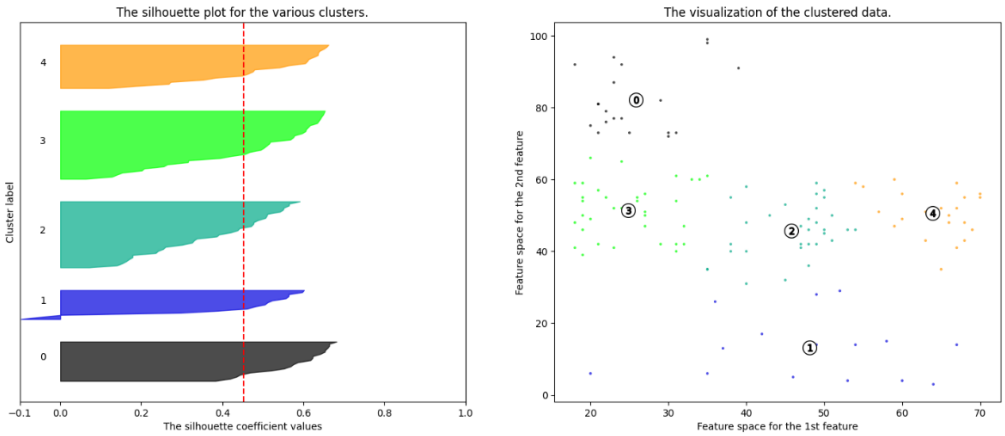
Silhouette analysis for KMeans clustering on sample data with n_clusters = 3



Silhouette analysis for KMeans clustering on sample data with n_clusters = 4



Silhouette analysis for KMeans clustering on sample data with n_clusters = 5



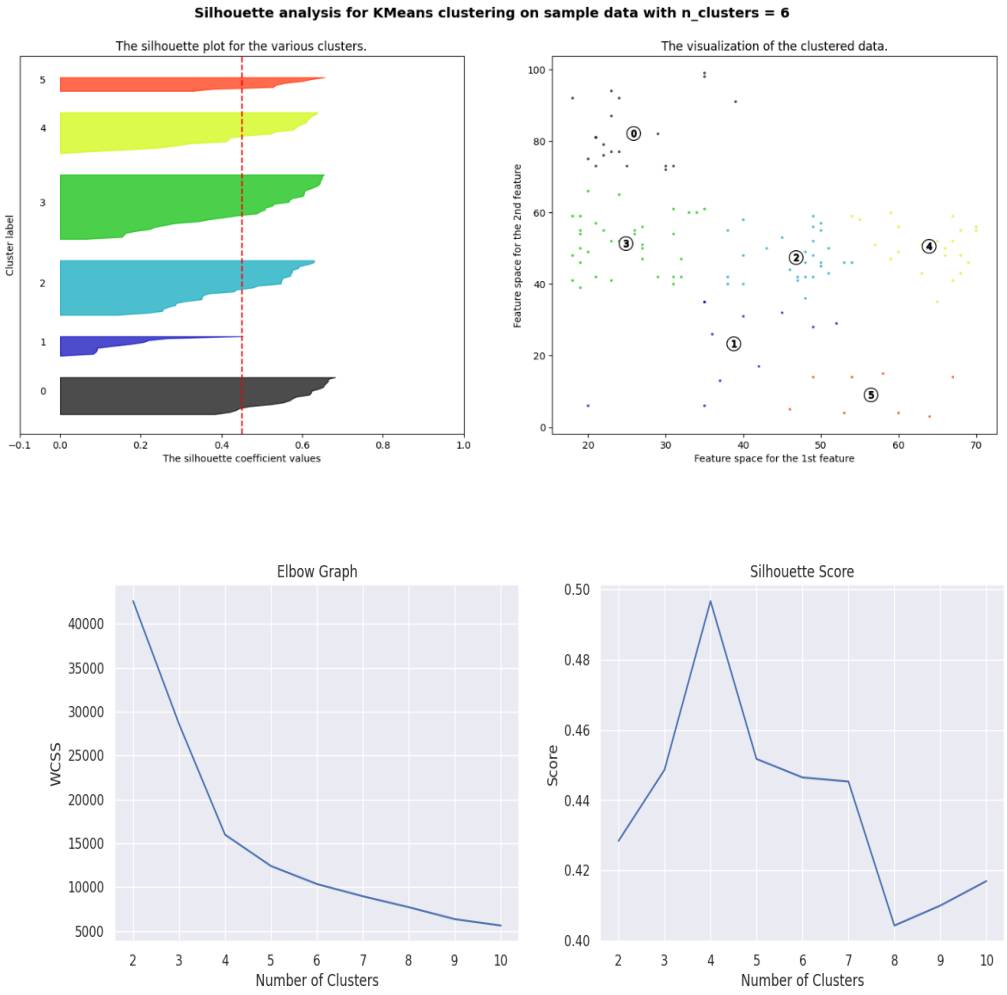


Figure 6.6: Elbow Method and Silhouette Score for optimal number of Clusters

From the elbow graph and silhouette score we can say that the optimum number of clusters here will be 4.

The final scatter plot clusters look like -

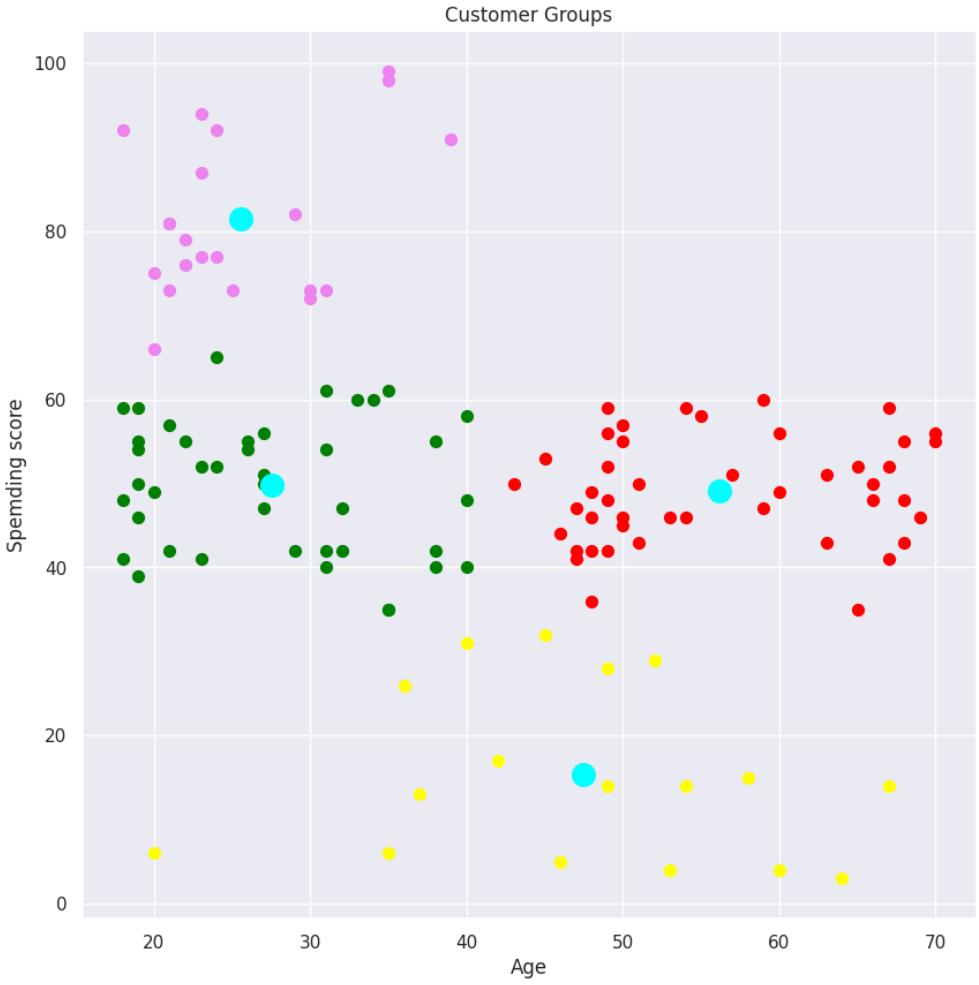


Figure 6.7: Final scatter plot

Bar graph representation of the Clusters :

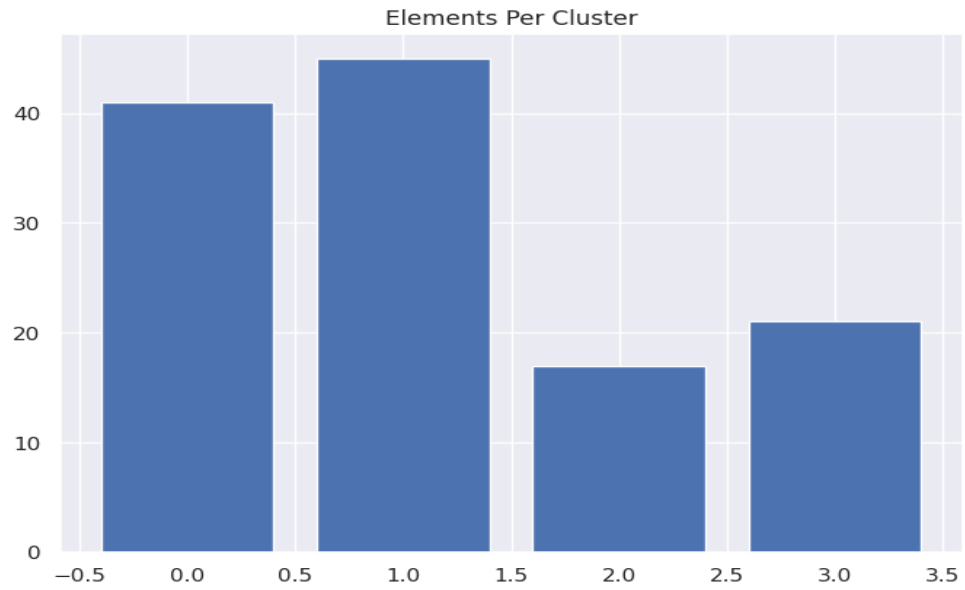


Figure 6.8: Elements per cluster

From this output, we can say that:

- 0.0 – Green Cluster – This is a concentrated cluster formed by people with medium spending score and less age, this group of people seem to be quite loyal to this business and thus to keep them invested flashy offers may be implemented.

- 1.0 – Red Cluster - This is a concentrated cluster formed by people with medium spending score and more age, this group of people seem to be quite loyal to this business and thus to keep them invested membership benefits may be implemented.

