Importing the libraries

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

Importing the dataset

```
df =
pd.read_csv('https://github.com/YBI-Foundation/Dataset/raw/main/Boston
.csv')
```

Get Information of Dataframe

```
df.head()
```

```
      CRIM    ZN  INDUS  CHAS     NX     RM   AGE     DIS  RAD    TAX
\
0  0.00632  18.0   2.31     0  0.538  6.575  65.2  4.0900    1  296.0

1  0.02731   0.0   7.07     0  0.469  6.421  78.9  4.9671    2  242.0

2  0.02729   0.0   7.07     0  0.469  7.185  61.1  4.9671    2  242.0

3  0.03237   0.0   2.18     0  0.458  6.998  45.8  6.0622    3  222.0

4  0.06905   0.0   2.18     0  0.458  7.147  54.2  6.0622    3  222.0


   PTRATIO       B  LSTAT  MEDV
0     15.3  396.90   4.98  24.0
1     17.8  396.90   9.14  21.6
2     17.8  392.83   4.03  34.7
3     18.7  394.63   2.94  33.4
4     18.7  396.90   5.33  36.2
```

```
df.shape
```

```
(506, 14)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     506 non-null    float64
 1   ZN       506 non-null    float64
 2   INDUS    506 non-null    float64
 3   CHAS     506 non-null    int64  
 4   NX       506 non-null    float64
 5   RM       506 non-null    float64
 6   AGE      506 non-null    float64
 7   DIS      506 non-null    float64
 8   RAD      506 non-null    int64  
 9   TAX      506 non-null    float64
 10  PTRATIO  506 non-null    float64
 11  B        506 non-null    float64
 12  LSTAT    506 non-null    float64
 13  MEDV     506 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 55.5 KB
```

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

```
CRIM       0
ZN         0
INDUS      0
CHAS       0
NX         0
RM         0
AGE        0
DIS        0
RAD        0
TAX        0
PTRATIO    0
B          0
LSTAT      0
MEDV       0
dtype: int64
```

```
df.describe()
```

```
             CRIM          ZN       INDUS        CHAS          NX  
RM  \
count  506.000000  506.000000  506.000000  506.000000  506.000000  
506.000000   
mean     3.613524   11.363636   11.136779    0.069170    0.554695  
6.284634   
std      8.601545   23.322453    6.860353    0.253994    0.115878  
0.702617   
min      0.006320    0.000000    0.460000    0.000000    0.385000  
3.561000   
25%      0.082045    0.000000    5.190000    0.000000    0.449000  
5.885500   
50%      0.256510    0.000000    9.690000    0.000000    0.538000  
6.208500   
75%      3.677083   12.500000   18.100000    0.000000    0.624000  
```

```
6.623500
max     88.976200  100.000000   27.740000    1.000000    0.871000
8.780000

              AGE         DIS         RAD         TAX     PTRATIO
B  \
count  506.000000  506.000000  506.000000  506.000000  506.000000
506.000000
mean    68.574901    3.795043    9.549407  408.237154   18.455534
356.674032
std     28.148861    2.105710    8.707259  168.537116    2.164946
91.294864
min      2.900000    1.129600    1.000000  187.000000   12.600000
0.320000
25%     45.025000    2.100175    4.000000  279.000000   17.400000
375.377500
50%     77.500000    3.207450    5.000000  330.000000   19.050000
391.440000
75%     94.075000    5.188425   24.000000  666.000000   20.200000
396.225000
max    100.000000   12.126500   24.000000  711.000000   22.000000
396.900000

            LSTAT        MEDV
count  506.000000  506.000000
mean    12.653063   22.532806
std      7.141062    9.197104
min      1.730000    5.000000
25%      6.950000   17.025000
50%     11.360000   21.200000
75%     16.955000   25.000000
max     37.970000   50.000000
```

```
df.columns
```

```
Index(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NX', 'RM', 'AGE', 'DIS', 'RAD',
'TAX',
       'PTRATIO', 'B', 'LSTAT', 'MEDV'],
      dtype='object')
```

Setting X and y

```
X= df[['CRIM', 'ZN', 'INDUS', 'CHAS', 'NX', 'RM', 'AGE', 'DIS', 'RAD',
'TAX',
       'PTRATIO', 'B', 'LSTAT']]
```

```
y=df['MEDV']
```

```
X.shape
```

```
(506, 13)
```

```
y.shape
```

```
(506,)
```

Splitting the dataset into the Training set and Test set

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.2, random_state = 1)
```

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((404, 13), (102, 13), (404,), (102,))
```

Feature Scaling

```
from sklearn.preprocessing import StandardScaler    #STANDARDIZATION
sc = StandardScaler()
X_train[:, 3:] = sc.fit_transform(X_train[:, 3:])  #as first 3 rows
are dummy variables
X_test[:, 3:] = sc.transform(X_test[:, 3:])
```

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
X_train_ss=ss.fit_transform(X_train)
X_test_ss=ss.fit_transform(X_test)
```

```
X_test_ss, X_train_ss
```

```
(array([[-0.541515  ,  0.95046258, -1.33166171, ..., -0.05711455,
          0.46657019, -0.67904682],
        [-0.54551801,  1.8887987 , -1.09985593, ..., -0.42995832,
          0.46657019, -0.73042217],
        [-0.5113626 , -0.45704162, -0.64638587, ..., -0.2901419 ,
          0.42049551,  0.97523937],
        ...,
        [-0.53169432, -0.45704162, -0.40733616, ...,  1.10802223,
          0.42880224,  0.12387647],
        [ 1.50236071, -0.45704162,  0.9748058 , ...,  0.78178393,
         -3.65633828,  0.51873271],
        [ 1.12021352, -0.45704162,  0.9748058 , ...,  0.78178393,
          0.35393089,  0.98257871]]),
 array([[-0.3892494 , -0.49559343, -0.60928978, ..., -0.24857777,
          0.28674182, -0.96685016],
        [-0.38783184,  0.57923879, -0.86952633, ...,  0.58214721,
          0.36669519, -0.82116789],
        [ 1.43554993, -0.49559343,  1.02669166, ...,  0.81290414,
          0.43472666,  2.50177533],
        ...,
        [ 0.23804008, -0.49559343,  1.02669166, ...,  0.81290414,
          0.43472666,  0.9145323 ],
        [-0.36856615, -0.49559343, -0.713092  , ..., -0.47933471,
          0.21433534, -0.26341291],
        [-0.39596611, -0.49559343, -0.74818007, ...,  0.35139027,
          0.43472666, -0.55616491]]))
```

mean of X_test_ss of rows

```
X_test_ss.mean(axis=0)
```

```
array([-2.02452434e-16,  5.55111512e-17, -1.63268092e-16, -
6.14976479e-17,
        3.26536184e-17,  1.27076998e-16, -8.27224999e-17, -
2.37282960e-16,
       -3.91843420e-17,  1.56737368e-16, -5.65996052e-16, -
1.07321559e-15,
       -5.11573354e-16])
```

Standard deviation of X_test_ss of rows

```
X_test_ss.std(axis=0)
```

```
array([1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.])
```