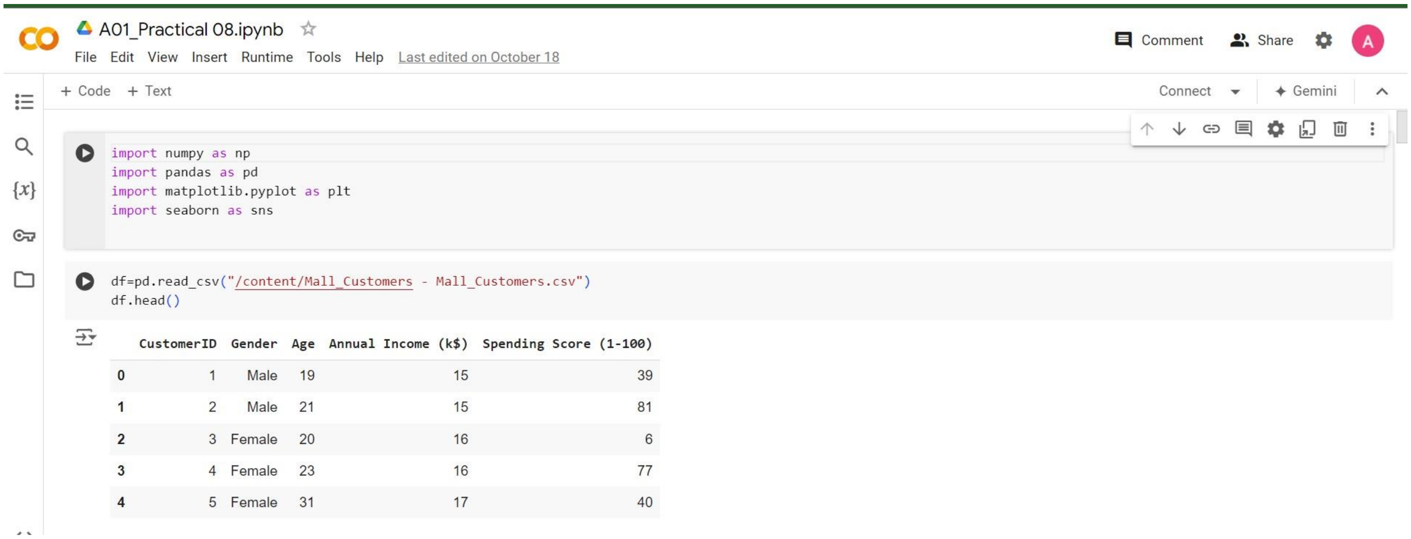
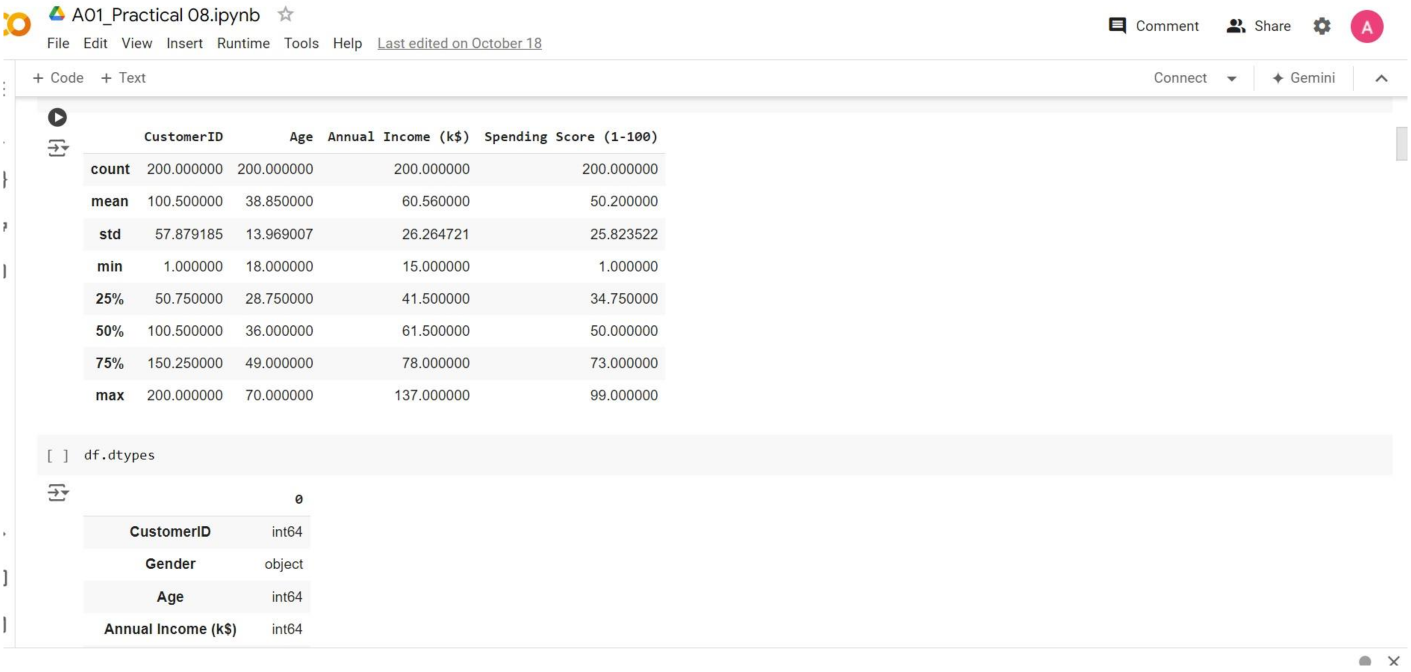
Prajjwal Mohan
A3-47
Practical Number 08


A01_Practical 08.ipynb
File Edit View Insert Runtime Tools Help Last edited on October 18
Comment Share
+ Code + Text
Connect
Gemini