  From Cluster 2 and 3 we can say that, overall people with medium spending score are more invested in this business rather than people with low and high spending scores.

- 2.0 – Yellow Cluster – This cluster is formed of customers who have lesser spending score i.e., <40 and they form a scattered cluster and mostly aged people from this group of customers show some interest in the

business. So, to attract younger people the business can offer flashy offers and discounts and for older people the business may incorporate membership benefits and loyalty points.

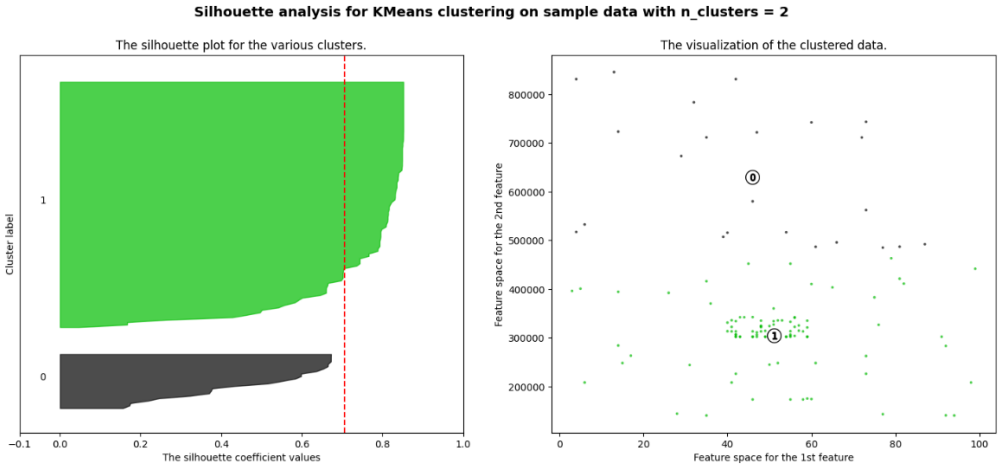
- 3.0 – Violet Cluster – This cluster is formed by people with high spending score and low age, to keep them invested and attract more customers from this groups membership benefits and flashy offers may be incorporated.

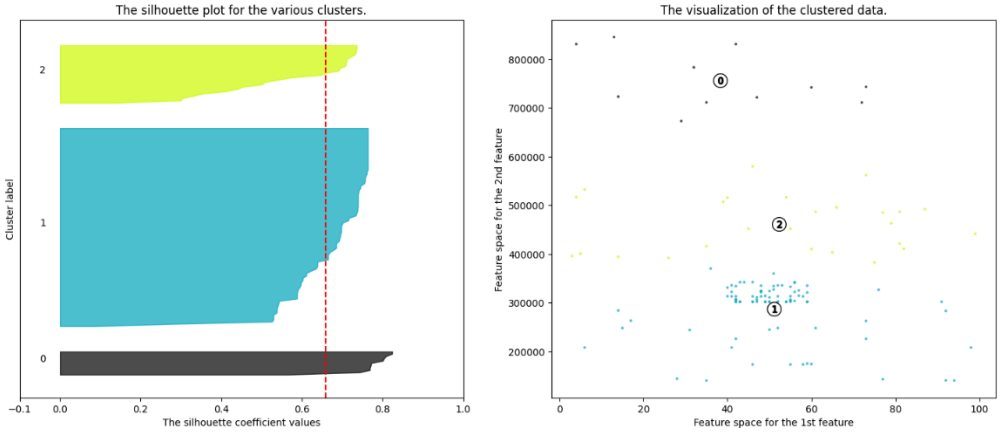
CASE 3 : In the third program we applied K-Means Clustering on the parameters –

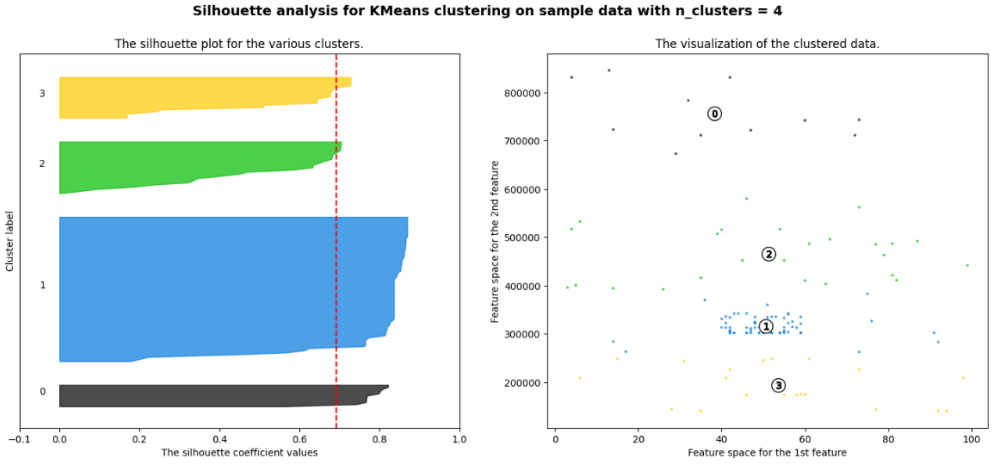
Pin-Code vs Spending score and the results looked like :-

Figure 6.9 :proves how 6 is the optimal number of clusters.

```
For n_clusters = 2 The average silhouette_score is:
0.7051708416017349

For n_clusters = 3 The average silhouette_score is:
0.6595557835052082

For n_clusters = 4 The average silhouette_score is:
0.6918601183884051

For n_clusters = 5 The average silhouette_score is :
0.6936287837961752

For n_clusters = 6 The average silhouette_score is :
0.6922766282170691
```


Silhouette analysis for KMeans clustering on sample data with n_clusters = 2
The silhouette plot for the various clusters.
Cluster label
The silhouette coefficient values
The visualization of the clustered data.
Feature space for the 2nd feature
Feature space for the 1st feature


Silhouette analysis for KMeans clustering on sample data with n_clusters = 3
The silhouette plot for the various clusters.
Cluster label
The silhouette coefficient values
The visualization of the clustered data.
Feature space for the 2nd feature
Feature space for the 1st feature


Silhouette analysis for KMeans clustering on sample data with n_clusters = 4
The silhouette plot for the various clusters.
Cluster label
The silhouette coefficient values
The visualization of the clustered data.
Feature space for the 2nd feature
Feature space for the 1st feature

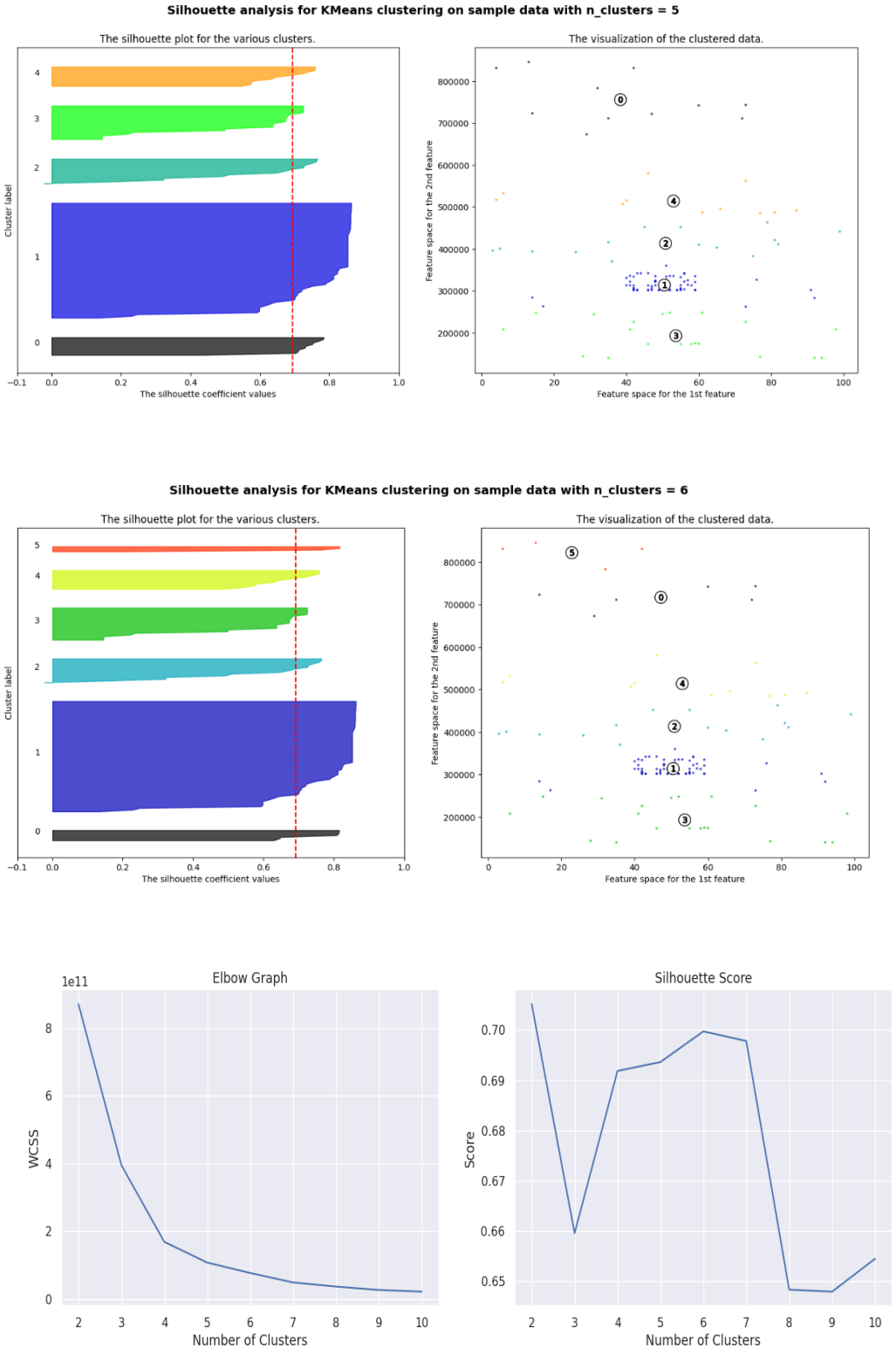


Figure 6.10: Elbow Method and Silhouette Score for optimal number of Clusters

From the elbow graph and silhouette method we can say that the optimum number of clusters are 6.

The final scatter plot clusters look like -

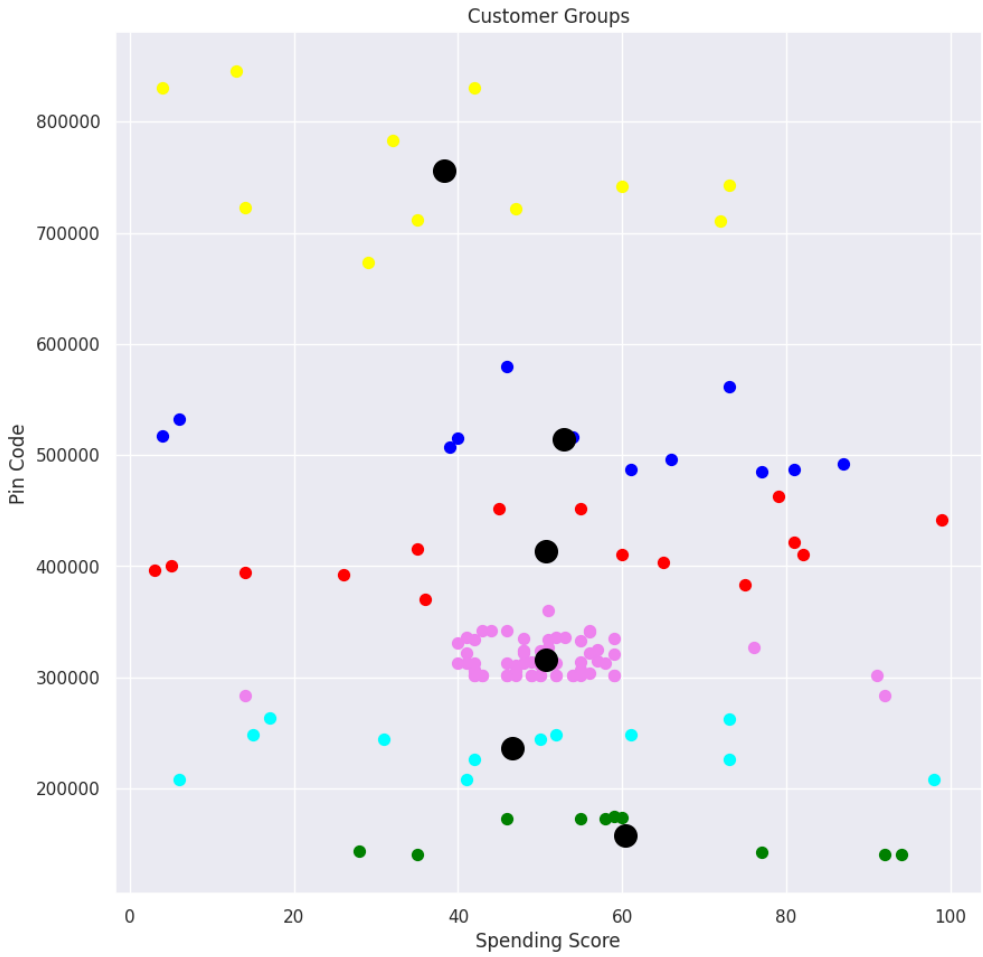


Figure 6.11: Final scatter plot

Bar graph representation of the Clusters :

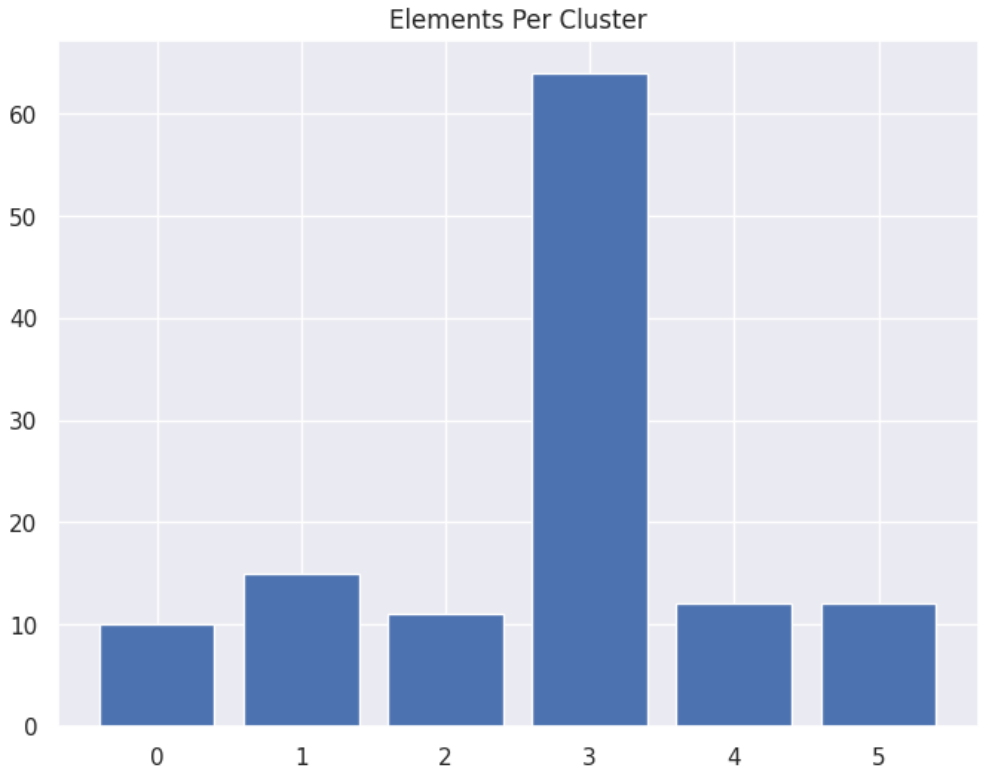


Figure 6.12: Elements per cluster

- 0.0 – Green Cluster - The customers who stay in this region form a scattered cluster which depicts that the customers from this region are less interested in buying products from this business.
  To get these customers more invested in the products we can avail free delivery services and discounted deals.
- 1.0 – Red Cluster – The customers who stay in this region form a scattered cluster which depicts that the customers from this region are less interested in buying products from this business.
  To get these customers more invested in the products we can avail free delivery services and discounted deals.
- 2.0 – Yellow Cluster – The customers who stay in this region form a very scattered cluster suggesting that the customers in this group are less invested in the products of this business.

We can assume that these customers are located far away from the business where either outlet are not there, or delivery options are not available. So, to invest these customers the business model may decide to open an outlet in this area or increase their delivery range.

- 3.0 – Violet Cluster – This is the most concentrated cluster . This depicts that the customers residing in this region with a moderate spending score of 40-60 are the most invested in buying products from this business.
  We may also assume that business outlets are located in this region which means there is an easier availability of the products in this region compared to the other regions. Also, to attract customers with a lower spending score the business model may implement discount offers and to attract customers with higher spending score the business may implement membership benefits and loyalty points.
- 4.0 – Blue Cluster – The customers who stay in this region form a scattered cluster which depicts that the customers from this region are less interested in buying products from this business.
  To get these customers more invested in the products we can avail free delivery services and discounted deals.
- 5.0 – Cyan Cluster - The customers who stay in this region form a scattered cluster which depicts that the customers from this region are less interested in buying products from this business.
  To get these customers more invested in the products we can avail free delivery services and discounted deals.

6.2 Dataset 2:

https://drive.google.com/file/d/17CBoeFRE9uVtYbJqaZs_ouCtz92KO6RK/view?usp=sharing

The given dataset has 8 columns – Customer ID, Gender, Age, Annual Income, Spending Score, Warehouse Pin-code, Customer Pin-code and Zone and data for 462 customers. The cluster value is appended to the dataset during the coding process.

Since, in this dataset, there are 462 customer data, which is quite large, we incorporated a bar graph to analyze the number of customers in each cluster.

	CustomerID	Gender	Age	Annual_Income	Spending_Score	Ware_Pin	Customer_Pin	Zone	clusters
0	1	Male	55	55	72	515631	434011	Kurnool Region	1
1	2	Male	43	57	44	515631	613712	Kurnool Region	3
2	3	Female	30	66	31	515631	403813	Kurnool Region	4
3	4	Female	30	38	92	515581	687163	Kurnool Region	2
4	5	Female	34	73	23	515581	353527	Kurnool Region	4

Table 6.2 : First five rows of the dataset 2

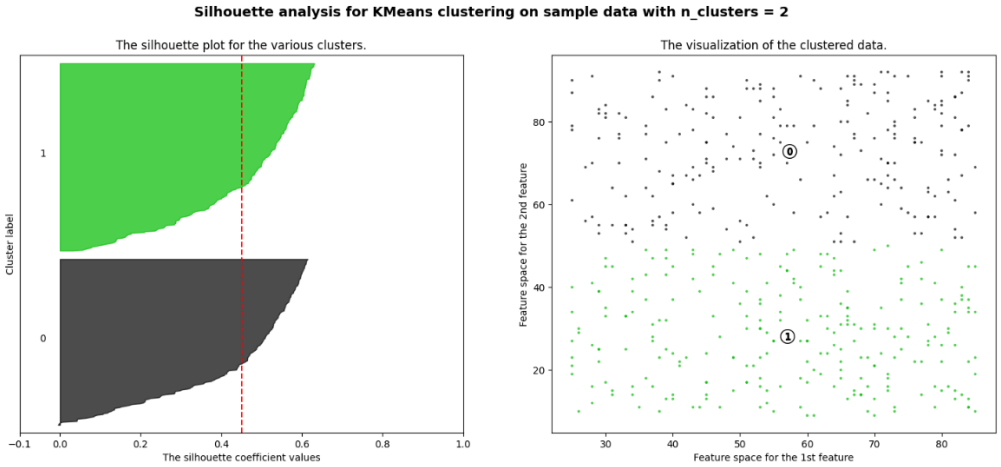
6.2.1 Analyzing the Results :

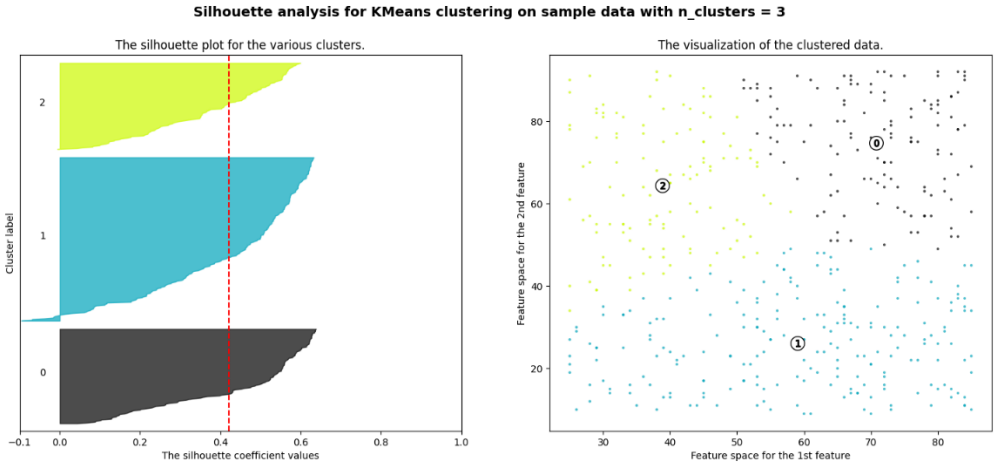
CASE 1: In the first program we applied K-Means Clustering on the parameters – Annual Income vs Spending Score and the results looked like:

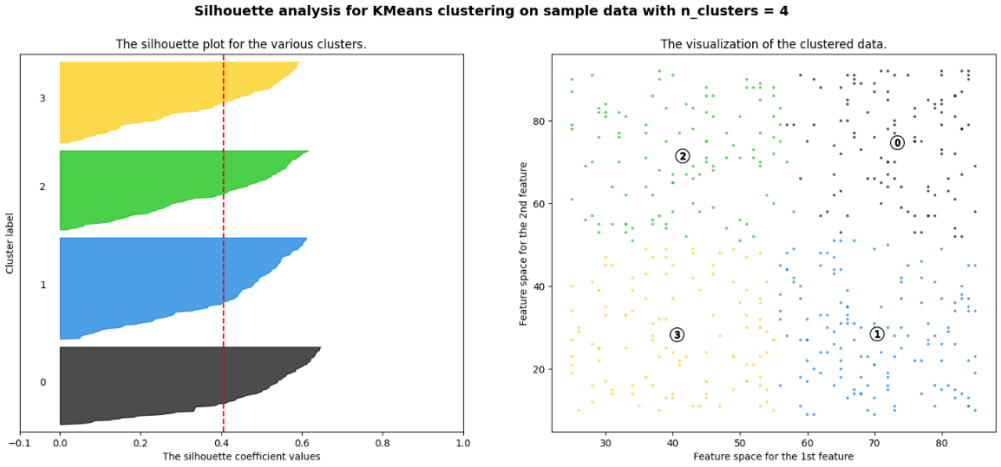
Figure 6.13 :proves how 5 is the optimal number of clusters.

```
For n_clusters = 2 The average silhouette_score is :
0.4501030908768755

For n_clusters = 3 The average silhouette_score is :
0.4204397683865185

For n_clusters = 4 The average silhouette_score is :
0.4049933674155039

For n_clusters = 5 The average silhouette_score is :
0.4116081952655662
```

For n_clusters = 6 The average silhouette_score is : 0.402872885781395







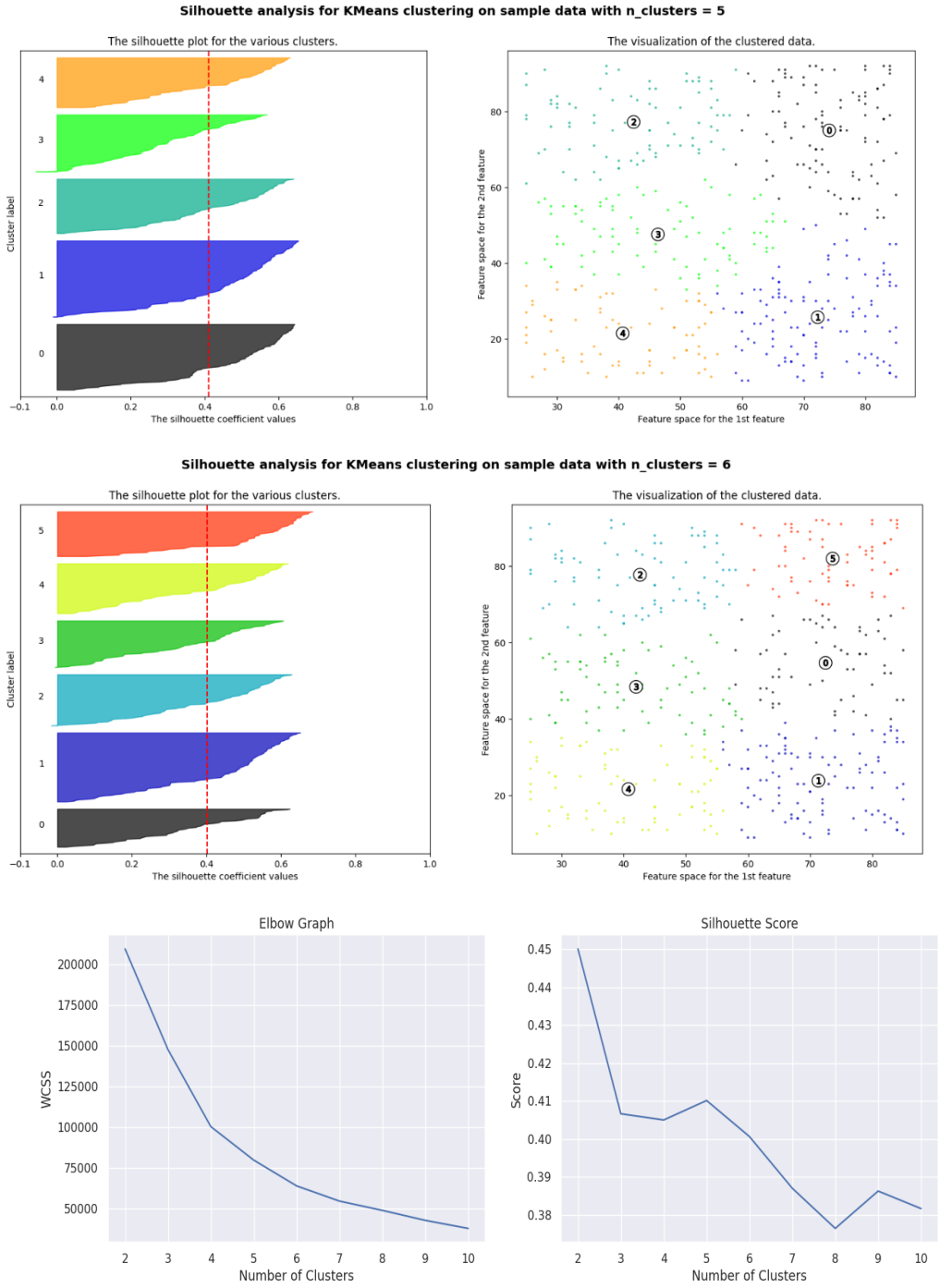


Figure 6.14: Elbow Method and Silhouette Score for optimal number of Clusters

From this elbow graph and silhouette score we can say that the optimum number of clusters in this case will be

The final scatter plot clusters look like –

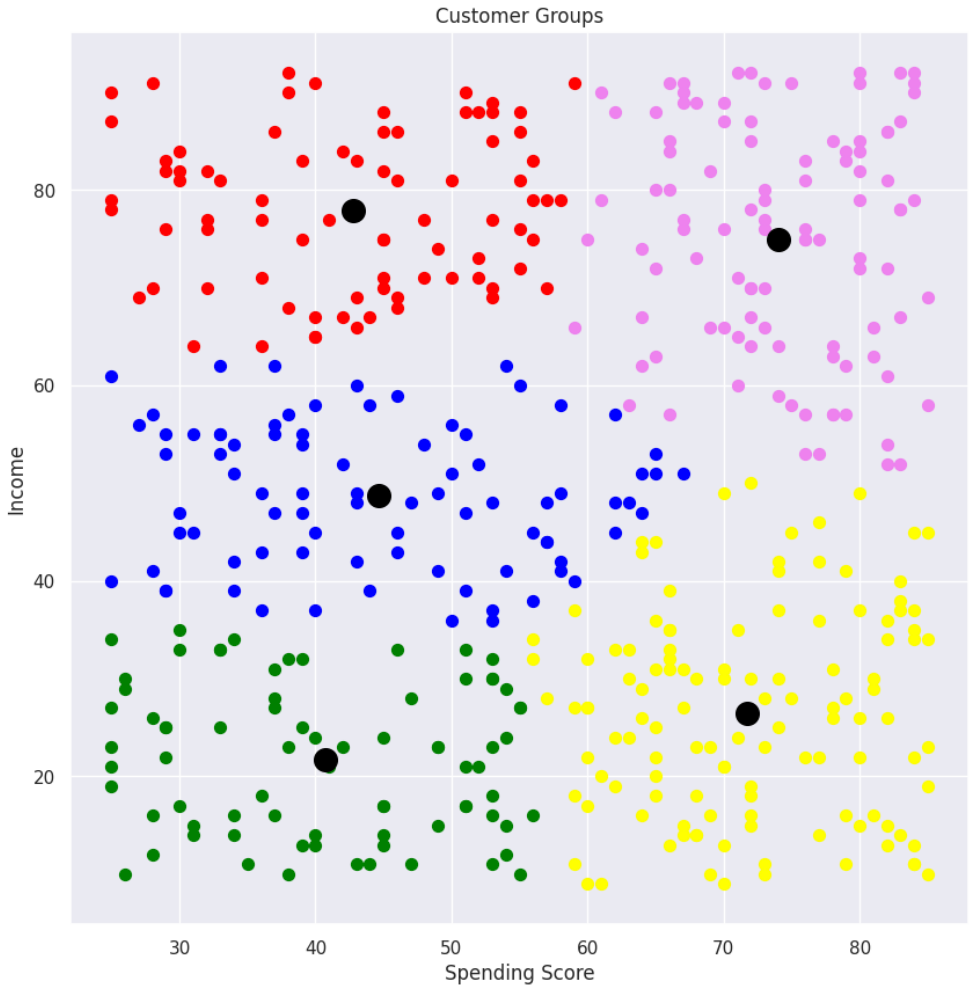


Figure 6.15: Final scatter plot

Bar graph representation of the Clusters :

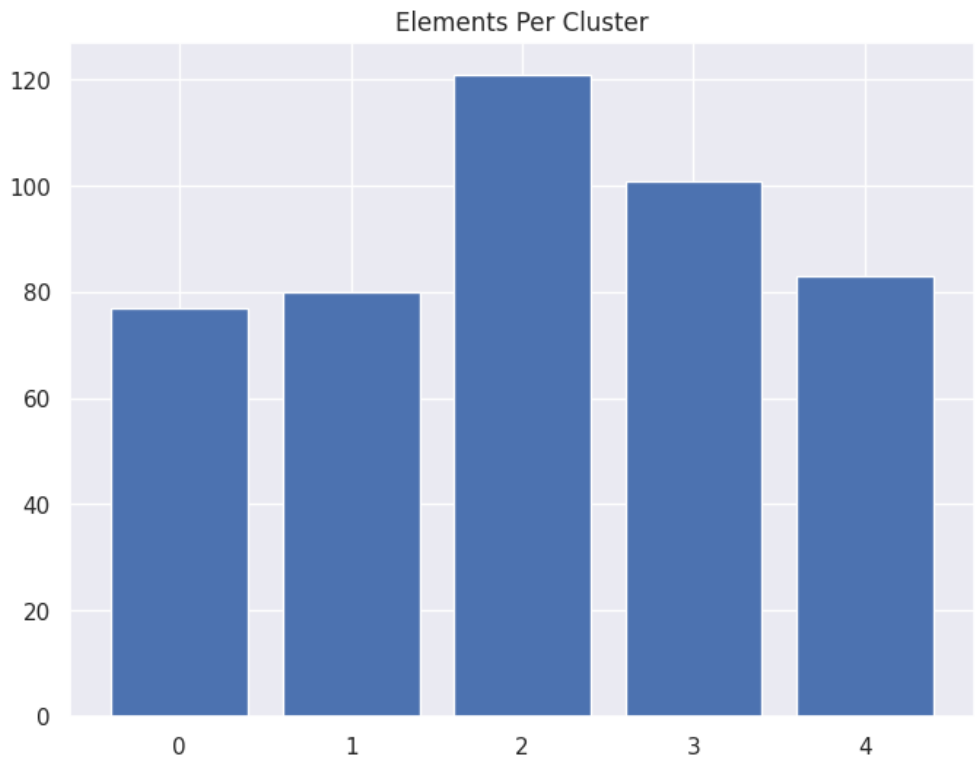


FIGURE 6.16: Elements per cluster

Here

0.0 - Green Cluster – Customers with low spending score and low income are less invested in buying products from this business as compared to other customer groups.

1.0 – Red Cluster – Customers with high income and a low spending score are a little more invested in buying products from this business than the previous cluster.

2.0 – Yellow Cluster – This is the most concentrated cluster, comprising of people with a high spending score and low income.

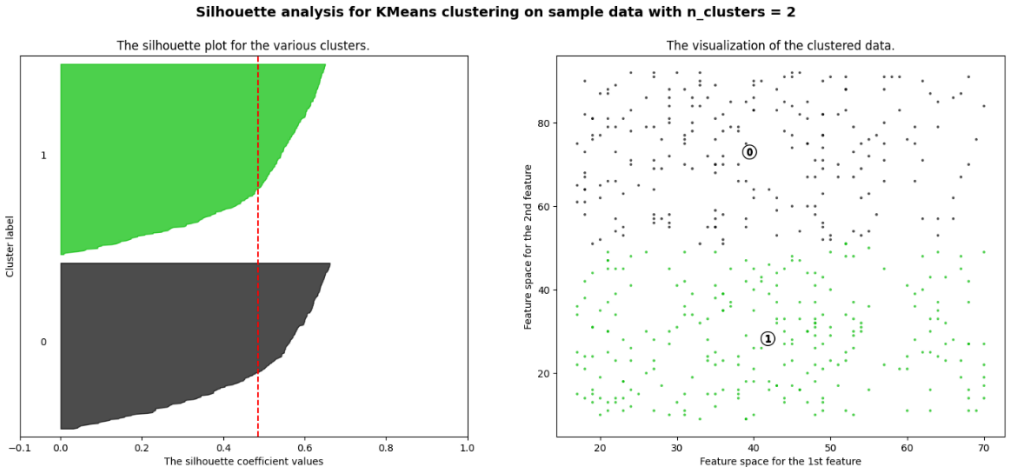
3.0 -Violet Cluster – This cluster has the customers with high spending score and high income and they are quite invested in buying from this business. It forms the second most concentrated cluster.

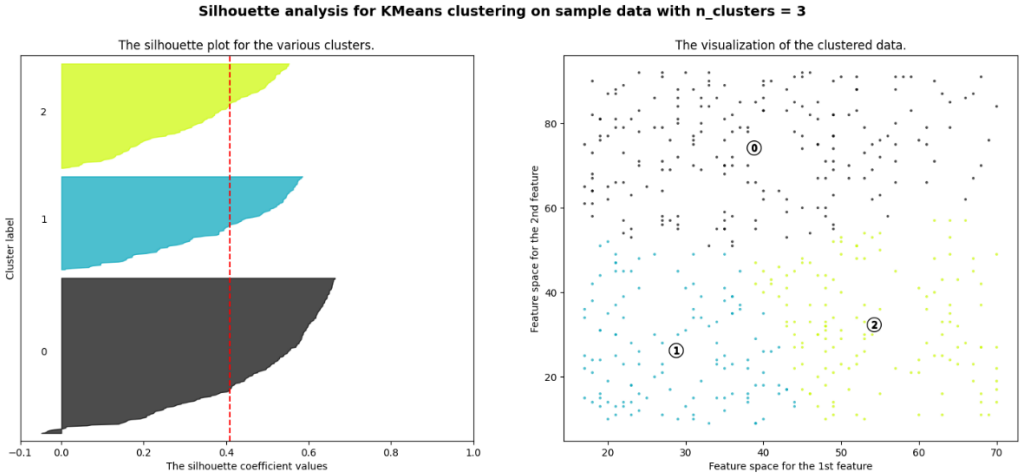
4.0 – Blue Cluster – This is the third most concentrated cluster with people of mediocre income and a low to medium spending score.

CASE 2 : In the second case we applied K-Means Clustering on the parameters – Spending Score vs Age and the results looked like :-

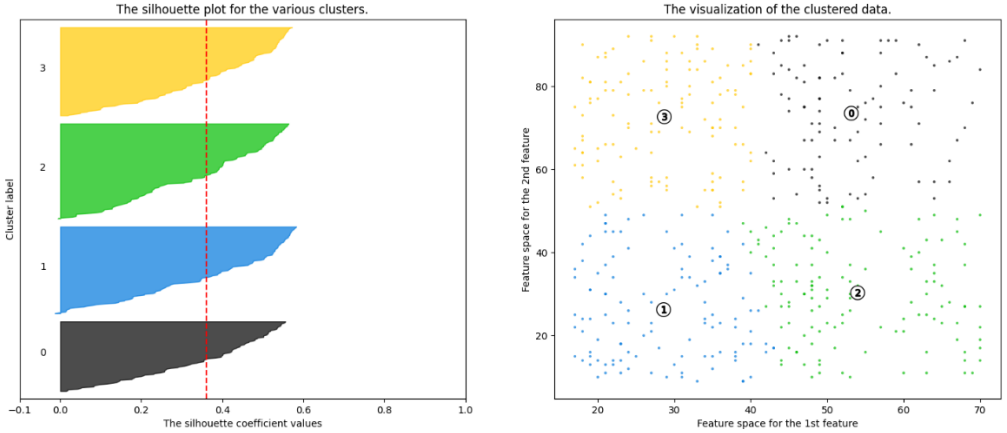
Figure 6.17 :proves how 5 is the optimal number of clusters.

```
For n_clusters = 2 The average silhouette_score is :
0.4858793595690374

For n_clusters = 3 The average silhouette_score is :
0.4093031312291 53

For n_clusters = 4 The average silhouette_score is :
0.3597251354382038

For n_clusters = 5 The average silhouette_score is :
0.37918852030118116

For n_clusters = 6 The average silhouette_score is :
0.3846591558553363
```

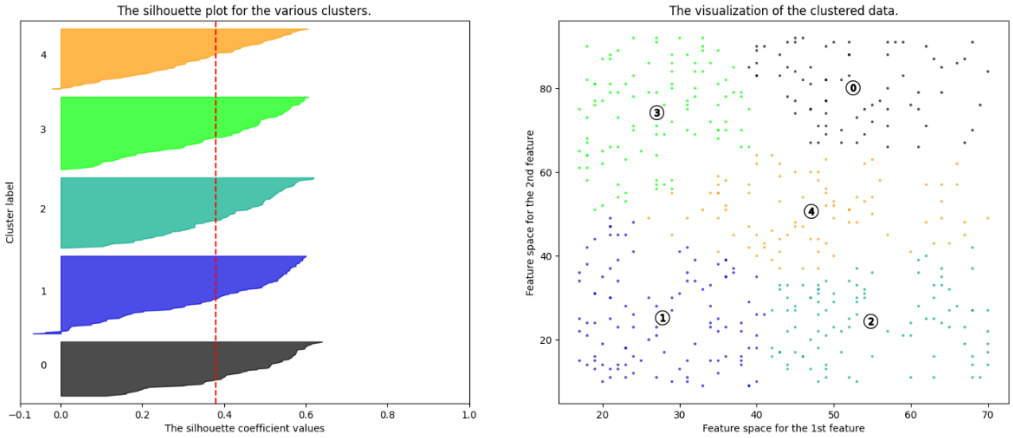




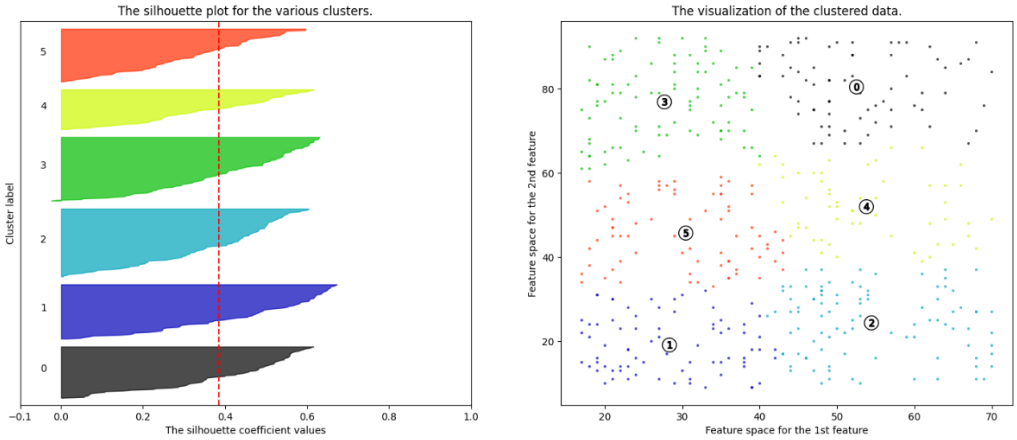
Silhouette analysis for KMeans clustering on sample data with n_clusters = 4



