```
import numpy as np
import pandas as pd
import sklearn



from sklearn.datasets import load_boston
df = load_boston()


df.keys()
```

```
      dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename'])
```

```
boston = pd.DataFrame(df.data, columns=df.feature_names)
boston.head()
```

```
boston['MEDV'] = df.target
boston.head()
```

```
boston.isnull()
```

```
boston.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
```

```
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,)
```

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

```
lin_model = LinearRegression()
lin_model.fit(X_train, Y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
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")
Y_test_predict = lin_model.predict(X_test)
rmse =(np.sqrt(mean_squared_error(Y_test , Y_test_predict)))
print("The model performance for training set")
print('RMSE is {}'.format(rmse))
```

```
The model performance for training set
RMSE is 4.710901797319796


The model performance for training set
RMSE is 4.687543527902972
```