Customer Segmentation for Big Bazaar

Mini Project Report

Department of Electronics & Communication Engineering
Thapar Institute of Engineering & Technology, Patiala

Group Members

Saksham Sohal	1024060114
Abhay Singh Minhas	1024060118
Ashmita	1024060121
Samaira Nayyar	1024060080
Swastik Bhanot	1024060116

Submitted To

Prof. Sandeep Mandia

Submitted in partial fulfillment of the requirements for the course:
UCS 321: AI Fundamentals for Engineers

15th September 2025

Contents

List of Figures

List of Tables

Abstract

In the competitive retail landscape, understanding customer behavior is paramount. This report presents a customer segmentation analysis for Big Bazaar, aimed at identifying and profiling distinct customer groups. The project leverages a public dataset of mall customers, which was rigorously pre-processed through outlier detection, feature engineering, and scaling. We conducted a comparative study of three clustering algorithms: K-Means, Agglomerative Hierarchical Clustering, and DBSCAN. The performance was evaluated using metrics like the Silhouette Score, where the K-Means model emerged as the most effective with a score of **0.501**. The model successfully partitioned the customer base into six meaningful segments. The primary outcome of this analysis is the identification and detailed profiling of these segments, providing Big Bazaar with a foundational understanding of their key customer archetypes.

Chapter 1

Introduction

1.1 Background

In today's hyper-competitive retail environment, the "one-size-fits-all" marketing approach is no longer effective. Modern businesses like Big Bazaar serve a diverse customer base, and treating everyone the same leads to wasted resources and missed opportunities. The key to unlocking growth lies in understanding the distinct subgroups within this broad audience. This is where the strategic implementation of unsupervised machine learning, specifically clustering models, becomes a critical business tool. Clustering algorithms are designed to analyze complex datasets and automatically identify natural groupings, or "clusters," of customers based on their shared characteristics. The primary objective of this project is to leverage these powerful algorithms to partition Big Bazaar's customers into well-defined personas. By doing so, we can provide a foundational, data-driven understanding of the customer base, enabling more precise and effective business decisions.

1.2 Problem Statement

Big Bazaar aims to segment its customers into distinct groups to optimize promotional strategies and improve sales. Using customer purchase history (annual income, spending score, and visit frequency etc.), develop a Python-based clustering model to identify customer segments. The project should include:Data cleaning and pre-processing, Performance parameters etc.

1.3 Objective

The primary contributions of this work are centered around the following objectives:

- **Develop a Robust Pre-processing Pipeline:** A key contribution was the establishment of a comprehensive pre-processing methodology. This involved not only standard procedures like data cleaning and scaling but also innovative feature engineering to create a novel Visit Frequency metric, adding a valuable behavioral dimension to the original dataset.
- **Conduct a Comparative Analysis of Clustering Algorithms:** Rather than relying on a single method, we have used three clustering methods to provide a comparative analysis of three distinct clustering paradigms—K-Means, Hierarchical, and DBSCAN. This multi-model approach ensures that the identified segments are robust and not merely an artifact of one specific algorithm.
- **Identify and Profile Data-Driven Customer Personas:** The central contribution of this work is the successful identification and detailed profiling of six distinct customer personas. The characteristics of each segment have been thoroughly analyzed and documented, transforming raw data into an understandable and actionable business asset.
- **Quantify Model Performance:** To validate our findings, we established a rigorous evaluation framework using the Silhouette Score metric, providing a quantitative justification for our final model selection.

Chapter 2

Data Pre-Processing and Exploratory Data Analysis

This chapter details the foundational steps taken to prepare the dataset for analysis. A rigorous pre-processing and exploration phase is essential to ensure the quality of the input data and to gain initial insights into its underlying structure, which in turn leads to a more robust and meaningful clustering model.

2.1 Dataset Description

The analysis is based on the "Mall Customer Segmentation" dataset, a publicly available resource from Kaggle. This dataset contains basic demographic and spending information for 200 anonymous customers of a shopping mall. The primary features include Age, Gender, Annual Income (in k$), and a proprietary Spending Score (ranging from 1 to 100) assigned by the mall based on customer behavior.

