

CUSTOMER SEGMENTATION

Project Overview

Project Title: Customer Segmentation using Data Science Techniques.

Project Phase: Phase 1 – Problem Definition and Design Thinking.

Dataset Link:

Project Description

The project aims to implement data science techniques to segment customers based on their behavior, preferences, and demographic attributes. The primary objective is to empower businesses to personalize marketing strategies and enhance overall customer satisfaction. The Project encompasses several essential steps, including data collection, data preprocessing, feature engineering, the application of clustering algorithms, visualization of results, and interpretation of customer segments.

Project Goals

- Segment customers into distinct groups based on their behavioral patterns, preferences, and demographic characteristics.
- Develop actionable insights to inform personalized marketing strategies.
- Enhance customer satisfaction and engagement through targeted marketing approaches.
- Enable data-driven decision-making for business growth.

Design Thinking

➤ Data Collection

Collect customer data, including attributes like purchase history, demographic information, and interaction behavior.

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Step 1: Import the libraries

```
dataset = pd.read_csv('/kaggle/input/mall-customers/Mall_Customers.csv')
```

Step 2: Using pandas libraries read the csv file

Step 3: Print the head of the csv file

```
dataset.head()
```

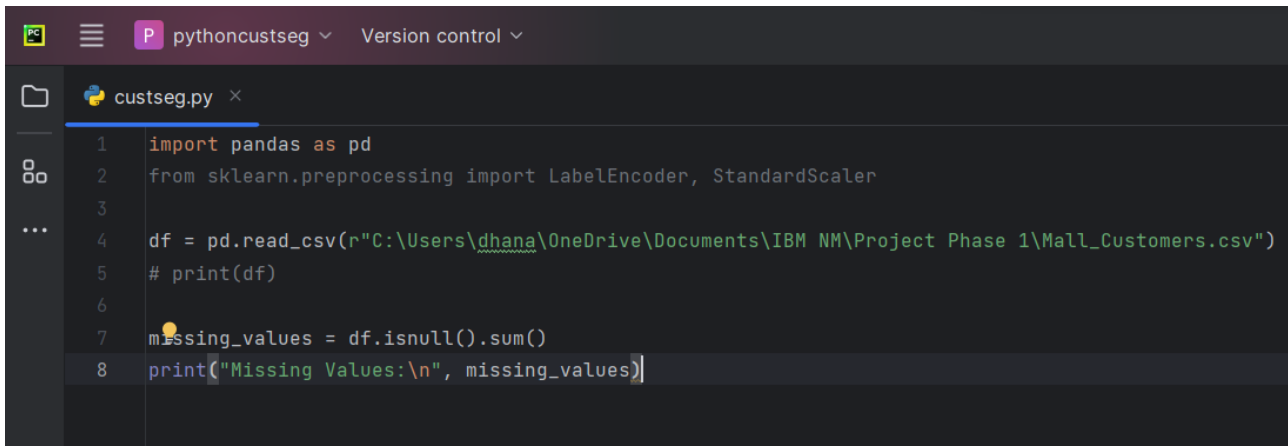
Output:

	CustomerID	Genre	Age	Annual Income (k\$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

➤ Data Preprocessing

Cleaning and preprocessing data for mall customers from a CSV file typically involves tasks like handling missing values, encoding categorical features, and scaling or normalizing numerical features. Here's a Python program using the pandas library to clean and preprocess a CSV file containing mall customer data:

(1) Check for missing values



```
1 import pandas as pd
2 from sklearn.preprocessing import LabelEncoder, StandardScaler
3
4 df = pd.read_csv(r"C:\Users\dhana\OneDrive\Documents\IBM NM\Project Phase 1\Mall_Customers.csv")
5 # print(df)
6
7 missing_values = df.isnull().sum()
8 print("Missing Values:\n", missing_values)
```