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

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

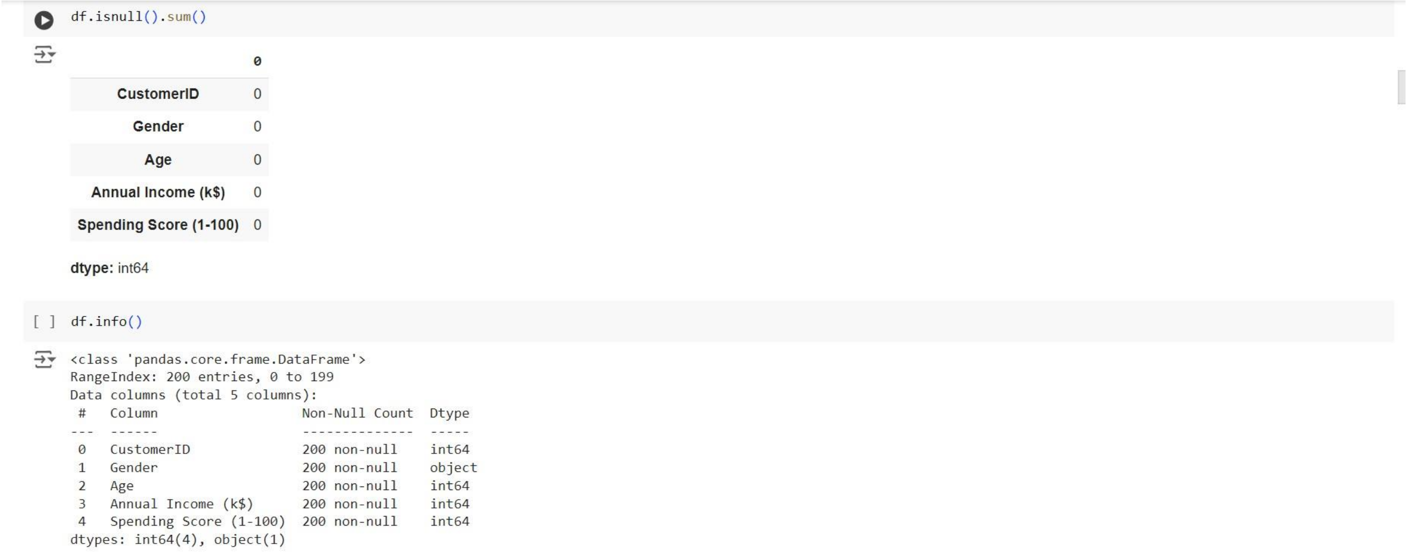
	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40


A01_Practical 08.ipynb
File Edit View Insert Runtime Tools Help Last edited on October 18
Comment Share
+ Code + Text
Connect
Gemini