Silhouette analysis for KMeans clustering on sample data with n_clusters = 5



Silhouette analysis for KMeans clustering on sample data with n_clusters = 6



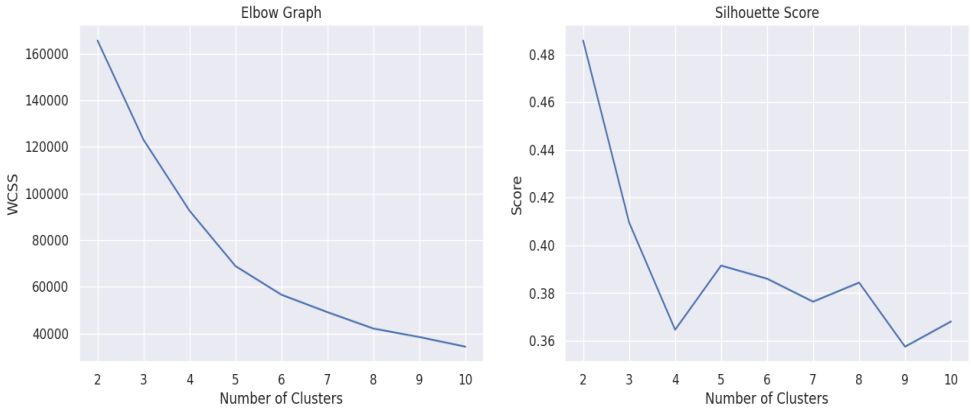


Figure 6.18: Elbow Method and Silhouette Score for optimal number of Clusters

From this elbow graph and silhouette graphs we can say that the optimum number of clusters is 5.

The final scatter plot clusters look like –

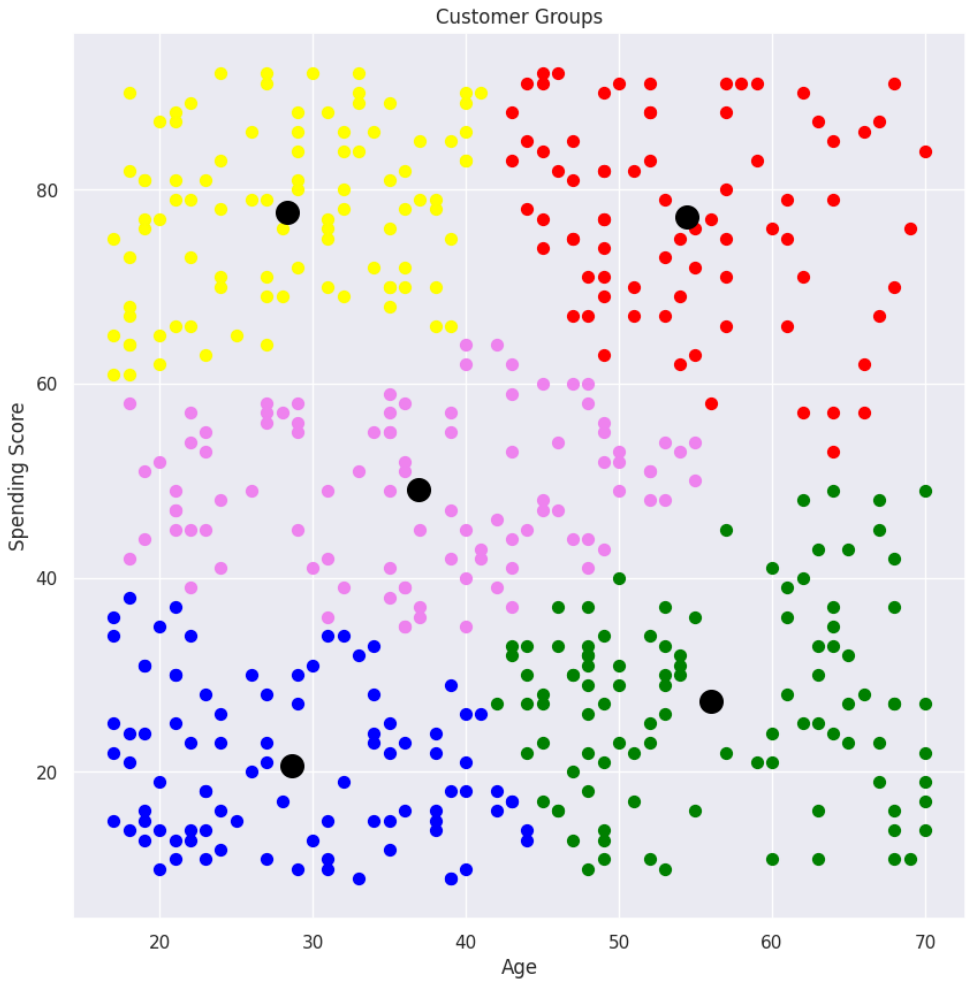


Figure 6.19: Final scatter plot

Bar graph representation of the Clusters :

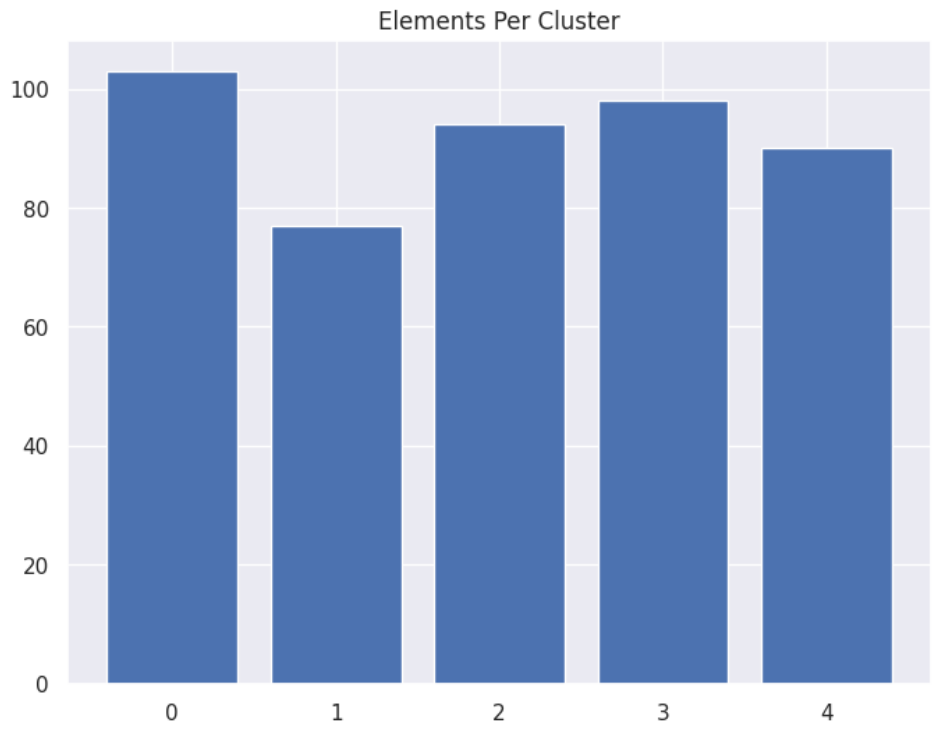


FIGURE 6.20: Elements per cluster

Here,

0.0 – Green Cluster – This cluster has most number of customers, which suggests that people of older ages with a low spending score tend to be more invested in this business which suggests that the products of this business are quite budget friendly.

1.0 – Red cluster – This is the least concentrated cluster depicting that customers with older age and higher spending score are less invested in buying products from this business as compared to other customer groups.

2.0 – Yellow Cluster – This is the third most concentrated cluster comprising of people of lower age and a high spending score.

3.0 – Violet Cluster – This is the second most concentrated cluster comprising of people of almost all ages with a medium spending score.

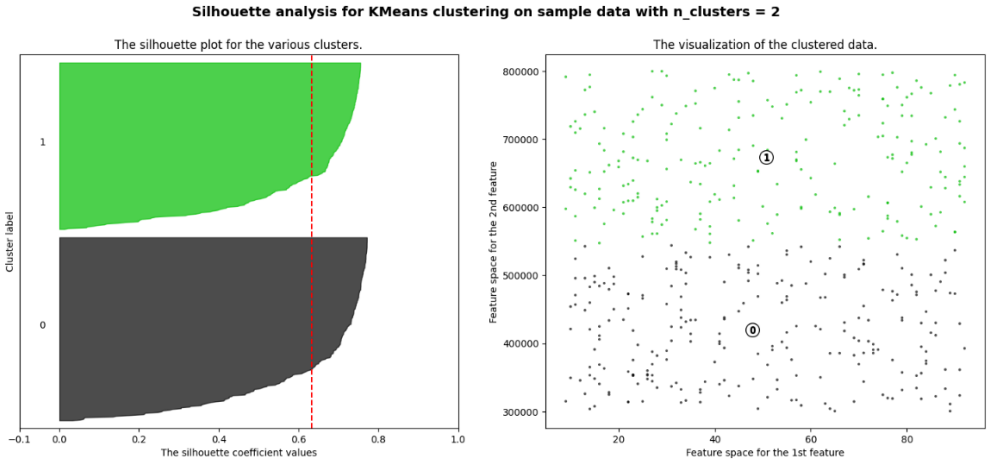
4.0 – Blue Cluster – This group is comprised of younger people with a low spending score and is the second most scattered after the red cluster.

CASE 3 : In the third program we applied K-Means Clustering on the parameters –

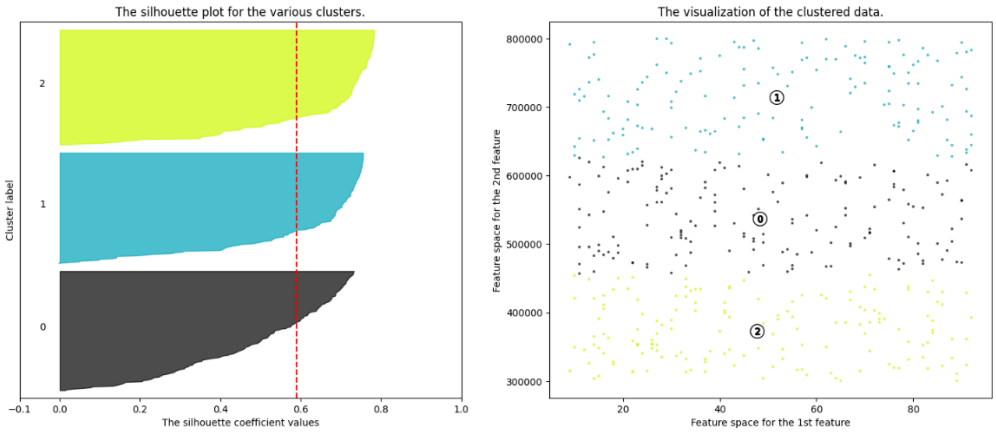
Pin-Code vs Spending score and the results looked like :-

Figure 6.21 :proves how 6 is the optimal number of clusters.

