Assignment 5:

Take all the columns in mall_customers.csv

gender age annual income spending score

perform label encoding on gender

train your data

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

```
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

```
print(df.head())
print(df.tail())
```

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

```
print(df.shape)
```

```
(200, 5)
```

```
print(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   Genre                   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
None
```

```
df.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

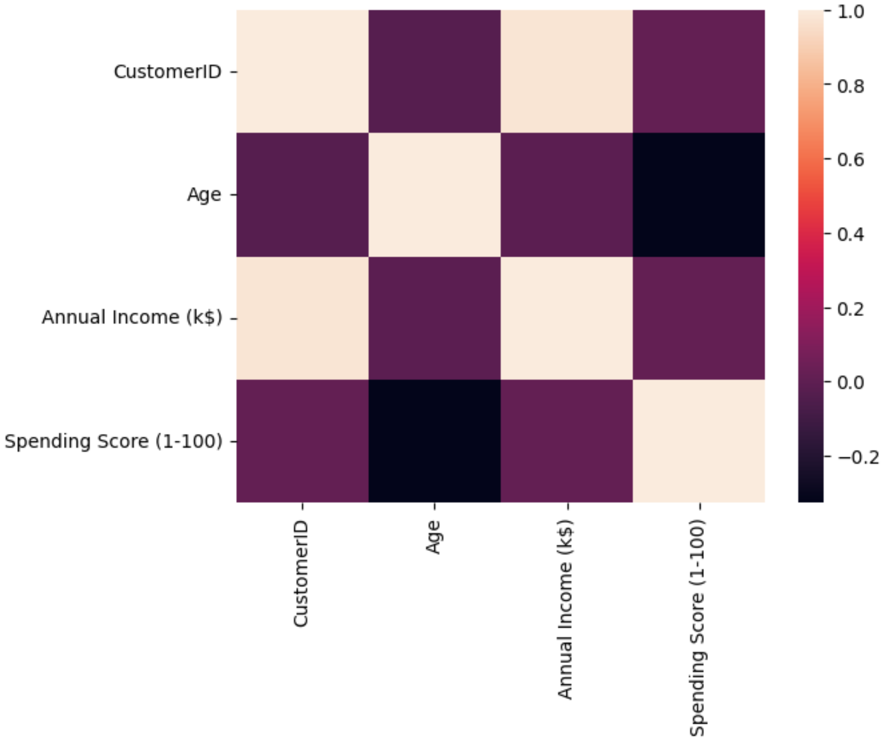
```
df.corr()
```

```
<ipython-input-9-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future vers
  df.corr()
```

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
CustomerID	1.000000	-0.026763	0.977548	0.013835
Age	-0.026763	1.000000	-0.012398	-0.327227
Annual Income (k$)	0.977548	-0.012398	1.000000	0.009903
Spending Score (1-100)	0.013835	-0.327227	0.009903	1.000000

```
sns.heatmap(df.corr())
```

```
<ipython-input-10-aa4f4450a243>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future ver
  sns.heatmap(df.corr())
<Axes: >
```



```
df.isnull().any()
```

```
CustomerID                False
Genre                     False
Age                       False
Annual Income (k$)        False
Spending Score (1-100)    False
dtype: bool
```