	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

```
df.dtypes
```

	0
CustomerID	int64
Gender	object
Age	int64
Annual Income (k$)	int64

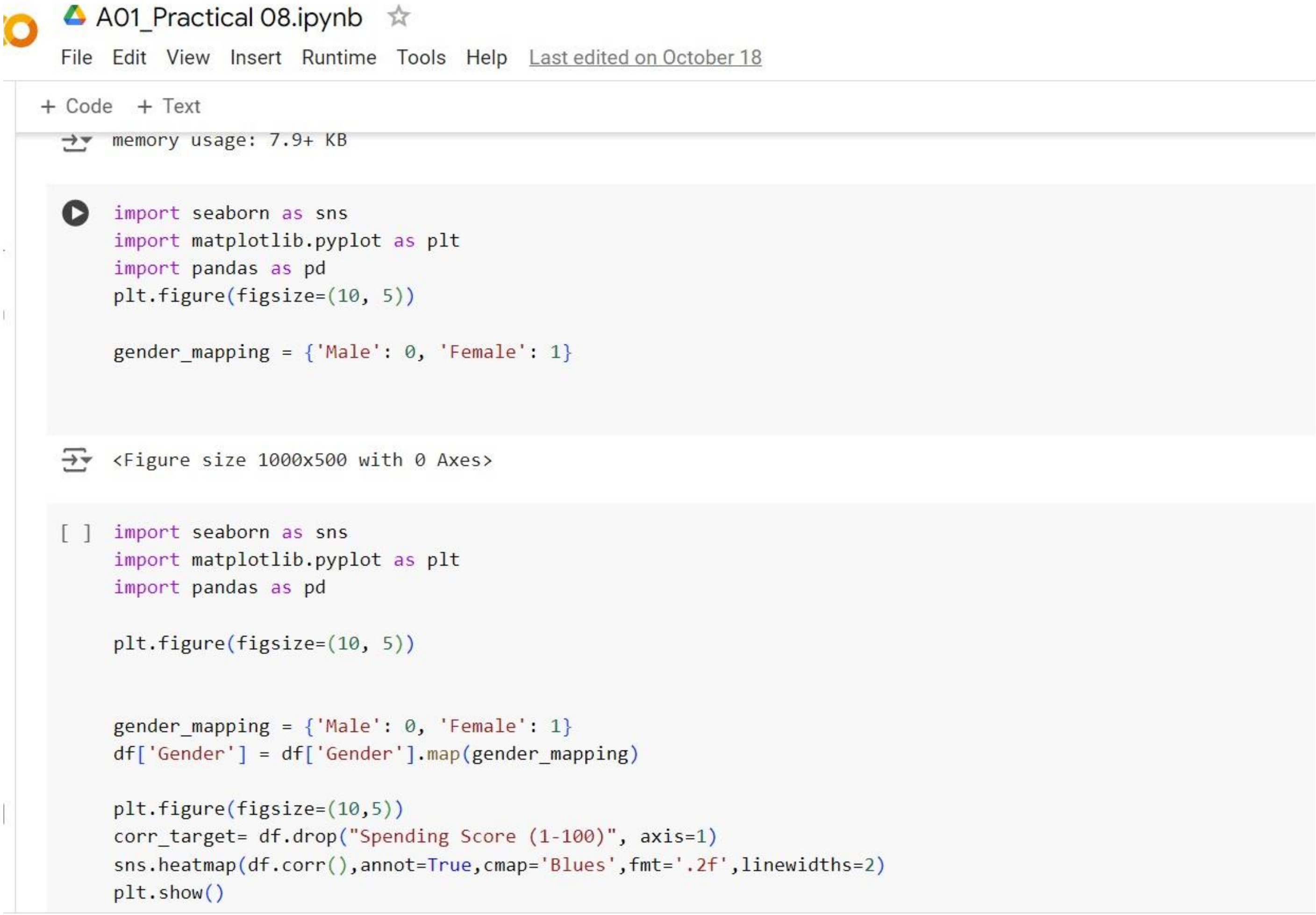


```
df.isnull().sum()
```

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

dtype: int64
```

```
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)
```



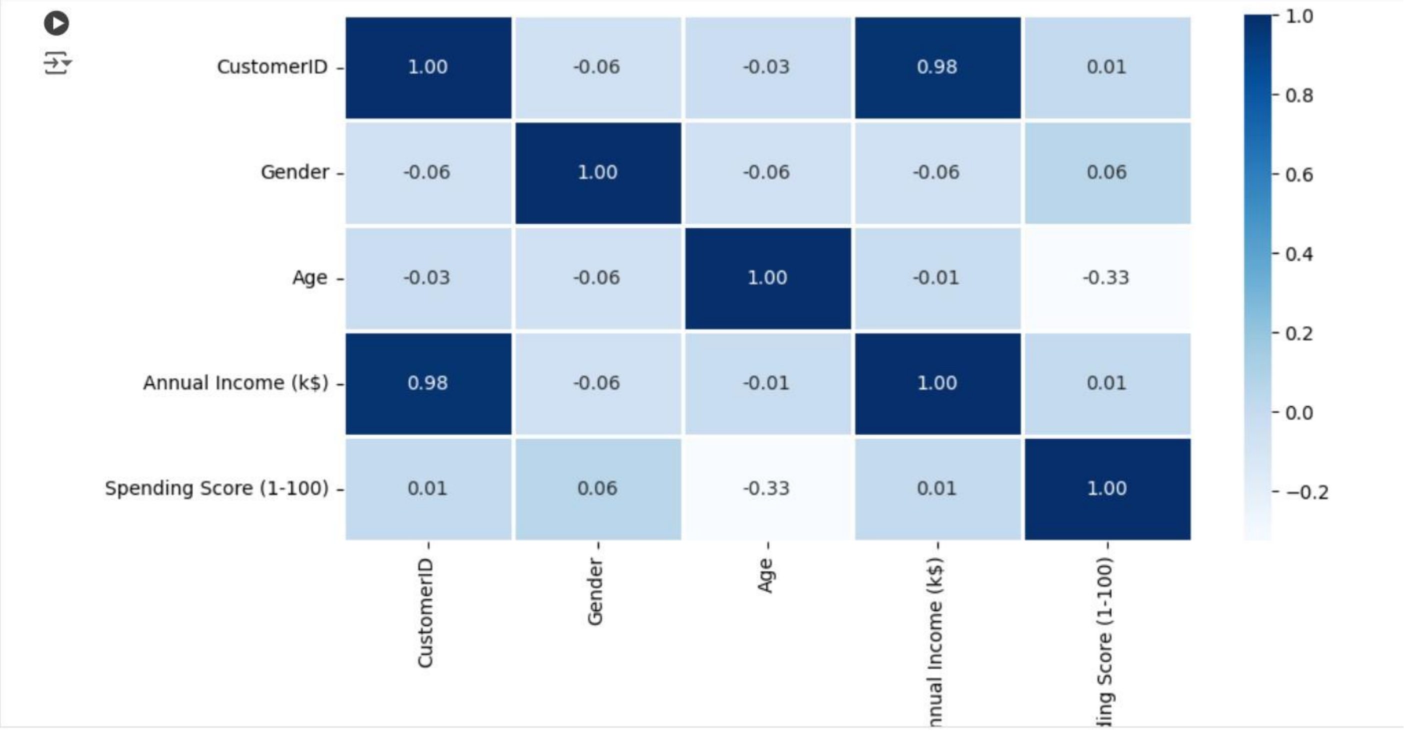
A01_Practical 08.ipynb
File Edit View Insert Runtime Tools Help Last edited on October 18
+ Code + Text

```
memory usage: 7.9+ KB
```

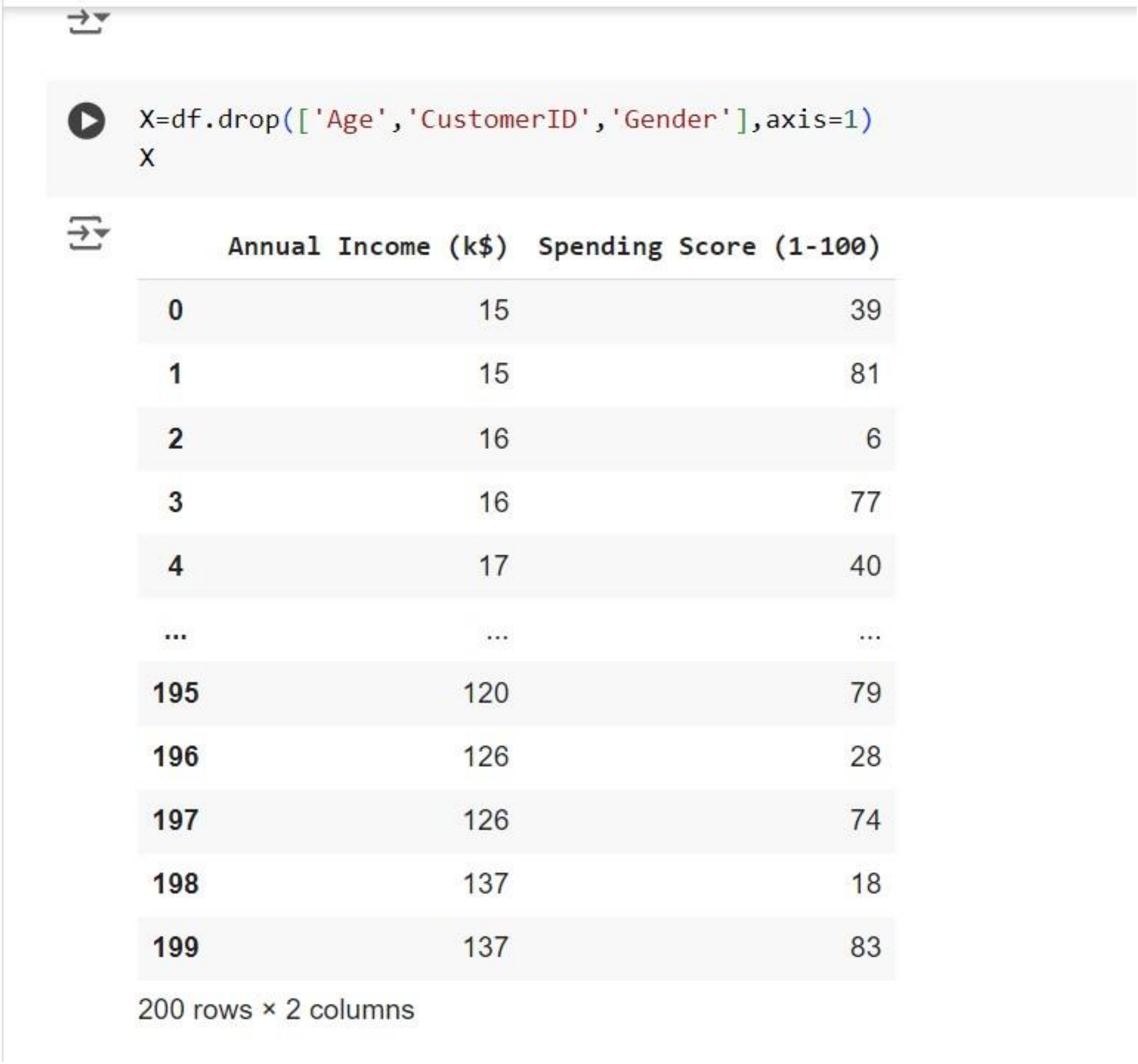
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
plt.figure(figsize=(10, 5))

gender_mapping = {'Male': 0, 'Female': 1}
```

```
<Figure size 1000x500 with 0 Axes>
```

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

plt.figure(figsize=(10, 5))


gender_mapping = {'Male': 0, 'Female': 1}
df['Gender'] = df['Gender'].map(gender_mapping)

plt.figure(figsize=(10,5))
corr_target= df.drop("Spending Score (1-100)", axis=1)
sns.heatmap(df.corr(),annot=True,cmap='Blues',fmt='.2f',linewidths=2)
plt.show()
```



	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
CustomerID	1.00	-0.06	-0.03	0.98	0.01
Gender	-0.06	1.00	-0.06	-0.06	0.06
Age	-0.03	-0.06	1.00	-0.01	-0.33
Annual Income (k$)	0.98	-0.06	-0.01	1.00	0.01
Spending Score (1-100)	0.01	0.06	-0.33	0.01	1.00

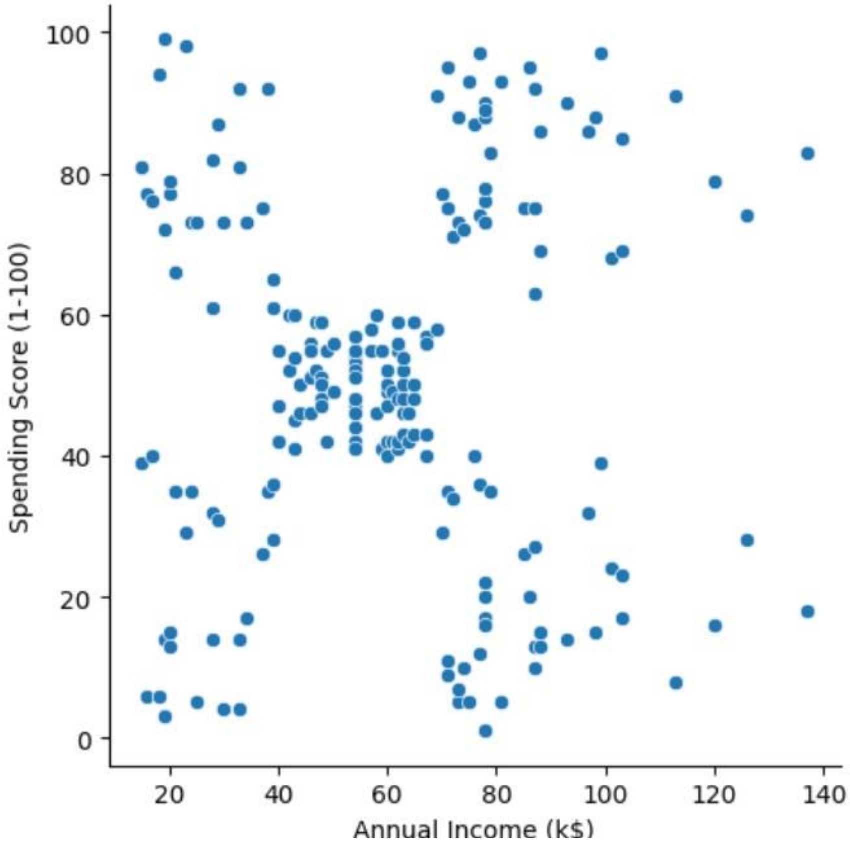


```
X=df.drop(['Age','CustomerID','Gender'],axis=1)
X
```

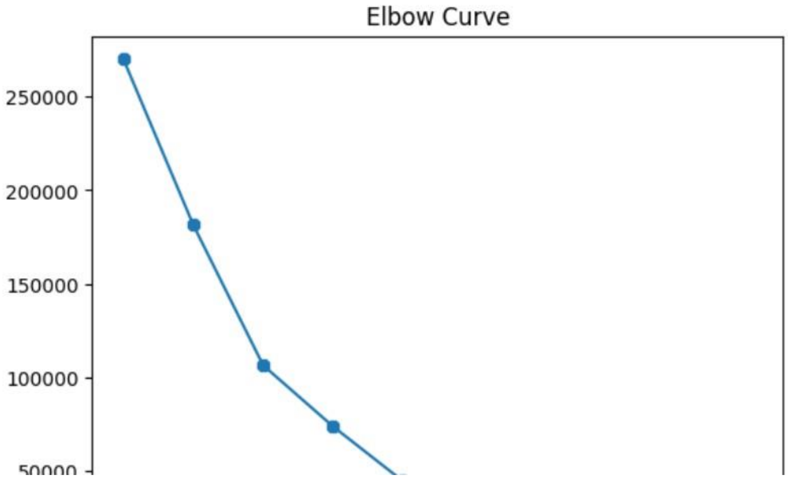
```
      Annual Income (k$)  Spending Score (1-100)
  0                   15                      39
  1                   15                      81
  2                   16                       6
  3                   16                      77
  4                   17                      40
...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

200 rows × 2 columns
```

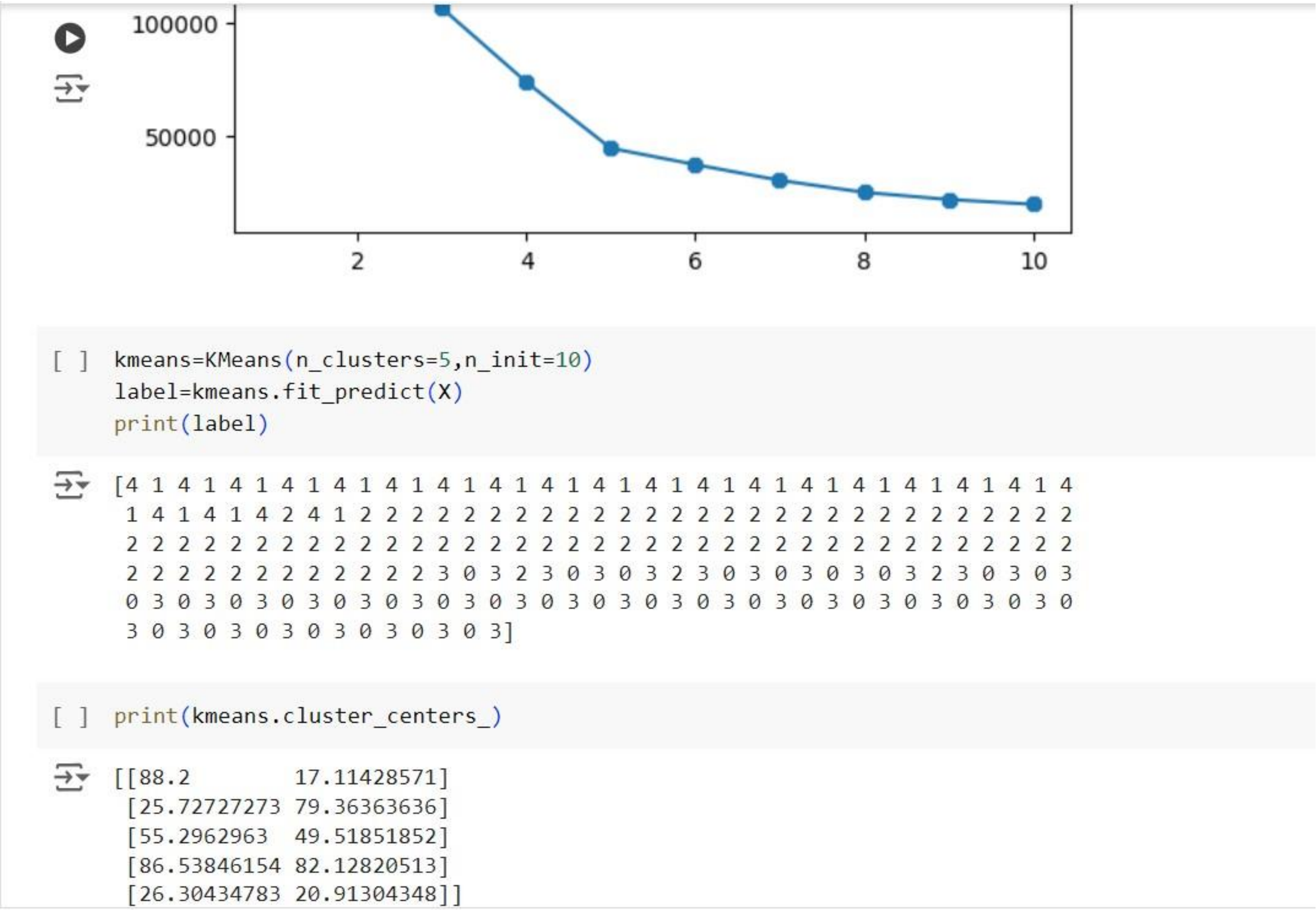
```
sns.relplot( x="Annual Income (k$)", y="Spending Score (1-100)", data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7b9cd5255990>
```



