MALL CUSTOMERS PROJECT

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

```
from google.colab import files
upload=files.upload()
```

Choose Files Mall_Customers.csv

- **Mall_Customers.csv**(text/csv) - 3981 bytes, last modified: 1/1/2025 - 100% done

```
Saving Mall_Customers.csv to Mall_Customers.csv
```

```
df=pd.read_csv("./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

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.shape
```

```
(200, 5)
```

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

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

dtype: int64

```
df["CustomerID"].duplicated().sum()
```

```
0
```

```
df=df.drop("CustomerID",axis=1)
```

```
df.head()
```

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

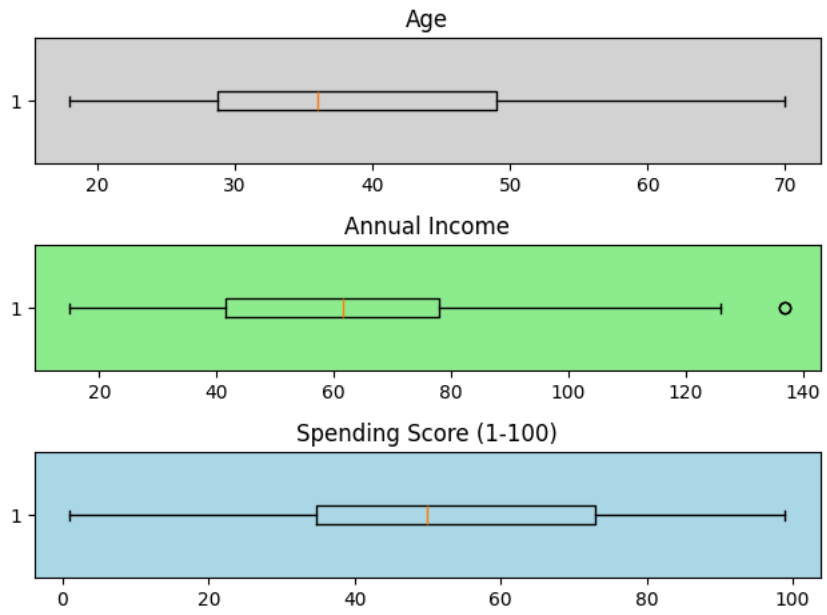
Next steps: Generate code with df | View recommended plots | New interactive sheet

```
plt.subplot(3,1,1,facecolor="lightgrey")
plt.boxplot(df['Age'], vert=False)
plt.title("Age")

plt.subplot(3,1,2,facecolor="lightgreen")
plt.boxplot(df['Annual Income (k$)'], vert=False)
plt.title("Annual Income")

plt.subplot(3,1,3,facecolor="lightblue")
plt.boxplot(df['Spending Score (1-100)'], vert=False)
plt.title("Spending Score (1-100)")

plt.tight_layout()
plt.show()
```




```
# calculate summary statistics
mean = df["Annual Income (k$)"].mean()
std  = df["Annual Income (k$)"].std()

# Calculate the lower and upper bounds
lower_bound = mean - std*2
upper_bound = mean + std*2

print("Annual Income")
```

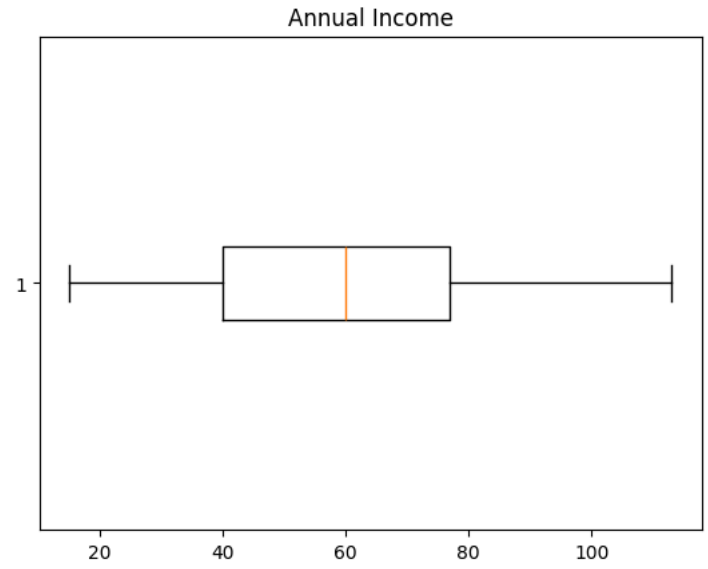
```
print('Lower Bound :',lower_bound)
print('Upper Bound :',upper_bound)

# Drop the outliers
df = df[(df["Annual Income (k$)"] >= lower_bound) & (df["Annual Income (k$)"] <= upper_bound)]
```

```
Annual Income
Lower Bound : 8.030557669457494
Upper Bound : 113.08944233054251
```

```
plt.boxplot(df['Annual Income (k$)'], vert=False)
plt.title("Annual Income")
```

```
Text(0.5, 1.0, 'Annual Income')
```




```
df.shape
```

```
(194, 4)
```

```
from sklearn.cluster import KMeans
```

```
df.head()
```