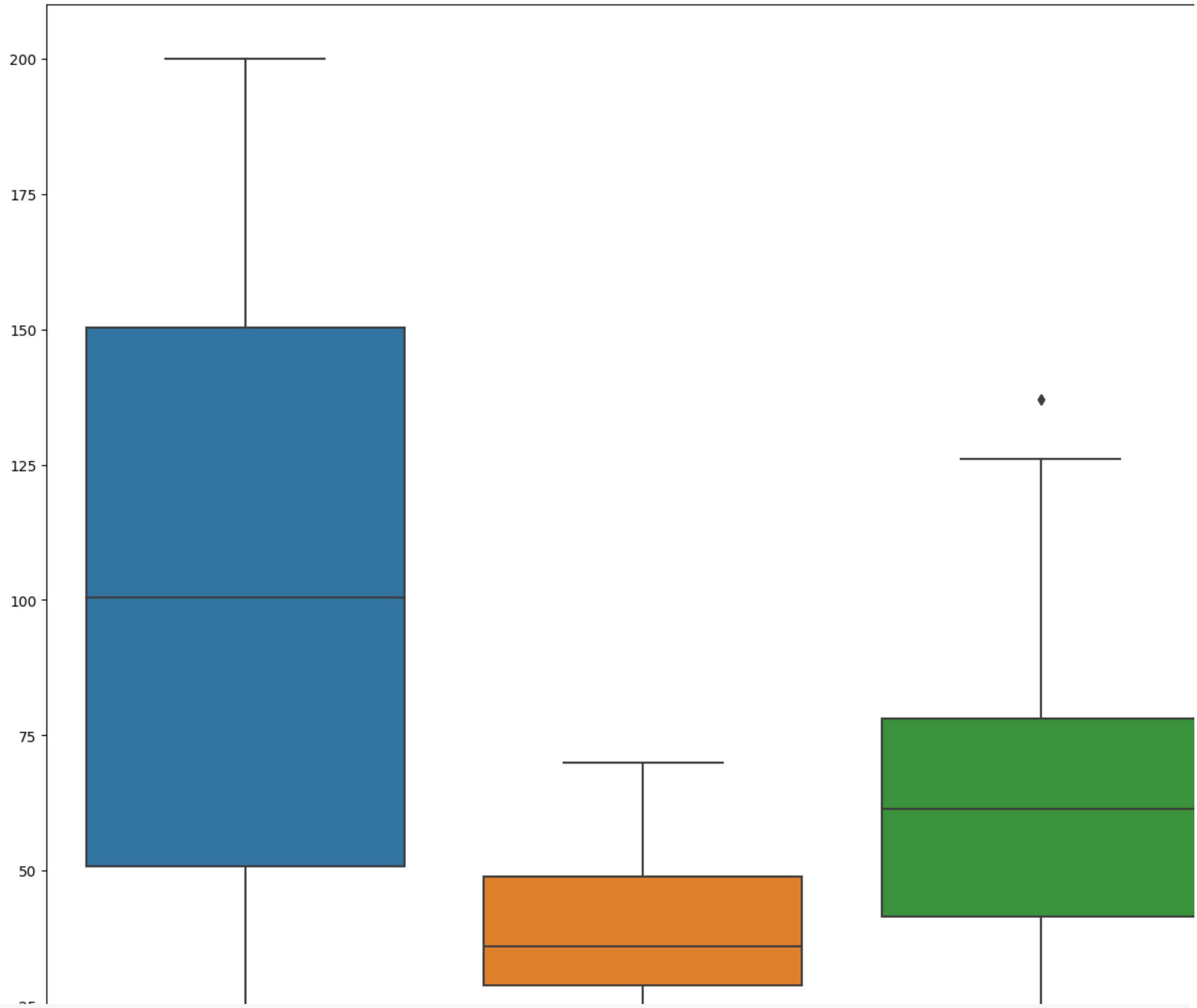
```
df.isnull().sum()
```

```
CustomerID                0
Genre                     0
Age                       0
Annual Income (k$)        0
```

```
Spending Score (1-100)    0
dtype: int64
```

```
plt.subplots(figsize=(20,15))
sns.boxplot(df)
```

```
<Axes: >
```



```
x=df.iloc[:,:3]
y=df.iloc[:,3:4]
```

```
x.head()
```

	CustomerID	Genre	Age
0	1	Male	19
1	2	Male	21
2	3	Female	20
3	4	Female	23
4	5	Female	31

```
y.head()
```

	Annual Income (k$)
0	15
1	15
2	16
3	16
4	17

```
print(x.shape)
print(y.shape)
```

```
(200, 3)
(200, 1)
```

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
x['Genre']=le.fit_transform(x['Genre'])
```

```
x[['Genre']]
```

	Genre
0	1
1	1
2	0
3	0
4	0
...	...
195	0
196	0
197	1
198	1
199	1

200 rows × 1 columns

