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

Choose files Mall_Customers.csv

- **Mall_Customers.csv**(text/csv) - 3981 bytes, last modified: 27/06/2025 - 100% done

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

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
```

```
#loading the data
data = pd.read_csv("Mall_Customers.csv")
data.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 data View recommended plots New interactive sheet

```
#checking the datatypes and missing values

data.info()
data.isnull().sum()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Gender                  200 non-null    int64
 1   Age                     200 non-null    int64
 2   Annual Income (k$)      200 non-null    int64
 3   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4)
memory usage: 6.4 KB
```

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

dtype: int64

```
#convert gender into numbers
data['Gender'] = data['Gender'].map({'Male': 0, 'Female': 1})
```

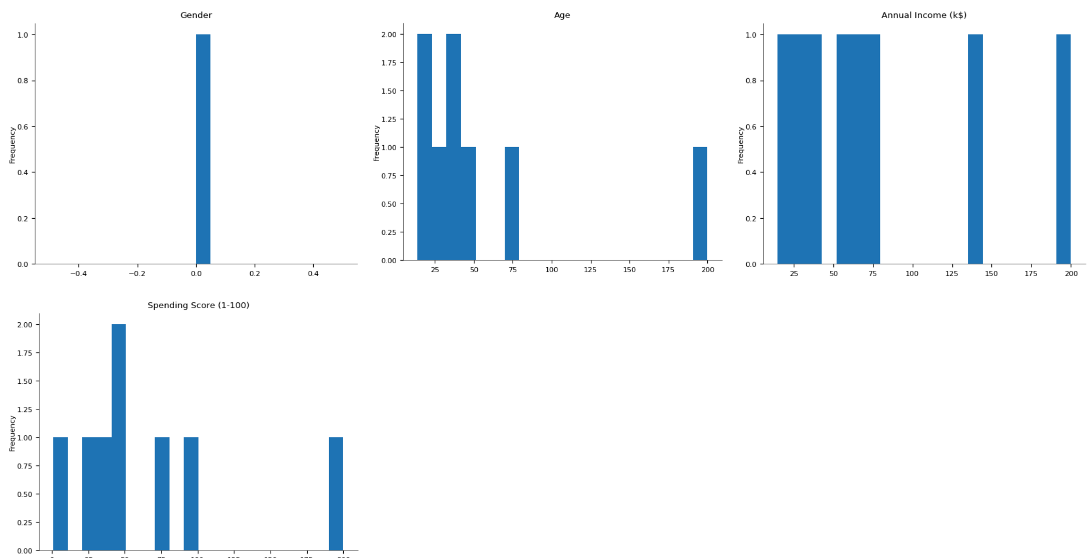
```
#data exploration and descriptive stats
data.describe()
```

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

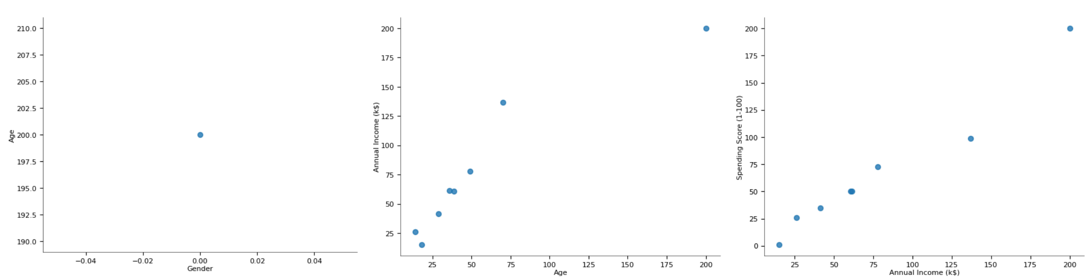
```
#decribing the data
data.describe()
```

