In [2]:
```
from sklearn.datasets import load_boston ## Import dataset Module
boston_dataset = load_boston() #Load dataset dictionary
```

In [15]:
```
print(boston_dataset['DESCR'])
```

```
.. _boston_dataset:

Boston house prices dataset
---------------------------

**Data Set Characteristics:**

    :Number of Instances: 506

    :Number of Attributes: 13 numeric/categorical predictive. Median Value (attribute 14) is usually the target.

    :Attribute Information (in order):
        - CRIM     per capita crime rate by town
        - ZN       proportion of residential land zoned for lots over 25,000 sq.ft.
        - INDUS    proportion of non-retail business acres per town
        - CHAS     Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
        - NOX      nitric oxides concentration (parts per 10 million)
        - RM       average number of rooms per dwelling
        - AGE      proportion of owner-occupied units built prior to 1940
        - DIS      weighted distances to five Boston employment centres
        - RAD      index of accessibility to radial highways
        - TAX      full-value property-tax rate per $10,000
        - PTRATIO  pupil-teacher ratio by town
        - B        1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
        - LSTAT    % lower status of the population
        - MEDV     Median value of owner-occupied homes in $1000's

    :Missing Attribute Values: None

    :Creator: Harrison, D. and Rubinfeld, D.L.

This is a copy of UCI ML housing dataset.
https://archive.ics.uci.edu/ml/machine-learning-databases/housing/ (https://archive.ics.uci.edu/ml/machine-learning-dat
abases/housing/)


This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University.

The Boston house-price data of Harrison, D. and Rubinfeld, D.L. 'Hedonic
```

```
prices and the demand for clean air', J. Environ. Economics & Management,
vol.5, 81-102, 1978.   Used in Belsley, Kuh & Welsch, 'Regression diagnostics
...', Wiley, 1980.   N.B. Various transformations are used in the table on
pages 244-261 of the latter.

The Boston house-price data has been used in many machine learning papers that address regression
problems.

.. topic:: References

   - Belsley, Kuh & Welsch, 'Regression diagnostics: Identifying Influential Data and Sources of Collinearity', Wiley,
 1980. 244-261.
   - Quinlan,R. (1993). Combining Instance-Based and Model-Based Learning. In Proceedings on the Tenth International Co
nference of Machine Learning, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.
```

In [3]:

```
import pandas as pd
boston = pd.DataFrame(boston_dataset.data, columns=boston_dataset.feature_names)# Converting the dictionary into a d
boston.head()##Printing out the head
```

Out[3]:

| | CRIM | ZN | INDUS | CHAS | NOX | RM | AGE | DIS | RAD | TAX | PTRATIO | B | LSTAT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.00632 | 18.0 | 2.31 | 0.0 | 0.538 | 6.575 | 65.2 | 4.0900 | 1.0 | 296.0 | 15.3 | 396.90 | 4.98 |
| **1** | 0.02731 | 0.0 | 7.07 | 0.0 | 0.469 | 6.421 | 78.9 | 4.9671 | 2.0 | 242.0 | 17.8 | 396.90 | 9.14 |
| **2** | 0.02729 | 0.0 | 7.07 | 0.0 | 0.469 | 7.185 | 61.1 | 4.9671 | 2.0 | 242.0 | 17.8 | 392.83 | 4.03 |
| **3** | 0.03237 | 0.0 | 2.18 | 0.0 | 0.458 | 6.998 | 45.8 | 6.0622 | 3.0 | 222.0 | 18.7 | 394.63 | 2.94 |
| **4** | 0.06905 | 0.0 | 2.18 | 0.0 | 0.458 | 7.147 | 54.2 | 6.0622 | 3.0 | 222.0 | 18.7 | 396.90 | 5.33 |

In [7]:

```
from sklearn.model_selection import train_test_split
```

In [8]:

```
x = boston
y = boston_dataset['target']
```

In [10]:

```
x_train,x_test,y_train,y_test  = train_test_split(x,y,test_size = 0.2)
```

In [11]:
```
from sklearn.linear_model import LinearRegression
model  = LinearRegression()## Initiate an object of the class LinearRegression
```

In [16]:
```
model.fit(x_train,y_train)
```

Out[16]:
```
LinearRegression()
```

In [17]:
```
model.coef_
```

Out[17]:
```
array([-9.77008335e-02,  5.48340243e-02,  1.13933515e-02,  2.39050225e+00,
       -1.69842810e+01,  3.48249249e+00,  1.49683660e-03, -1.39774194e+00,
        2.94125048e-01, -1.04164138e-02, -9.72248399e-01,  1.10809460e-02,
       -5.41757798e-01])
```

In [18]:
```
y_pred = model.predict(x_test)
```

In [19]:
```
from sklearn.metrics import r2_score,mean_squared_error
```

In [20]:
```
r2_score(y_pred,y_test)
```

Out[20]:
```
0.6922415690090088
```

In [16]:
```
j = mean_squared_error(y_pred,y_test)
```

In [19]:
```
import numpy as np
j/np.mean(y_test) * 100
```

Out[19]:
```
114.64320722496011
```

In [ ]:
```

```