Clustering and Data Fitting Report

Prepared by : Harshitha Mokide

Student ID : 23065647

https://github.com/harshi460/Cluster-and-Fitting

Introduction

The goal of this analysis is to explore and apply clustering techniques and data-fitting methods to segment customers and analyze their spending behaviors. The dataset used is the Mall Customer Segmentation dataset from Kaggle, focusing on clustering based on annual income and spending scores, and fitting a linear regression model to analyze relationships between these variables.

Data Preprocessing

Dataset Overview: The dataset contains 200 entries, each representing a customer, with attributes such as Gender, Age, Annual Income (k$), and Spending Score (1-100).

Selected Features:

- Annual Income (k$)

- Spending Score (1-100)

Standardization: To ensure fair clustering, features were standardized to remove scale biases, transforming both variables to have a mean of 0 and standard deviation of 1.

Key Insight: Standardization is crucial to balance the influence of features with different scales (e.g., income in thousands vs. spending scores on a scale of 1-100).

Clustering Using K-Means

Elbow Method

This code from line 3 to 7 is testing how good clustering is for different numbers of groups (clusters) ranging from 1 to 10.

- n_clusters=k: This means the number of clusters is being set for each value of k (e.g., 2 clusters, 3 clusters, etc.).
- kmeans.fit(features_scaled): The K-Means algorithm tries to split the data into these groups by assigning similar data points to the same cluster.
- inertia.append(kmeans.inertia_): After clustering, we calculate "inertia," which tells us how close the points in a cluster are to each other. Lower values mean better clustering.

Implementation:

- The Elbow Method evaluates the Within-Cluster Sum of Squares (WCSS) or "inertia" for different cluster counts.

- Lower WCSS indicates tighter clusters, but adding too many clusters may lead to overfitting.

- By plotting WCSS against the number of clusters, the "elbow point" reveals the optimal number where diminishing returns set in.

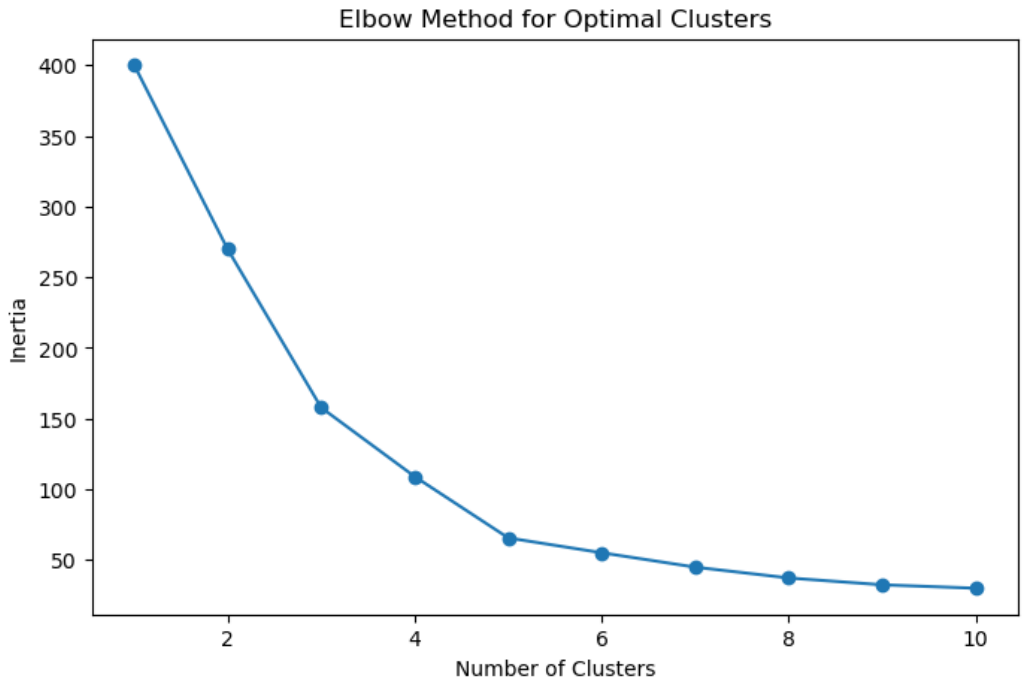


Figure 4 : Elbow method for optimal number of clusters.