	Gender	Age	Annual Income (k$)	Spending Score (1-100)
count	0.0	200.000000	200.000000	200.000000
mean	NaN	38.850000	60.560000	50.200000
std	NaN	13.969007	26.264721	25.823522
min	NaN	18.000000	15.000000	1.000000
25%	NaN	28.750000	41.500000	34.750000
50%	NaN	36.000000	61.500000	50.000000
75%	NaN	49.000000	78.000000	73.000000
max	NaN	70.000000	137.000000	99.000000

Distributions



2-d distributions

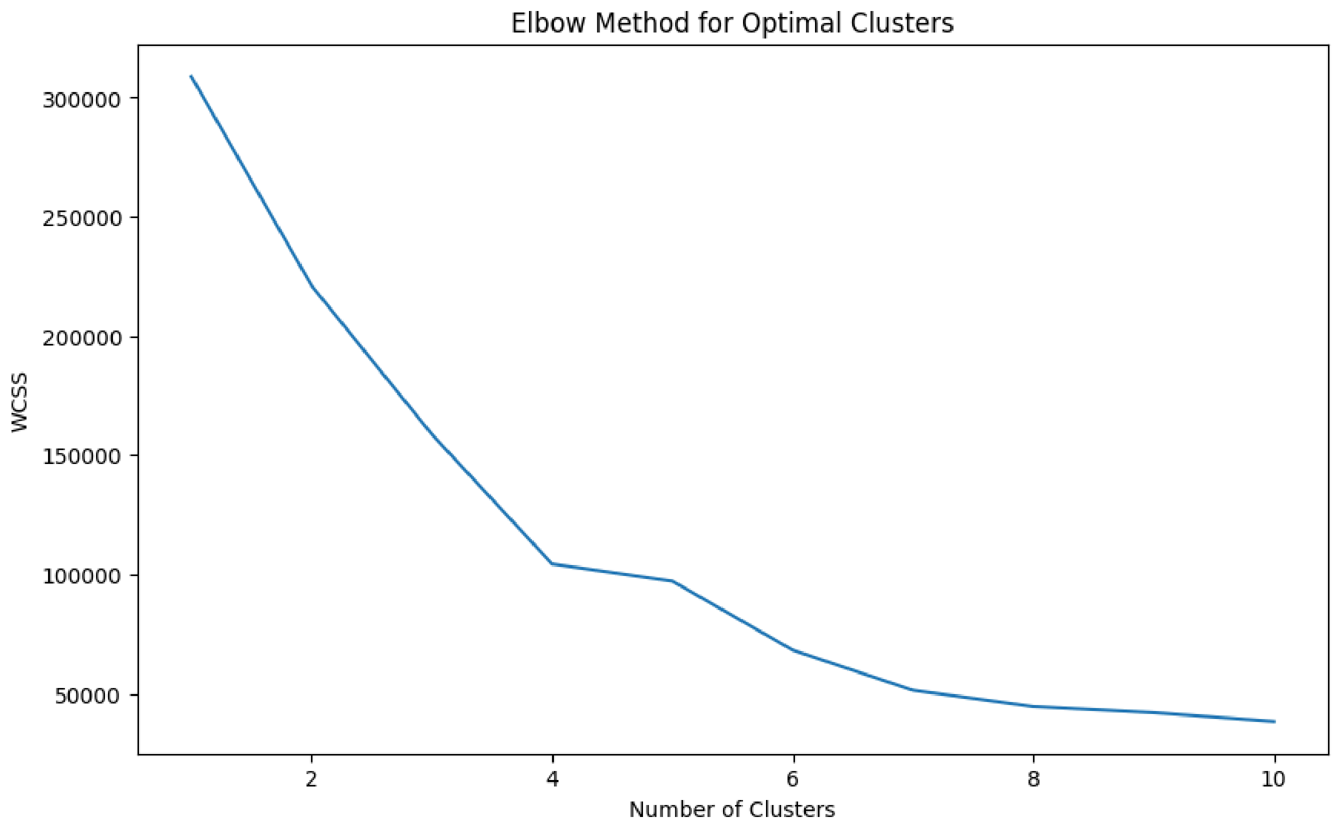



```
#elbow method to decide cluster count
X = data[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']]
```

```
#running the elboow method loop
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(X)
```

```
    wcss.append(kmeans.inertia_)
```

```
#plot the elbow loop
plt.figure(figsize=(10,6))
plt.plot(range(1,11), wcss)
plt.title("Elbow Method for Optimal Clusters")
plt.xlabel("Number of Clusters")
plt.ylabel("WCSS")
plt.show()
```




```
#apply the kmeans algorithm
kmeans = KMeans(n_clusters=5, init='k-means++', random_state=42)
data['Cluster'] = kmeans.fit_predict(X)
```

```
data.head()
```

