# PRODIGY-ML-02

# TASK-02

## Import Necessary Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
```

## Load the Dataset

```
url = 'https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial-in-python'
data = pd.read_csv('Mall_Customers.csv')
print(data.head())
```

## Data Preprocessing

```
print(data.isnull().sum())
data = data.drop('CustomerID', axis=1)
data['Gender'] = data['Gender'].map({'Male': 0, 'Female': 1})
print(data.head())
```

## Data Standardization

```
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data)
print(data_scaled[:5])
```

## Finding the Optimal Number of Clusters using Elbow Method

```
sse = []
k_range = range(1, 11)
```

```
for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(data_scaled)
    sse.append(kmeans.inertia_)

plt.figure(figsize=(10, 6))
plt.plot(k_range, sse, marker='o')
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('SSE')
plt.show()
```

## Applying K-Means Clustering

```
k = 5
kmeans = KMeans(n_clusters=k, random_state=42)
kmeans.fit(data_scaled)
data['Cluster'] = kmeans.labels_
print(data.head())
```

## Visualizing the Clusters

```
plt.figure(figsize=(12, 8))
sns.scatterplot(x='Annual Income (k$)', y='Spending Score (1-100)',
hue='Cluster', palette='viridis', data=data, s=100)
plt.title('Customer Segments')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend(title='Cluster')
plt.show()
```

## Evaluating the Clustering using Silhouette Score

```
silhouette_avg = silhouette_score(data_scaled, data['Cluster'])
print(f'Silhouette Score: {silhouette_avg}')
```

## Summary and Insights

```
cluster_centers = scaler.inverse_transform(kmeans.cluster_centers_)
cluster_centers_df = pd.DataFrame(cluster_centers, columns=data.columns[:-
1])
cluster_centers_df['Cluster'] = range(k)
print(cluster_centers_df)
```

## Save the Clustered Data

```
data.to_csv('Clustered_Customers.csv', index=False)
```