```
import numpy as np
import pandas as pd

data =
pd.read_csv("https://raw.githubusercontent.com/selva86/datasets/master
/BostonHousing.csv")
data.head()

      crim    zn  indus  chas    nox     rm   age     dis  rad  tax
ptratio  \
0  0.00632  18.0   2.31     0  0.538  6.575  65.2  4.0900    1  296
15.3
1  0.02731   0.0   7.07     0  0.469  6.421  78.9  4.9671    2  242
17.8
2  0.02729   0.0   7.07     0  0.469  7.185  61.1  4.9671    2  242
17.8
3  0.03237   0.0   2.18     0  0.458  6.998  45.8  6.0622    3  222
18.7
4  0.06905   0.0   2.18     0  0.458  7.147  54.2  6.0622    3  222
18.7

        b  lstat  medv
0  396.90   4.98  24.0
1  396.90   9.14  21.6
2  392.83   4.03  34.7
3  394.63   2.94  33.4
4  396.90   5.33  36.2

data.tail()

        crim   zn  indus  chas    nox     rm   age     dis  rad  tax
ptratio  \
501  0.06263  0.0  11.93     0  0.573  6.593  69.1  2.4786    1  273
21.0
502  0.04527  0.0  11.93     0  0.573  6.120  76.7  2.2875    1  273
21.0
503  0.06076  0.0  11.93     0  0.573  6.976  91.0  2.1675    1  273
21.0
504  0.10959  0.0  11.93     0  0.573  6.794  89.3  2.3889    1  273
21.0
505  0.04741  0.0  11.93     0  0.573  6.030  80.8  2.5050    1  273
21.0

          b  lstat  medv
501  391.99   9.67  22.4
502  396.90   9.08  20.6
503  396.90   5.64  23.9
504  393.45   6.48  22.0
505  396.90   7.88  11.9
```

```
print("The shape of the data is: ")
data.shape

The shape of the data is:

(506, 14)
```

Hence, we can see that there are no NULL values

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

crim       0
zn         0
indus      0
chas       0
nox        0
rm         0
age        0
dis        0
rad        0
tax        0
ptratio    0
b          0
lstat      0
medv       0
dtype: int64
```

Define the independent and dependent variables from the dataset

```
data.dropna()

        crim    zn  indus  chas    nox     rm   age     dis  rad  tax
\
0    0.00632  18.0   2.31     0  0.538  6.575  65.2  4.0900    1  296

1    0.02731   0.0   7.07     0  0.469  6.421  78.9  4.9671    2  242

2    0.02729   0.0   7.07     0  0.469  7.185  61.1  4.9671    2  242

3    0.03237   0.0   2.18     0  0.458  6.998  45.8  6.0622    3  222

4    0.06905   0.0   2.18     0  0.458  7.147  54.2  6.0622    3  222

..       ...   ...    ...   ...    ...    ...   ...     ...  ...  ...

501  0.06263   0.0  11.93     0  0.573  6.593  69.1  2.4786    1  273

502  0.04527   0.0  11.93     0  0.573  6.120  76.7  2.2875    1  273

503  0.06076   0.0  11.93     0  0.573  6.976  91.0  2.1675    1  273
```

```
504  0.10959   0.0  11.93     0  0.573  6.794  89.3  2.3889    1  273

505  0.04741   0.0  11.93     0  0.573  6.030  80.8  2.5050    1  273


     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
..       ...     ...    ...   ...
501     21.0  391.99   9.67  22.4
502     21.0  396.90   9.08  20.6
503     21.0  396.90   5.64  23.9
504     21.0  393.45   6.48  22.0
505     21.0  396.90   7.88  11.9

[506 rows x 14 columns]
```

```
X = data.iloc[:,0:13]
y = data.iloc[:,-1]
```

Splitting data into traing and testing dataset

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.20,random_state=42)
```

Shapes of the training and testing dataset

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

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

Importing LinearRegression() function

```
from sklearn.linear_model import LinearRegression

from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
model = make_pipeline(StandardScaler(with_mean=False),
LinearRegression())
model.fit(X_train, y_train)
```

```
Pipeline(steps=[('standardscaler', StandardScaler(with_mean=False)),
                ('linearregression', LinearRegression())])
```

```
model.score(X_test,y_test)
```

```
0.668759493535632
```