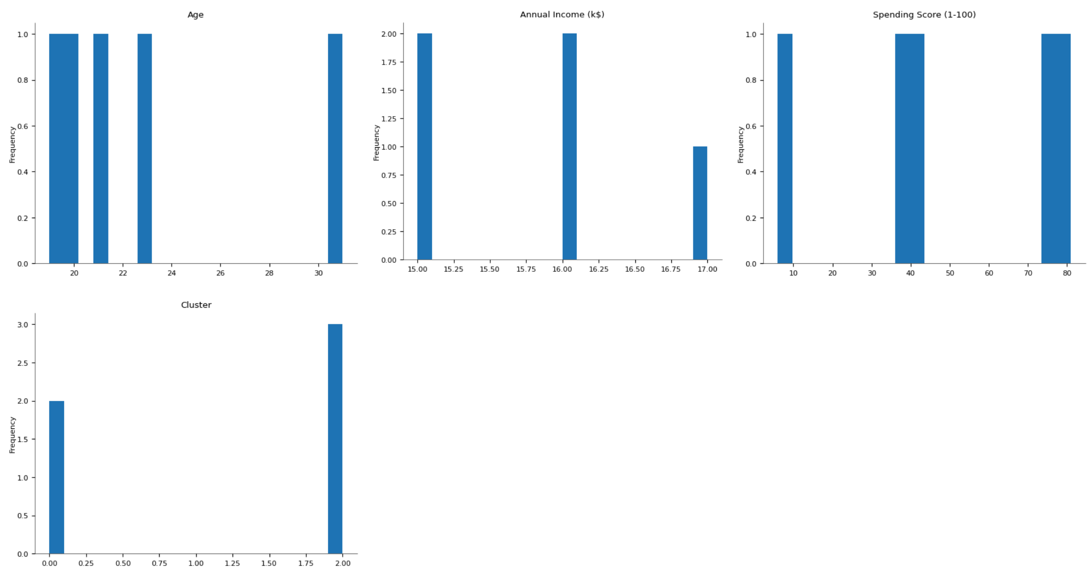
1 to 5 of 5 entries Filter

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

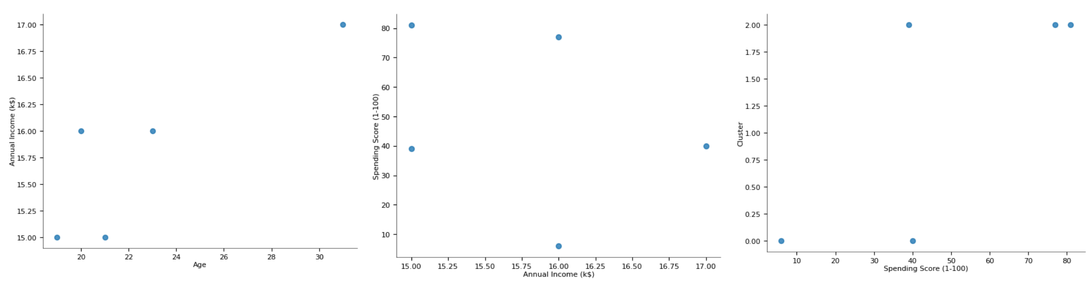
Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

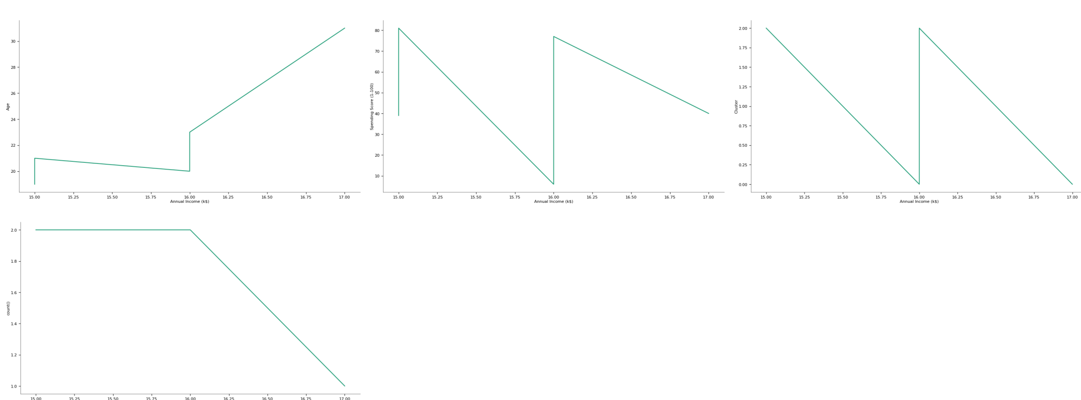
Distributions



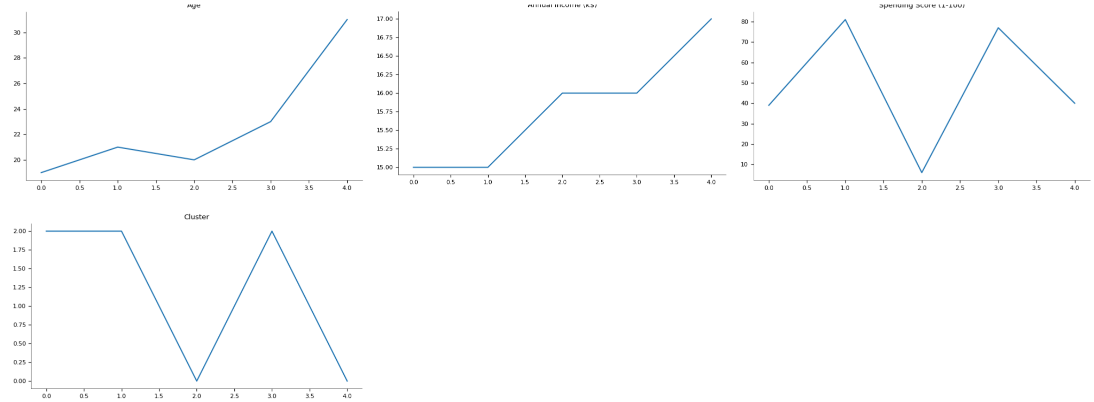
2-d distributions



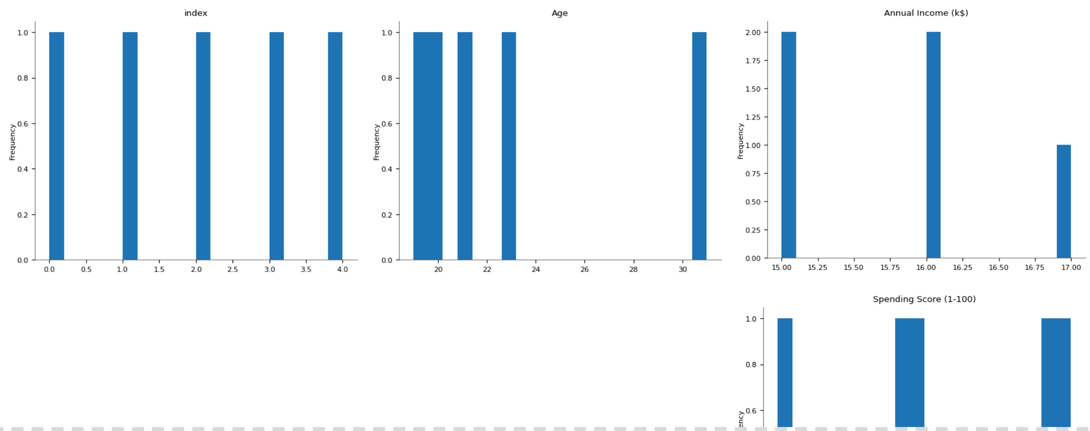
Time series



Values



Distributions



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

Age

Annual Income (k$)

Spending Score (1-100)

Cluster

Age vs Annual Income (k$)

Annual Income (k$) vs Spending Score (1-100)

Spending Score (1-100) vs Cluster

Annual Income (k$) vs Age

Annual Income (k$) vs Spending Score (1-100)

Annual Income (k$) vs Cluster

Annual Income (k$) vs count()

