Assignment No:04

Aim: Data Analytics I Create a Linear Regression Model using Python/R to predict home prices using Boston Housing Dataset (https://www.kaggle.com/c/boston-housing (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.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
x=np.array([95,85,80,70,60])
```

In [3]:
```
y=np.array([85,95,70,65,70])
```

In [4]:
```
model= np.polyfit(x, y, 1)
```

In [5]:
```
model
```

Out[5]:
```
array([ 0.64383562, 26.78082192])
```

In [6]:
```
predict = np.poly1d(model)
```

In [7]:
```
predict(65)
```

Out[7]:
```
68.63013698630137
```

In [8]:
```
y_pred= predict(x)
y_pred
```

Out[8]:
```
array([87.94520548, 81.50684932, 78.28767123, 71.84931507, 65.4109589 ])
```

In [9]:
```
from sklearn.metrics import r2_score
```

In [10]:
```
r2_score(y, y_pred)
```

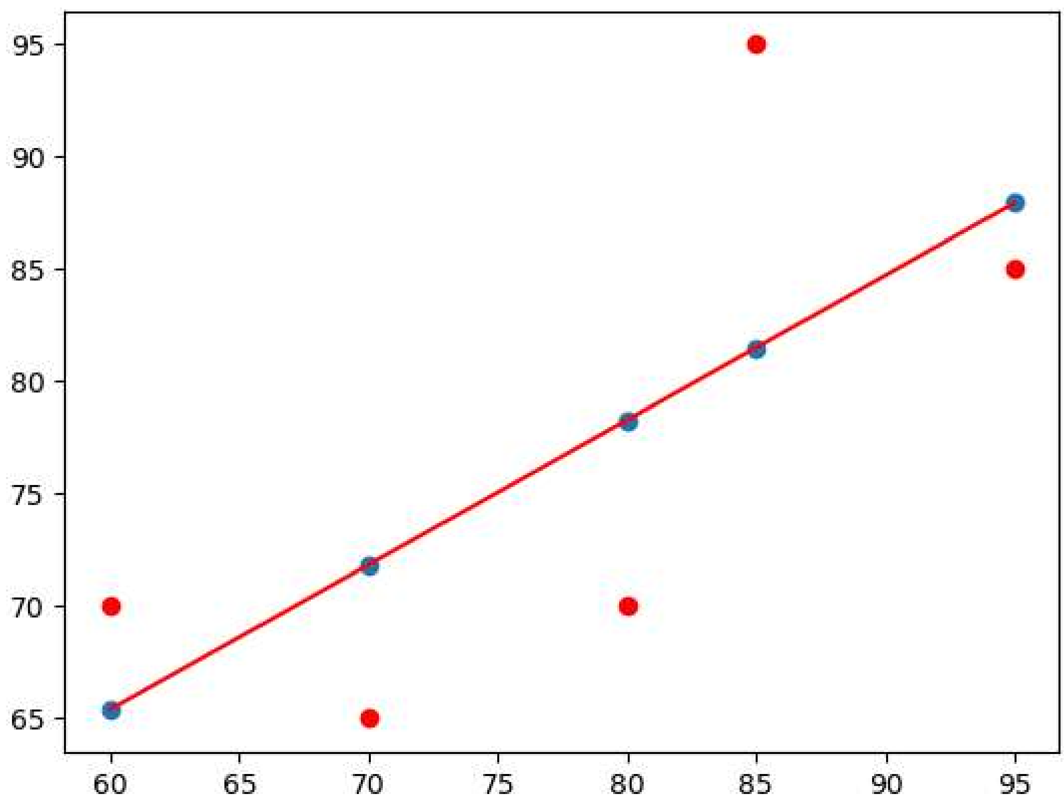
Out[10]:
```
0.4803218090889326
```

In [11]:

```
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')
```

Out[11]:

```
<matplotlib.collections.PathCollection at 0x12ebf394fd0>
```



In [12]:

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

In [3]:

```
import pandas as pd
from sklearn.datasets import fetch_openml
from sklearn.datasets import fetch_california_housing
housing = fetch_california_housing()
```

In [4]:
```
1 housing
```

Out[4]:
```
{'data': array([[   8.3252    ,   41.        ,    6.98412698, ...,    2.55555
556,
           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 In
stances: 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 grou
p\n        - AveRooms      average number of rooms per household\n        - A
veBedrms     average number of bedrooms per household\n        - Population
block group population\n        - AveOccup      average number of household m
embers\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/R
egression/cal_housing.html\n\nThe target variable is the median house value f
or California districts,\nexpressed in hundreds of thousands of dollars ($10
0,000).\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 peopl
e 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, suc
h as vacation resorts.\n\nIt can be downloaded/loaded using the\n:func:`sklea
rn.datasets.fetch_california_housing` function.\n\n.. topic:: References\n\n
- Pace, R. Kelley and Ronald Barry, Sparse Spatial Autoregressions,\n      St
atistics and Probability Letters, 33 (1997) 291-297\n'}
```

In [7]:

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

Out[7]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25
...	...	...	...	...	...	...	...	...
20635	1.5603	25.0	5.045455	1.133333	845.0	2.560606	39.48	-121.09
20636	2.5568	18.0	6.114035	1.315789	356.0	3.122807	39.49	-121.21
20637	1.7000	17.0	5.205543	1.120092	1007.0	2.325635	39.43	-121.22
20638	1.8672	18.0	5.329513	1.171920	741.0	2.123209	39.43	-121.32
20639	2.3886	16.0	5.254717	1.162264	1387.0	2.616981	39.37	-121.24

20640 rows × 8 columns

In [9]:

```
df.head()
```

Out[9]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25

In [10]:

```
df['PRICE'] = housing.target

```

In [11]:

```
df.isnull().sum()
```

Out[11]:

```
MedInc        0
HouseAge      0
AveRooms      0
AveBedrms     0
Population    0
AveOccup      0
Latitude      0
Longitude     0
PRICE         0
dtype: int64
```

In [16]:
```
x = df.drop(['PRICE'], axis = 1)
y = df['PRICE']
```

In [19]:
```
from sklearn.model_selection import train_test_split

xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.2, rando

```

In [20]:
```
 import sklearn
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
model=lm.fit(xtrain, ytrain)
```

In [21]:
```
ytrain_pred = lm.predict(xtrain)
ytest_pred = lm.predict(xtest)
```

In [22]:
```
df=pd.DataFrame(ytrain_pred,ytrain)
df=pd.DataFrame(ytest_pred,ytest)
```

In [23]:
```
from sklearn.metrics import mean_squared_error, r2_score
```

In [24]:
```
mse = mean_squared_error(ytest, ytest_pred)
print(mse)
```

```
0.5289841670367221
```

In [25]:
```
mse = mean_squared_error(ytrain_pred,ytrain)
print(mse)
```

```
0.5234413607125449
```

In [26]:
```
mse = mean_squared_error(ytest, ytest_pred)
print(mse)
```

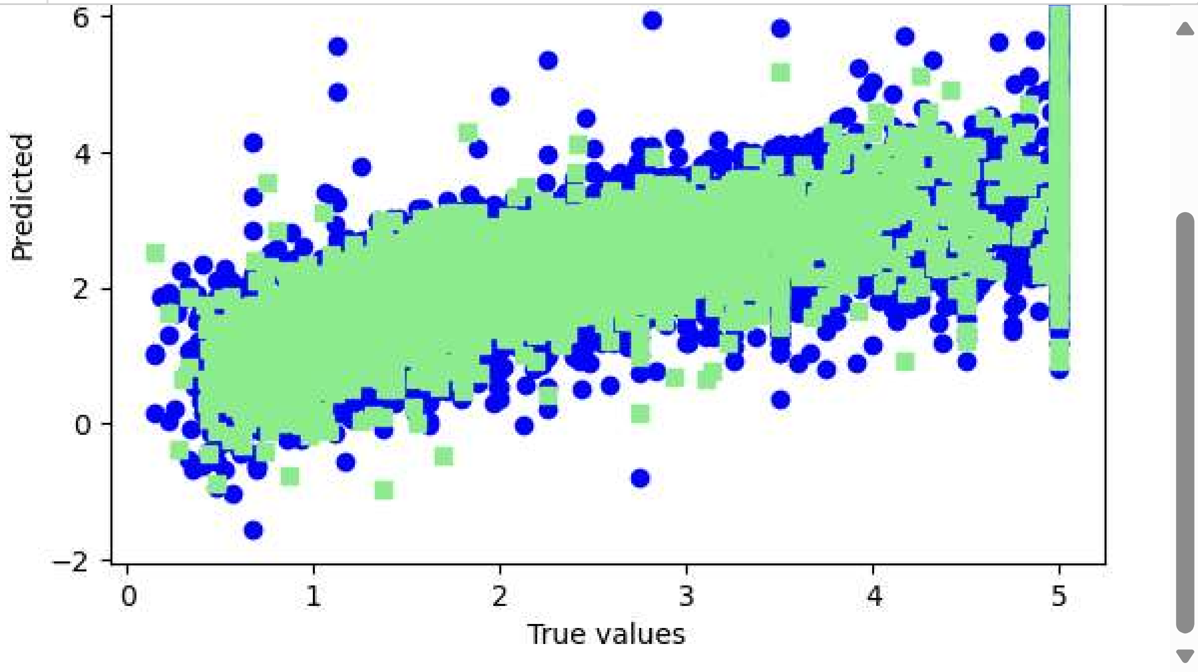
```
0.5289841670367221
```

In [28]:

```
import matplotlib.pyplot as plt


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

```



Name : Sneha Navgire

Roll no :13246