Result

- The elbow curve shows a clear "elbow" at 4 clusters, indicating this as the optimal choice.

Real-World Implications:

- Businesses can segment customers into manageable groups to better understand and target them.

- For example, retail stores can use clustering to personalize promotions for high-value customers (e.g., high-income, high-spending groups).

K-Means Clustering

Objective: Partition customers into distinct segments based on spending behavior and income.

The line 2 and 3 is applying K-Means clustering with 4 clusters (based on the Elbow Method result).

- fit_predict(features_scaled): The algorithm divides the data into 4 groups and assigns each data point to a cluster.
- data['Cluster']: The cluster assignments are added to the dataset, so we know which group each customer belongs to.

Line 6 creates a scatter plot where:

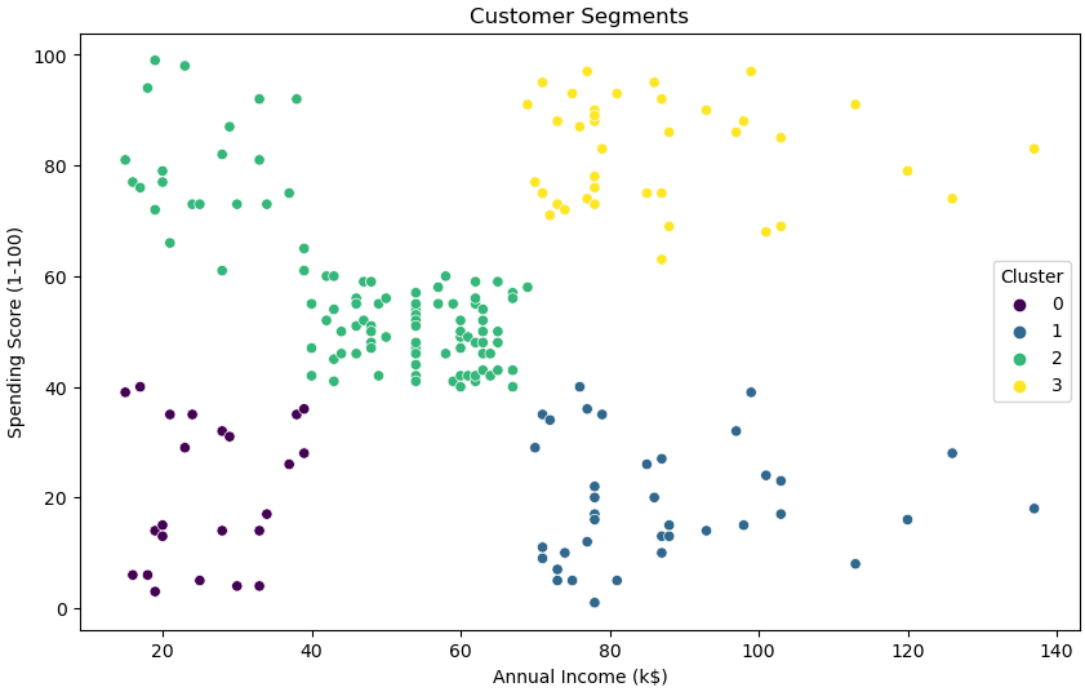
- The X-axis is **Annual Income**.
- The Y-axis is **Spending Score**.
- Each point represents a customer, and the color (hue) shows which cluster they belong to.

Implementation:

- K-Means clustering assigns each data point to the nearest cluster centroid.

- Iteratively, the centroids are recalculated to minimize the sum of squared distances within each cluster.

Figure 6 : Spending score across customer segments



Result

- Data points were grouped into 4 distinct clusters.

- Clusters showed clear separation, indicating meaningful groupings based on income and spending scores.

Real-World Implications:

- Retail chains can use clustering to create targeted loyalty programs for specific customer groups.

- Travel companies can identify high-spending customers for exclusive packages.

Linear Regression Analysis

Objective

To determine the relationship between annual income and spending scores.

Line 6 and 7 calculates a straight-line relationship between annual income (X) and spending score (y).

- fit(X, y): The algorithm finds the best-fitting straight line that minimizes the difference between the actual spending scores and the predicted scores from the line.

