Lab Work 3: K-Means Clustering on Customer Sales Data

Objective:

In this lab, we will use the K-Means clustering algorithm to group customers based on their sales and profit data. The goal is to segment customers into clusters to identify distinct purchasing behaviors.

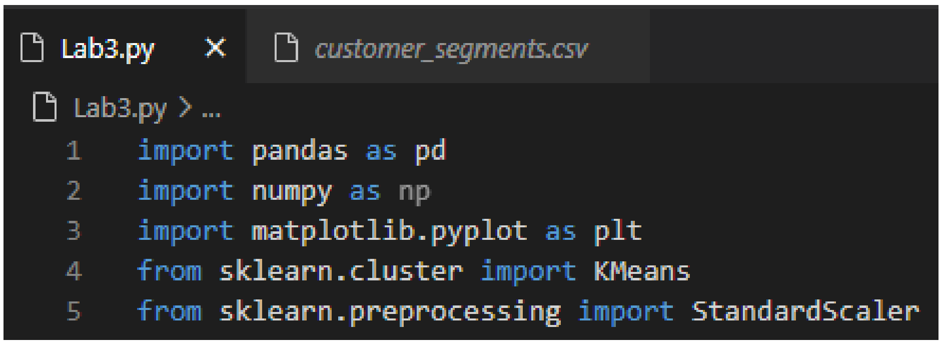
Problem Statement:

A retail company wants to segment its customers into groups based on their total sales and profit. This clustering can help in identifying various groups.

Steps:

1. Import necessary libraries.

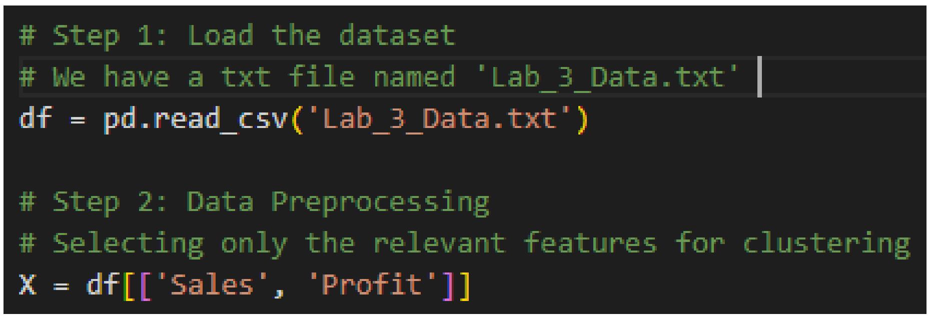
- Import all the necessary python libraries like pandas, numpy, matplotlib, and scikit-learn.



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
```

2. Load the dataset into a pandas Data Frame and perform data pre processing.

- Load the provided CSV data into a DataFrame using pandas and preprocess the data by ensuring all necessary columns (Sales, Profit) are correctly formatted.



```python
# Step 1: Load the dataset
# We have a txt file named 'Lab_3_Data.txt'
df = pd.read_csv('Lab_3_Data.txt')

# Step 2: Data Preprocessing
# Selecting only the relevant features for clustering
X = df[['Sales', 'Profit']]
```

3. Apply K-Means clustering to group the customers.

- Determine the optimal number of clusters using the Elbow Method.