```
from sklearn.cluster import KMeans
wcss=[] #within clusters Sum of squared distances of samples to their closest cluster center,
for i in range(1,11):
    kmeans=KMeans(n_clusters=i,n_init=10) # Number of times the k-means algorithm is run with different centroid seeds.
    kmeans.fit(X)
    wcss.append(kmeans.inertia_) #seggregate the datapoints into clusters
plt.plot(range(1,11),wcss,marker="8")
plt.title('Elbow Curve')
plt.show()
```



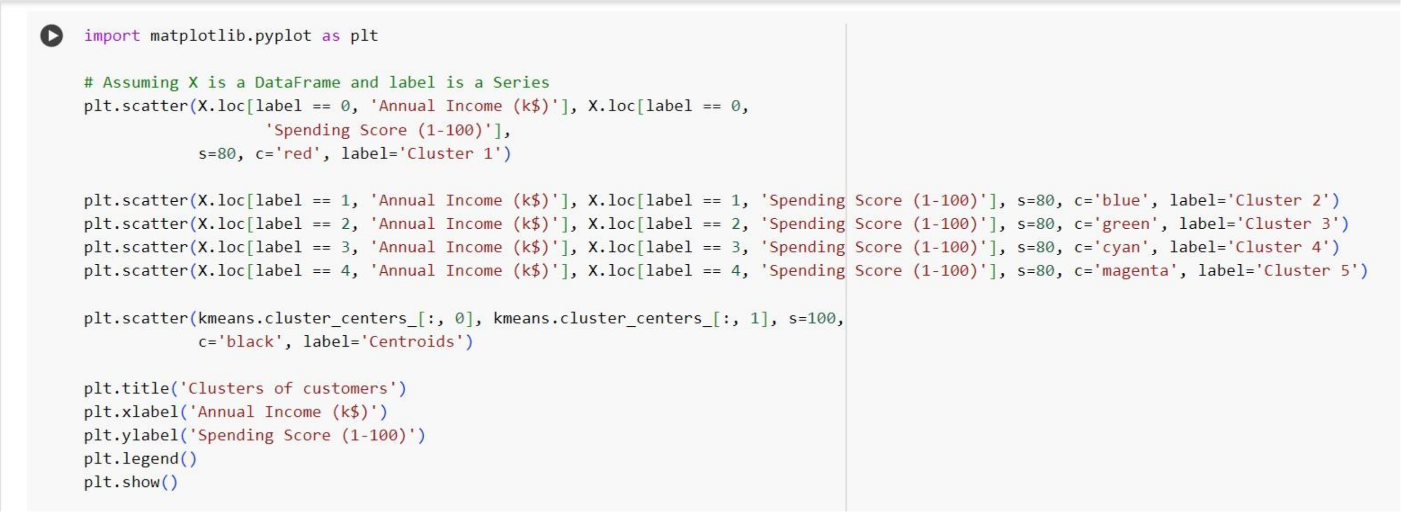