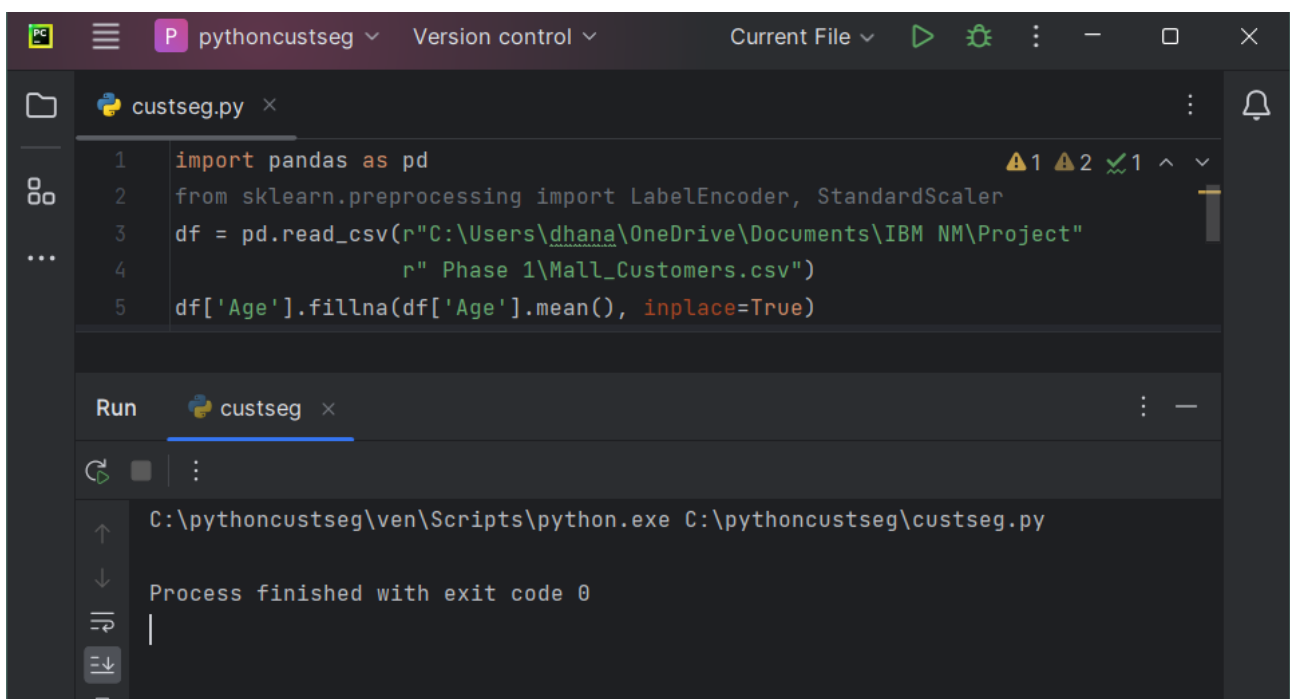


```
Run custseg x
C:\pythoncustseg\ven\Scripts\python.exe C:\pythoncustseg\custseg.py
Missing Values:
CustomerID      0
Genre           0
Age             0
Annual Income (k$)  0
Spending Score (1-100)  0
dtype: int64

Process finished with exit code 0
```

Output:

(2) Handling Missing values (if any)



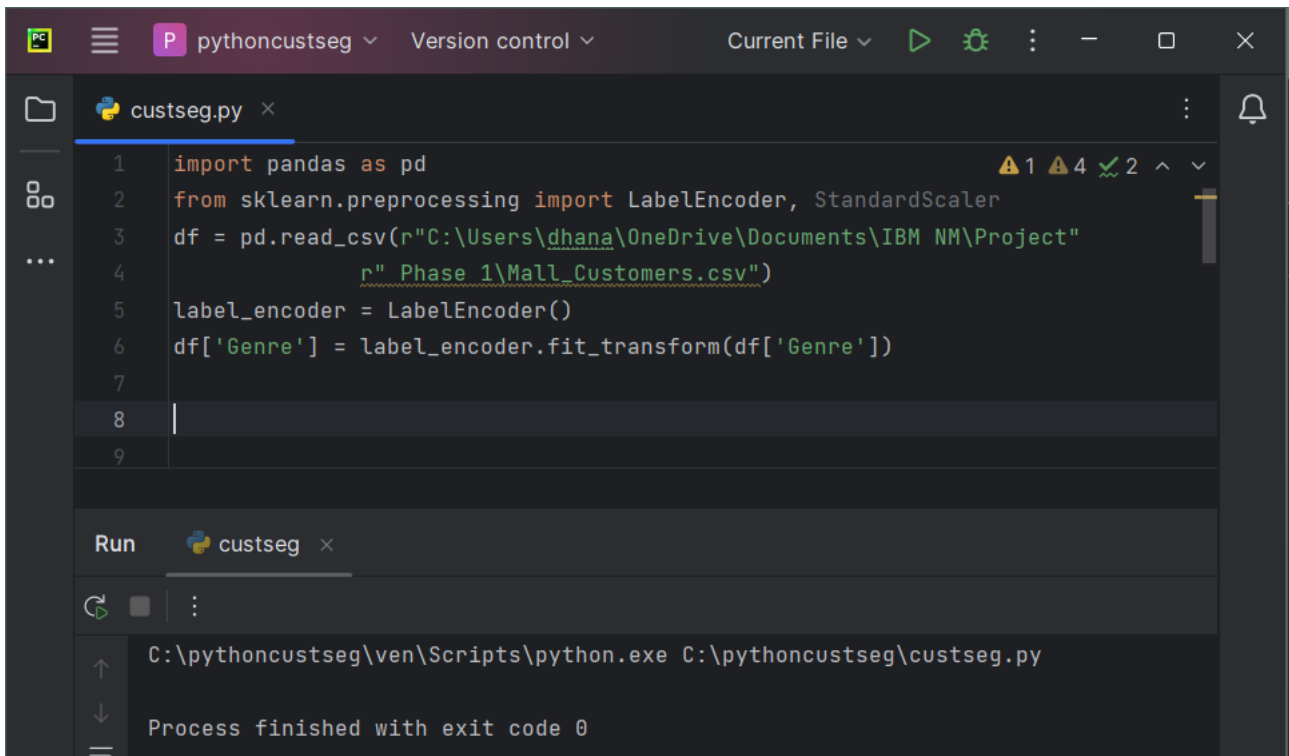
```
1 import pandas as pd
2 from sklearn.preprocessing import LabelEncoder, StandardScaler
3 df = pd.read_csv(r"C:\Users\dhana\OneDrive\Documents\IBM NM\Project"
4                  r" Phase 1\Mall_Customers.csv")
5 df['Age'].fillna(df['Age'].mean(), inplace=True)
```

```
Run custseg x
C:\pythoncustseg\ven\Scripts\python.exe C:\pythoncustseg\custseg.py

Process finished with exit code 0
```

(3) Encode categorical features(if any)

Example: Encode the 'Genre' column using Label Encoding

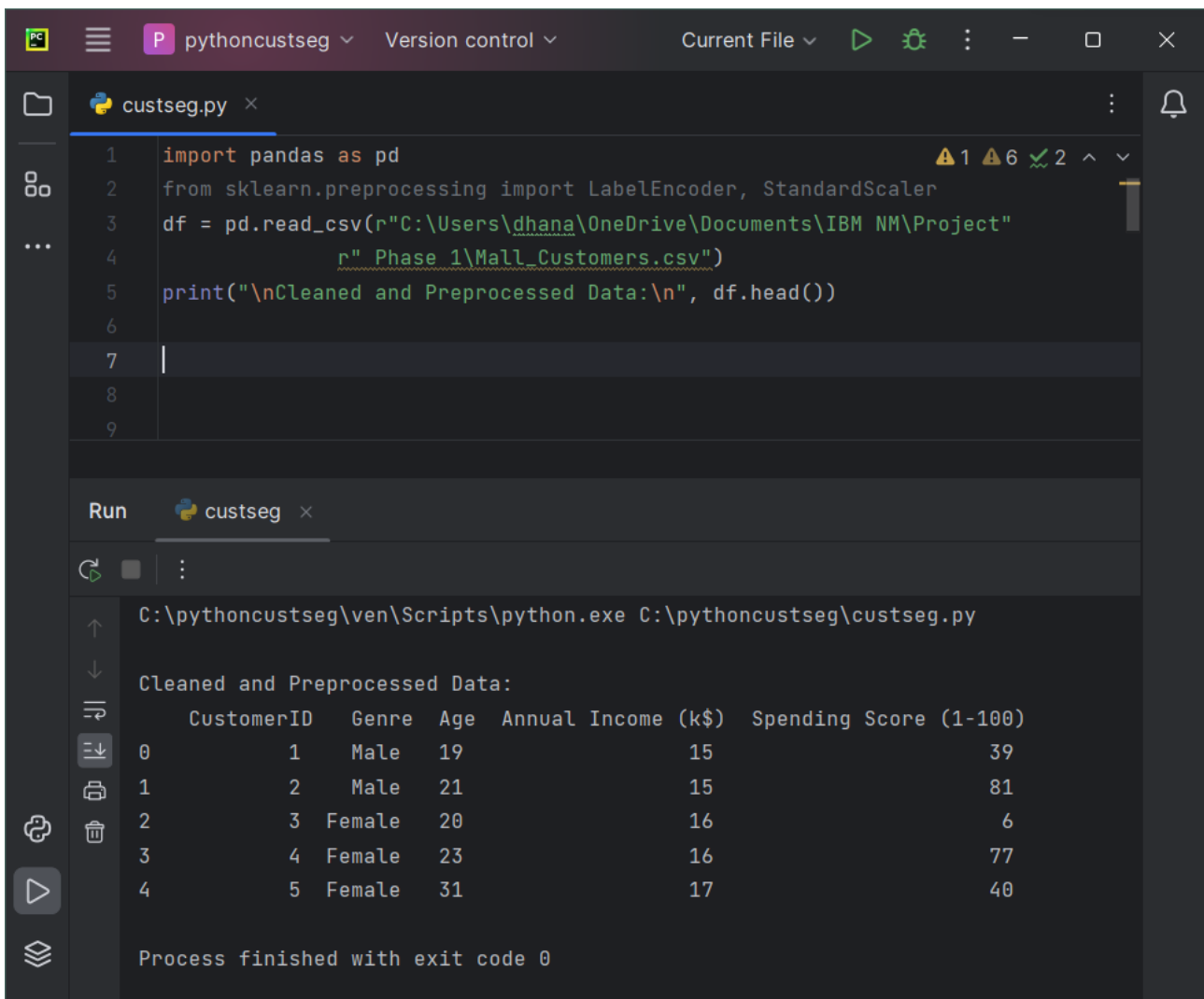


```
1 import pandas as pd
2 from sklearn.preprocessing import LabelEncoder, StandardScaler
3 df = pd.read_csv(r"C:\Users\dhana\OneDrive\Documents\IBM NM\Project"
4                 r" Phase 1\Mall_Customers.csv")
5 label_encoder = LabelEncoder()
6 df['Genre'] = label_encoder.fit_transform(df['Genre'])
7
8
9
```

