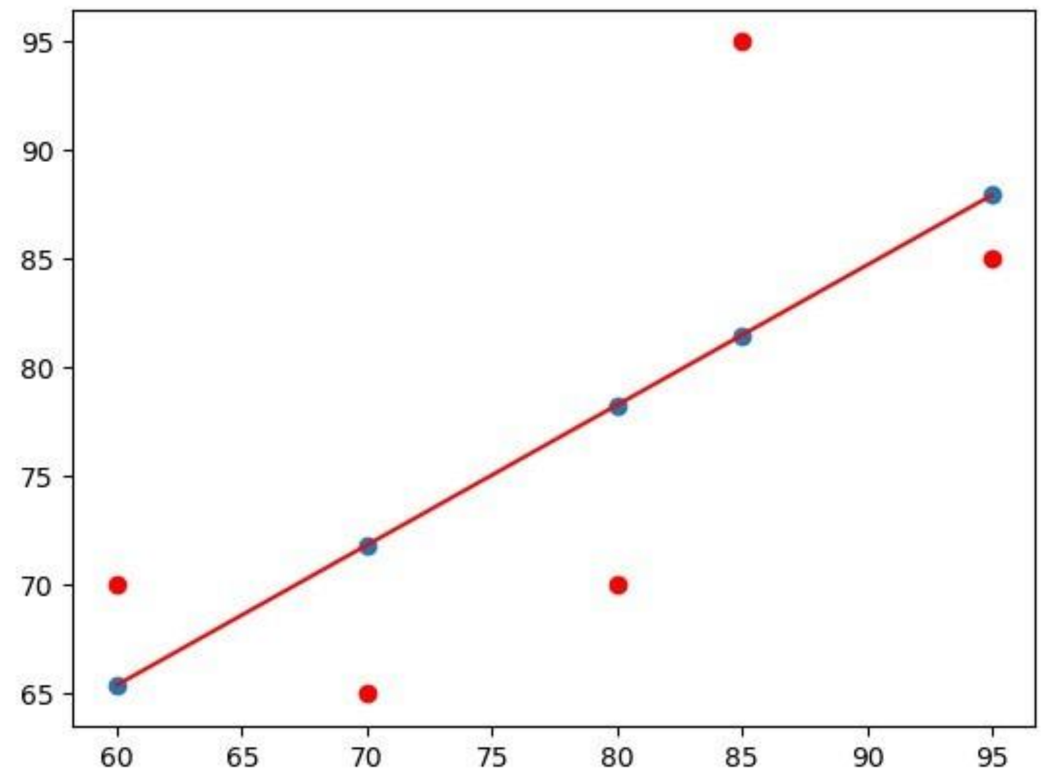
Lab Assignment NO 4

AIM:-

Create a Linear Regression Model using Python/R to predict home prices using Boston Housing
Dataset (https://www.kaggle.com/c/boston-housing). The Boston Housing dataset contains
information about various houses in Boston through different parameters. There are 506 samples and 14 feature variables
in this dataset.
The objective is to predict the value of prices of the house using the given features

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

x=np.array([95,85,80,70,60])
y=np.array([85,95,70,65,70])
model=np.polyfit(x, y, 1)
model
array([ 0.64383562, 26.78082192])
predict=np.poly1d(model)
predict(65)
68.63013698630137
y_pred= predict (x)
y_pred
array([87.94520548, 81.50684932, 78.28767123, 71.84931507,
65.4109589 ])

from sklearn.metrics import r2_score
r2_score(y, y_pred)
0.4803218090889326
y_line= model[1] + model[0]*x
plt.plot(x, y_line, c='r')
plt.scatter(x, y_pred)
plt.scatter(x, y, c='r')

<matplotlib.collections.PathCollection at 0x1e79c2ba890>
```




```
#import numpy as np
#import pandas as pd
#import matplotlib.pyplot as plt

from sklearn.datasets import fetch_openml
from sklearn.datasets import fetch_california_housing
housing = fetch_california_housing()
housing

{'data': array([[   8.3252    ,   41.        ,    6.98412698, ...,
2.55555556,
           37.88      , -122.23      ],
        [   8.3014    ,   21.        ,    6.23813708, ...,
2.10984183,
           37.86      , -122.22      ],
        [   7.2574    ,   52.        ,    8.28813559, ...,
2.80225989,
           37.85      , -122.24      ],
        ...,
        [   1.7       ,   17.        ,    5.20554273, ...,
2.3256351 ,
           39.43      , -121.22      ],
        [   1.8672    ,   18.        ,    5.32951289, ...,
2.12320917,
           39.43      , -121.32      ],
        [   2.3886    ,   16.        ,    5.25471698, ...,
```

```
2.61698113,
           39.37      , -121.24      ]]),
 'target': array([4.526, 3.585, 3.521, ..., 0.923, 0.847, 0.894]),
 'frame': None,
 'target_names': ['MedHouseVal'],
 'feature_names': ['MedInc',
  'HouseAge',
  'AveRooms',
  'AveBedrms',
  'Population',
  'AveOccup',
  'Latitude',
  'Longitude'],
 'DESCR': '.. _california_housing_dataset:\n\nCalifornia Housing
dataset\n--------------------------\n\n**Data Set Characteristics:**\
n\n    :Number of Instances: 20640\n\n    :Number of Attributes: 8
numeric, predictive attributes and the target\n\n    :Attribute
Information:\n        - MedInc        median income in block group\n
- HouseAge      median house age in block group\n        - AveRooms
average number of rooms per household\n        - AveBedrms     average
number of bedrooms per household\n        - Population    block group
population\n        - AveOccup      average number of household
members\n        - Latitude      block group latitude\n        -
Longitude     block group longitude\n\n    :Missing Attribute Values:
None\n\nThis dataset was obtained from the StatLib repository.\
nhttps://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html\n\nThe
target variable is the median house value for California districts,\
nexpressed in hundreds of thousands of dollars ($100,000).\n\nThis
dataset was derived from the 1990 U.S. census, using one row per
census\nblock group. A block group is the smallest geographical unit
for which the U.S.\nCensus Bureau publishes sample data (a block group
typically has a population\nof 600 to 3,000 people).\n\nA household is
a group of people residing within a home. Since the average\nnumber of
rooms and bedrooms in this dataset are provided per household, these\
ncolumns may take surprisingly large values for block groups with few
households\nand many empty houses, such as vacation resorts.\n\nIt can
be downloaded/loaded using the\
n:func:`sklearn.datasets.fetch_california_housing` function.\n\n..
topic:: References\n\n    - Pace, R. Kelley and Ronald Barry, Sparse
Spatial Autoregressions,\n      Statistics and Probability Letters, 33
(1997) 291-297\n'}

data = pd.DataFrame(fetch_california_housing().data)

data.columns =fetch_california_housing().feature_names

data.head()

   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup
Latitude  \
```

```
0  8.3252       41.0  6.984127   1.023810       322.0  2.555556
37.88
1  8.3014       21.0  6.238137   0.971880      2401.0  2.109842
37.86
2  7.2574       52.0  8.288136   1.073446       496.0  2.802260
37.85
3  5.6431       52.0  5.817352   1.073059       558.0  2.547945
37.85
4  3.8462       52.0  6.281853   1.081081       565.0  2.181467
37.85

   Longitude
0    -122.23
1    -122.22
2    -122.24
3    -122.25  4    -122.25
```

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

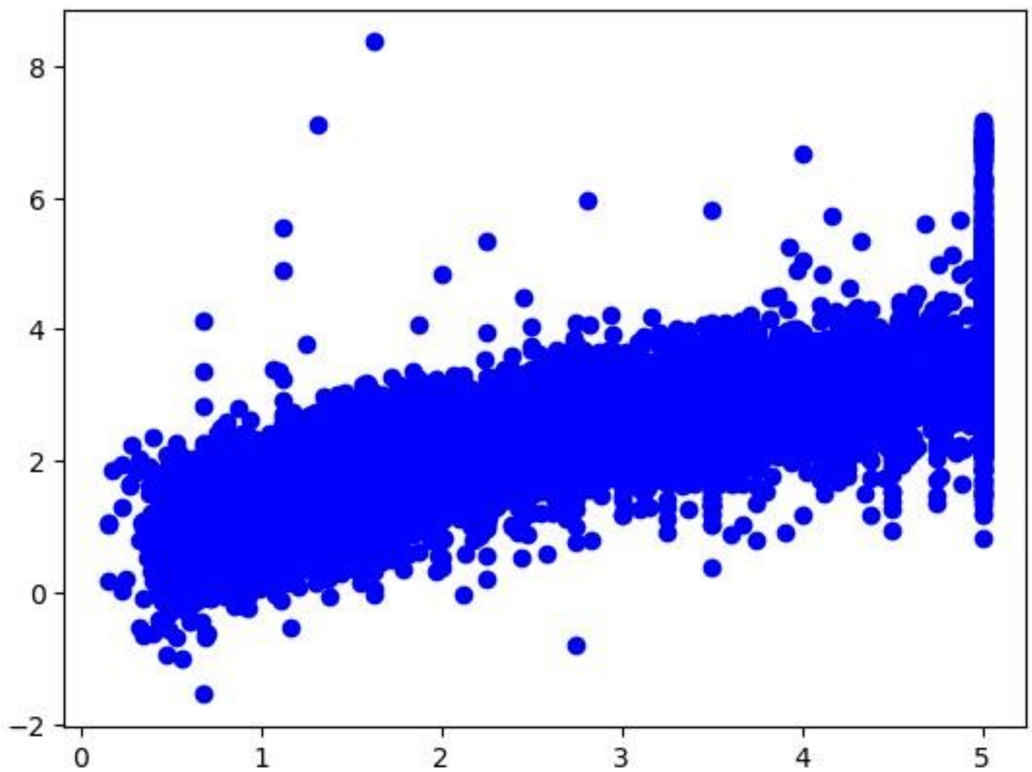
```
df
```