```
[ ] kmeans=KMeans(n_clusters=5,n_init=10)
    label=kmeans.fit_predict(X)
    print(label)
```

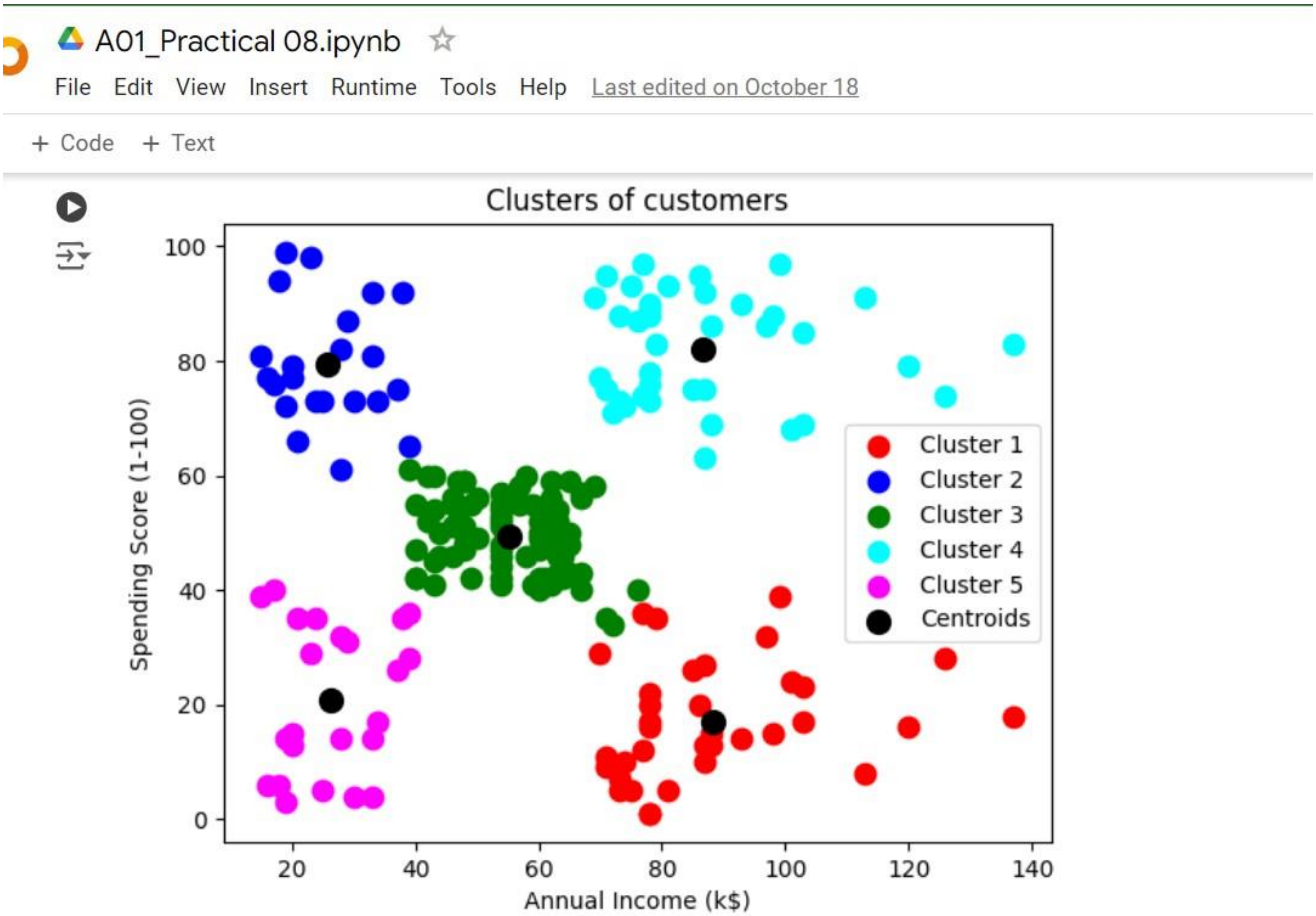
```
[4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4 1 4
 1 4 1 4 1 4 2 4 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 3 0 3 2 3 0 3 0 3 2 3 0 3 0 3 0 3 0 3 2 3 0 3 0 3
 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3 0
 3 0 3 0 3 0 3 0 3 0 3 0 3 0 3]
```

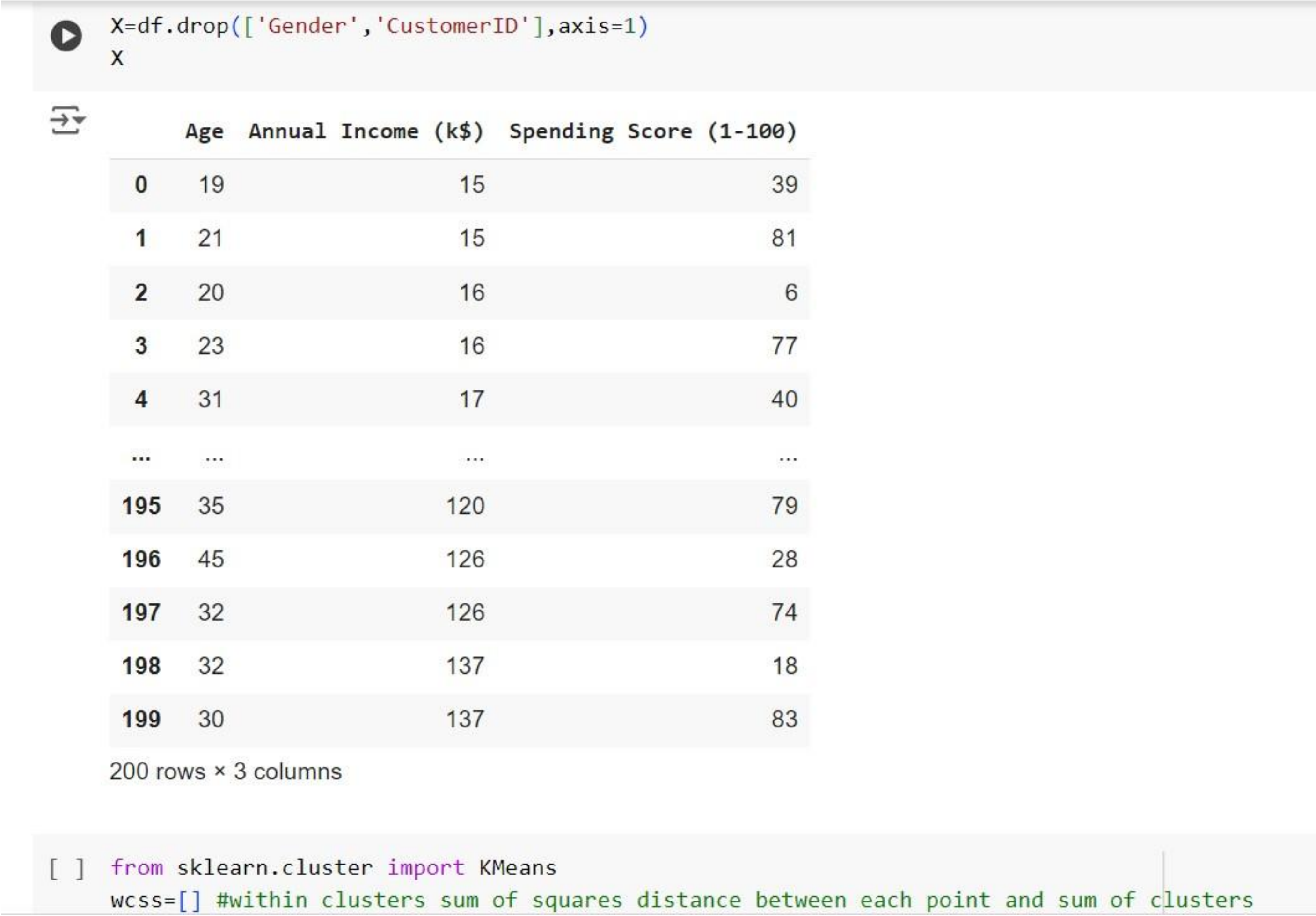
```
[ ] print(kmeans.cluster_centers_)
```

```
[[88.2        17.11428571]
 [25.72727273 79.36363636]
 [55.2962963  49.51851852]
 [86.53846154 82.12820513]
 [26.30434783 20.91304348]]
```



```
import matplotlib.pyplot as plt

# Assuming X is a DataFrame and label is a Series
plt.scatter(X.loc[label == 0, 'Annual Income (k$)'], X.loc[label == 0,
                   'Spending Score (1-100)'],
            s=80, c='red', label='Cluster 1')

plt.scatter(X.loc[label == 1, 'Annual Income (k$)'], X.loc[label == 1, 'Spending Score (1-100)'], s=80, c='blue', label='Cluster 2')
plt.scatter(X.loc[label == 2, 'Annual Income (k$)'], X.loc[label == 2, 'Spending Score (1-100)'], s=80, c='green', label='Cluster 3')
plt.scatter(X.loc[label == 3, 'Annual Income (k$)'], X.loc[label == 3, 'Spending Score (1-100)'], s=80, c='cyan', label='Cluster 4')
plt.scatter(X.loc[label == 4, 'Annual Income (k$)'], X.loc[label == 4, 'Spending Score (1-100)'], s=80, c='magenta', label='Cluster 5')

plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=100,
            c='black', label='Centroids')

plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```


A01_Practical 08.ipynb
File
Edit
View
Insert
Runtime
Tools
Help
Last edited on October 18
+ Code
+ Text
Clusters of customers
Spending Score (1-100)
100
80
60
40
20
0
20
40
60
80
100
120
140
Annual Income (k$)
● Cluster 1
● Cluster 2
● Cluster 3
● Cluster 4
● Cluster 5
● Centroids

