Muhammad Umer Adeeb

Question 1: Mall_Customer Dataset

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv('/content/Mall_Customers.csv')
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 200,\n  \"fields\": [\
n    {\n      \"column\": \"CustomerID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 57,\n        \"min\": 1,\n
\"max\": 200,\n        \"num_unique_values\": 200,\n
\"samples\": [\n          96,\n          16,\n          31\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Gender\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"Female\",\n          \"Male\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 13,\n        \"min\": 18,\n
\"max\": 70,\n        \"num_unique_values\": 51,\n        \"samples\":
[\n          55,\n          26\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Annual Income (k$)\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 26,\n        \"min\": 15,\n
\"max\": 137,\n        \"num_unique_values\": 64,\n
\"samples\": [\n          87,\n          101\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Spending Score (1-100)\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
25,\n        \"min\": 1,\n        \"max\": 99,\n
\"num_unique_values\": 84,\n        \"samples\": [\n          83,\n
39\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```


Objective: Group retail store customers based on their purchase history.

```
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
```

```
df.shape
```

```
(200, 5)
```

```
df.isnull().sum()
```

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

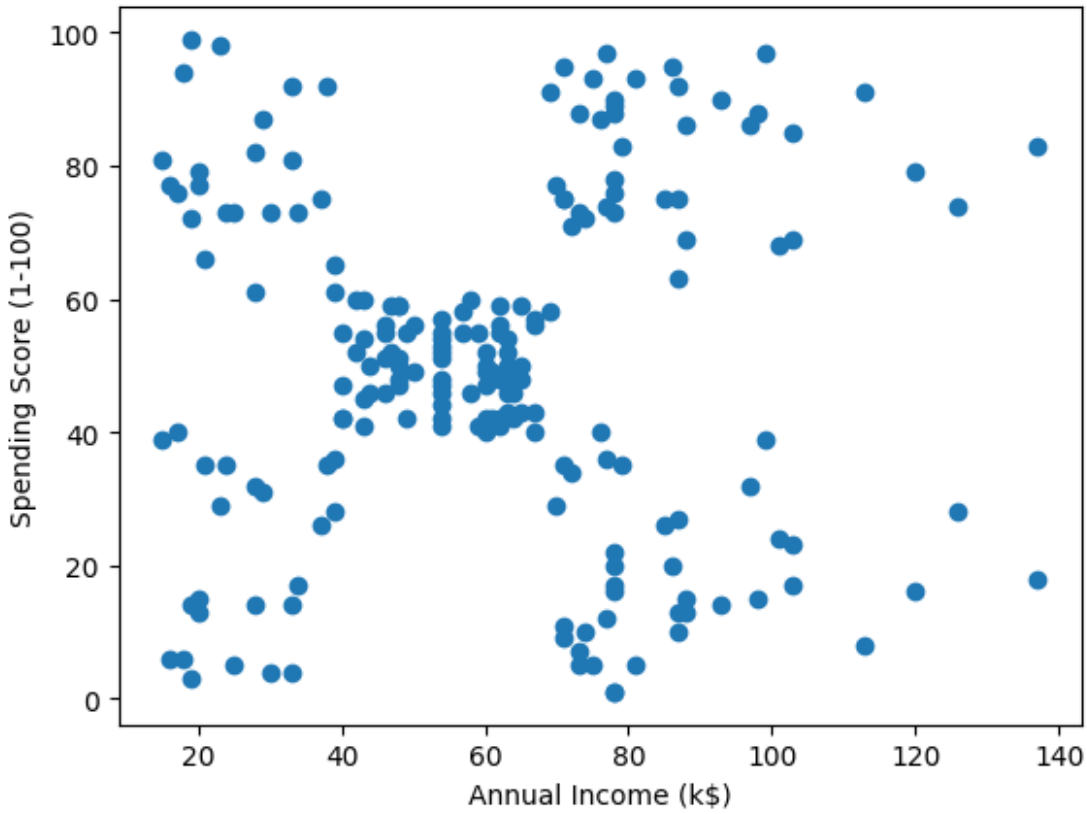
```
duplicate_rows_df = df[df.duplicated()]
print("Number of duplicate rows: ", duplicate_rows_df.shape)
duplicate_rows_df
```

```
Number of duplicate rows:  (0, 5)
```


```
{"repr_error":"Out of range float values are not JSON compliant:
nan","type":"dataframe","variable_name":"duplicate_rows_df"}
```


```
X = df[['Annual Income (k$)', 'Spending Score (1-100)']]
```