CustomerID	**Gender**	**Age**	**Annual Income (k$)**	**Spending Score (1–100)**
1	Male	19	15	39
2	Male	21	15	81
3	Female	20	16	6
4	Female	23	16	77
5	Female	31	17	40

Table 2.1: *Sample of the Raw Dataset*

2.2 Exploratory Data Analysis (EDA)

The initial phase of EDA was conducted to understand the distributions of the key features and the relationships between them. Histograms revealed that both Annual Income and Spending Score have reasonably normal distributions, indicating a good spread of customer types. The most revealing insight from the EDA came from visualizing the relationship between Annual Income and Spending Score, as shown in Figure 2.1. This two-dimensional plot immediately suggested the presence of natural, distinct groupings within the data, providing a strong preliminary indication that clustering would be an effective approach for segmentation.

Figure 2.1: *Annual Income vs. Spending Score*

2.3 Pre-Processing Pipeline

To ensure the data was of high quality and properly formatted for our clustering algorithms, we implemented a comprehensive, multi-step pre-processing pipeline.

2.3.1 Flowchart

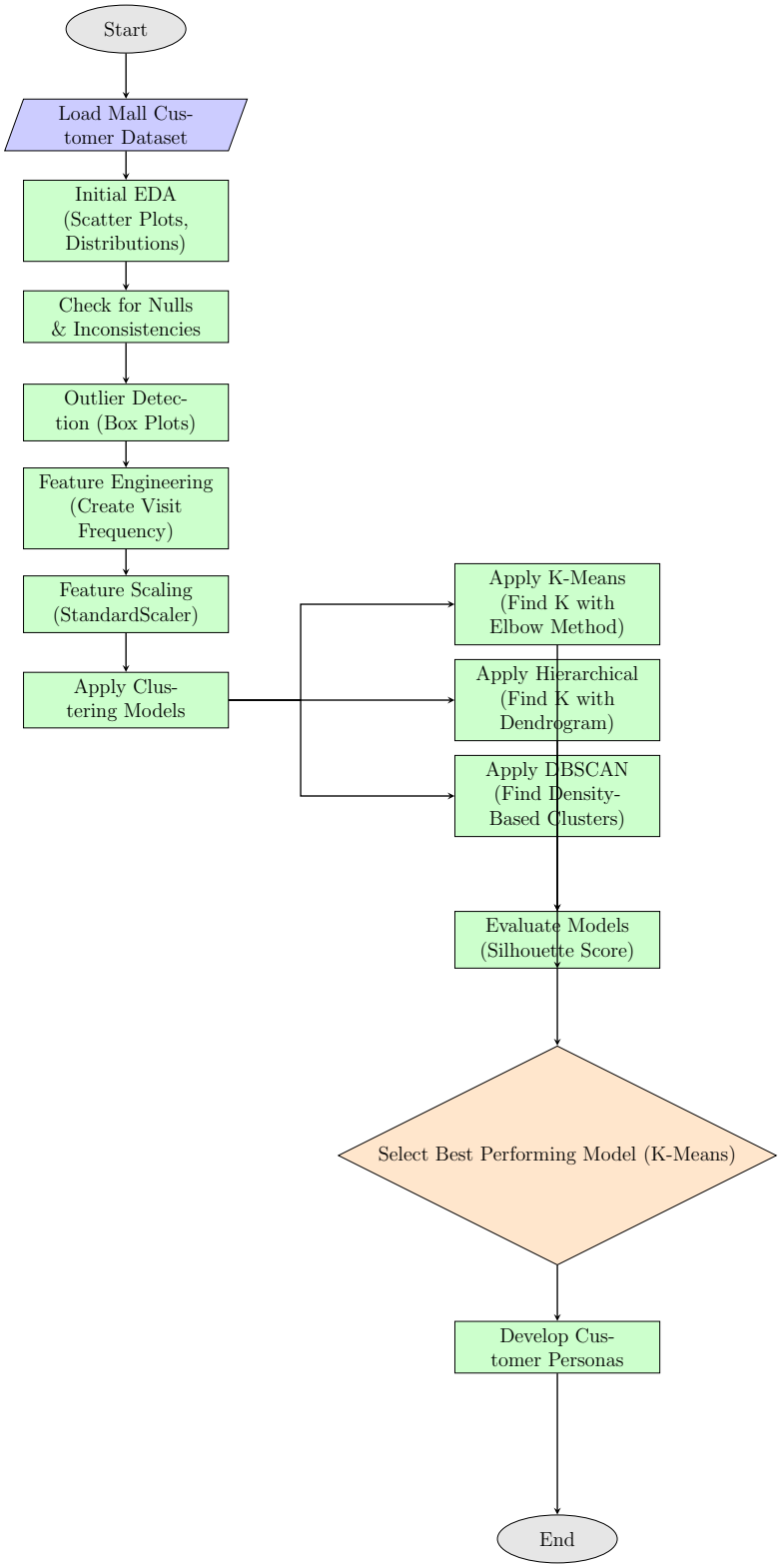

Figure 2.2: *Flowchart of the Clustering Model*

2.3.2 Data Cleaning

The first step was to check for missing or null values. Using the df.info() command, we verified that all 200 entries were complete across all features. The dataset was found to be clean, requiring no imputation or removal of records.

2.3.3 Outlier Detection

We utilized box plots to visually inspect the numerical features for statistical outliers. While the distributions for Age and Spending Score were well-contained, a few potential outliers were observed at the upper end of the Annual Income feature. After careful consideration, the decision was made to retain these data points. In the context of customer segmentation, these "outliers" represent the highest-earning customers and are a legitimate and valuable segment of the customer base that must be included in the analysis.

Figure 2.3: *Box Plots for Outlier Detection*

2.3.4 Inconsistency Check

The categorical Gender column was checked for inconsistencies (e.g., typos, mixed case). The data was found to be consistent, containing only 'Male' and 'Female' as unique entries.

2.3.5 Feature Engineering

To enrich the dataset and provide more dimensions for analysis, two new features were engineered:

- **Visit Frequency (Monthly):** A synthetic behavioral metric was created to simulate how often customers visit per month. This feature was logically correlated with the Spending Score to add a realistic layer of behavioral data.