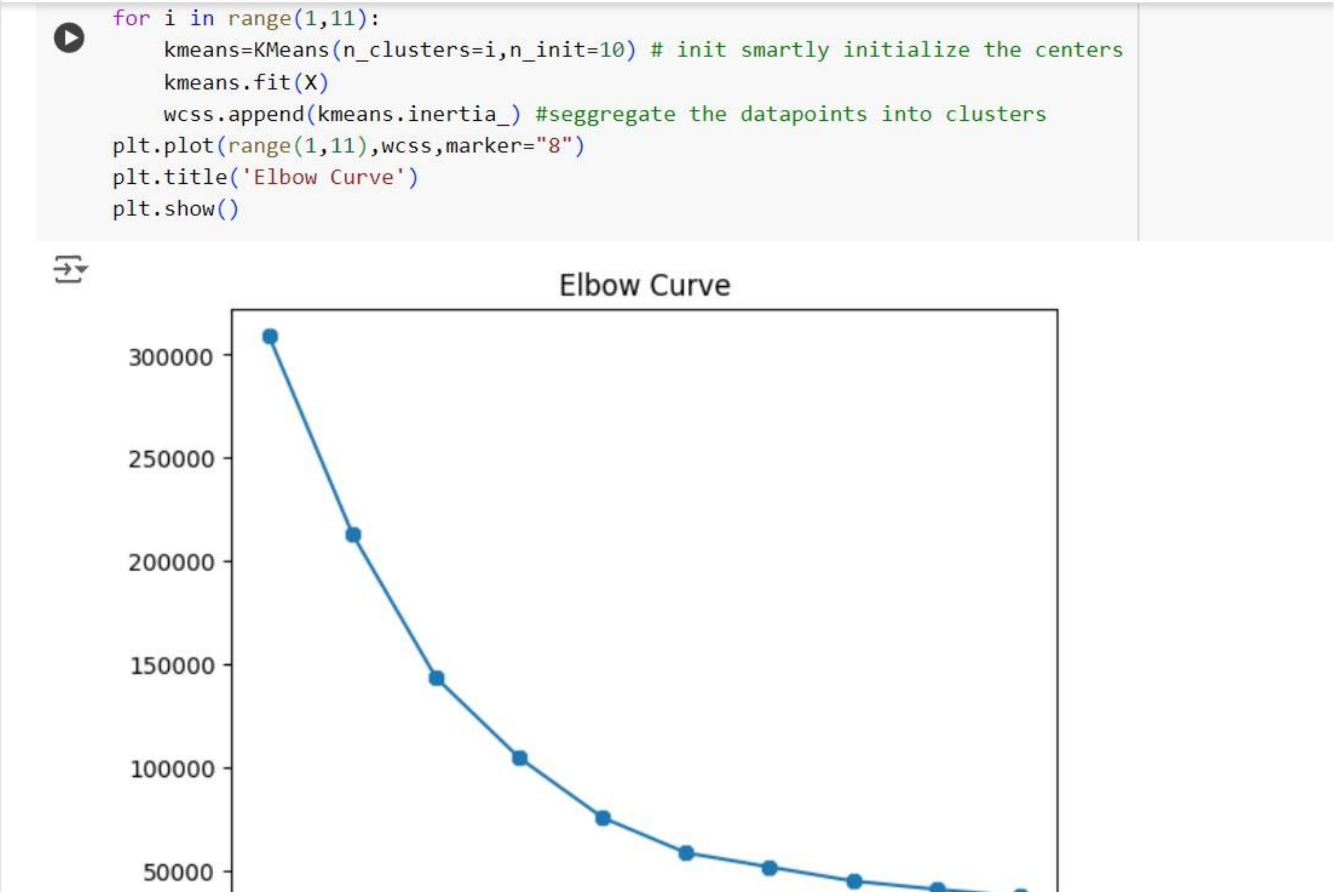


```python
X=df.drop(['Gender','CustomerID'],axis=1)
X
```

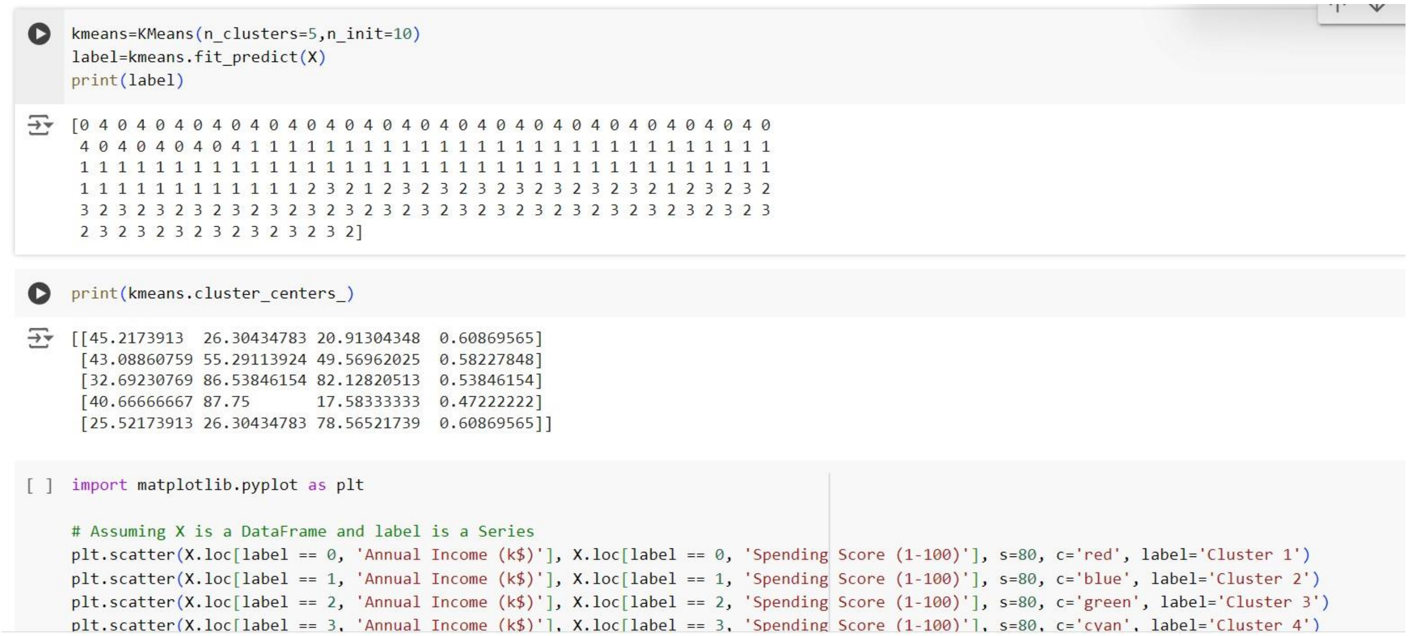
	Age	Annual Income (k$)	Spending Score (1-100)
0	19	15	39
1	21	15	81
2	20	16	6
3	23	16	77
4	31	17	40
...	...	...	...
195	35	120	79
196	45	126	28
197	32	126	74
198	32	137	18
199	30	137	83