Run custseg x

C:\pythoncustseg\ven\Scripts\python.exe C:\pythoncustseg\custseg.py

Process finished with exit code 0



```
1 import pandas as pd
2 from sklearn.preprocessing import LabelEncoder, StandardScaler
3 df = pd.read_csv(r"C:\Users\dhana\OneDrive\Documents\IBM NM\Project"
4                 r" Phase 1\Mall_Customers.csv")
5 print("\nCleaned and Preprocessed Data:\n", df.head())
6
7
8
9
```

Run custseg x

C:\pythoncustseg\ven\Scripts\python.exe C:\pythoncustseg\custseg.py

Cleaned and Preprocessed Data:

	CustomerID	Genre	Age	Annual Income (k\$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Process finished with exit code 0

(4) Display the first few rows of the cleaned data

➤ Feature Engineering

Create additional features that capture customer behavior and preferences, such as total spending, frequency of purchases, etc.

```
In [1]: import pandas as pd

In [5]: data = pd.read_csv(r"C:\Users\dhana\OneDrive\Documents\IBM NM\Project Phase 1\Mall_Customers.csv")

In [6]: data['Total_Spending'] = data['Annual Income (k$)'] * data['Spending Score (1-100)']

In [7]: print(data['Total_Spending'])

0      585
1      600
2      656
3      672
4      731
...
195    28080
196    29610
197    29736
198    32469
199    32606
Name: Total_Spending, Length: 200, dtype: int64

In [8]: data.to_csv("modified_mall_customers.csv", index=False)
```

In the above code shows,

1. Import packages
2. Load the dataset from the provided path
3. Feature Engineering
4. Save the modified DataFrame back to a CSV file

This code will create a new CSV file named “modified_mall_customers.csv” in the current directory, containing the original columns and the newly added ‘Total_Spending’ column. The ‘index=False’ argument that the DataFrame index is not saved as a separate column in the CSV file.