- **Age Group:** The continuous Age variable was binned into descriptive categories (e.g., '18-25', '26-40'). This transformation was performed to facilitate the final interpretation and creation of customer personas.

2.3.6 Feature Scaling

As a final and critical pre-processing step, the numerical features selected for clustering (Annual Income, Spending Score, and Visit Frequency) were scaled. This was accomplished using the **Z-score method**, which was implemented with the **StandardScaler** object from the scikit-learn library. This transformation is essential because distance-based algorithms, such as K-Means, are highly sensitive to the scale of the data. Without scaling, features with larger numerical ranges would disproportionately influence the distance calculations, biasing the model's outcome. The Z-score method, also known as standardization, addresses this by transforming the data to have a mean of 0 and a standard deviation of 1. This is achieved by applying the following formula to each data point x:

$$z = \frac{x - \mu}{\sigma}$$

Chapter 3

Model Implementation and Evaluation

This chapter details the core of the analytical work, from determining the optimal number of customer segments to the implementation and rigorous evaluation of three distinct clustering algorithms. The goal of this comparative approach is to ensure that the final segmentation model is not only effective but also robust and well-justified.

3.1 Clustering Algorithms

To achieve a comprehensive analysis, we selected three algorithms representing different clustering paradigms:

3.1.1 K-Means Clustering

An unsupervised machine learning algorithm used to partition a dataset into a specified number of clusters (k), where each data point belongs to the cluster with the nearest mean (centroid). The algorithm works by iteratively assigning data points to the closest centroid and then recalculating the centroid's position, continuing until the cluster assignments no longer change. It's widely used for tasks like customer segmentation, data exploration, and image analysis, grouping similar data points based on their features to reveal hidden patterns.

Algorithm 1 K-Means Algorithm

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$
repeat
  for $i = 1$ to m **do**
    Cluster assignment:
      $c^{(i)} := \arg\min_{k \in \{1,\ldots,K\}} \|x^{(i)} - \mu_k\|^2$
  end for
  for $k = 1$ to K **do**
    Move centroid:
      $\mu_k := \frac{1}{|C_k|} \sum_{i \in C_k} x^{(i)}$
  end for
until convergence

3.1.2 Agglomerative Hierarchical Clustering

A bottom-up hierarchical method that starts with each data point as its own cluster and progressively merges the closest pairs of clusters until only one remains. It is particularly useful for visualizing the cluster hierarchy.