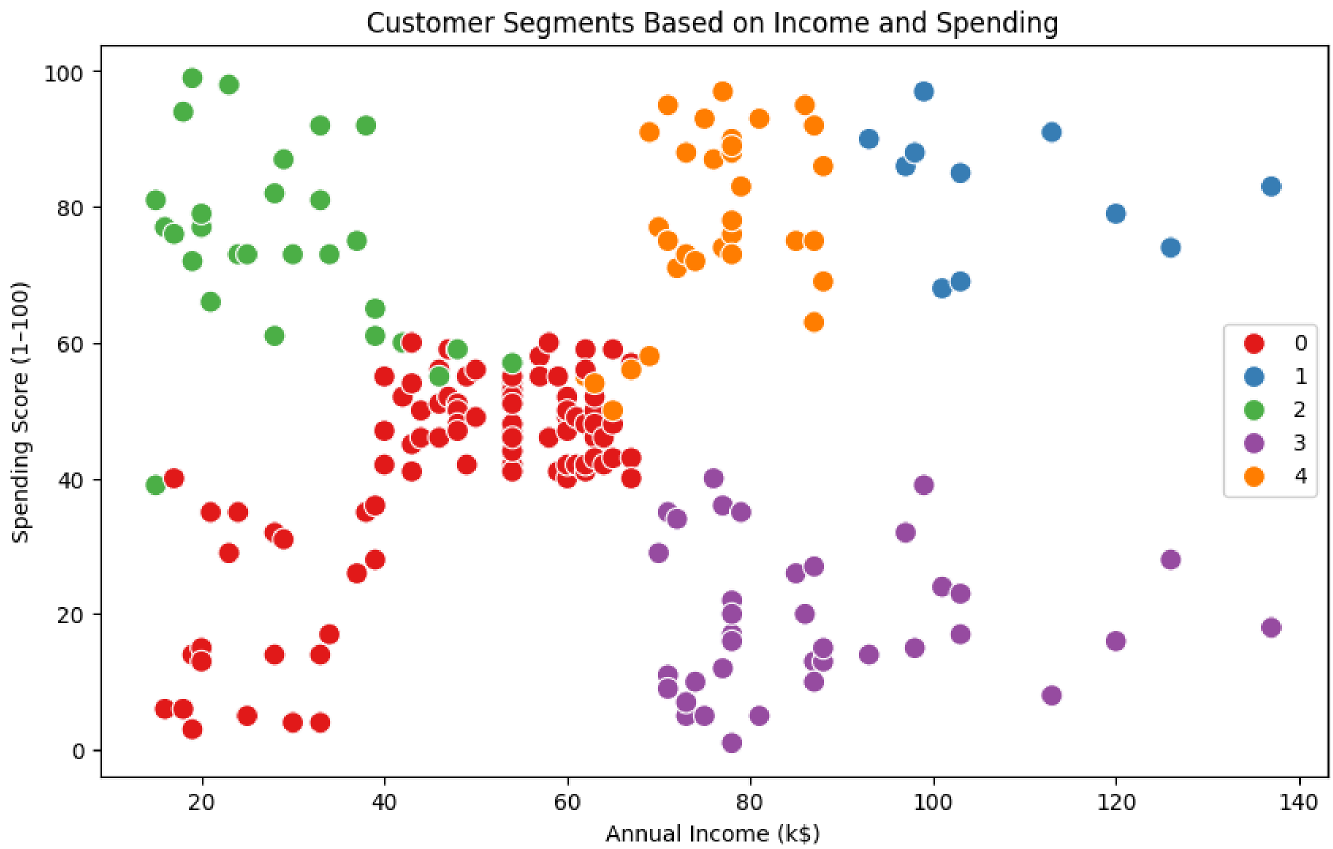
Age

Annual Income (k$)

