customersegmenpredic-pixproject

June 21, 2025

1 Customer Segmentation and Prediction project

2 Steps of project:-

1. Loading Data
2. Data Cleaning
3. Exploratory Data Analysis (EDA)
4. Feature Engineering
5. Model Building
6. Evaluation
7. Graphical user interface development (GUI)

3 Loading Data

[1]:
```
# Importing Some important library
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

[2]:
```
#Loading datasets
df = pd.read_csv("Mall_Customers.csv")
df
```

[2]:

	CustomerID	Gender	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
..	…	…	…	…	…
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18

```
199         200    Male   30                 137                      83

[200 rows x 5 columns]
```

[3]:
```
#seeing starting five rows
df.head()
```

[3]:

	CustomerID	Gender	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

[4]:
```
#seeing ending five rows
df.tail()
```

[4]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

4 Data Cleaning & EDA

[5]:
```
# check information of this dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              200 non-null    int64
 1   Gender                  200 non-null    object
 2   Age                     200 non-null    int64
 3   Annual Income (k$)      200 non-null    int64
 4   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

[6]:
```
#checking shape of this dataset
print("Number of Rows =",df.shape[0])
print("Number of Columns =",df.shape[1])
```

```
Number of Rows = 200
Number of Columns = 5
```

[7]:
```
#checking null value of this dataset
df.isnull().sum()
```

[7]:
```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

[8]:
```
# Check duplicate value of this datasets
df.duplicated()
```

[8]:
```
0      False
1      False
2      False
3      False
4      False
       …
195    False
196    False
197    False
198    False
199    False
Length: 200, dtype: bool
```

5 Feature Engineering

[9]:
```
# Changing the columns name Gender to Genre
df.rename(columns={"Gender":"Genre"},inplace=True)
```

[10]:
```
df.head()
```

[10]:

	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

[11]:
```
# let's check categorical and Numerical feature or columns of this dataset
Categorical_Feature=[feature for feature in df.columns if df[feature].
 ↪dtype=="object"]
```

```
Numerical_Feature=[feature for feature in df.columns if df[feature].dtype in␣
  ↪["int64","float64"]]
print(f"Hear are categorical feature =",Categorical_Feature)
print(f"Hear are Numerical feature =",Numerical_Feature)
```