Algorithm 2 Agglomerative Hierarchical Clustering

Initialize each data point as its own cluster
while number of clusters $> K$ **do**
  Compute pairwise distances between all clusters
  Find the two closest clusters C_i and C_j
  Merge C_i and C_j into a new cluster
end while
Construct a dendrogram from the sequence of merges

3.1.3 DBSCAN

A density-based clustering algorithm that groups together points that are closely packed together, marking other points as outliers. It is effective at finding clusters of arbitrary shapes and doesn't require you to pre-specify the number of clusters. Instead, you provide two parameters: epsilon (ϵ), the maximum distance between two points for one to be considered as in the neighborhood of the other, and minPoints, the minimum number of points required to form a dense region.

Algorithm 3 DBSCAN Algorithm

Input: Dataset D, parameters ϵ, MinPts
for each point p in D **do**
  if p is a core point and not processed **then**
    $C := \{$all points density-reachable from $p\}$
    Mark all points in C as processed
    Report C as a cluster
  else
    Mark p as an outlier
  end if
end for

3.2 Finding the Optimal Number of Clusters

For the K-Means and Hierarchical models, a critical preliminary step is to determine the most appropriate number of clusters (K) for the dataset. Two distinct methods were employed to arrive at a data-driven decision.

3.2.1 Elbow Method (For K-Means)

The Elbow Method involves plotting the Within-Cluster Sum of Squares (WCSS) for a range of K values. The "elbow" of the resulting curve, where the rate of WCSS reduction sharply decreases, indicates the optimal K. As shown in Figure 3.1, the elbow is clearly visible at K=6, suggesting this is the most effective number of segments.

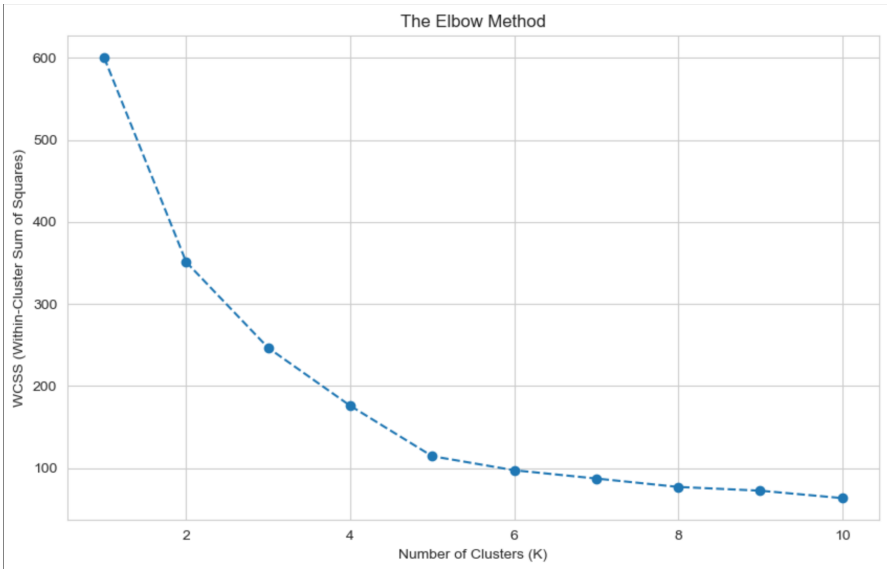

Figure 3.1: *The Elbow Method Plot*

3.2.2 The Dendrogram (Hierarchical Clustering)

The dendrogram provides a visual representation of the hierarchical merging process. By identifying the longest vertical line and making a horizontal "cut" across it, we can determine the optimal number of clusters. The dendrogram in Figure 3.2 corroborates the finding from the Elbow Method, also strongly suggesting that 6 clusters is the most natural grouping for the data.

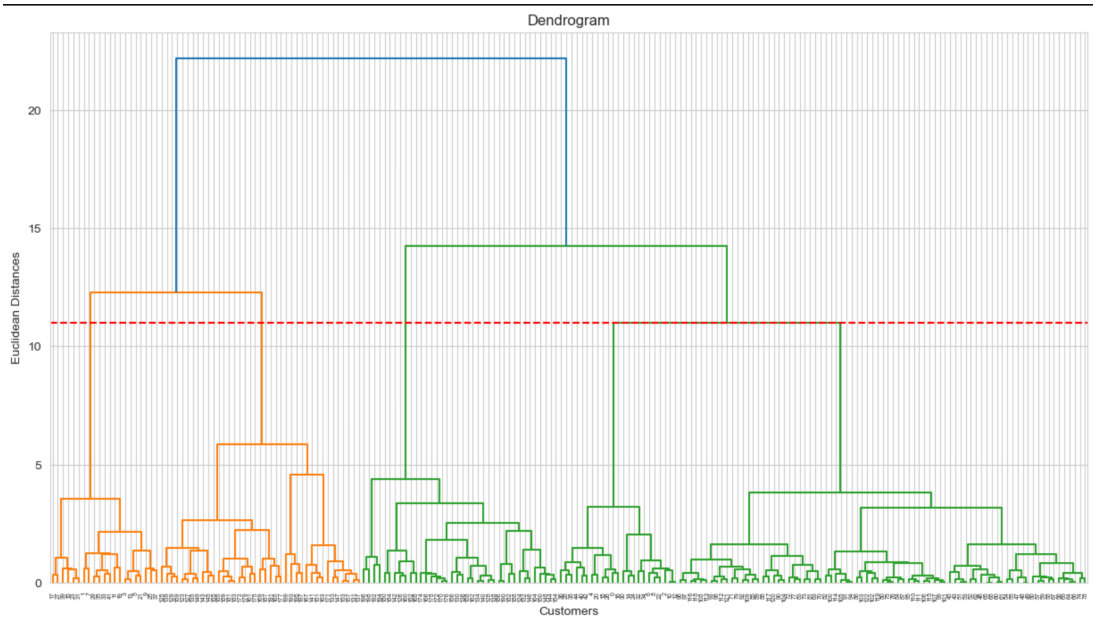

Figure 3.2: *Dendrogram for Cluster Number Validation*

3.3 Model Implementation and Results

All models were implemented in Python using the scikit-learn library on the pre-processed and scaled dataset.

- **K-Means & Hierarchical Models:** Both algorithms were configured to partition the data into the predetermined 6 clusters.
- **DBSCAN and Outlier Analysis:** The DBSCAN model, operating on density principles, provided a unique and powerful insight. It identified 5 core, dense clusters and classified 44 customers as outliers. This is a significant finding, as it reveals a substantial subgroup of customers with anomalous behaviours that do not conform to any primary segment.

3.4 Performance Evaluation

To objectively evaluate and compare the quality of the clusters produced by our models, we used the **Silhouette Score** as our primary performance metric. The Silhouette Score is a measure of how well-defined and distinct the clusters are. It calculates a score for each data point based on its similarity to its own cluster compared to other clusters. The score ranges from -1 to +1, where a value closer to +1 indicates that the clusters are dense and well-separated.

Model	**Silhouette Score (Higher is better)**	**Key Insight**
K-Means (k=6)	**0.501**	6 well-defined clusters
Hierarchical (k=6)	0.480	Confirmed 6-cluster structure
DBSCAN	N/A*	Found 5 core groups + 44 outliers

Table 3.1: *Model Performance using Silhouette Score*

*The Silhouette Score is not typically used as the primary metric for DBSCAN, as its main strength is identifying outliers.

Chapter 4

Segment Analysis and Persona Development

This chapter presents the final clustering results. Based on the previous evaluation, the K-Means model with six clusters was chosen to define customer segments. We visualize these clusters, analyze their key characteristics, and develop detailed personas for each group.

4.1 Visualization of Final Customer Segments

Figure 4.1 shows a 2D scatter plot of the six clusters using Annual Income and Spending Score with their centroids, while Figure 4.2 presents a 3D view including Visit Frequency for a multi-dimensional perspective.