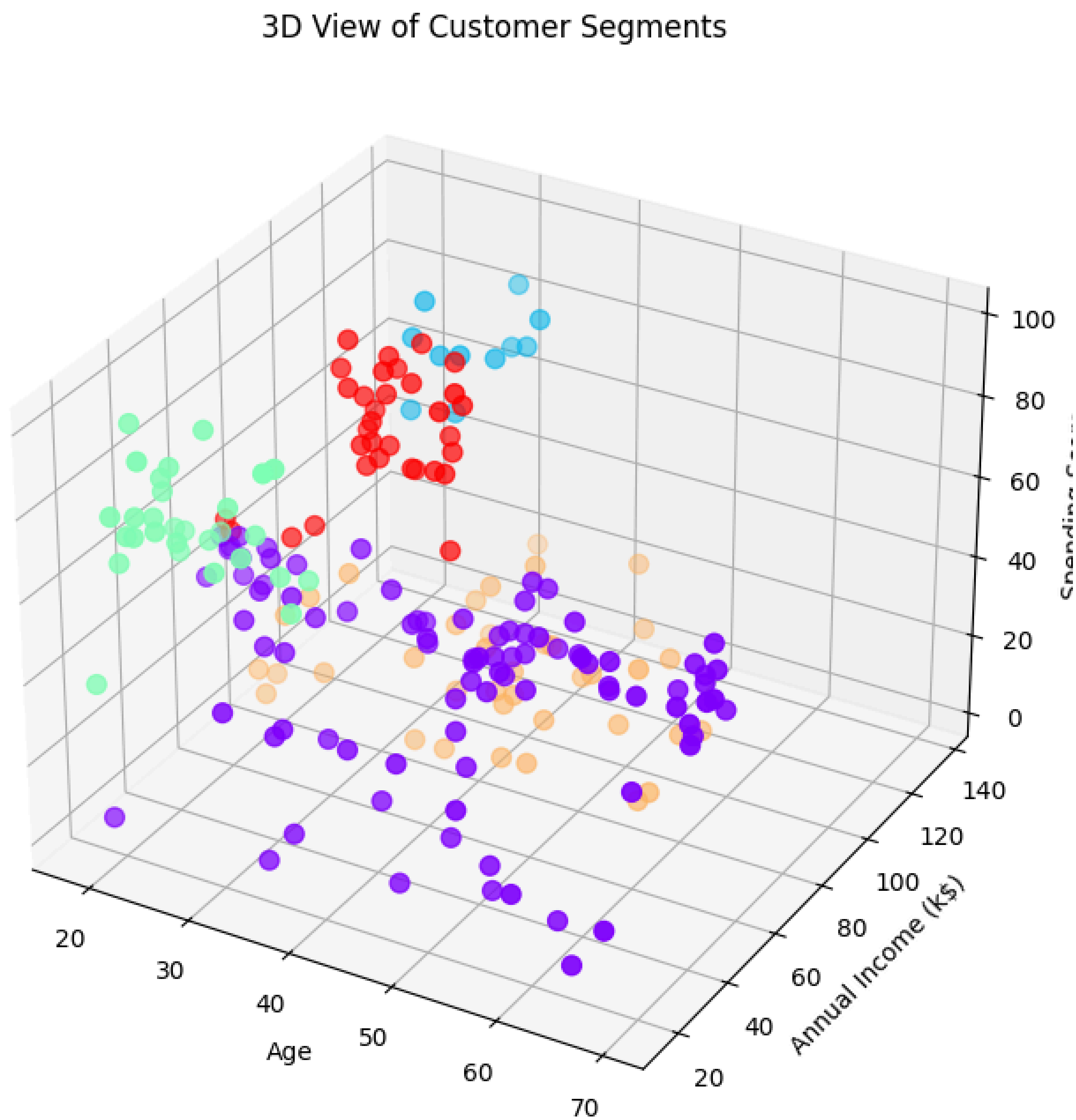
Spending Score (1-100)

