```
import pandas as pd
import numpy as np
df = pd.read_csv("/content/Mall_Customers.csv")
```

```
df
```

	CustomerID	Genre	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
...	...	...	...	...	...
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

200 rows × 5 columns

```
df.mean()
```

```
<ipython-input-33-c61f0c8f89b5>:1: FutureWarning: The default value of numeric_only in DataFrame.mean is deprecated. In a futur
  df.mean()
CustomerID                100.50
Age                        38.85
Annual Income (k$)         60.56
Spending Score (1-100)     50.20
dtype: float64
```

```
df.loc[:,'Age'].mean()
```

```
38.85
```

```
df.mean(axis = 1)[0:4]
```

```
<ipython-input-5-5a17804ef008>:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None'
  df.mean(axis = 1)[0:4]
0    18.50
1    29.75
2    11.25
3    30.00
dtype: float64
```

```
df.median()
```

```
<ipython-input-6-6d467abf240d>:1: FutureWarning: The default value of numeric_only in DataFrame.median is deprecated. In a futu
  df.median()
CustomerID                100.5
Age                        36.0
Annual Income (k$)         61.5
Spending Score (1-100)     50.0
dtype: float64
```

```
df.loc[:,'Age'].median()
```

```
36.0
```

```
df.median(axis = 1)[0:4]
```

```
<ipython-input-8-b75ab1b1d07a>:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None'
  df.median(axis = 1)[0:4]
0    17.0
1    18.0
2    11.0
3    19.5
dtype: float64
```

```
df.mode()
```

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Female	32.0	54.0	42.0
1	2	NaN	NaN	78.0	NaN
2	3	NaN	NaN	NaN	NaN
3	4	NaN	NaN	NaN	NaN
4	5	NaN	NaN	NaN	NaN
...	...	...	...	...	...
195	196	NaN	NaN	NaN	NaN
196	197	NaN	NaN	NaN	NaN
197	198	NaN	NaN	NaN	NaN
198	199	NaN	NaN	NaN	NaN
199	200	NaN	NaN	NaN	NaN

200 rows × 5 columns

```
df.loc[:,'Age'].mode()
```

```
0    32
Name: Age, dtype: int64
```

```
df.min()
```

```
CustomerID                    1
Genre                    Female
Age                          18
Annual Income (k$)           15
Spending Score (1-100)        1
dtype: object
```

```
df.loc[:,'Age'].min(skipna = False)
```

```
18
```

```
df.max()
```

```
CustomerID                200
Genre                    Male
Age                        70
Annual Income (k$)        137
Spending Score (1-100)     99
dtype: object
```

```
df.loc[:,'Age'].max(skipna = False)
```

```
70
```

```
df.std()
```

```
<ipython-input-15-ce97bb7eaef8>:1: FutureWarning: The default value of numeric_only in DataFrame.std is deprecated. In a future
  df.std()
CustomerID                57.879185
Age                       13.969007
Annual Income (k$)        26.264721
Spending Score (1-100)    25.823522
dtype: float64
```

```
df.loc[:,'Age'].std()
```

```
13.96900733155888
```

```
df.std(axis=1)[0:4]
```

```
<ipython-input-17-18b37211c5c6>:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None
  df.std(axis=1)[0:4]
0    15.695010
1    35.074920
2     8.057088
3    32.300671
dtype: float64
```

```
df.groupby(['Genre'])['Age'].mean()
```

```
Genre
Female    38.098214
Male      39.806818
Name: Age, dtype: float64
```

```
df_u=df.rename(columns= {'Annual Income (k$)':'Income'},inplace=False)
df_u
```

	CustomerID	Genre	Age	Income	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
...	...	...	...	...	...
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

200 rows × 5 columns

```
df_u.groupby(['Genre'])['Income'].mean()
```

```
Genre
Female    59.250000
Male      62.227273
Name: Income, dtype: float64
```

```
from sklearn import preprocessing
enc = preprocessing.OneHotEncoder()
enc_df = pd.DataFrame(enc.fit_transform(df[['Genre']]).toarray())
enc_df
```

	0	1
0	0.0	1.0
1	0.0	1.0
2	1.0	0.0
3	1.0	0.0
4	1.0	0.0
...	...	...
195	1.0	0.0
196	1.0	0.0
197	0.0	1.0
198	0.0	1.0
199	0.0	1.0

200 rows × 2 columns

```
df_encode =df_u.join(enc_df)
df_encode
```

	CustomerID	Genre	Age	Income	Spending Score (1-100)	0	1
0	1	Male	19	15	39	0.0	1.0
1	2	Male	21	15	81	0.0	1.0
2	3	Female	20	16	6	1.0	0.0
3	4	Female	23	16	77	1.0	0.0
4	5	Female	31	17	40	1.0	0.0
...	...	...	...	...	...	...	...
198	199	Male	32	137	18	0.0	1.0

```
iris = pd.read_csv("/content/IRIS.csv")
```

```
col_names = ['Sepal_Length','Sepal_Width','Petal_Length','Petal_Width','Species']
```

```
iris
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

```
irisSet = (iris['species'] == 'Iris-setosa')
```

```
print('Iris-setosa')
print(iris[irisSet].describe())
```

```
Iris-setosa
       sepal_length  sepal_width  petal_length  petal_width
count      50.00000    50.000000     50.000000     50.00000
mean        5.00600     3.418000      1.464000      0.24400
std         0.35249     0.381024      0.173511      0.10721
min         4.30000     2.300000      1.000000      0.10000
25%         4.80000     3.125000      1.400000      0.20000
50%         5.00000     3.400000      1.500000      0.20000
75%         5.20000     3.675000      1.575000      0.30000
max         5.80000     4.400000      1.900000      0.60000
```

```
irisVer = (iris['species'] == 'Iris-versicolor')
print('Iris-versicolor')
print(iris[irisVer].describe())
```

```
Iris-versicolor
       sepal_length  sepal_width  petal_length  petal_width
count     50.000000    50.000000     50.000000    50.000000
mean       5.936000     2.770000      4.260000     1.326000
std        0.516171     0.313798      0.469911     0.197753
min        4.900000     2.000000      3.000000     1.000000
25%        5.600000     2.525000      4.000000     1.200000
50%        5.900000     2.800000      4.350000     1.300000
75%        6.300000     3.000000      4.600000     1.500000
max        7.000000     3.400000      5.100000     1.800000
```

```
irisVir = (iris['species'] == 'Iris-virginica')
print('Iris-virginica')
print(iris[irisVir].describe())
```

```
Iris-virginica
       sepal_length  sepal_width  petal_length  petal_width
count      50.00000    50.000000     50.000000     50.00000
mean        6.58800     2.974000      5.552000      2.02600
```

std	0.63588	0.322497	0.551895	0.27465
min	4.90000	2.200000	4.500000	1.40000
25%	6.22500	2.800000	5.100000	1.80000
50%	6.50000	3.000000	5.550000	2.00000
75%	6.90000	3.175000	5.875000	2.30000
max	7.90000	3.800000	6.900000	2.50000

```
iris[irisVir]
```

	sepal_length	sepal_width	petal_length	petal_width	species
100	6.3	3.3	6.0	2.5	Iris-virginica
101	5.8	2.7	5.1	1.9	Iris-virginica
102	7.1	3.0	5.9	2.1	Iris-virginica
103	6.3	2.9	5.6	1.8	Iris-virginica
104	6.5	3.0	5.8	2.2	Iris-virginica
105	7.6	3.0	6.6	2.1	Iris-virginica
106	4.9	2.5	4.5	1.7	Iris-virginica
107	7.3	2.9	6.3	1.8	Iris-virginica
108	6.7	2.5	5.8	1.8	Iris-virginica
109	7.2	3.6	6.1	2.5	Iris-virginica
110	6.5	3.2	5.1	2.0	Iris-virginica
111	6.4	2.7	5.3	1.9	Iris-virginica
112	6.8	3.0	5.5	2.1	Iris-virginica
113	5.7	2.5	5.0	2.0	Iris-virginica
114	5.8	2.8	5.1	2.4	Iris-virginica
115	6.4	3.2	5.3	2.3	Iris-virginica
116	6.5	3.0	5.5	1.8	Iris-virginica
117	7.7	3.8	6.7	2.2	Iris-virginica

✓ 0s completed at 12:34 AM