```
import pandas as pd
import seaborn as sns
```

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

```
dt.info()
```

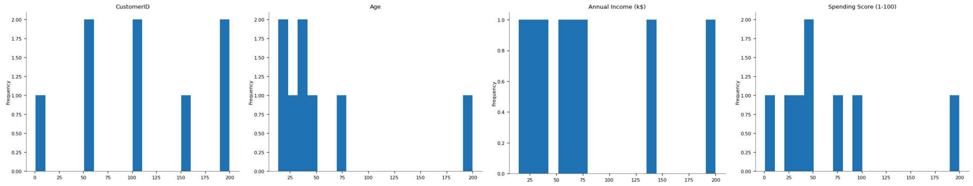
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   CustomerID              200 non-null    int64 
 1   Gender                  200 non-null    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
```

```
x = dt["Annual Income (k$)"]
y = dt["Spending Score (1-100)"]
```

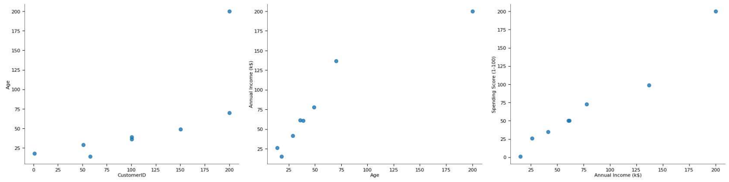
```
dt.describe()
```

	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

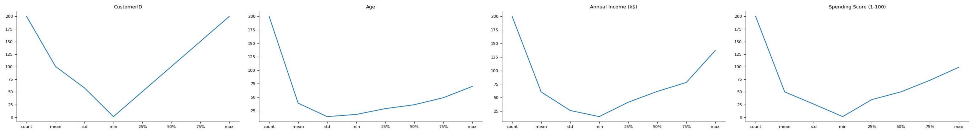
Distributions



2-d distributions



Values



```
dt.dtypes
```

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

dtype: object
```

```
dt['Gender'].value_counts()
```

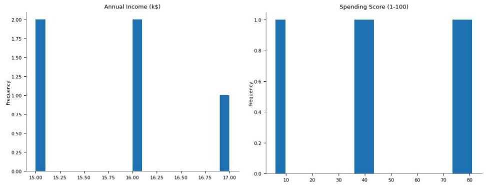
```
          count
Gender
Female      112
  Male       88

dtype: int64
```

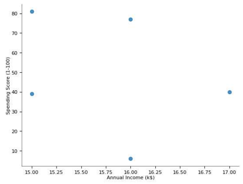
```
x = dt.iloc[:,3:]
x.head()
```

```
   Annual Income (k$)  Spending Score (1-100)
0                  15                      39
1                  15                      81
2                  16                       6
3                  16                      77
4                  17                      40
```

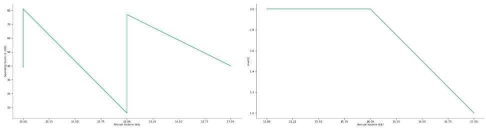
Distributions



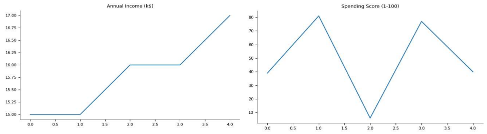
2-d distributions



Time series

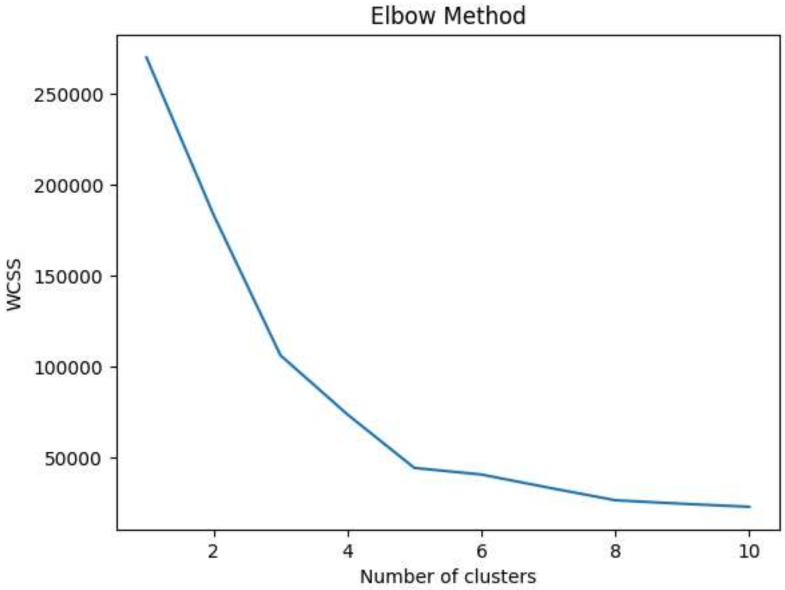


Values



```
from sklearn.cluster import KMeans
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_)
```

```
import matplotlib.pyplot as plt
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



```
kmeans = kmeans = KMeans(n_clusters=5, init='k-means++', random_state = 42)
y_kmeans = kmeans.fit(x)
```

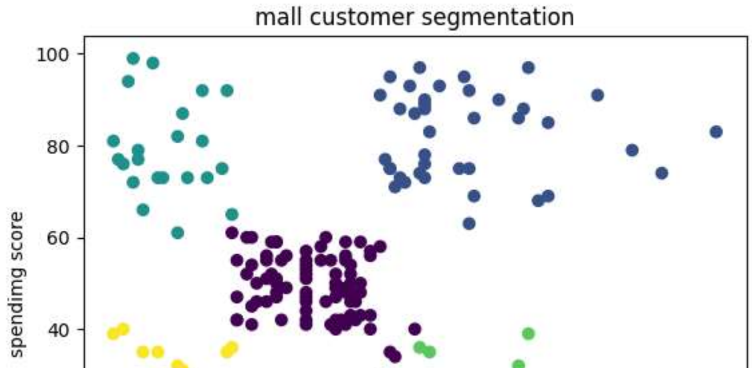
```
label = kmeans.labels_
dt['cluster'] = label
```

```
plt.scatter(dt['Annual Income (k$)'], dt['Spending Score (1-100)'], c=dt['cluster'])
plt.title('mall customer segmentation')
plt.xlabel('age')
plt.ylabel('spendimg score')
plt.show
```

```
matplotlib.pyplot.show
def show(*args, **kwargs) -> None
```

```
Display all open figures.

Parameters
----------
block : bool, optional
    Whether to wait for all figures to be closed before returning.
```


mall customer segmentation
spendimg score
100
80
60
40