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

Next steps: Generate code with df | View recommended plots | New interactive sheet

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

```
df.head()
```

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

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
X=df.iloc[:,:]
```

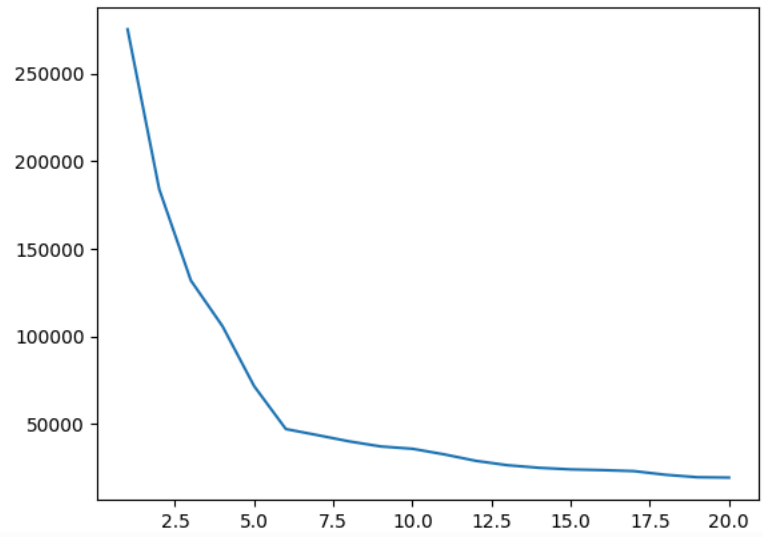
```
wcss=[]
for i in range(1,21):
    km=KMeans(n_clusters=i)
    km.fit_predict(X)
    wcss.append(km.inertia_)
```

```
wcss
```

```
[275244.6288659795,
 183967.6023519163,
 131884.22015098727,
 105767.72929482804,
 71544.15593423716,
 47102.70317650184,
 43631.47211225872,
 40070.0450261629,
 37162.23065756082,
 35845.48626373627,
 32654.713152065313,
 28960.391486068103,
 26462.460721231804,
 24981.80213813962,
 24074.57333503098,
 23659.940890215734,
 23082.371428571423,
 20994.91627768517,
 19599.809944684952,
 19391.345422910425]
```

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

```
[<matplotlib.lines.Line2D at 0x7bd08d8c85b0>]
```




```
km=KMeans(n_clusters=5)
```

```
km.fit(X)
```

```
KMeans
KMeans(n_clusters=5)
```

```
Y_pred=km.predict(X)
```

```
Y_pred
```

```
array([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, 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, 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, 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],
      dtype=int32)
```