```
Hear are categorical feature = ['Genre']
Hear are Numerical feature = ['CustomerID', 'Age', 'Annual Income (k$)',
'Spending Score (1-100)']
```

[12]:
```
# Get overall statistics of this dataset
df.describe()
```

[12]:

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

[13]:
```
# Lets see all columns name present of this dataset
df.columns
```

[13]:
```
Index(['CustomerID', 'Genre', 'Age', 'Annual Income (k$)',
       'Spending Score (1-100)'],
      dtype='object')
```

[14]:
```
# So this time we creating project purpose i only use this feature of this␣
  ↪dataset 'Annual Income (k$)','Spending Score (1-100)'
x = df[['Annual Income (k$)','Spending Score (1-100)']]
x
```

[14]:

	Annual Income (k$)	Spending Score (1-100)
0	15	39
1	15	81
2	16	6
3	16	77
4	17	40
..	…	…
195	120	79
196	126	28
197	126	74
198	137	18
199	137	83

[200 rows x 2 columns]

6 Model Building

[15]:
```
# Applying k-means clustering algorithmns & train it
from sklearn.cluster import KMeans
k_means = KMeans()
k_means.fit(x)
```

[15]:
```
KMeans()
```

[16]:
```
# identify cluster
k_means = KMeans(n_clusters=8)
k_means.fit_predict(x)#this fit_predict(x) is not only train but also create␣
 ↪dependent variable clusters
```

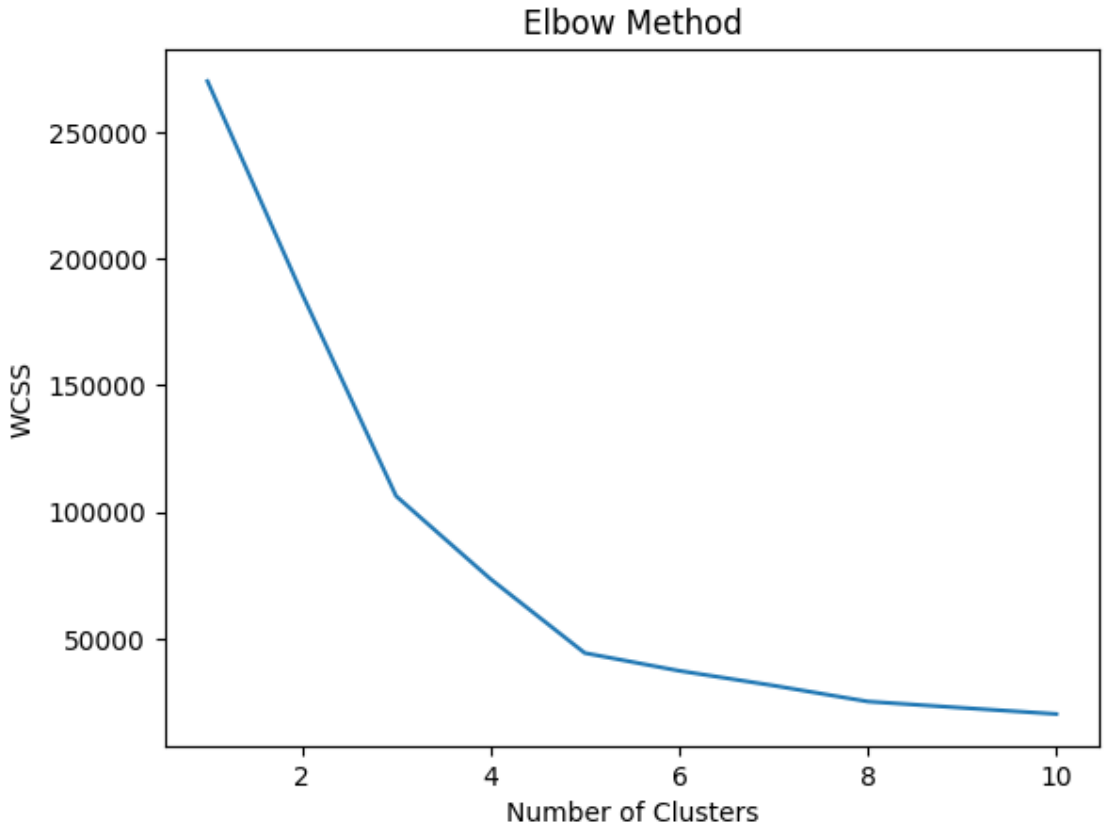
[16]:
```
array([7, 3, 4, 3, 7, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 7, 3, 7, 3, 7, 3,
       4, 3, 4, 3, 7, 3, 7, 3, 4, 3, 4, 3, 4, 3, 4, 3, 7, 3, 7, 3, 7, 5,
       7, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 0, 5, 5, 0, 0, 5, 5, 5, 5,
       5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 2, 1, 2, 1,
       0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 0, 1, 2, 1, 0, 1, 2, 1, 2, 1, 2, 1,
       2, 1, 2, 1, 2, 1, 0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
       2, 1, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6,
       2, 6])
```

[17]:
```
# We are going to use elbow method to find optimal number of clusters
wcss=[]
for i in range(1,11):
    k_means=KMeans(n_clusters=i)
    k_means.fit(x)
    wcss.append(k_means.inertia_)
```

[18]:
```
wcss
```

[18]:
```
[269981.28,
 186362.95600651758,
 106348.37306211119,
 73679.78903948834,
 44454.47647967974,
 37455.98455516028,
 31631.182088744594,
 25336.946861471864,
 22842.00465346013,
 20411.07505289899]
```

[19]:
```
# visualize it
plt.plot(range(1,11),wcss)
plt.title("Elbow Method")
plt.xlabel("Number of Clusters")
plt.ylabel("WCSS")
plt.show()
```



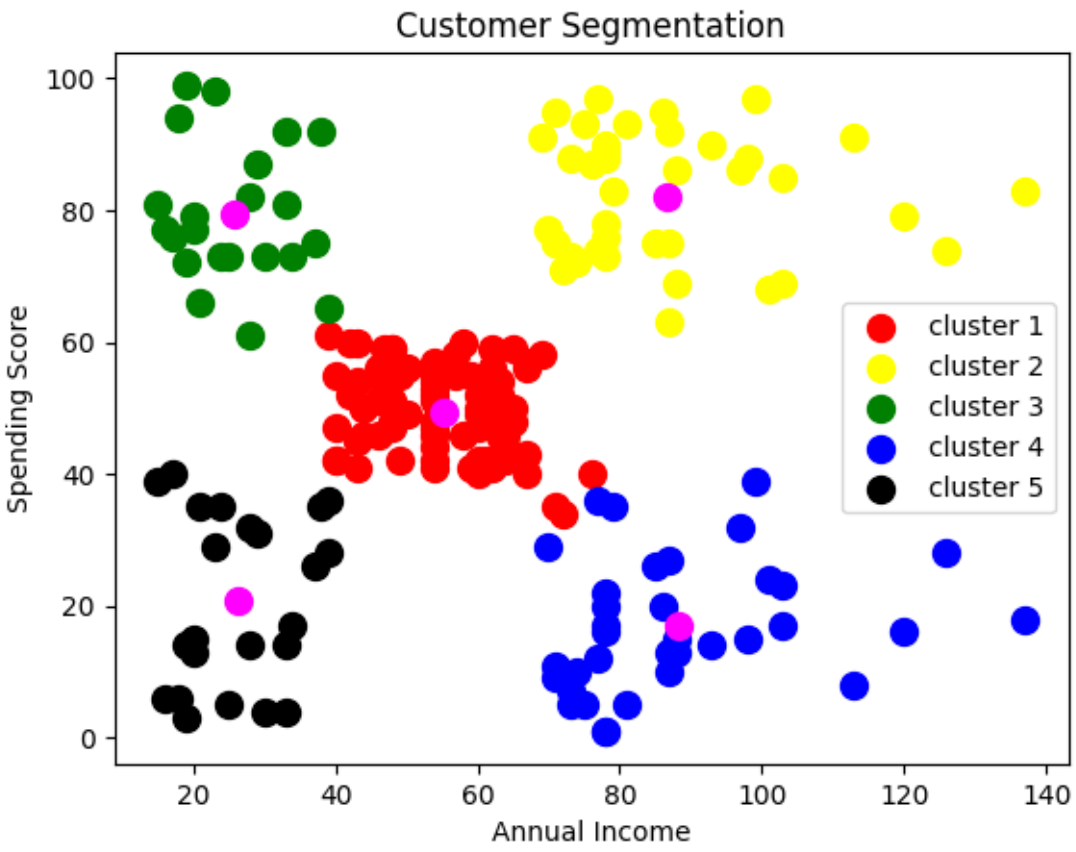
7 Evaluation

[20]:
```
# We are going to train kmeans clustering algorithmns with optimal number of␣
 ↪cluster
x = df[['Annual Income (k$)','Spending Score (1-100)']]
#Create instance of kmeans clustering algorithmns
k_means=KMeans(n_clusters=5,random_state=42)
y_means=k_means.fit_predict(x)#As we know this fit_predict method not only␣
 ↪train but also return dependent variable means clusters
```

[21]:
```
y_means
```

[21]:
```