➤ Clustering Algorithm

Apply clustering algorithms (e.g., K-Means, DBSCAN, hierarchical clustering) to segment customers effectively

By using the K-Means Algorithm

```
kmeansmodel = KMeans(n_clusters = 5 , init = "k-means+  
+", random_state=0)  
y_kmeans = kmeansmodel.fit_predict(x)
```

```
plt.scatter(x[y_kmeans==0,0],x[y_kmeans==0,1],s=100,c  
="red",label = "cluster 1")  
plt.scatter(x[y_kmeans==1,0],x[y_kmeans==1,1],s=100,c  
="blue",label = "cluster 2")  
plt.scatter(x[y_kmeans==2,0],x[y_kmeans==2,1],s=100,c  
="green",label = "cluster 3")  
plt.scatter(x[y_kmeans==3,0],x[y_kmeans==3,1],s=100,c  
="cyan",label = "cluster 4")  
plt.scatter(x[y_kmeans==4,0],x[y_kmeans==4,1],s=100,c  
="magenta",label = "cluster 5")  
plt.scatter(kmeans.cluster_centers_[ :,0],kmeans.cluste  
r_centers_[ :,1],s=300,c="yellow",label="centroids")  
plt.title("clusters of customers")  
plt.xlabel("Yıllık gelir")  
plt.ylabel("harcama skoru")  
plt.legend()  
plt.show()
```



➤ Visualization

To visualize customer segments using various techniques such as scatter plots, bar charts and heatmaps.

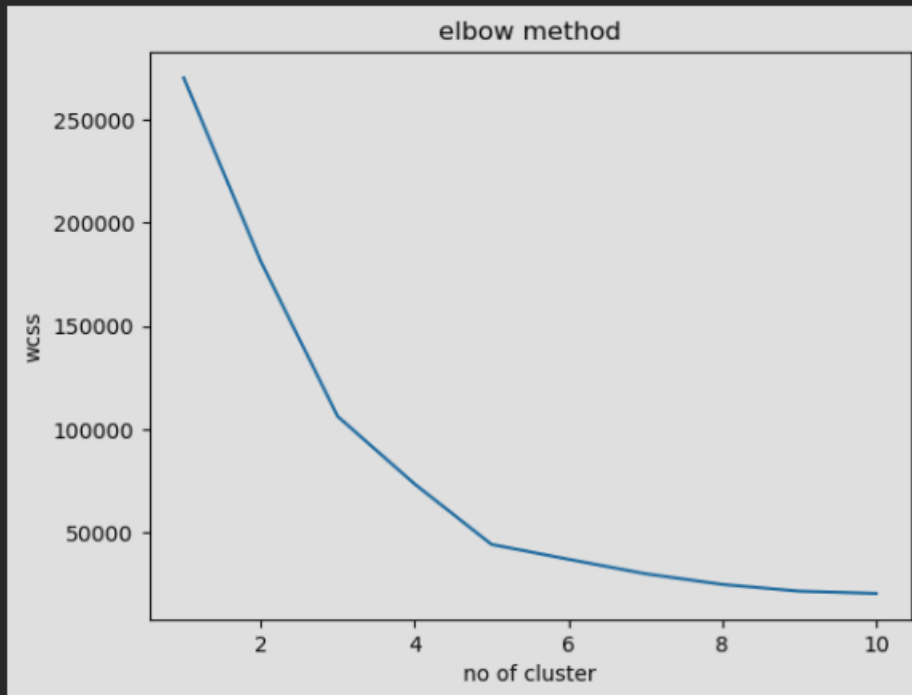
```
In [7]: x=dataset.iloc[:,[3,4]].values

In [8]: from sklearn.cluster import KMeans

In [9]: wcss=[]
for i in range(1,11):
    kmeans=KMeans(n_clusters = i , init="k-means++",random_state=0)
    kmeans.fit(x)
    wcss.append(kmeans.inertia_)
```

```
In [10]:
```

```
plt.plot(range(1,11),wcss)  
plt.title("elbow method")  
plt.xlabel("no of cluster")  
plt.ylabel("wcss")  
plt.show()
```



➤ Interpretation

Creating a Python program for the interpretation

Phase of your customer segmentation .

1. Import the Libraries
2. Load the preprocessed dataset
3. Assuming already performed clustering (K-means)
4. Fit the Clustering model
5. Now, we have a 'Cluster' column in our Dataframe indicating which cluster each customer belongs to.
6. Interpretate and Analyze the characterisitics of each cluster
7. Perform further analysis to derive actionable insights

Expected Deliverables

- Cleaned and preprocessed customer dataset
- Feature e-engineered dataset
- Customer segmentation using clustering algorithms
- Visualizaion illustrating customer segments
- Interpretation and insights delivered from the segmentation.

Submitted By:

S. Dhanalakshmi B.tech information Technology

IBM Naan Mudhalvan Applied Data Science Group 2