```
X[Y_pred==3].head()
```

	Gender	Age	Annual Income (k$)	Spending Score (1-100)
124	0	23	70	29
126	1	43	71	35
128	1	59	71	11
130	1	47	71	9
132	0	25	72	34

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
```

```
pca.fit(df)
```

```
PCA(n_components=2)
```

```
transformed_data=pca.transform(df)
transformed_data
```

```
array([[-8.86067439e+00, -4.31613041e+01],
       [ 3.14357334e+01, -4.55207435e+01],
       [-4.10603372e+01, -4.02589799e+01],
       [ 2.71486432e+01, -4.42551682e+01],
       [-1.05674532e+01, -4.09833845e+01],
       [ 2.64715800e+01, -4.32198554e+01],
       [-4.44388562e+01, -3.79629154e+01],
       [ 4.37683647e+01, -4.32300229e+01],
       [-5.40563285e+01, -3.62128112e+01],
       [ 2.08437233e+01, -4.08351512e+01],
       [-4.40801130e+01, -3.67812800e+01],
       [ 4.58821485e+01, -4.22777143e+01],
       [-4.09485709e+01, -3.60211516e+01],
       [ 2.71562555e+01, -4.02425210e+01],
       [-3.79925259e+01, -3.63254492e+01],
       [ 2.95624726e+01, -4.03956727e+01],
       [-1.61123627e+01, -3.66251476e+01],
       [ 1.74721202e+01, -3.86947982e+01],
       [-2.57820886e+01, -3.39452769e+01],
       [ 4.51524785e+01, -3.82279206e+01],
       [-1.59331083e+01, -3.36295789e+01],
       [ 2.32802808e+01, -3.60002893e+01],
       [-4.75537709e+01, -3.06988194e+01],
       [ 2.19409170e+01, -3.48822302e+01],
       [-4.05043096e+01, -2.80586214e+01],
       [ 3.13230165e+01, -3.24418536e+01],
       [-2.09351058e+01, -2.92667765e+01],
       [ 9.54196593e+00, -3.11222590e+01],
       [-2.06791138e+01, -2.83113922e+01],
       [ 3.76369515e+01, -3.18502934e+01],
       [-5.14903178e+01, -2.53700395e+01],
       [ 2.45760529e+01, -3.00922902e+01],
       [-4.96768141e+01, -2.25154157e+01],
       [ 4.38954743e+01, -2.82421124e+01],
       [-3.90368954e+01, -2.31677008e+01],
       [ 3.25210787e+01, -2.75548207e+01],
       [-3.44321915e+01, -2.24807954e+01],
       [ 2.27175548e+01, -2.59197859e+01],
       [-2.41170368e+01, -2.01203474e+01],
       [ 2.71719421e+01, -2.32393658e+01],
       [-2.20865534e+01, -1.90568571e+01],
       [ 4.27969866e+01, -2.31313883e+01],
       [-1.70917063e+01, -1.84544459e+01],
```

```
            [ 1.11388332e+01, -2.02234286e+01],
            [-2.50879360e+01, -1.79785044e+01],
            [ 1.66538642e+01, -2.05918182e+01],
            [ 9.43763410e-01, -1.85028298e+01],
            [-1.45553772e+00, -1.85053897e+01],
            [-6.77479574e+00, -1.81797821e+01],
            [-7.24132335e+00, -1.81398178e+01],
            [-1.61417297e+00, -1.63550934e+01],
            [ 9.88101389e+00, -1.71307669e+01],
            [ 4.58588673e+00, -1.58308629e+01],
            [ 3.87637397e+00, -1.56130651e+01],
            [-8.58104032e+00, -1.49370444e+01],
            [-1.17648359e+01, -1.47674063e+01],
            [-3.90135468e+00, -1.42045393e+01],
```

```
temp_df = pd.DataFrame({
    "PC1(Feature1)" : transformed_data[:,0],
    "PC2(Fearure2)" : transformed_data[:,1]
})
temp_df.head()
```

	PC1(Feature1)	PC2(Fearure2)
0	-8.860674	-43.161304
1	31.435733	-45.520743
2	-41.060337	-40.258980
3	27.148643	-44.255168
4	-10.567453	-40.983385

Next steps: Generate code with temp_df | View recommended plots | New interactive sheet

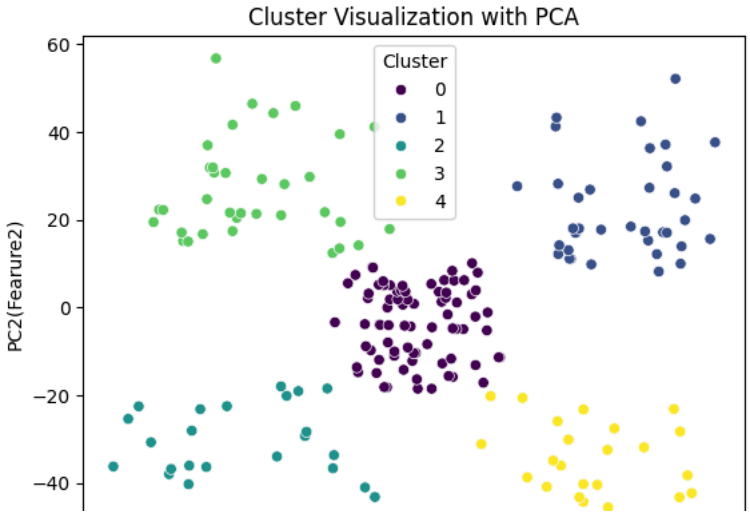
```
temp_df['Cluster'] = Y_pred
temp_df.head()
```

	PC1(Feature1)	PC2(Fearure2)	Cluster
0	-8.860674	-43.161304	2
1	31.435733	-45.520743	4
2	-41.060337	-40.258980	2
3	27.148643	-44.255168	4
4	-10.567453	-40.983385	2

Next steps: Generate code with temp_df | View recommended plots | New interactive sheet

```
sns.scatterplot(x=temp_df.iloc[:,0], y=temp_df.iloc[:,1], hue=temp_df['Cluster'], palette='viridis')
plt.title('Cluster Visualization with PCA')
plt.show()
```

[4]


Cluster Visualization with PCA
Cluster
0
1
2
3
4
PC2(Fearure2)
60
40
20
0
−20
−40