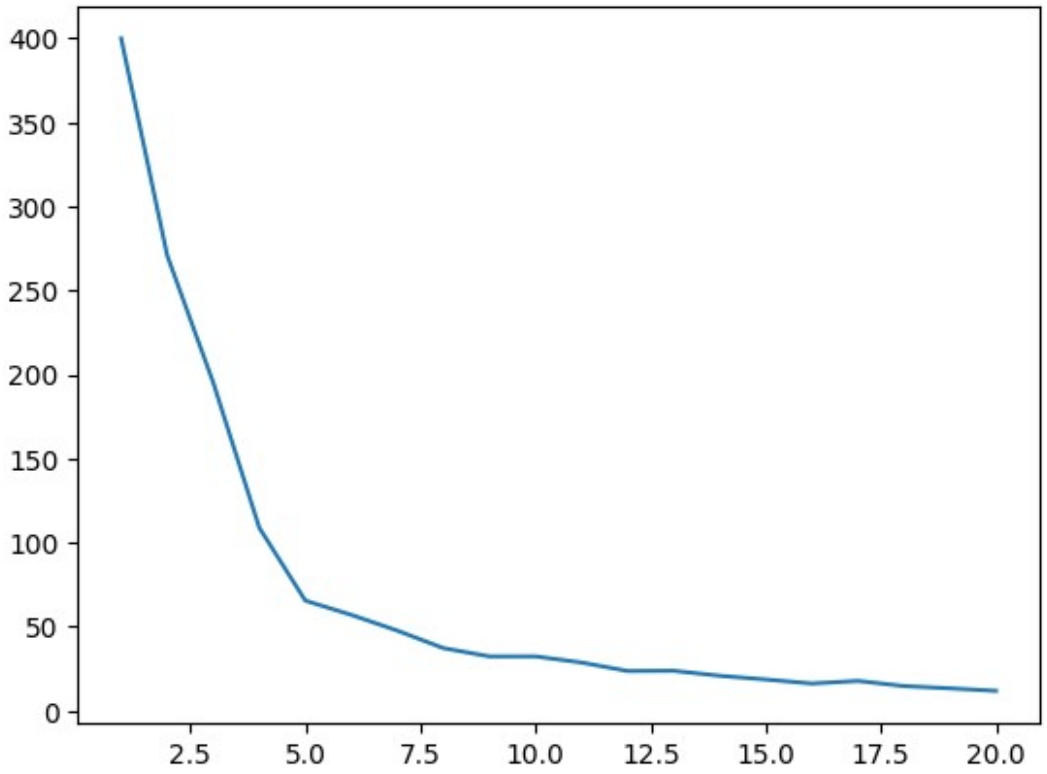
```
plt.scatter(X['Annual Income (k$)'], X['Spending Score (1-100)'])
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.show()
```




```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

from sklearn.cluster import KMeans
wcss = []
for i in range(1, 21):
    kmeans = KMeans(n_clusters=i)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

wcss
```

```
[399.99999999999994,
 270.89235946739063,
 195.2466301907915,
 108.92131661364358,
 65.57885579985046,
 57.11147724296594,
 47.710583761307916,
 37.31912287833882,
 32.39226763033118,
 32.40246298115112,
 28.751291042159014,
 23.710344944514176,
```

```
 23.848393399486483,
 20.877880368245425,
 18.681695442209257,
 16.27860877692049,
 17.83632250802142,
 14.763920183001115,
 13.42240191419025,
 11.900757680608644]

plt.plot(range(1,21), wcss)

[<matplotlib.lines.Line2D at 0x7f7811484950>]
```




```
km = KMeans(n_clusters=5)
y_means = km.fit_predict(X_scaled)

y_means

array([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,
0,
       3, 1, 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, 0, 0, 0, 2, 4, 2, 0, 2, 4, 2, 4,
2,
       0, 2, 4, 2, 4, 2, 4, 2, 4, 2, 0, 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, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4, 2, 4,
2,
       4, 2], dtype=int32)
```

```
X_scaled[y_means ==0]
```

```
array([[-0.82293289,  0.41927286],
       [-0.78476346,  0.18634349],
       [-0.78476346, -0.12422899],
       [-0.78476346, -0.3183368 ],
       [-0.78476346, -0.3183368 ],
       [-0.70842461,  0.06987881],
       [-0.70842461,  0.38045129],
       [-0.67025518,  0.14752193],
       [-0.67025518,  0.38045129],
       [-0.67025518, -0.20187212],
       [-0.67025518, -0.35715836],
       [-0.63208575, -0.00776431],
       [-0.63208575, -0.16305055],
       [-0.55574689,  0.03105725],
       [-0.55574689, -0.16305055],
       [-0.55574689,  0.22516505],
       [-0.55574689,  0.18634349],
       [-0.51757746,  0.06987881],
       [-0.51757746,  0.34162973],
       [-0.47940803,  0.03105725],
       [-0.47940803,  0.34162973],
       [-0.47940803, -0.00776431],
       [-0.47940803, -0.08540743],
       [-0.47940803,  0.34162973],
       [-0.47940803, -0.12422899],
       [-0.4412386 ,  0.18634349],
       [-0.4412386 , -0.3183368 ],
       [-0.40306917, -0.04658587],
       [-0.40306917,  0.22516505],
       [-0.25039146, -0.12422899],
       [-0.25039146,  0.14752193],
       [-0.25039146,  0.10870037],
       [-0.25039146, -0.08540743],
       [-0.25039146,  0.06987881],
       [-0.25039146, -0.3183368 ],
```

```
       [-0.25039146,  0.03105725],
       [-0.25039146,  0.18634349],
       [-0.25039146, -0.35715836],
       [-0.25039146, -0.24069368],
       [-0.25039146,  0.26398661],
       [-0.25039146, -0.16305055],
       [-0.13588317,  0.30280817],
       [-0.13588317,  0.18634349],
       [-0.09771374,  0.38045129],
       [-0.09771374, -0.16305055],
       [-0.05954431,  0.18634349],
       [-0.05954431, -0.35715836],
       [-0.02137488, -0.04658587],
       [-0.02137488, -0.39597992],
       [-0.02137488, -0.3183368 ],
       [-0.02137488,  0.06987881],
       [-0.02137488, -0.12422899],
       [-0.02137488, -0.00776431],
       [ 0.01679455, -0.3183368 ],
       [ 0.01679455, -0.04658587],
       [ 0.05496398, -0.35715836],
       [ 0.05496398, -0.08540743],
       [ 0.05496398,  0.34162973],
       [ 0.05496398,  0.18634349],
       [ 0.05496398,  0.22516505],
       [ 0.05496398, -0.3183368 ],
       [ 0.09313341, -0.00776431],
       [ 0.09313341, -0.16305055],
       [ 0.09313341, -0.27951524],
       [ 0.09313341, -0.08540743],
       [ 0.09313341,  0.06987881],
       [ 0.09313341,  0.14752193],
       [ 0.13130284, -0.3183368 ],
       [ 0.13130284, -0.16305055],
       [ 0.16947227, -0.08540743],
       [ 0.16947227, -0.00776431],
       [ 0.16947227, -0.27951524],
       [ 0.16947227,  0.34162973],
       [ 0.24581112, -0.27951524],
       [ 0.24581112,  0.26398661],
       [ 0.24581112,  0.22516505],
       [ 0.24581112, -0.39597992],
       [ 0.32214998,  0.30280817],
       [ 0.39848884, -0.59008772],
       [ 0.43665827, -0.62890928],
       [ 0.58933599, -0.39597992]])

plt.scatter(X_scaled[y_means==0,0] , X_scaled[y_means==0,1],
color='blue')
plt.scatter(X_scaled[y_means==1,0], X_scaled[y_means==1,1],
```

```
color='red')
plt.scatter(X_scaled[y_means==2,0], X_scaled[y_means==2,1],
color='green')
plt.scatter(X_scaled[y_means==3,0], X_scaled[y_means==3,1],
color='yellow')
plt.scatter(X_scaled[y_means==4,0], X_scaled[y_means==4,1],
color='black')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.show()
```