```python
# Step 3: Apply K-Means Clustering
# Determine the optimal number of clusters using the Elbow Method
wcss = []  # Within-cluster sum of squares
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, init='k-means++', random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)
```

4. Visualize the clusters and interpret the output.

Expected Output:

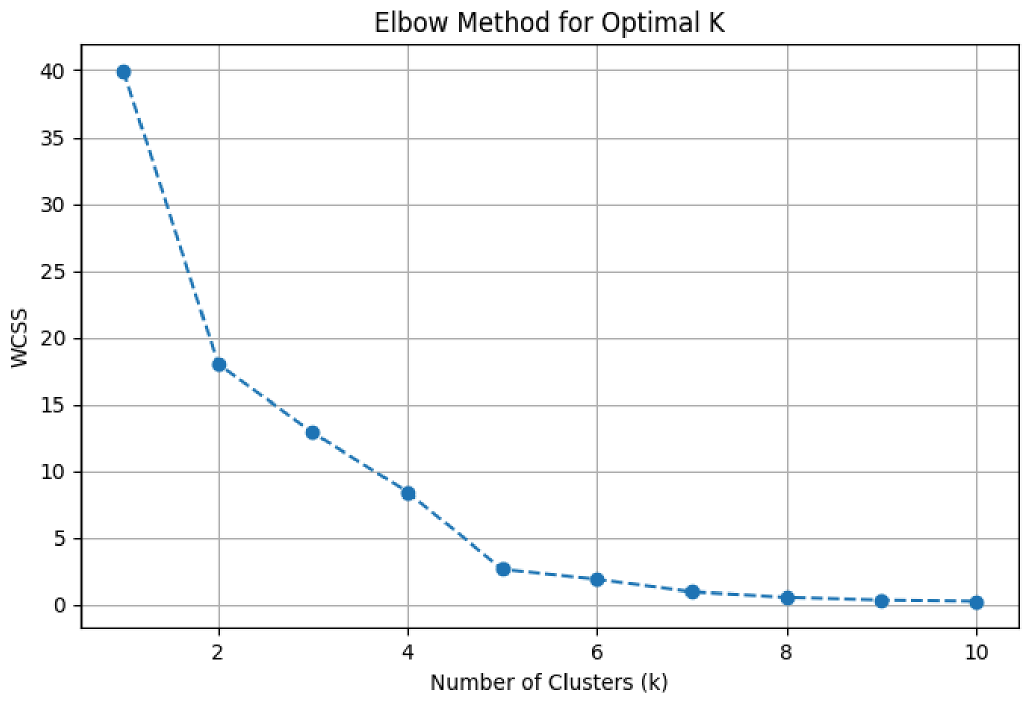


Fig: Elbow Method

Graph Overview

- X-axis (Number of Clusters, k): Ranges from 2 to 10.
- Y-axis (WCSS - Within-Cluster Sum of Squares): Measures cluster compactness (lower = tighter clusters).

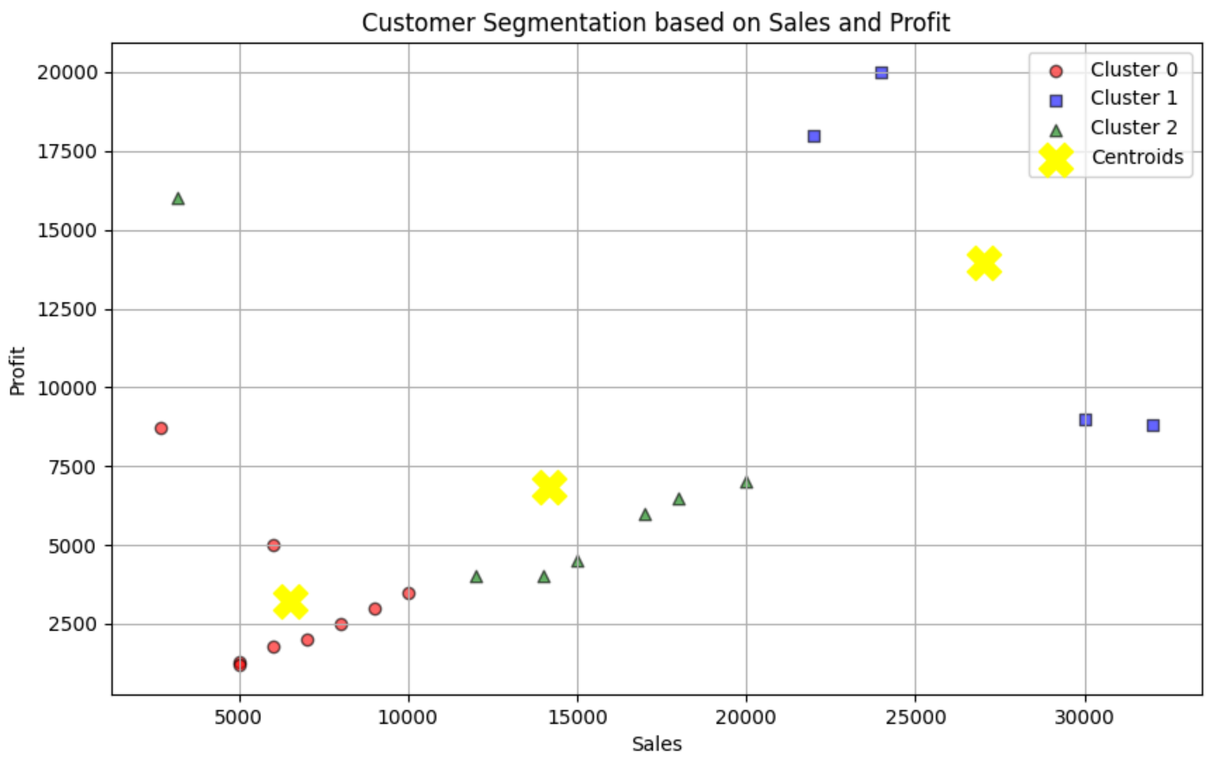


Fig: Scatter Plot Visualization for clusters

a. **Axes Explanation**

- **X-axis (Sales):** Total sales made by each customer (2700-32,000).
- **Y-axis (Profit):** Total profit generated from each customer (1200-20,000).
- **Each point** = one customer.
- **Each centroid** ("heart" of its cluster) = the average sales and profit for that customer group, helping us see what's normal for each segment and spot who's performing better or worse.

b. **Clusters Identified (k = 3):**

Cluster 0: Circles

- Position: Low Sales (<=10k) & Low-to-Moderate Profit (<10k)
- Likely occasional or small-scale buyers
- May represent low-margin transactions

Cluster 1: Squares

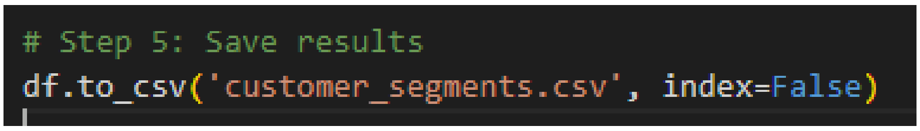
- Position: High Sales (>20k) & High Profit (>15k)
- Most valuable customers
- High revenue and profit contributors

Cluster 2: Triangles

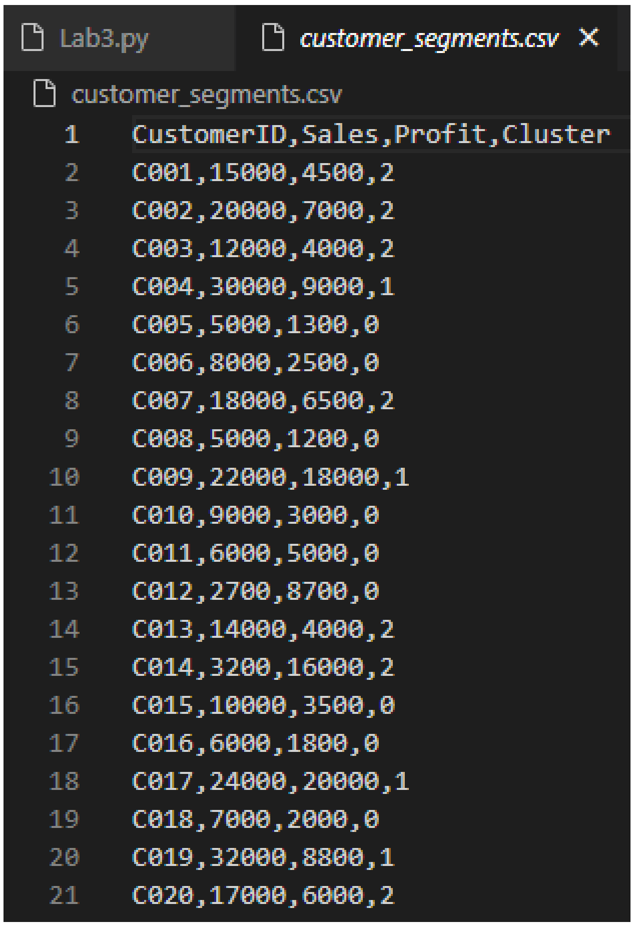
- Position: Moderate Sales (3.2k-25k) & Moderate Profit (5k-16k)
- Possibly frequent buyers with lower margins
- Could be bulk purchasers

5. Save Results:

- Save the updated data to a new CSV file.