Cluster

```
#visualise the clusters(scatter plot)
plt.figure(figsize=(10,6))
sns.scatterplot(x='Annual Income (k$)', y='Spending Score (1-100)',
                hue='Cluster', data=data, palette='Set1', s=100)
plt.title("Customer Segments Based on Income and Spending")
plt.xlabel("Annual Income (k$)")
plt.ylabel("Spending Score (1–100)")
plt.legend()
plt.show()
```




```
#3d visualisation
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(10,8))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(data['Age'], data['Annual Income (k$)'], data['Spending Score (1-100)'],
           c=data['Cluster'], cmap='rainbow', s=60)
ax.set_xlabel('Age')
ax.set_ylabel('Annual Income (k$)')
ax.set_zlabel('Spending Score')
plt.title("3D View of Customer Segments")
plt.show()
```




```
#view cluster averages
data.groupby('Cluster').mean(numeric_only=True)
```

	Gender	Age	Annual Income (k$)	Spending Score (1-100)
3	NaN	40.394737	87.000000	18.631579
4	NaN	31.787879	76.090909	77.757576

```
#cluster summary

cluster_summary = data.groupby('Cluster').mean(numeric_only=True)
print("📊 Average values per cluster:")
print(cluster_summary)
```

```
📊 Average values per cluster:
         Gender        Age  Annual Income (k$)  Spending Score (1-100)
Cluster
0           NaN  46.213483           47.719101               41.797753
1           NaN  32.454545          108.181818               82.727273
2           NaN  24.689655           29.586207               73.655172
3           NaN  40.394737           87.000000               18.631579
4           NaN  31.787879           76.090909               77.757576
```

```
print("\n📌 Insights per Cluster:\n")

for i in range(cluster_summary.shape[0]):
    print(f"🔷 Cluster {i}:")
    income = cluster_summary.iloc[i]['Annual Income (k$)']
    score = cluster_summary.iloc[i]['Spending Score (1-100)']

    if income >= 70 and score >= 60:
        print("💰 High income, high spending – Target with luxury or premium offers.")
    elif income >= 70 and score < 40:
        print("📉 High income, low spending – Upsell premium or loyalty programs.")
    elif income < 40 and score >= 60:
        print("😳 Low income, high spending – Possibly impulsive buyers.")
    elif income < 40 and score < 40:
        print("🧩 Low income, low spending – Budget-sensitive customers.")
    else:
        print("📊 Mid-range group – Explore flexible strategies like discounts or combo dea

    print(f"📍 Avg Age: {cluster_summary.iloc[i]['Age']:.1f} | Avg Income: {income:.1f}k | /
    print("-" * 60)
```

```
📌 Insights per Cluster:
```