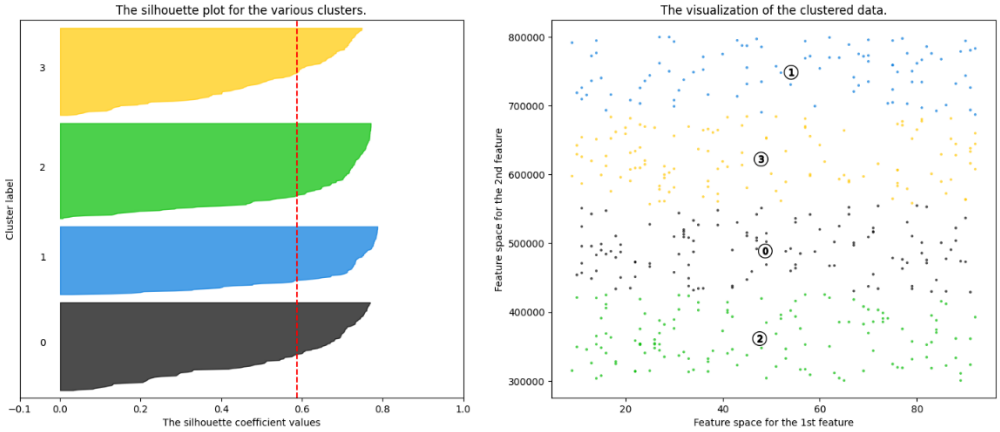
```
For n_clusters = 2 The average silhouette_score is :
                 0.6332393184473132

For n_clusters = 3 The average silhouette_score is :
                 0.5888768884614077

For n_clusters = 4 The average silhouette_score is :
                 0.5869725890224581

For n_clusters = 5 The average silhouette_score is :
                 0.5738630509939165

For n_clusters = 6 The average silhouette_score is :
                 0.5851475760744748
```



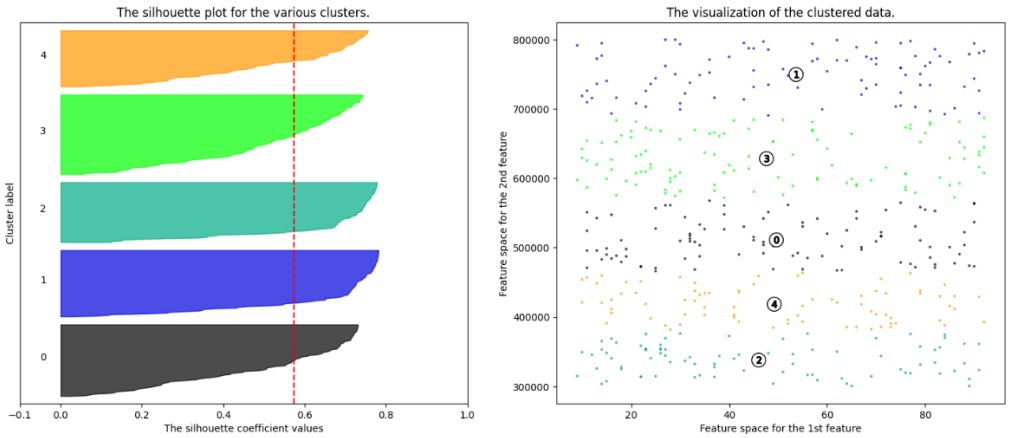
Silhouette analysis for KMeans clustering on sample data with n_clusters = 3



Silhouette analysis for KMeans clustering on sample data with n_clusters = 4



Silhouette analysis for KMeans clustering on sample data with n_clusters = 5



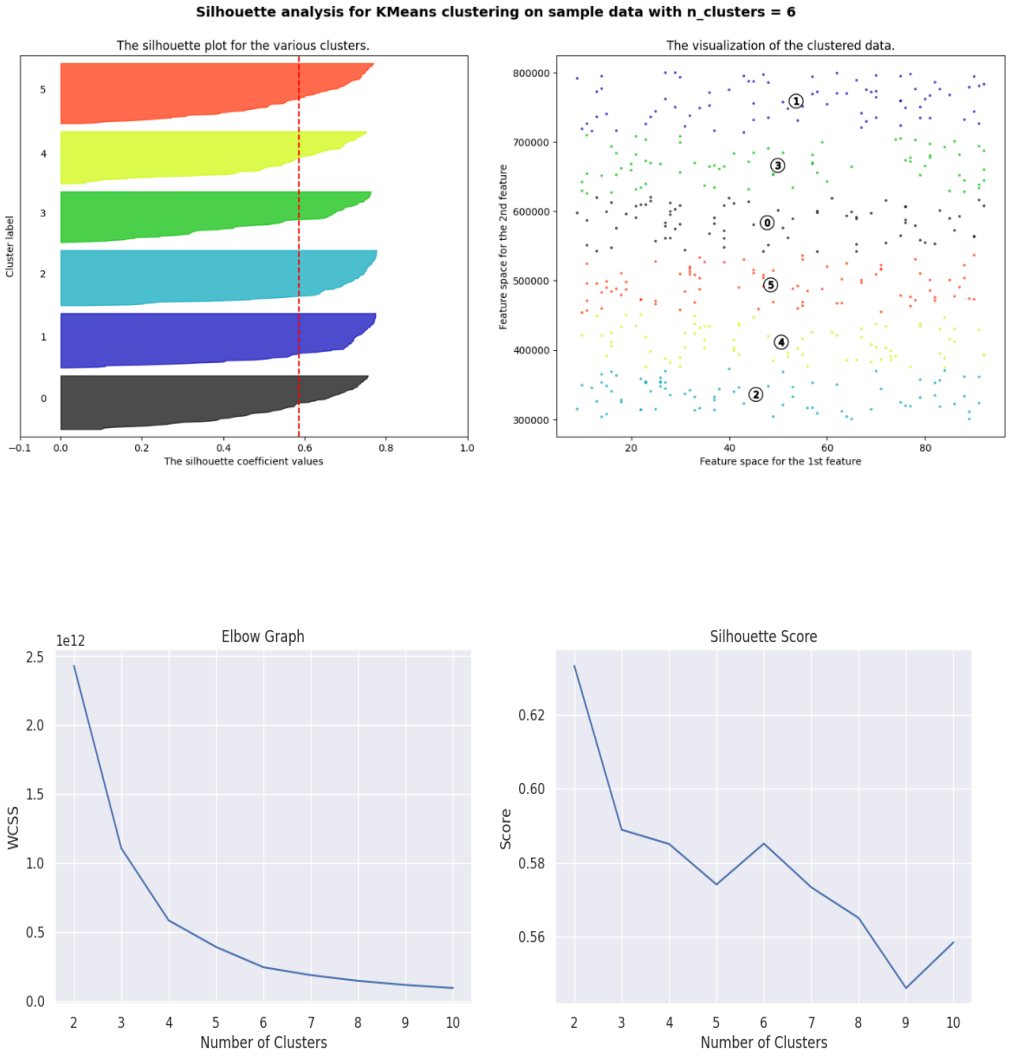


Figure 6.22: Elbow Method and Silhouette Score for optimal number of Clusters

From the elbow graph and silhouette method we can say that the optimum number of clusters here are 6.

The final scatter plot clusters look like –

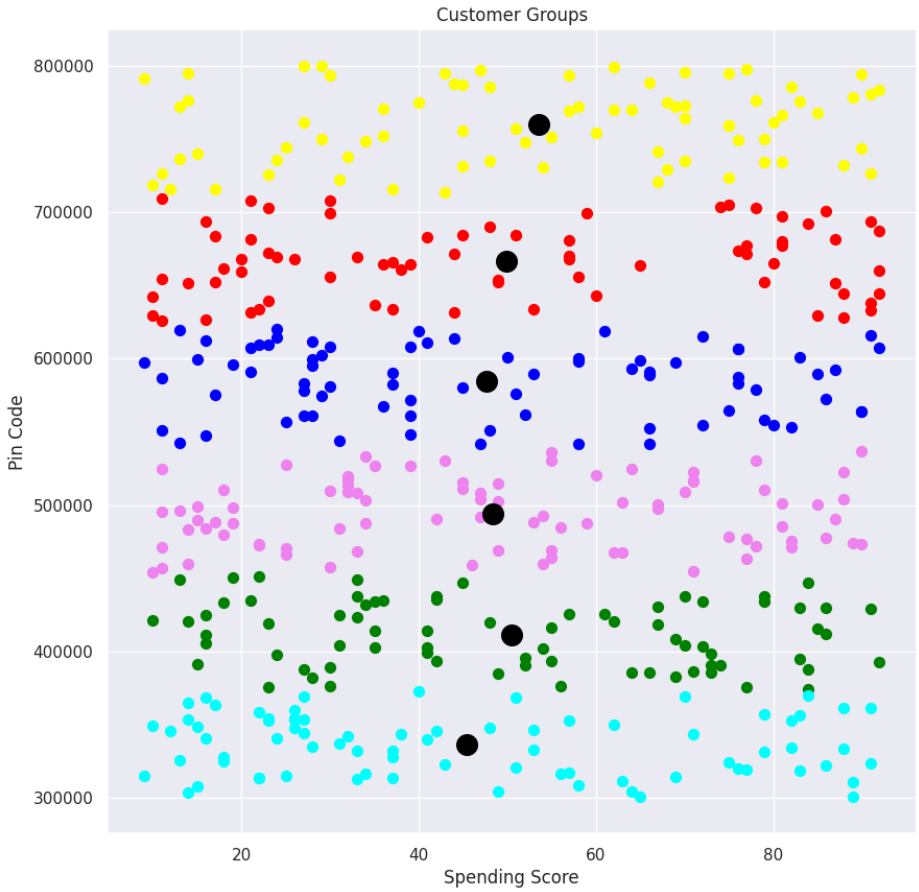


Figure 6.23: Final scatter plot

Bar graph representation of the Clusters :