```python
# Step 5: Save results
df.to_csv('customer_segments.csv', index=False)
```



```
CustomerID,Sales,Profit,Cluster
C001,15000,4500,2
C002,20000,7000,2
C003,12000,4000,2
C004,30000,9000,1
C005,5000,1300,0
C006,8000,2500,0
C007,18000,6500,2
C008,5000,1200,0
C009,22000,18000,1
C010,9000,3000,0
C011,6000,5000,0
C012,2700,8700,0
C013,14000,4000,2
C014,3200,16000,2
C015,10000,3500,0
C016,6000,1800,0
C017,24000,20000,1
C018,7000,2000,0
C019,32000,8800,1
C020,17000,6000,2
```

Fig: customer_segments.csv

Discussion Questions:

a. How can the company use these clusters to improve customer engagement?

- **Cluster 0 (Low Sales, Low Profit):** Likely price-sensitive or occasional buyers.
  - Strategies: Targeted discounts, bundled deals, loyalty incentives, or personalized recommendations to encourage more frequent purchases.
- **Cluster 1 (High Sales, High Profit):** Top-tier, high-value customers.
  - Strategies: VIP perks, early access to new products, dedicated account managers, or premium rewards to reinforce loyalty.
- **Cluster 2 (Moderate Sales, Varying Profit):** Customers with growth potential.

    - Strategies: Upsell higher-margin products, analyze profit drivers (e.g., product mix), and tailor promotions to boost profitability.

b. What additional features could improve clustering accuracy?

- Adding relevant features can reveal deeper customer insights like:
    - **Behavioral data:** Purchase frequency, product preferences, or customer tenure (Duration of relationship).
    - **Transactional metrics:** Average order value, return rates, or seasonal buying patterns.
    - **Demographic/situational data:** Geographic location or region, Business size (B2B), or Return rate/Customer satisfaction scores.
      These features could reveal hidden patterns, such as why some moderate-spenders are more profitable than others.

c. What would happen if the number of clusters (k) is increased or decreased?

- Increasing k like if we use more groups (e.g., 5 instead of 3):
    - More refined segments: We might spot smaller, more specific customer types.
    - Risk of overfitting: Too many groups can make things messy—like splitting customers into tiny categories leading to too much fragmentation and less clarity.
- Decreasing k like if we use fewer groups (e.g., 2 instead of 3):
    - Simplifies segmentation: Easier to manage just a few broad categories.
    - Risk of underfitting: Important differences might get ignored or distinct behaviors might get merged—like treating occasional buyers and regular customers the same leading to losing useful insights.

Conclusion:

This lab provided hands-on experience with K-Means clustering for customer segmentation, revealing three distinct groups (low, medium, and high-value customers) with actionable business insights.

Lab Work 2: Linear Regression for Predicting Profit and Transactions

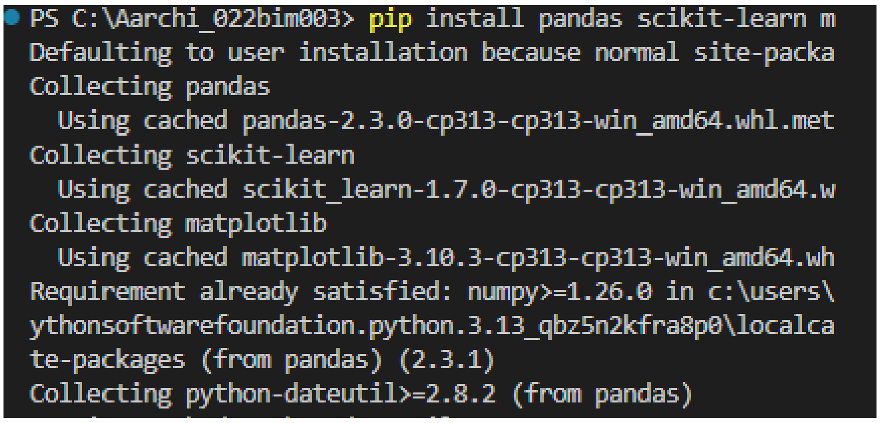
Objective:

To apply linear regression for predicting Profit and Transactions based on Sales data.

Steps:

1. Set Up Python Environment:

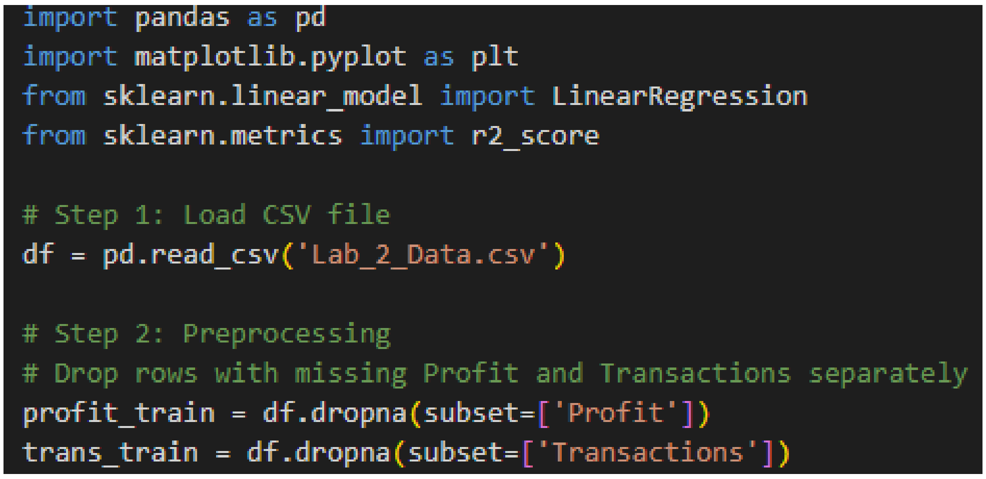
- Ensure that Python and necessary libraries like pandas, sklearn, and matplotlib are installed.
- Use Python's package manager (pip) to install required libraries if not already done:



```
PS C:\Aarchi_022bim003> pip install pandas scikit-learn m
Defaulting to user installation because normal site-packa
Collecting pandas
  Using cached pandas-2.3.0-cp313-cp313-win_amd64.whl.met
Collecting scikit-learn
  Using cached scikit_learn-1.7.0-cp313-cp313-win_amd64.w
Collecting matplotlib
  Using cached matplotlib-3.10.3-cp313-cp313-win_amd64.wh
Requirement already satisfied: numpy>=1.26.0 in c:\users\
ythonsoftwarefoundation.python.3.13_qbz5n2kfra8p0\localca
te-packages (from pandas) (2.3.1)
Collecting python-dateutil>=2.8.2 (from pandas)
```

2. Load and Prepare data:

- Load the provided CSV data into a DataFrame using pandas.
- Preprocess the data by ensuring all necessary columns (Sales, Profit, & Transactions) are correctly formatted.



```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

# Step 1: Load CSV file
df = pd.read_csv('Lab_2_Data.csv')

# Step 2: Preprocessing
# Drop rows with missing Profit and Transactions separately
profit_train = df.dropna(subset=['Profit'])
trans_train = df.dropna(subset=['Transactions'])
```