```
x.head()
```

	CustomerID	Genre	Age
0	1	1	19
1	2	1	21
2	3	0	20
3	4	0	23
4	5	0	31

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
```

```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(140, 3)
(60, 3)
(140, 1)
(60, 1)
```

```
from sklearn.preprocessing import MinMaxScaler
mms=MinMaxScaler()
x_train_scaled=mms.fit_transform(x_train)
x_test_scaled=mms.fit_transform(x_test)
```

```
x_train_scaled
```

```
       [0.19095477, 0.        , 0.34615385],
       [0.66834171, 0.        , 0.25      ],
       [0.26633166, 1.        , 0.78846154],
       [0.78894472, 0.        , 0.23076923],
       [0.64321608, 1.        , 0.78846154],
       [0.17085427, 0.        , 0.59615385],
       [0.14070352, 0.        , 0.42307692],
       [0.57286432, 0.        , 0.        ],
       [0.75879397, 1.        , 0.40384615],
       [0.15577889, 0.        , 0.05769231],
       [0.83417085, 1.        , 0.46153846],
       [0.63819095, 1.        , 0.42307692],
       [0.88442211, 1.        , 0.76923077],
       [0.16080402, 1.        , 0.67307692],
       [0.71356784, 0.        , 0.19230769],
       [0.84924623, 1.        , 0.26923077],
       [0.73869347, 0.        , 0.26923077],
       [0.14572864, 0.        , 0.09615385],
       [0.49748744, 1.        , 0.03846154],
       [0.4120603 , 1.        , 0.94230769],
       [0.39698492, 0.        , 0.59615385],
       [0.57788945, 0.        , 0.01923077],
       [0.74371859, 0.        , 0.30769231],
       [0.96984925, 0.        , 0.38461538],
       [0.36180905, 0.        , 0.80769231],
       [0.38693467, 1.        , 0.42307692],
       [0.12562814, 1.        , 0.21153846],
       [0.82914573, 0.        , 0.34615385],
       [0.40703518, 1.        , 0.38461538],
       [0.94472362, 0.        , 0.44230769],
       [0.87437186, 0.        , 0.65384615],
       [0.95477387, 0.        , 0.30769231],
       [0.1959799 , 0.        , 0.03846154],
       [0.29145729, 0.        , 0.17307692],
       [0.70351759, 0.        , 0.75      ],
       [0.44221106, 0.        , 0.30769231],
       [0.35175879, 1.        , 1.        ],
       [0.43718593, 0.        , 0.07692308],
       [0.18090452, 0.        , 0.46153846],
       [0.10552764, 1.        , 0.13461538],
       [0.04522613, 0.        , 0.23076923],
       [0.51758794, 1.        , 0.15384615],
       [0.33668342, 0.        , 0.96153846],
       [0.96482412, 1.        , 0.28846154],
       [0.5879397 , 0.        , 0.59615385],
       [0.2361809 , 0.        , 0.17307692],
       [0.86432161, 1.        , 0.34615385]])
```

```
print(x_test_scaled)
```

```
 [0.55675676 1.         0.69230769]
 [0.50810811 1.         0.57692308]
 [0.93513514 1.         0.17307692]
 [0.96216216 1.         0.53846154]
 [0.00540541 0.         0.07692308]
 [0.76756757 1.         0.57692308]
 [0.04324324 0.         0.76923077]
 [0.8        0.         0.5       ]
 [0.30810811 1.         0.01923077]
 [0.65405405 0.         0.25      ]
 [0.95135135 0.         0.36538462]
 [0.81081081 0.         0.55769231]
 [0.41081081 1.         0.75      ]
 [0.01621622 0.         0.09615385]
 [0.15675676 1.         0.        ]
 [0.68108108 1.         0.55769231]
 [0.17837838 0.         0.23076923]
 [0.37837838 1.         0.78846154]
 [0.96756757 0.         0.21153846]
 [0.76216216 1.         0.19230769]
 [0.22162162 0.         0.11538462]
 [0.83783784 0.         0.23076923]
 [0.3027027  1.         1.        ]
 [0.64324324 1.         0.40384615]
 [0.94594595 1.         0.32692308]
 [0.97837838 1.         0.23076923]
 [0.63783784 0.         0.42307692]
 [0.21621622 0.         0.59615385]
 [0.06486486 0.         0.32692308]
 [0.27567568 1.         0.55769231]
 [0.78918919 1.         0.48076923]
 [0.57837838 0.         0.01923077]
 [0.0972973  0.         0.53846154]
 [1.         0.         0.34615385]
 [0.67567568 1.         0.38461538]
 [0.         0.         0.25      ]
 [0.42702703 0.         0.53846154]
 [0.55135135 0.         0.92307692]
 [0.11891892 0.         0.51923077]
 [0.58918919 1.         0.01923077]
 [0.88648649 0.         0.34615385]
 [0.31891892 0.         0.69230769]
 [0.02162162 1.         0.88461538]
 [0.38378378 1.         0.15384615]
 [0.61621622 0.         0.63461538]
 [0.75135135 0.         0.26923077]
 [0.36216216 0.         0.55769231]
 [0.64864865 0.         0.09615385]
 [0.97297297 0.         0.44230769]
 [0.5027027  0.         0.17307692]
 [0.78378378 1.         0.30769231]
 [0.10810811 0.         0.69230769]
 [0.14054054 1.         0.80769231]
 [0.84324324 0.         0.73076923]
 [0.19459459 0.         0.90384615]
 [0.28108108 0.         0.63461538]]
```