```
        MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup
Latitude  \
0       8.3252      41.0  6.984127   1.023810       322.0  2.555556
37.88
1       8.3014      21.0  6.238137   0.971880      2401.0  2.109842
37.86
2       7.2574      52.0  8.288136   1.073446       496.0  2.802260
37.85
3       5.6431      52.0  5.817352   1.073059       558.0  2.547945
37.85
4       3.8462      52.0  6.281853   1.081081       565.0  2.181467
37.85
...        ...       ...       ...        ...         ...       ...
...
20635  1.5603      25.0  5.045455   1.133333       845.0  2.560606
39.48
20636  2.5568      18.0  6.114035   1.315789       356.0  3.122807
39.49
20637  1.7000      17.0  5.205543   1.120092      1007.0  2.325635
39.43
20638  1.8672      18.0  5.329513   1.171920       741.0  2.123209
39.43
20639  2.3886      16.0  5.254717   1.162264      1387.0  2.616981
39.37

        Longitude
0         -122.23
1         -122.22
```

```
2     -122.24
3     -122.25
4     -122.25
...
...
20635    -
         121.09
20636    -
         121.21
20637    -
         121.22
20638    -
         121.32
20639    -
         121.24
[20640 rows x 8
columns]
data['PRICE'] = housing.target
data.isnull().sum()
MedInc        0
HouseAge      0
AveRooms      0
AveBedrms     0
Population    0
AveOccup      0
Latitude      0
Longitude     0
PRICE         0
dtype: int64

x = data.drop(['PRICE'], axis = 1)
y = data['PRICE']

from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size
=0.2,random_state = 0)

import sklearn

from sklearn.linear_model import LinearRegression
lm = LinearRegression() model=lm.fit(xtrain,
ytrain) ytrain_pred = lm.predict(xtrain)
ytest_pred = lm.predict(xtest)
df=pd.DataFrame(ytrain_pred,ytrain)
df=pd.DataFrame(ytest_pred,ytest)
```

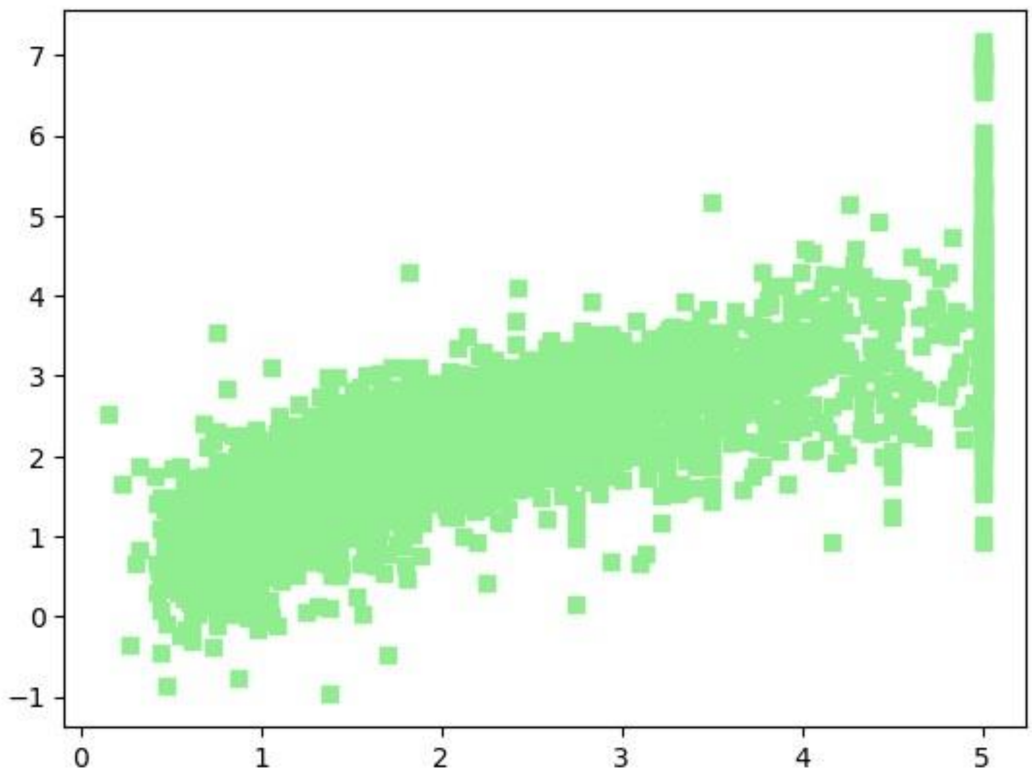
```
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(ytest, ytest_pred) print(mse)
0.5289841670367192
mse = mean_squared_error(ytrain_pred,ytrain)
print(mse)
0.5234413607125448
plt.scatter(ytrain ,ytrain_pred,c='blue',marker='o',label='Training
data')
```

```
<matplotlib.collections.PathCollection at 0x1e79e542c90>
```

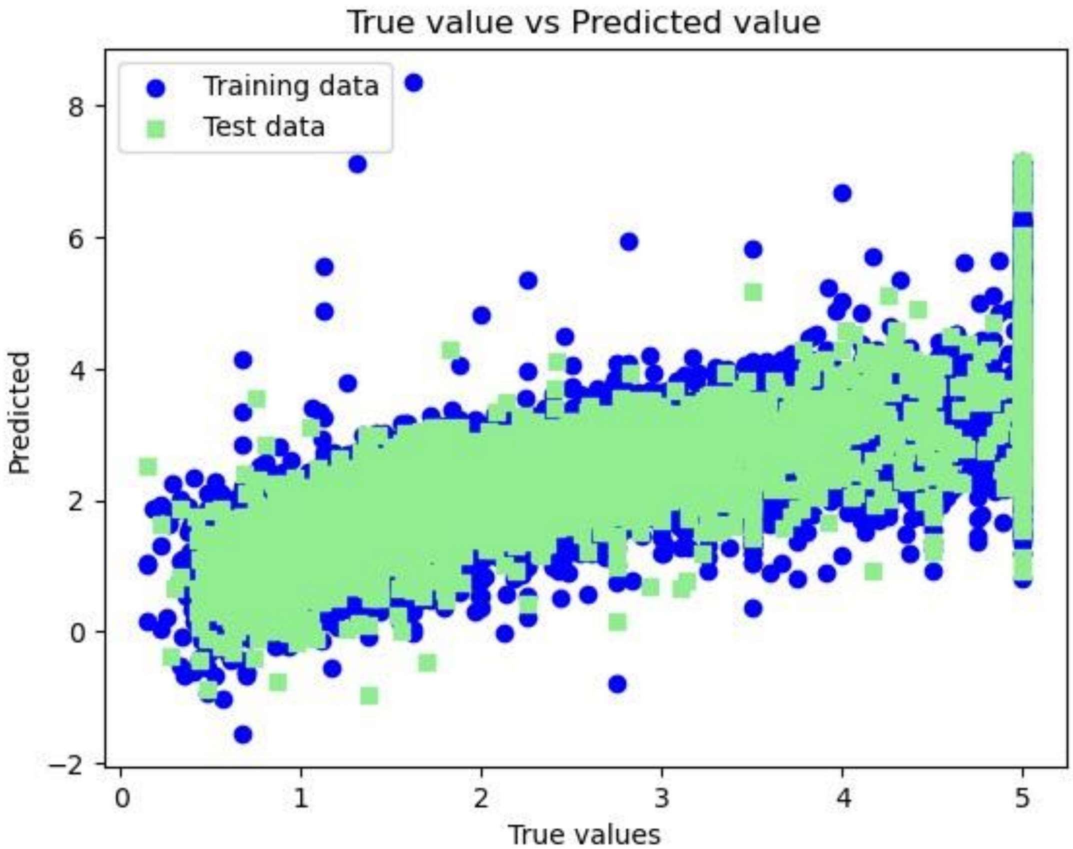



```
plt.scatter(ytest,ytest_pred ,c='lightgreen',marker='s',label='Test
data')
```

```
<matplotlib.collections.PathCollection at 0x1e79e5387d0>
```




```
plt.scatter(ytrain ,ytrain_pred,c='blue',marker='o',label='Training
data')
plt.scatter(ytest,ytest_pred ,c='lightgreen',marker='s',label='Test
data')
plt.xlabel('True values')
plt.ylabel('Predicted')
plt.title("True value vs Predicted value")
plt.legend(loc= 'upper left') plt.plot()
plt.show()
```



Name :Shruti Manwar Roll no: B2_13229