In [ ]:

import numpy as np

import pandas as pd

import sklearn

In [ ]:

from sklearn.datasets import load\_boston

df = load\_boston()

In [ ]:

df.keys()

Out[ ]:

dict\_keys(['data', 'target', 'feature\_names', 'DESCR', 'filename'])

In [ ]:

boston = pd.DataFrame(df.data, columns=df.feature\_names)

boston.head()

Out[ ]:

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 [ ]:

boston['MEDV'] = df.target

boston.head()

Out[ ]:

CRIM ZN INDUS CHAS NOX RM AGE DIS RAD TAX PTRATIO B LSTAT MEDV

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 24.0

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 21.6

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 34.7

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 33.4

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 36.2

In [ ]:

boston.isnull()

Out[ ]:

CRIM ZN INDUS CHAS NOX RM AGE DIS RAD TAX PTRATIO B LSTAT MEDV

0 False False False False False False False False False False False False False False

1 False False False False False False False False False False False False False False

2 False False False False False False False False False False False False False False

3 False False False False False False False False False False False False False False

4 False False False False False False False False False False False False False False

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

501 False False False False False False False False False False False False False False

502 False False False False False False False False False False False False False False

503 False False False False False False False False False False False False False False

504 False False False False False False False False False False False False False False

505 False False False False False False False False False False False False False False

506 rows × 14 columns

In [ ]:

boston.isnull().sum()

Out[ ]:

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

In [ ]:

from sklearn.model\_selection import train\_test\_split

X = boston.drop('MEDV', axis=1)

Y = boston['MEDV']

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size = 0.15, random\_state=5)

print(X\_train.shape)

print(X\_test.shape)

print(Y\_train.shape)

print(Y\_test.shape)

(430, 13)

(76, 13)

(430,)

(76,)

In [ ]:

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

In [ ]:

## FITTING MODEL ON THE TRAINING DATASET

lin\_model = LinearRegression()

lin\_model.fit(X\_train, Y\_train)

Out[ ]:

LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)

In [ ]:

y\_train\_predict = lin\_model.predict(X\_train)

rmse = (np.sqrt(mean\_squared\_error(Y\_train, y\_train\_predict)))

print("The model performance for training set")

print('RMSE is {}'.format(rmse))

print("\n")

# on testing set

y\_test\_predict = lin\_model.predict(X\_test)

rmse = (np.sqrt(mean\_squared\_error(Y\_test, y\_test\_predict)))

print("The model performance for testing set")

print('RMSE is {}'.format(rmse))

The model performance for training set

RMSE is 4.710901797319796

The model performance for testing set

RMSE is 4.687543527902972