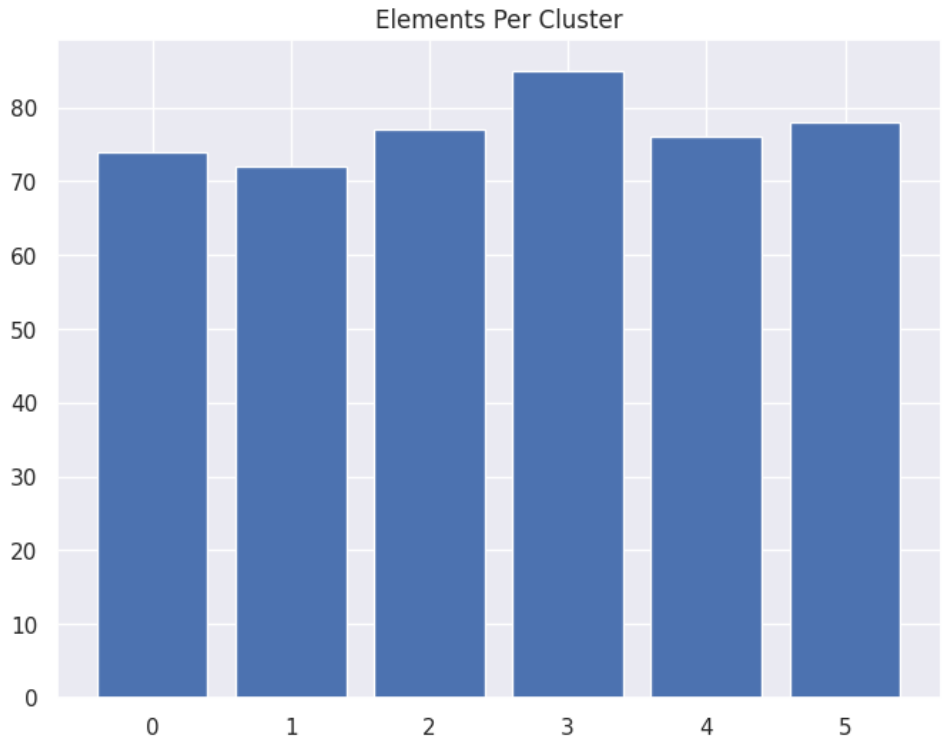


Figure 6.24: Elements per cluster

Here,

- 0.0– Green Cluster –Customers from this region form a concentrated cluster suggesting that they are fairly invested in buying products from this business.
- 1.0– Red Cluster –This is the least concentrated cluster suggesting that customers from this region are less interested in buying products from this business as compared to customers from other regions.
- 2.0– Yellow Cluster –This is the third most concentrated cluster suggesting that they are quite invested in buying products from this business.
- 3.0– Violet Cluster –Most people from this region are invested in buying products from this business as this forms the most concentrated cluster.
- 4.0 - Blue Cluster- Customers from this region form a fairly concentrated cluster suggesting that they are fairly invested in buying products from this business.
- 5.0 – Cyan Cluster – This group of customers, residing in this region form the second most concentrated cluster suggesting that they are quite invested in buying products from this business.

CHAPTER 7: CONCLUSION

Customer segmentation is an essential marketing tactic that aims to divide a broader target market into smaller groups of consumers with shared characteristics and needs. This enables businesses to understand the distinct preferences and behaviors of different customer groups, allowing them to tailor their marketing strategies to meet the specific needs of each segment. The methods used in this research paper include behavioral and geographic segmentation.

A *k*-means clustering algorithm, which is a non-hierarchical method, was used. For computing *k*-means clustering, the initial cluster centers were chosen, and then final stable cluster centers were computed by continuing number of iterations until means had stopped further changing with next iterations. This convergent condition was also achieved by setting a threshold value for change in the mean. The final cluster centers contained the mean values for each variable in each cluster. Computing based system developed was intelligent and it automatically presented results to the mangers to infer for quick and fast decision-making process. The simulation tests were also computed for cluster brands and other characteristics of the cluster representing a particular class of people .Benefits segmentation is another approach that focuses on the benefits customers seek from a product or service. By identifying the primary benefits that customers seek, businesses can cater to the specific needs of each customer group more effectively. In conclusion, customer segmentation is an invaluable marketing strategy that enables businesses to understand the diverse needs and preferences of their customers. By implementing different approaches to segmentation, businesses can tailor their marketing campaigns to meet the specific needs of each customer segment, leading to greater customer engagement, loyalty, advertisement and sales.

7.1 Limitations of the System

The number of clusters is determined arbitrarily: Our code uses the elbow method and silhouette score to determine the number of clusters, but this is not always the most effective method. Sometimes, a better way to determine the number of clusters is to use domain knowledge or hierarchical clustering.

It only considers two features: Our code only uses two features, spending score and pin code(in geographical segmentation), age and spending score, income and spending(in behavioral segmentation), to cluster the customers. This may not be sufficient to capture the complexity of customer behavior. It is important to consider more features that are relevant to the business problem.

It assumes that clusters are spherical and equally sized: K-means assumes that clusters are spherical and equally sized, which may not be true in all cases. If the clusters have complex shapes or different sizes, K-means may not be the best clustering algorithm to use.

7.2 Future Scope of the Project

This project has four broad scopes that can be worked upon more in the near future.

1. Different methods could be implemented to determine the number of clusters: In addition to the elbow method and silhouette score, other methods such as the gap statistic, or hierarchical clustering could have been tried. These methods may provide a better estimate of the number of clusters.
2. More features could have been considered: More relevant features that can help capture the complexity of customer behavior could have been included. This can improve the accuracy and usefulness of the clustering results.
3. Different clustering algorithms: If K-means is not suitable for the dataset, you other clustering algorithms such as DBSCAN, hierarchical clustering, or Gaussian mixture models could also have been implemented. These algorithms can handle clusters with complex shapes and different sizes.
4. Evaluation of the clustering results: Always evaluating the quality of the clustering results to ensure that they are meaningful and useful for the business problem. Metrics such as purity, or completeness could be used to evaluate the results. Additionally, we could use visualization techniques to explore the clusters and identify any anomalies or outliers.

REFFERENCES

[1] Jayant Tikmani, Sudhanshu Tiwari, Sujata Khedkar "Telecom customer segmentation based on cluster analysisAn Approach to Customer Classification using k-means", IJIRCCE,Year: 2015.

[2] Tushar Kansal, Suraj Bahuguna ,Vishal Singh, Tanupriya Choudhury, "Customer Segmentation using K-means Clustering", International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS),2018.

[3] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient K-means clustering algorithm," IEEE Trans. Pattern Analysis and Machine Intelligence,vol. 24, pp. 881-892, 2002.

[4] M. Inaba, N. Katoh, and H. Imai, "Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering," in
Proc.10th ACM Symposium on Computational Geometry, 1994, pp.332-339.

[5] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "NP-hard Euclidean sum-of-squares clustering," Machine Learning, vol. 75, pp. 245-249, 2009

[6] Chinedu Pascal Ezenkwu, Simeon Ozuomba, Constance kalu Electrical/Electronics & Computer Engineering Department, University of Uyo, Uyo, Akwa Ibom State, Nigeria "Application of K-Means Algorithm for Efficient Customer Segmentation: A Strategy for Targeted Customer Services", IJARAI,Year: 2015.

[7] SulekhaGoyat"The basis of market segmentation: a critical review of literature", EJBM,Year: 2011.

[8] Vaishali R. Patel and Rupa G. Mehta "Impact of Outlier Removal and Normalization Approach in Modified k-Means Clustering Algorithm", IJCSI,Year: 2011.

[9]Scikit-learn: https://scikit-learn.org

[10] Tanupriya Choudhury, Vivek Kumar, Darshika Nigam, Intelligent Classification & Clustering Of Lung & Oral Cancer through Decision Tree & Genetic Algorithm, International Journal of Advanced Research in Computer Science and Software Engineering,2015

[11] Tanupriya Choudhury, Vivek Kumar, Darshika Nigam, An Innovative and Automatic Lung and Oral Cancer Classification Using Soft Computing Techniques, International Journal of Computer Science & Mobile Computing,2015

[12]Puwanenthiren Premkanth, ―Market Segmentation and Its Impact on Customer Satisfaction with Especial Reference to Commercial Bank of Ceylon PLC.‖ Global Journal of Management and Business Research Publisher: Global Journals Inc. (USA). 2012. Print ISSN: 0975-5853. Volume 12 Issue 1.

[13]Xu, R., and Wunsch, D. (2005). "Clustering in Data Mining: A Survey." IEEE Transactions on Knowledge and Data Engineering, 16(3), 303-316. doi: 10.1109/TKDE.2004.68

[14]MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, 1(14), 281-297.

[15]Tan, P.N., Steinbach, M., and Kumar, V. (2006). "Introduction to Data Mining." Addison-Wesley, Boston, MA.

[16]Jain, A., Murty, M.N., and Flynn, P.J. (1999). "Data Clustering: A Review." ACM Computing Surveys, 31(3), 264-323.

[17] B. Huang, M. T. Kechadi, and B. Buckley, "Customer churn prediction in telecommunications, "Expert Systems with Applications, vol. 39, no. 1, pp. 1414–1425.

[18] M. Farhadloo, R. A. Patterson, and E. Rolland, "Modelling customer satisfaction from unstructured data using a Bayesian approach," Decision Support Systems, vol. 90, pp. 1–11.

[19] L. Luan and H. Shu, "Integration of data mining techniques to evaluate promotion for mobile
customers' data traffic in data plan," 2016 13th International Conference on Service Systems and
Service Management (ICSSSM), 2016.

[20] A. A. Khan, S. Jamwal, and M. Sepehri, "Applying Data Mining to Customer Churn Prediction in an Internet Service Provider," International Journal of Computer Applications, vol. 9, no. 7, pp. 8–14.

[21] Y. Huang and T. Kechadi, "An effective hybrid learning system for telecommunication churn prediction," Expert Systems with Applications, vol. 40, no. 14, pp. 5635–5647.

[22] Kishana R. Kashwan," Customer Segmentation Using Clustering and Data Mining Techniques" in International Journal of Computer Theory and Engineering · January 2013.

[23] A. Vattani, "K-means exponential iterations even in the plane", Discrete and Computational Geometry, vol. 45, no. 4, pp. 596-616,
2011.

[24] C. Elkan, "Using the triangle inequality to accelerate K-means," in Proc. the 12th International Conference on Machine Learning (ICML), 2003.

[25] H. Zha, C. Ding, M. Gu, X. He, and H. D. Simon, "Spectral Relaxation for K-means Clustering," Neural Information Processing Systems, Vancouver, Canada, vol.14, pp. 1057-1064, 2001.