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


from sklearn.database import load.boaston
df = load_bkaston()


df.keys()


Print(df.feature_names)
Print(df.target)
Print(df.filename)
Print(df.data)


boaston = pd.DataFrame(df.data, columns = 1, df.feature_names)
boaston.head()


//Adding a new column of target valuesto the dataframe
boaston['MEDV'] = df.target
boaston.head()


boaston.isnull().sum()


from sklearn.mod_selection import train_test_split
X = boaston.drop('MEDV', axis = 1)
Y = boaston['MEDV']
X_train, X_test, Y_train, Ytest = train_test_split x,y,test_size = 0.15)
Print(X_train.shape)
Print(X_test.shape)
Print(Y_train.shape)
Print(Y_test.shape)


from sklearn.linear_model import LinearRegression
from sklearn.matrices import mean_squared_error


##FITTING MODEL ON THE TRAINING DATASET
lin_model = LinearRegression ()
lin_mkdel.fit(X_train,Y_train)


y_train_predict = lin_model.predict(X_train)
rmse = ( no.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 = (no.sqrt(mean_squarer_error(Y_test, y_test_predict)))
```

```
Print("The model performance for testing set")
Print("RMSE is {}'.Format(rmse))
```