200 rows × 3 columns

```python
from sklearn.cluster import KMeans
wcss=[] #within clusters sum of squares distance between each point and sum of clusters
```



```
for i in range(1,11):
    kmeans=KMeans(n_clusters=i,n_init=10) # init smartly initialize the centers
    kmeans.fit(X)
    wcss.append(kmeans.inertia_) #seggregate the datapoints into clusters
plt.plot(range(1,11),wcss,marker="8")
plt.title('Elbow Curve')
plt.show()
```



```
kmeans=KMeans(n_clusters=5,n_init=10)
label=kmeans.fit_predict(X)
print(label)
```

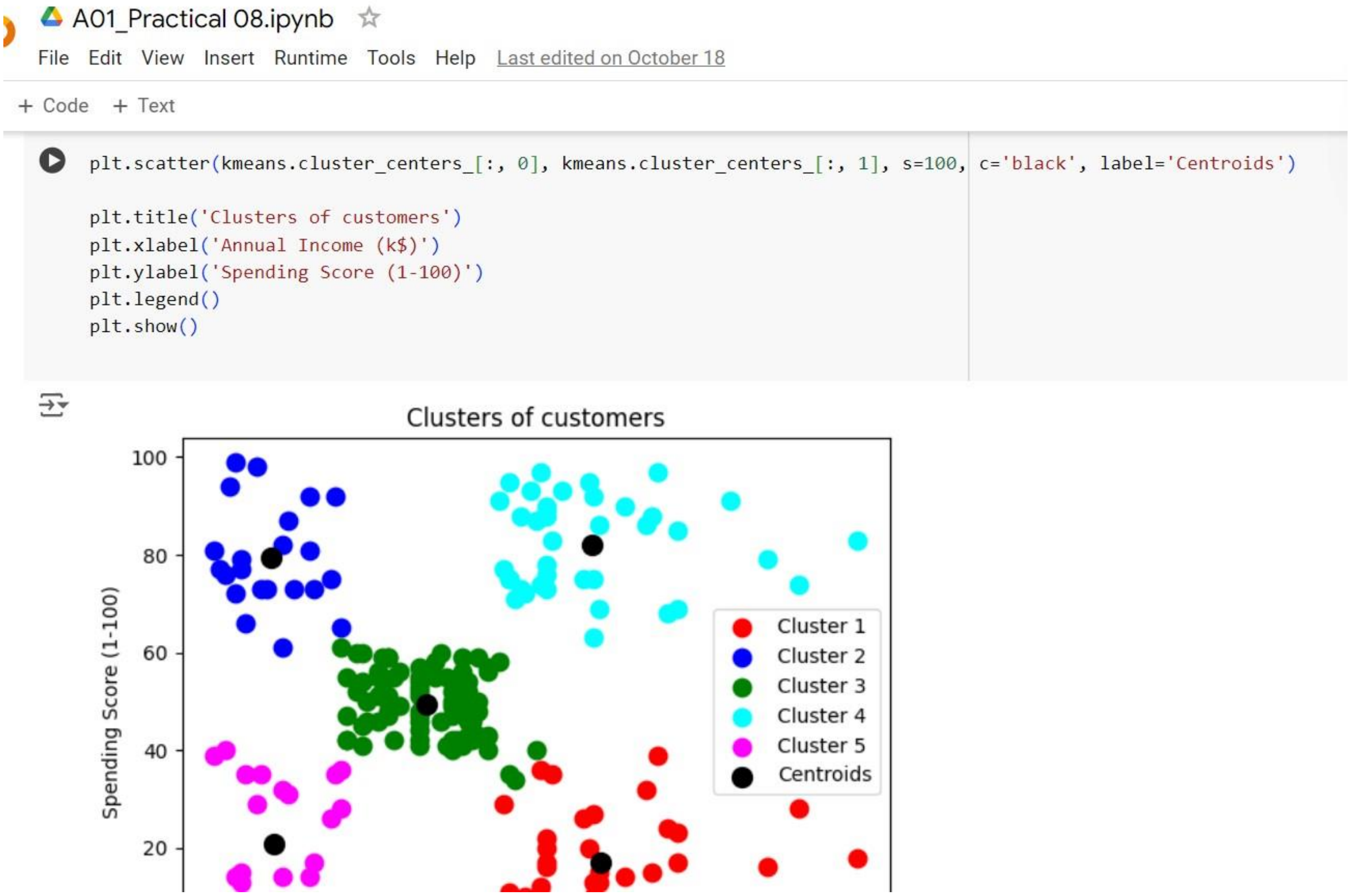
```
[0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0 4 0
 4 0 4 0 4 0 4 0 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 2 3 2 1 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 1 2 3 2 3 2
 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3
 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2]
```

```
print(kmeans.cluster_centers_)
```

```
[[45.2173913  26.30434783 20.91304348  0.60869565]
 [43.08860759 55.29113924 49.56962025  0.58227848]
 [32.69230769 86.53846154 82.12820513  0.53846154]
 [40.66666667 87.75        17.58333333  0.47222222]
 [25.52173913 26.30434783 78.56521739  0.60869565]]
```

[ ]

```
import matplotlib.pyplot as plt

# Assuming X is a DataFrame and label is a Series
plt.scatter(X.loc[label == 0, 'Annual Income (k$)'], X.loc[label == 0, 'Spending Score (1-100)'], s=80, c='red', label='Cluster 1')
plt.scatter(X.loc[label == 1, 'Annual Income (k$)'], X.loc[label == 1, 'Spending Score (1-100)'], s=80, c='blue', label='Cluster 2')
plt.scatter(X.loc[label == 2, 'Annual Income (k$)'], X.loc[label == 2, 'Spending Score (1-100)'], s=80, c='green', label='Cluster 3')
plt.scatter(X.loc[label == 3, 'Annual Income (k$)'], X.loc[label == 3, 'Spending Score (1-100)'], s=80, c='cyan', label='Cluster 4')
```



```
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=100, c='black', label='Centroids')

plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```