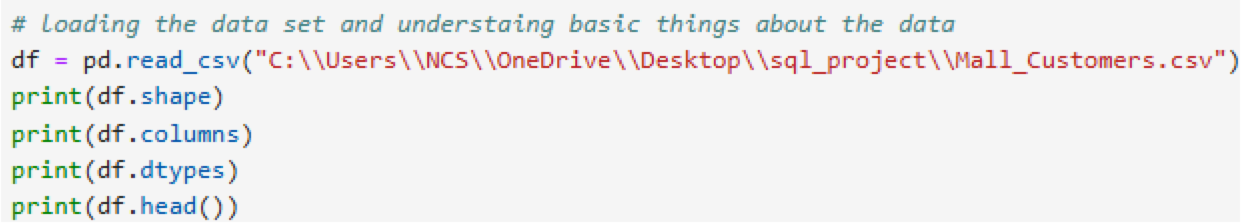
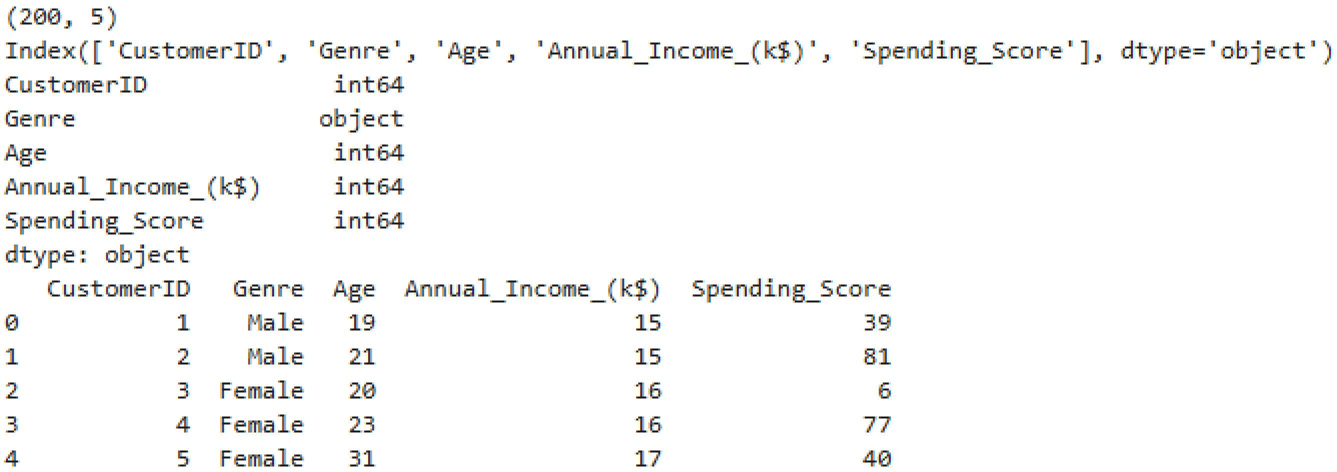
Customer Segmentation using KMeans Clustering