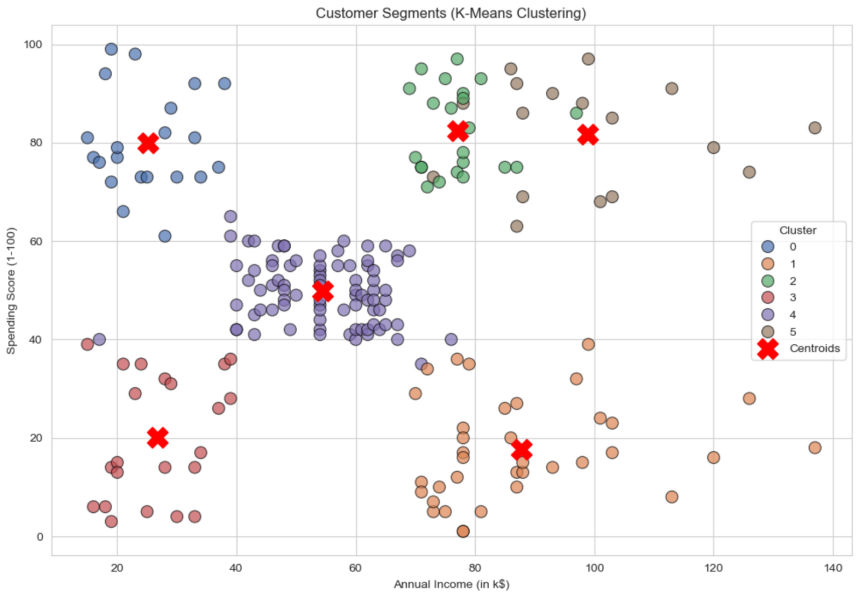

Figure 4.1: *2D Visualization of K-Means Clusters and Centroids*

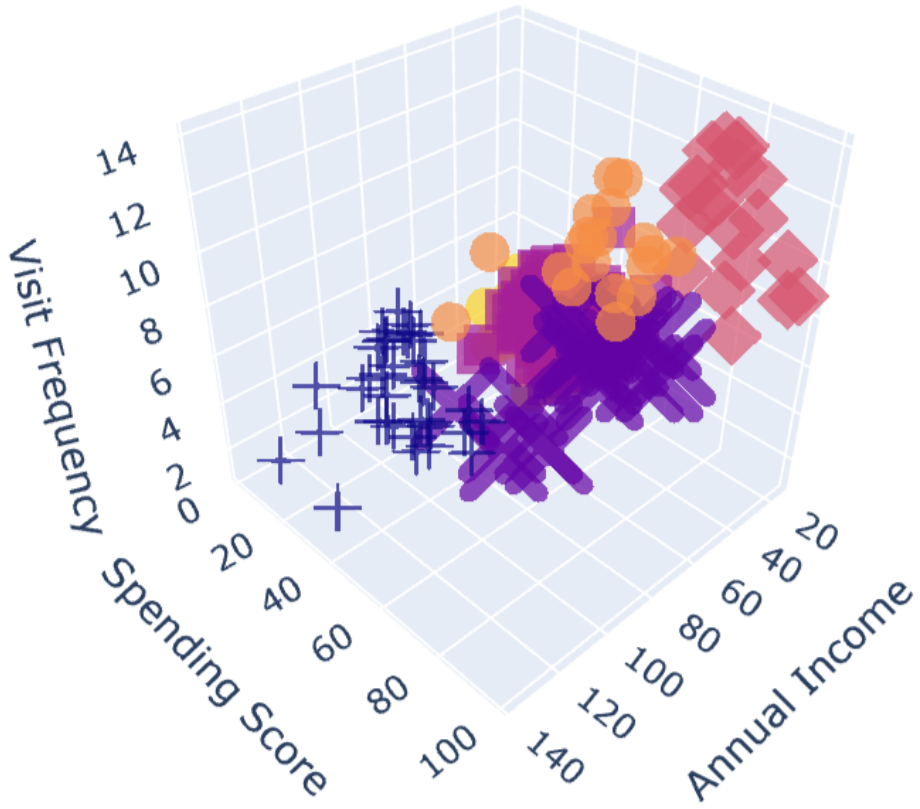

Figure 4.2: *3D Visualization of Customer Segments*

4.2 Customer Persona Profiles

To translate these clusters into actionable business insights, we analyzed the average characteristics of the customers in each group. The summary in Table 4.1 provides a quantitative foundation for developing the personas.