Line 10 - The model uses the straight line it calculated earlier to predict spending scores for each customer based on their income.

Line 15 - This draws the straight line (fitted line) on the graph of income vs. spending scores.

Line 23 and 24 -

mean_squared_error(y, y_pred): Calculates how far off the predicted spending scores are from the actual ones (on average). A smaller value is better.

r2_score(y, y_pred): Measures how well the straight line explains the data. An R-squared value close to 1 means the line is a good fit, and a value close to 0 means it's a bad fit.

Imagine guessing someone's spending based on their income. These numbers tell you how often you're wrong (MSE) and how useful your guesses are overall (R-squared).

Implementation

- A linear regression model was used to predict spending scores from annual income.

- Linear regression assumes a straight-line relationship between the independent (annual income) and dependent (spending score) variables.

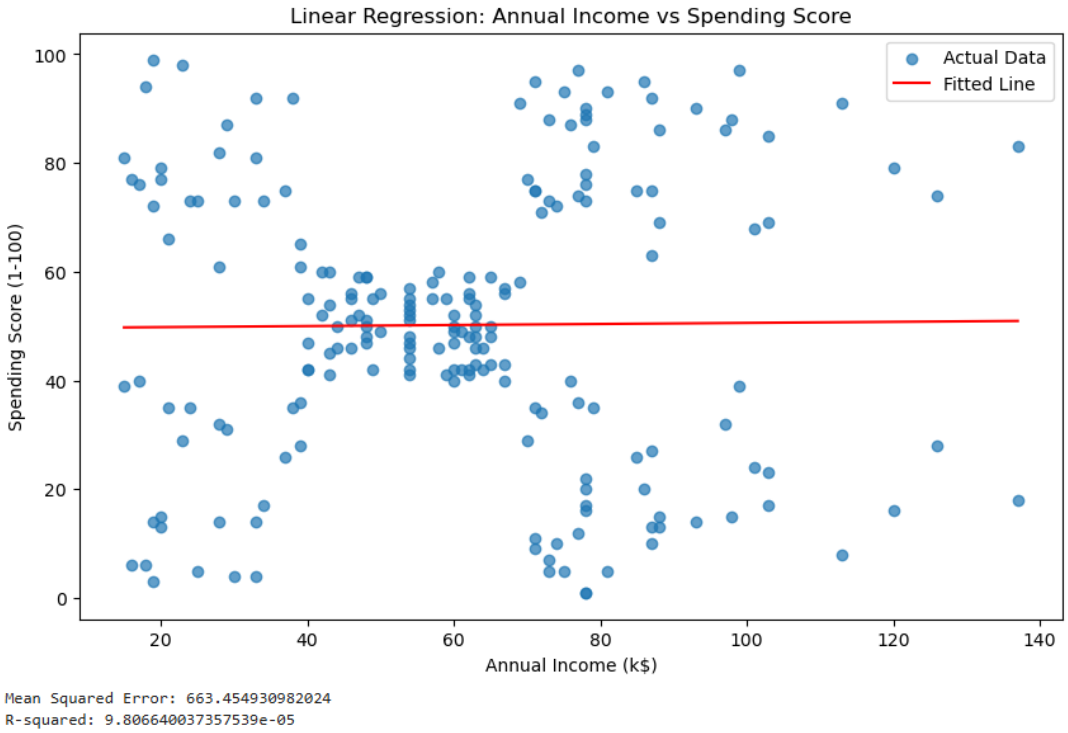


Figure 8 : Annual Income and spending score.

Result

- Regression Line: A fitted line was plotted over the actual data.

- Evaluation Metrics:

  - Mean Squared Error (MSE): 663.45 (indicating significant errors in predictions).

  - R-squared: ~0.00 (indicating no linear relationship).

Limitations:

- A linear model oversimplifies relationships that might be non-linear.

- Additional features and advanced models (e.g., polynomial regression, machine learning) could improve predictive power.

Conclusion- Clustering Insights: Four customer segments were identified, each with distinct income and spending patterns, providing actionable insights for customer targeting. Regression Analysis: Income alone does not explain spending scores, emphasizing the need for more comprehensive models to understand customer behavior.