array([4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2,
       4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 0,
       4, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 3, 1, 0, 1, 3, 1, 3, 1,
       0, 1, 3, 1, 3, 1, 3, 1, 3, 1, 0, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1,
       3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1,
       3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1,
       3, 1])
```

[22]:
```
#Lets visualize this cluster in 2-D plot
plt.scatter(x.iloc[y_means==0,0],x.
 ↪iloc[y_means==0,1],s=100,c='red',label="cluster 1")
plt.scatter(x.iloc[y_means==1,0],x.
 ↪iloc[y_means==1,1],s=100,c='yellow',label="cluster 2")
plt.scatter(x.iloc[y_means==2,0],x.
 ↪iloc[y_means==2,1],s=100,c='green',label="cluster 3")
plt.scatter(x.iloc[y_means==3,0],x.
 ↪iloc[y_means==3,1],s=100,c='blue',label="cluster 4")
plt.scatter(x.iloc[y_means==4,0],x.
 ↪iloc[y_means==4,1],s=100,c='black',label="cluster 5")
plt.scatter(k_means.cluster_centers_[:,0],k_means.cluster_centers_[:
 ↪,1],s=100,c="magenta")# this way be can display cluster centroid using␣
 ↪k-means attribute
plt.title("Customer Segmentation")
plt.xlabel("Annual Income")
plt.ylabel("Spending Score")
plt.legend()
plt.show()
```



8 insight that we can seen of this distribution visualization.

- Cluster 1 customers with medium annual income and medium annual spend.
- Cluster 2 customers with heigh annual income and Low annual spend.
- Cluster 3 customers with Low annual income and Low annual spend.
- Cluster 4 customers with Low annual income and Heigh annual spend.
- Cluster 5 customers with High annual income and High annual spend.

9 Observation:-

According to this customer groups stratigic team will decided for which product we have to target with customer show it is a very good way to organization to understand the customer knowing them difference between customer group it's easier to make strategic decision regarding product growth and marketing

[23]:
```
# let's perform prediction for k-means clustering algorithmns
k_means.predict([[15,39]])# this prediction 2-D list we have to pass annual␣
↪income and spending score so am taking 15 annual income 39 spending score
```

[23]:
```
array([4])
```

[24]:
```
# Save the model
import joblib
joblib.dump(k_means,'Customer_Segmentation')
```

[24]:
```
['Customer_Segmentation']
```

[25]:
```
#We can perform prediction using this save model
model=joblib.load('Customer_Segmentation') # and we know that be save our model␣
 ↪by using this name which is:-"Customer_Segmentation"
```

[26]:
```
model.predict([[15,39]])#perform prediction of this save model we can pass 15␣
 ↪as annual income and 39 as a spending score
```

[26]:
```
array([4])
```

10 Creating GUI / Graphical user interface development (GUI)

[27]:
```
from tkinter import *
import joblib
```

[30]:
```
def show_entry_fields():
    p1 = int(e1.get())
    p2 = int(e2.get())