Cluster	Annual Income (k$)	Spending Score	Visit Frequency	Age	Age Group
4	85.84	82.58	12.63	32.05	26-40
1	87.20	81.70	8.35	33.30	26-40
3	24.95	81.00	10.95	24.85	18-25
2	54.12	49.90	6.52	42.25	41-60
5	26.73	20.05	2.86	45.86	41-60
0	87.72	17.61	3.00	41.17	41-60

Table 4.1: *Mean Characteristics of Customer Segments*

Based on this data, we have defined the following five customer personas:

4.2.1 Clusters 1 & 5: Affluent Spenders

This segment represents the most valuable customers. They are characterized by **high annual incomes** (averaging ∼$77k for Cluster 1 and ∼$108k for Cluster 5) and correspondingly **high spending scores (both ∼82)**. They are also frequent shoppers, visiting the mall often. These customers are the primary drivers of revenue.

4.2.2 Cluster 4: Young High Spenders

This group is defined by its youthful demographic (predominantly **18-25** years old). Despite having **low annual incomes (∼$25k)**, they exhibit a **very high spending score (∼81)**. They are highly engaged and likely purchase trendy, non-essential items.

4.2.3 Cluster 2: Core Customers

This is the central, middle-ground segment. They have **average annual incomes (∼$55k)** and **average spending scores (∼50)**. These customers are the stable and predictable backbone of the business, representing the "average" shopper.

4.2.4 Cluster 3: Budget-Conscious Segment

Characterized by **low annual incomes (∼$26k)** and **low spending scores (∼21)**, this segment is highly price-sensitive. They are likely focused on essential goods and value-driven purchases, visiting the least frequently of all segments.

4.2.5 Cluster 0: Discerning High Earners

This persona is one of the most intriguing. They possess **high annual incomes (∼$87k)** but have a **very low spending score (∼18)**. This indicates significant purchasing power combined with a discerning and needs-based shopping behavior. They are not easily swayed by mass-market promotions.

Chapter 5

Conclusion

In this project, we successfully developed and implemented a machine learning pipeline for customer segmentation. Through a comprehensive process of data pre-processing, innovative feature engineering, and a comparative analysis of three distinct clustering algorithms, we transformed raw customer data into a clear and structured understanding of the customer base.

The K-Means model proved most effective, and its application resulted in the identification of six distinct and well-characterized customer personas, ranging from the high-value "Affluent Spenders" to the price-sensitive "Budget-Conscious Segment." Furthermore, our analysis highlighted a significant "wildcard" segment of 44 anomalous shoppers via the DBSCAN model, revealing a nuanced layer of customer behavior that a single-model approach would have missed. Ultimately, this work provides a detailed, data-driven foundation for understanding the key archetypes within the customer population.

Bibliography

[1] E. Alpaydin, *Introduction to Machine Learning*, 4th ed. MIT Press, 2020.

[2] Kaggle Dataset: https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial-in-python

[3] Scikit-learn Documentation: https://scikit-learn.org/stable/unsupervised_learning.html

[4] Project Repository: https://github.com/SakshamSohal/UCS-321-AI-for-Engineers-Mini-Project

[5] Data and Pre-Processing: https://docs.google.com/document/d/1md14Dv1mXvwxx-QnbmBbfEylqNQoJxO85UcXB4XHO48/edit

Appendix A

Complete Python Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df = pd.read_csv("Mall_Customers.csv")
df.head()
```

```
df.info()
```

```
df.describe()
```

```
plt.figure(figsize=(15, 5))
plt.subplot(1, 3, 1)
sns.boxplot(y=df['Annual Income (k$)'])
plt.title('Box Plot for Annual Income')
plt.subplot(1, 3, 2)
sns.boxplot(y=df['Spending Score (1-100)'])
plt.title('Box Plot for Spending Score')
plt.subplot(1, 3, 3)
sns.boxplot(y=df['Age'])
plt.title('Box Plot for Age')
plt.tight_layout()
plt.show()
```

```
print("Unique values in 'Gender' column:", df['Gender'].unique())
```

```
np.random.seed(42)
df['Visit Frequency (Monthly)'] = np.random.randint(1, 6, size=len(df))
df.loc[df['Spending Score (1-100)'] > 70, 'Visit Frequency (Monthly)']
    = np.random.randint(8, 15, size=len(df[df['Spending Score (1-100)']
    > 70]))
```

```
df.loc[df['Spending Score (1-100)'].between(40, 70), 'Visit Frequency (Monthly)'] = np.random.randint(5, 9, size=len(df[df['Spending Score (1-100)'].between(40, 70)]))
```

```
bins = [18, 25, 40, 60, 70]
labels = ['18-25', '26-40', '41-60', '60+']
df['Age Group'] = pd.cut(df['Age'], bins=bins, labels=labels, right=False)
```

```
import os
os.environ["OMP_NUM_THREADS"] = "1"
from sklearn.preprocessing import StandardScaler

features_for_clustering = ['Annual Income (k$)', 'Spending Score (1-100)', 'Visit Frequency (Monthly)']
X = df[features_for_clustering]
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
from sklearn.cluster import KMeans

wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42, n_init=10)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

plt.figure(figsize=(10, 6))
plt.plot(range(1, 11), wcss, marker='o', linestyle='--')
plt.title('The Elbow Method')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('WCSS (Within-Cluster Sum of Squares)')
plt.grid(True)
plt.show()
```

```
from sklearn.metrics import silhouette_score

optimal_k = 6
kmeans = KMeans(n_clusters=optimal_k, init='k-means++', random_state=42, n_init=10)
y_kmeans = kmeans.fit_predict(X_scaled)
df['KMeans_Cluster'] = y_kmeans
kmeans_silhouette = silhouette_score(X_scaled, y_kmeans)
print(f"K-Means Silhouette Score: {kmeans_silhouette:.3f}")
```

```
from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=0.6, min_samples=8)
y_dbscan = dbscan.fit_predict(X_scaled)
df['DBSCAN_Cluster'] = y_dbscan
n_clusters_ = len(set(y_dbscan)) - (1 if -1 in y_dbscan else 0)
n_noise_ = list(y_dbscan).count(-1)
print(f'DBSCAN found {n_clusters_} clusters and {n_noise_} outliers.')
```

```
import scipy.cluster.hierarchy as sch

plt.figure(figsize=(15, 8))
dendrogram = sch.dendrogram(sch.linkage(X_scaled, method='ward'))
plt.title('Dendrogram')
plt.xlabel('Customers')
plt.ylabel('Euclidean Distances')
plt.show()
```

```
from sklearn.cluster import AgglomerativeClustering

agg_cluster = AgglomerativeClustering(n_clusters=6, linkage='ward')
y_agg = agg_cluster.fit_predict(X_scaled)
df['Agg_Cluster'] = y_agg
agg_silhouette = silhouette_score(X_scaled, y_agg)
print(f"Agglomerative Clustering Silhouette Score:
    {agg_silhouette:.3f}")
```

```
plt.figure(figsize=(12, 8))
sns.scatterplot(
    data=df,
    x='Annual Income (k$)',
    y='Spending Score (1-100)',
    hue='KMeans_Cluster',
    palette='deep',
    s=100,
    alpha=0.7,
    edgecolor='k'
)
centroids = scaler.inverse_transform(kmeans.cluster_centers_)
plt.scatter(
    x=centroids[:, 0],
    y=centroids[:, 1],
    s=300,
    c='red',
    marker='X',
    label='Centroids'
```

```
)
plt.title('Customer Segments (K-Means Clustering)')
plt.legend(title='Cluster')
plt.savefig('kmeans_clusters.png', dpi=300)
plt.show()
```

```
import plotly.express as px

fig = px.scatter_3d(df,
    x='Annual Income (k$)',
    y='Spending Score (1-100)',
    z='Visit Frequency (Monthly)',
    color='KMeans_Cluster',
    symbol='KMeans_Cluster',
    size_max=10,
    opacity=0.7
)
fig.update_layout(
    title_text='Interactive 3D Customer Segments',
    scene=dict(
        xaxis=dict(title='Annual Income'),
        yaxis=dict(title='Spending Score'),
        zaxis=dict(title='Visit Frequency')
    )
)
fig.show()
```

```
cluster_analysis = df.groupby('KMeans_Cluster').agg({
    'Annual Income (k$)': 'mean',
    'Spending Score (1-100)': 'mean',
    'Visit Frequency (Monthly)': 'mean',
    'Age': 'mean',
    'Age Group': lambda x: x.mode()[0]
}).reset_index()

cluster_analysis.sort_values(by='Spending Score (1-100)',
   ascending=False)
```