Customer Segmentation using KMeans Clustering

- Dataset : customer.csv from Kaggle
- About Dataset

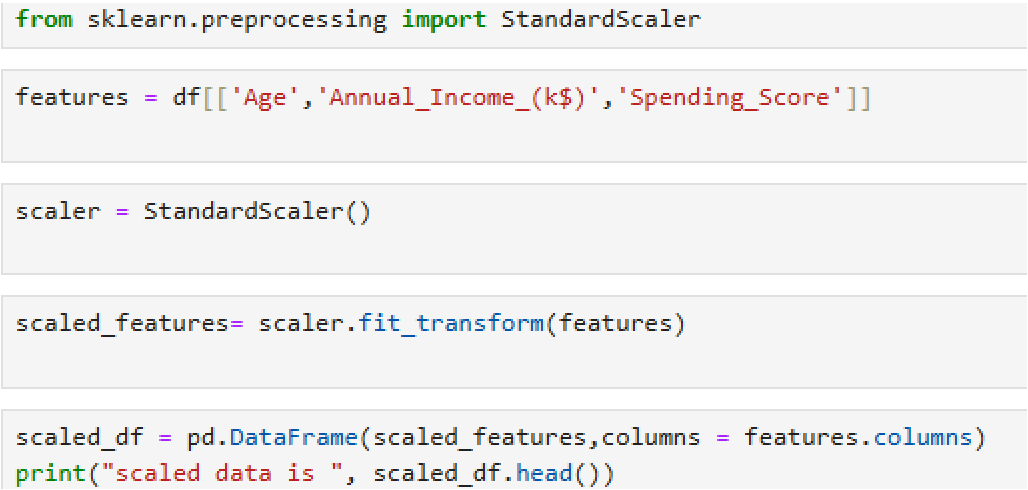


```
# Loading the data set and understaing basic things about the data
df = pd.read_csv("C:\\Users\\NCS\\OneDrive\\Desktop\\sql_project\\Mall_Customers.csv")
print(df.shape)
print(df.columns)
print(df.dtypes)
print(df.head())
```



```
(200, 5)
Index(['CustomerID', 'Genre', 'Age', 'Annual_Income_(k$)', 'Spending_Score'], dtype='object')
CustomerID             int64
Genre                 object
Age                    int64
Annual_Income_(k$)     int64
Spending_Score         int64
dtype: object
   CustomerID   Genre  Age  Annual_Income_(k$)  Spending_Score
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
```

Scaling- Standardscalar



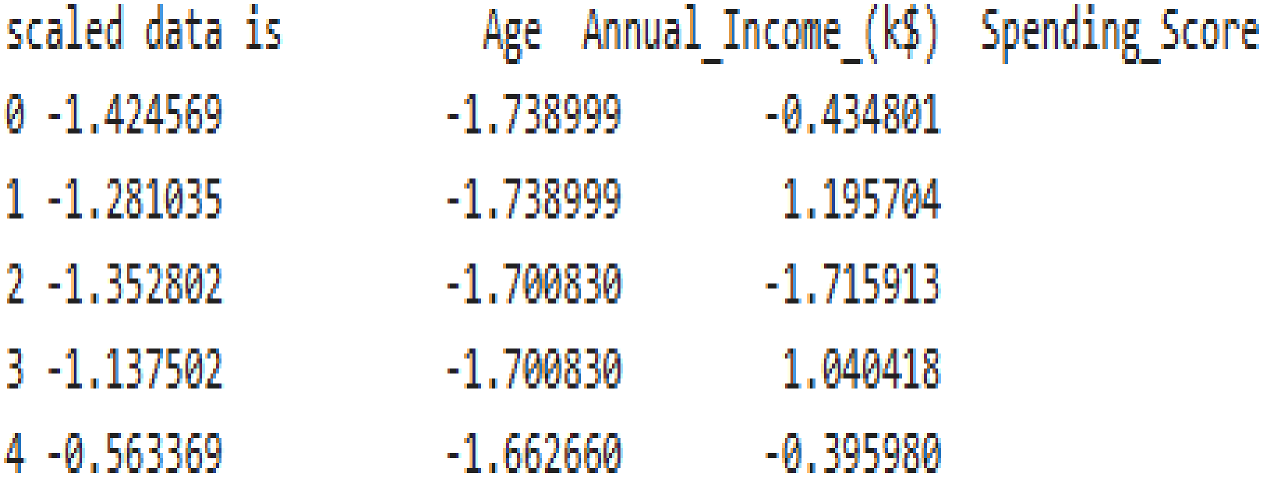
```
from sklearn.preprocessing import StandardScaler
```

```
features = df[['Age','Annual_Income_(k$)','Spending_Score']]
```

```
scaler = StandardScaler()
```

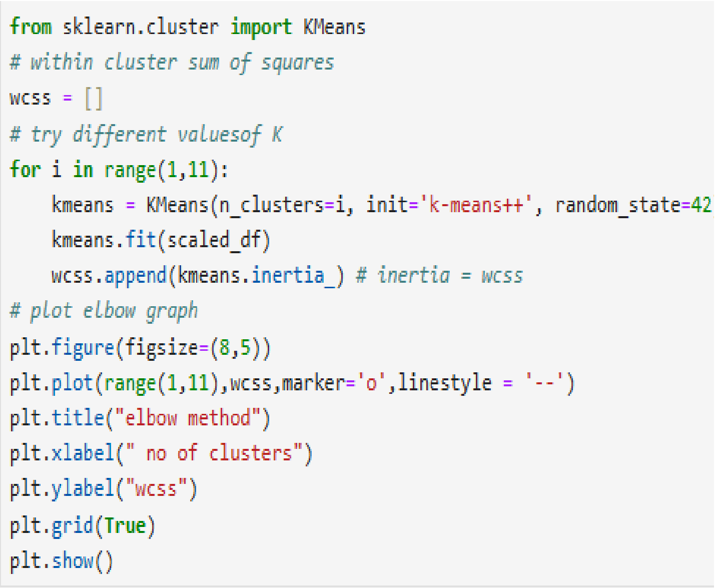
```
scaled_features= scaler.fit_transform(features)
```

```
scaled_df = pd.DataFrame(scaled_features,columns = features.columns)
print("scaled data is ", scaled_df.head())
```

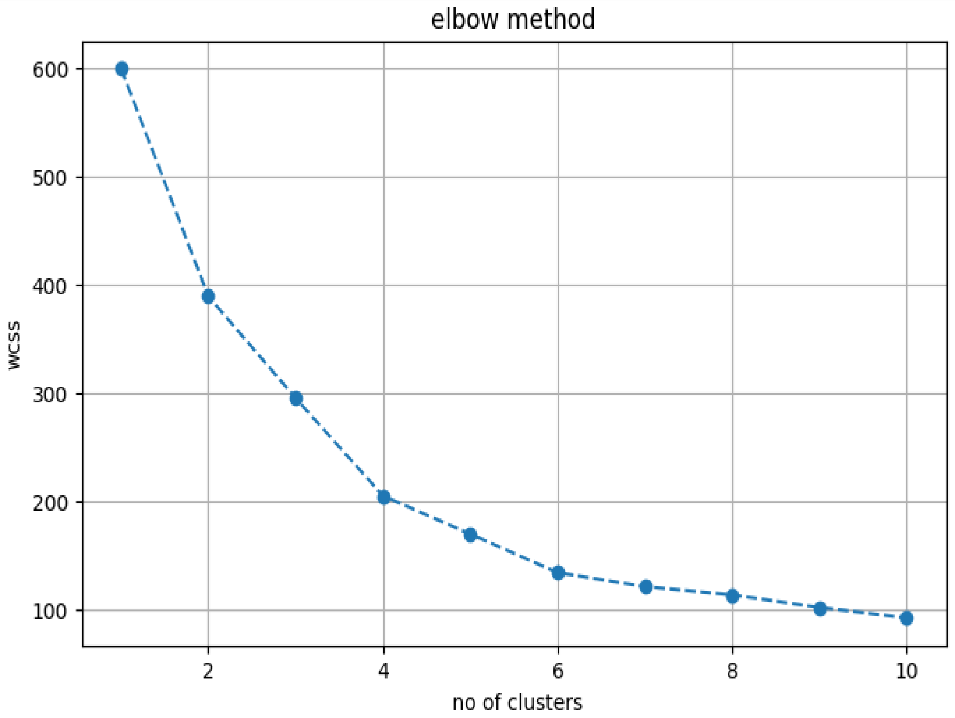


```
scaled data is         Age  Annual_Income_(k$)  Spending_Score
0 -1.424569           -1.738999       -0.434801
1 -1.281035           -1.738999        1.195704
2 -1.352802           -1.700830       -1.715913
3 -1.137502           -1.700830        1.040418
4 -0.563369           -1.662660       -0.395980
```

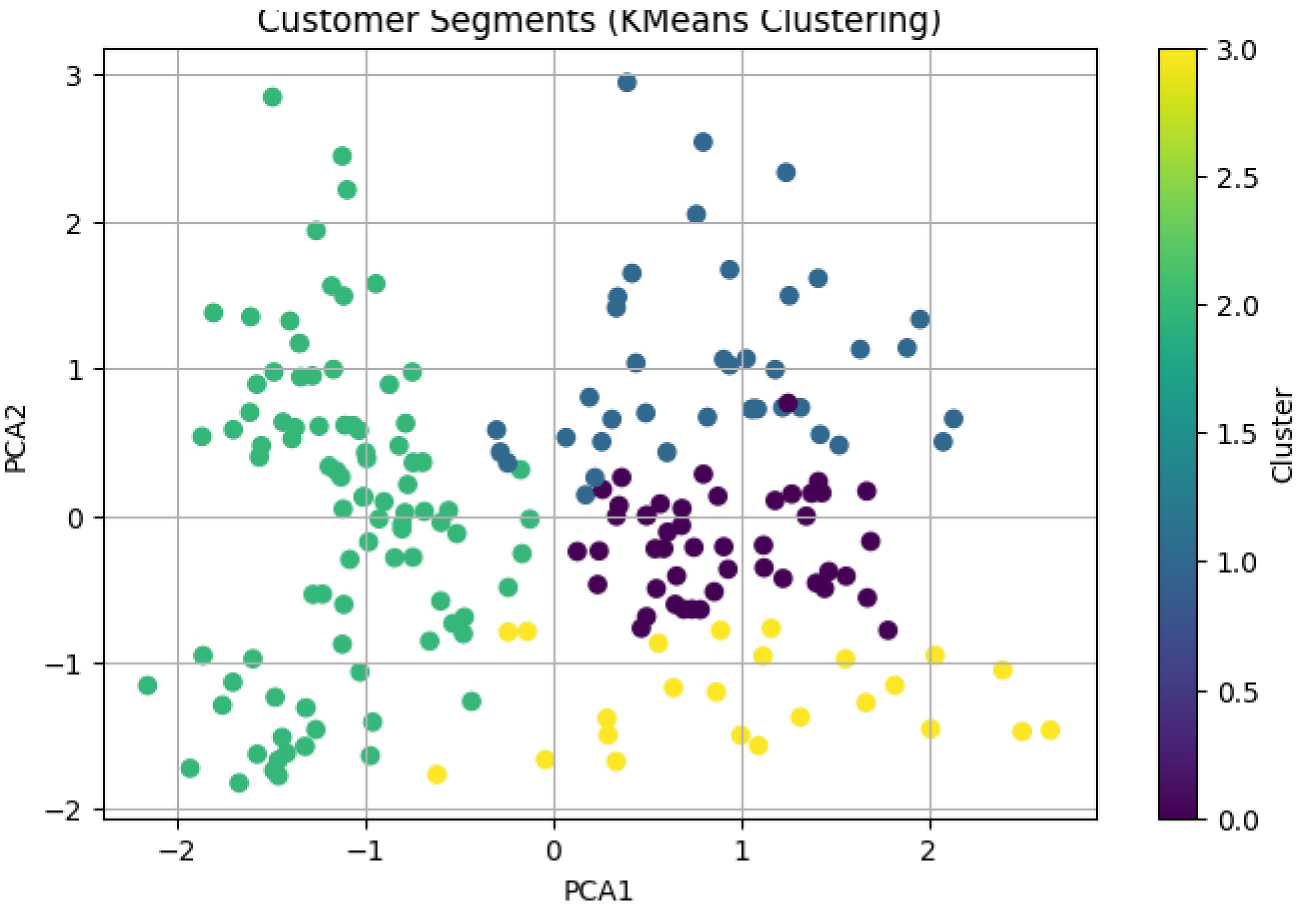
K-MEANS CLUSTRING



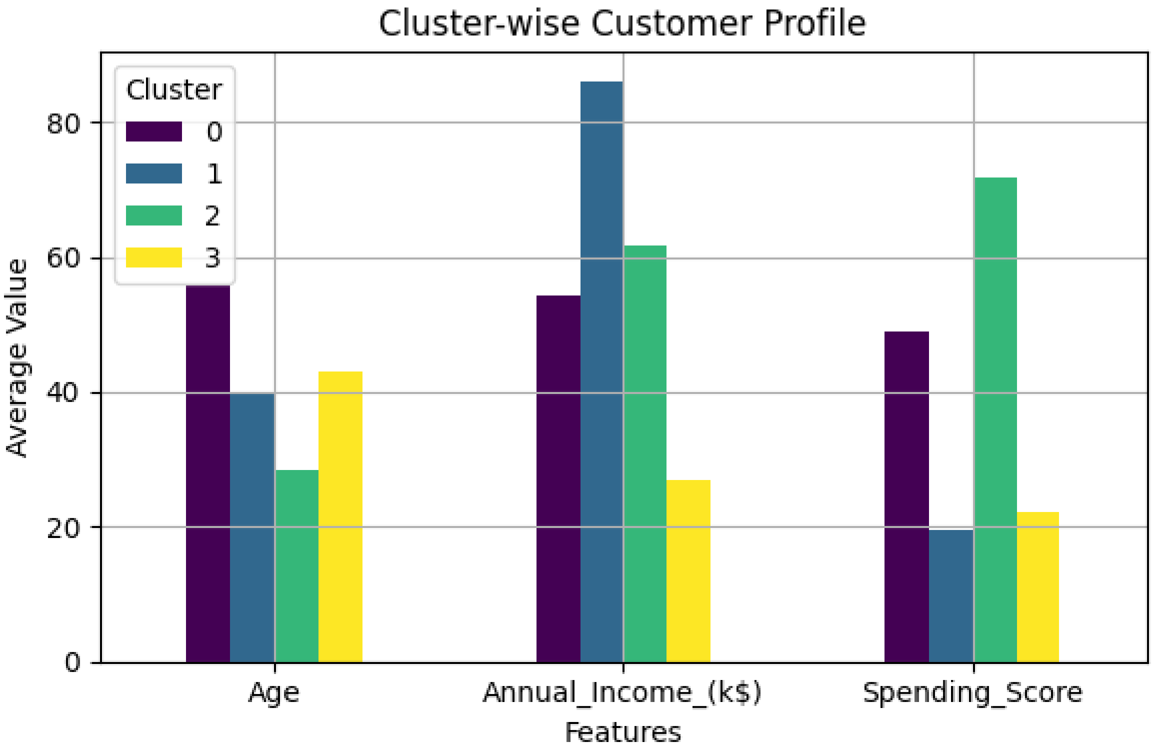
```
from sklearn.cluster import KMeans
# within cluster sum of squares
wcss = []
# try different valuesof K
for i in range(1,11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42
    kmeans.fit(scaled_df)
    wcss.append(kmeans.inertia_) # inertia = wcss
# plot elbow graph
plt.figure(figsize=(8,5))
plt.plot(range(1,11),wcss,marker='o',linestyle = '--')
plt.title("elbow method")
plt.xlabel(" no of clusters")
plt.ylabel("wcss")
plt.grid(True)
plt.show()
```



Cluster representation using PCA



Cluster-wise customer profile


Cluster-wise Customer Profile
Cluster
Average Value
Age
Annual_Income_(k$)
Spending_Score
Features