    model = joblib.load('Customer_Segmentation')
    result = model.predict([[p1, p2]])

    print("This Customer belongs to cluster no:", result[0])

    # Clear previous label (if any)
    for widget in master.winfo_children():
        if isinstance(widget, Label) and widget.grid_info()['row'] == 4:
            widget.destroy()

    if result[0] == 0:
        Label(master, text="Customers with medium annual income and medium␣
 ↪spending score").grid(row=4, columnspan=2)
    elif result[0] == 1:
        Label(master, text="Customers with high annual income but low spending␣
 ↪score").grid(row=4, columnspan=2)
    elif result[0] == 2:
        Label(master, text="Customers with low annual income and low spending␣
 ↪score").grid(row=4, columnspan=2)
    elif result[0] == 3:
```

```
        Label(master, text="Customers with average annual income and high␣
 ↪spending score").grid(row=4, columnspan=2)
    elif result[0] == 4:
        Label(master, text="Customers with high annual income and high spending␣
 ↪score").grid(row=4, columnspan=2)

master = Tk()
master.title("Customer Segmentation Using Machine Learning")

Label(master, text="Customer Segmentation Using Machine Learning", bg="black",␣
 ↪fg="white").grid(row=0, columnspan=2)
Label(master, text="Annual Income").grid(row=1)
Label(master, text="Spending Score").grid(row=2)

e1 = Entry(master)
e2 = Entry(master)

e1.grid(row=1, column=1)
e2.grid(row=2, column=1)

Button(master, text='Predict', command=show_entry_fields).grid(row=3,␣
 ↪columnspan=2)

mainloop()
```

```
This Customer belongs to cluster no: 1
```

[ ]:

[ ]: