A Project Report

On

House Price Estimation

Submitted by

Abhay Goyal, 2115000009

Ayush Anuroop,2115000251

Anushka Singh,2115000188

Atharv Bharadwaj,2115000241

Anurag Sharma,2115000184

Supervisor

Dr. Subhash Chand Agrawal
Assistant Professor

Department of Computer Engineering & Applications,
GLA University, Mathura

GLA University, Mathura - 281406

Date of Submission – 28/05/2022

DECLARATION

We ***Atharv Bharadwaj B.Tech Ist Year 2115000241, Anurag Sharma B.Tech Ist Year 2115000184, Abhay Goyal B.Tech Ist Year 2115000009, Ayush Anuroop B.Tech Ist Year 2115000251, Anushka Singh B.Tech Ist Year 2115000188*** hereby declare that the work presented in this project report entitled House Price Estimation is an authentic record of our own work carried out under supervision of Dr.Subhash Chandra Agrawal.

Atharv Bharadwaj, 2115000241

Anurag Sharma, 2115000184

Abhay Goyal, 2115000009

Ayush Anuroop, 2115000251

Anushka Singh, 2115000188

CERTIFICATE

This is to certify that the above statement made by the students are correct to the best of my knowledge and belief.

Date: 28/05/2022

Place: Mathura

Dr. Subhash Chand Agrawal

Asst. Professor

Contents

Chapter - 1
Introduction, Motivation and Objective

1.1 Introduction

Accurately estimating the value of real estate is an important problem for many stakeholders including house owners, house buyers, agents, creditors, and investors .It is also a difficult one. Though it is common knowledge that factors such as the size, number of rooms and location affect the price, there are many other things at play. Additionally, prices are sensitive to changes in market demand and the peculiarities of each situation, such as when a property need to be urgently sold.

The sales price of a property can be predicted in various ways, but is often based on regression techniques. All regression techniques essentially involve one or more predictor variables as input and a single target variable as output. In this paper, we compare different machine learning methods performance in predicting the selling price of houses based on a number of features such as average number of rooms per dwelling, proportion of owner occupied units built prior to 1940, weighted distances to five Boston employment centres, percentage lower status of the populations.

1.2 Motivations

Being extremely interested in everything having a relation with the Machine Learning, the independent project was a great occasion to give me the time to learn and confirm my interest for this field. The fact that we can make estimations, predictions and give the ability for machines to learn by themselves is both powerful and limitless in term of application possibilities. We can use Machine Learning in Finance, Medicine, almost everywhere. That's why we decided to conduct my project around the Machine Learning.

As a first experience, we wanted to make my project as much didactic as possible by approaching every different Steps of the machine learning process and trying to understand them deeply. Known as "toy problem" defining the problems that are not immediate scientific interest but useful to illustrate and practice, we chose to take House Pricing Prediction as approach. The goal was to predict the price of a given apartment according to the market prices taking into account different "features" that will be developed in the following sections.

1.3 Objective

The goal of this assignment is to estimate the price of a house of Boston area in the United States. We will use publicly available data to answer the following questions: what is the average price of a house in the area you chose, and how does that compare to the national average price of a house; and how do different factors affect the price of a house in the area you chose; , what is the relationship between the price of a house and the distance a house is from the ocean in the area you chose; and , what are some ways that you could use additional data to answer the above questions, or other questions you may have about the area you chose.

Chapter - 2
Description and Work done

2.1 Description

House prices changes and manipulates every year, so there is a need for a system to predict house prices in the future. House price prediction can help the developers to determine the price of a house and can help the customer to arrange the right time and information to purchase a house. There are certain factors that influence the price of a house which include Average No of Rooms, proportion of owner-occupied units built prior to 1940,weighted distances to five Boston employment centres,% lower status of the population. This research aims to predict house prices based on supervised model with regression task and this model is based on Batch learning i.e. On the Dataset of Boston. The result from this research proved combination regression and get the minimum prediction error and getting the optimized result.

Getting the Data and Previous Pre-process

The dataset used in this project comes from the UCI Machine Learning Repository. This data was collected in 1978 and each of the 506 entries represents aggregate information about 14 features of homes from various suburbs located in Boston.

The features can be summarized as follows:

- CRIM: This is the per capita crime rate by town
- ZN: This is the proportion of residential land zoned for lots larger than 25,000 sq.ft.
- INDUS: This is the proportion of non-retail business acres per town.
- CHAS: This is the Charles River dummy variable (this is equal to 1 if tract bounds river; 0 otherwise)
- NOX: This is the nitric oxides concentration (parts per 10 million)
- RM: This is the average number of rooms per dwelling
- AGE: This is the proportion of owner-occupied units built prior to 1940

- DIS: This is the weighted distances to five Boston employment centres
- RAD: This is the index of accessibility to radial highways
- TAX: This is the full-value property-tax rate per $10,000
- PTRATIO: This is the pupil-teacher ratio by town
- B: This is calculated as $1000(Bk — 0.63)^2$, where Bk is the proportion of people of African American descent by town
- LSTAT: This is the percentage lower status of the population
- MEDV: This is the median value of owner-occupied homes in $1000s

This is an overview of the original dataset, with its original features:

	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
5	0.02985	0.0	2.18	0.0	0.458	6.430	58.7	6.0622	3.0	222.0	18.7	394.12	5.21	28.7

For the purpose of the project the dataset has been preprocessed as follows:

- The essential features for the project are: ‘RM’, ‘LSTAT’, ‘PTRATIO’ and ‘MEDV’. The remaining features have been excluded.
- 16 data points with a ‘MEDV’ value of 50.0 have been removed. As they likely contain censored or missing values.
- 1 data point with a ‘RM’ value of 8.78 it is considered an outlier and has been removed for the optimal performance of the model.

- As this data is out of date, the 'MEDV' value has been scaled multiplicatively to account for 35 years of market inflation.

We'll now open a python 3 Jupyter Notebook and execute the following code snippet to load the dataset and remove the non-essential features. Receiving a success message if the actions were correctly performed.

As our goal is to develop a model that has the capacity of predicting the value of houses, we will split the dataset into features and the target variable. And store them in features and prices variables, respectively

- The features 'RM', 'LSTAT' and 'PTRATIO', give us quantitative information about each data point. We will store them in *features*.
- The target variable, 'MEDV', will be the variable we seek to predict. We will store it in *prices*.

Feature Observation

Data Science is the process of making some assumptions and hypothesis on the data, and testing them by performing some tasks. Initially we could make the following intuitive assumptions for each feature:

- Houses with more rooms (higher 'RM' value) will worth more. Usually houses with more rooms are bigger and can fit more people, so it is reasonable that they cost more money. They are directly proportional variables.
- Neighbourhoods with more lower class workers (higher 'LSTAT' value) will worth less. If the percentage of lower working class people is higher, it is likely that they have low purchasing power and therefore, they houses will cost less. They are inversely proportional variables.
- Neighbourhoods with more students to teachers ratio (higher 'PTRATIO' value) will be worth less. If the percentage of students to teachers ratio people is higher, it is likely that in the neighbourhood there are less schools, this could be because there is less tax income which could be because in that neighbourhood people earn less money. If people earn less money it is likely that their houses are worth less. They are inversely proportional variables.
- We'll find out if these assumptions are correct through the project.

2.2 Work Done:

Loading Data Using Pandas Library

We can print the data using head () that returns the first n rows of an object (default n= 5). It helps in knowing the data and datatype of the object.

```
In [1]: import pandas as pd
        housing=pd.read_csv("newdata.csv")
        housing.head()
```

Out[1]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2

Describing the attribute's of Dataset

With the help of describe method we came to know about the count ,mean ,std, min,25%,50%,75%,max of the dataset

housing.Describe()

```
In [2]: housing.describe()
```

Out[2]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	L
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.0
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901	3.795043	9.549407	408.237154	18.455534	356.674032	12.6
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.105710	8.707259	168.537116	2.164946	91.294864	7.1
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600	1.000000	187.000000	12.600000	0.320000	1.7
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.025000	2.100175	4.000000	279.000000	17.400000	375.377500	6.9
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.500000	3.207450	5.000000	330.000000	19.050000	391.440000	11.3
75%	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	94.075000	5.188425	24.000000	666.000000	20.200000	396.225000	16.9
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500	24.000000	711.000000	22.000000	396.900000	37.9

Getting Each Attribute's Data Type

Using housing. D types gives us the each attribute data type as a result. In the output mentioned below we can observe the data types for each of the attribute in our newdata dataset.

housing.dtypes

```
In [3]: housing.dtypes

Out[3]: CRIM       float64
        ZN         float64
        INDUS      float64
        CHAS         int64
        NOX        float64
        RM         float64
        AGE        float64
        DIS        float64
        RAD          int64
        TAX          int64
        PTRATIO    float64
        B          float64
        LSTAT      float64
        MEDV       float64
        dtype: object
```

Dimensions(Number of Rows and Columns) in our Dataset

Using *datasetnameshape* gives us the dimension as result. In output below we can see (506,14) that indicates newdata dataset has 506 rows and 14 columns.

housing.shape

```
In [4]: housing.shape

Out[4]: (506, 14)
```

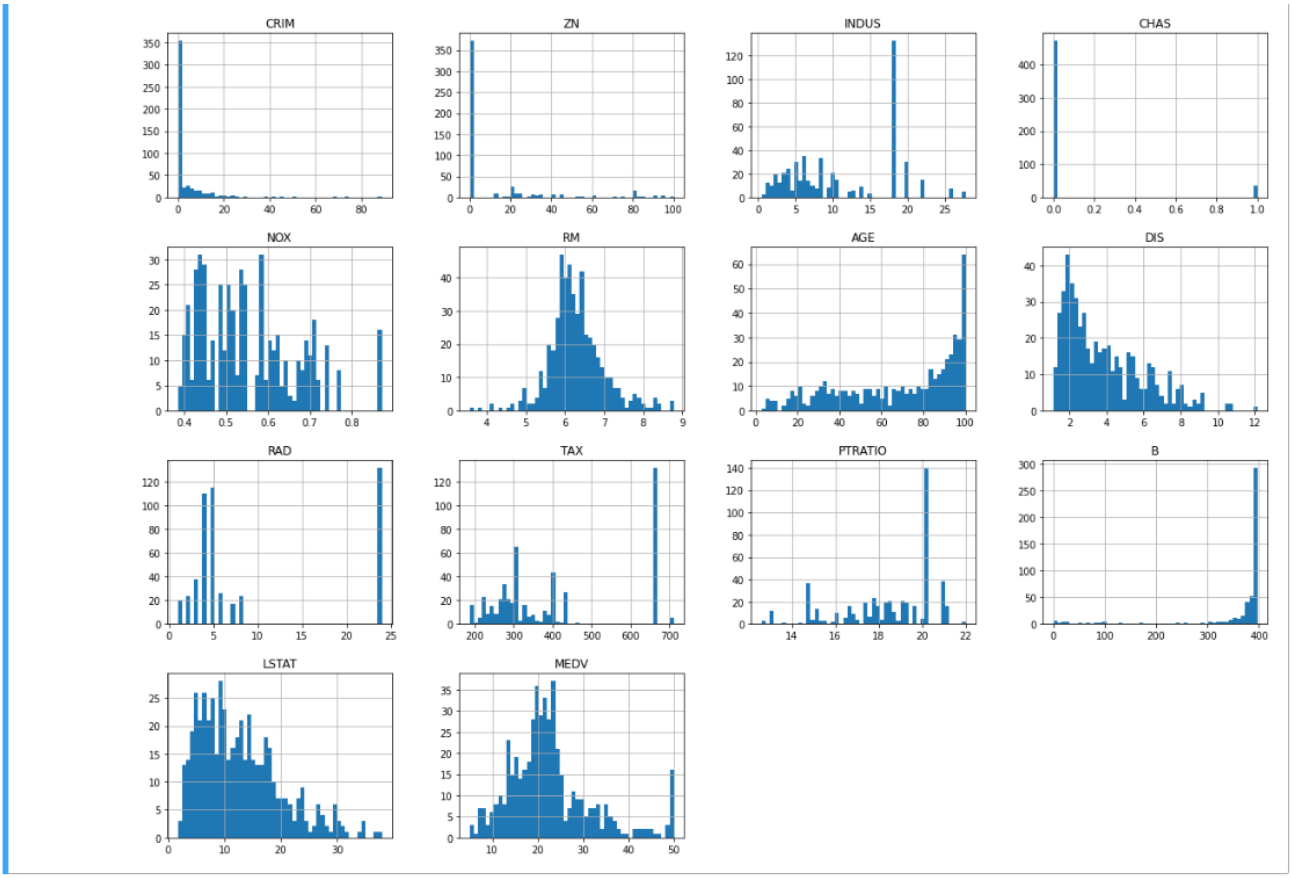
Visualization Of Data using Matplotlib

we need to import matplotlib library and use *plot()* function, we can make our diagram more informative by providing Title, Labels on x and y axis using *title(),xlabel()* and *ylabel()* respectively.

import matplotlib.pyplot as plt

housing.Hist(bins=50,figsize=(20,15))

```
In [5]: import matplotlib.pyplot as plt
        housing.hist(bins=50,figsize=(20,15))

Out[5]: array([[<AxesSubplot:title={'center':'CRIM'}>,
                <AxesSubplot:title={'center':'ZN'}>,
                <AxesSubplot:title={'center':'INDUS'}>,
                <AxesSubplot:title={'center':'CHAS'}>],
               [<AxesSubplot:title={'center':'NOX'}>,
                <AxesSubplot:title={'center':'RM'}>,
                <AxesSubplot:title={'center':'AGE'}>,
                <AxesSubplot:title={'center':'DIS'}>],
               [<AxesSubplot:title={'center':'RAD'}>,
                <AxesSubplot:title={'center':'TAX'}>,
                <AxesSubplot:title={'center':'PTRATIO'}>,
                <AxesSubplot:title={'center':'B'}>],
               [<AxesSubplot:title={'center':'LSTAT'}>,
                <AxesSubplot:title={'center':'MEDV'}>, <AxesSubplot:>,
                <AxesSubplot:>]], dtype=object)
```



From the histograms we came to know that the value of Dependent Variable i.e. MEDV Depends mostly on 4 Attributes that are RM,AGE,DIS,LSTAT

ATTRIBUTE INFORMATION:

1.CRIM per capita crime rate by town
2.ZN proportion of residential land zoned for lots over 25,000 sq. ft.
3.INDUS proportion of non-retail business acres per town
4.CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
5.NOX nitric oxides concentration (parts per 10 million)
6.RM average number of rooms per dwelling
7.AGE proportion of owner-occupied units built prior to 1940
8.DIS weighted distances to five Boston employment centers
9.RAD index of accessibility to radial highways
10.TAX full-value property-tax rate per $10,000
11.PTRATIO pupil-teacher ratio by town
12.B 1000(Bk –0.63)^2 where Bk is the proportion of blacks by town
13.LSTAT % lower status of the population
14.MEDV Median value of owner-occupied homes in $1000's

Now we are assigning both Independent Variables and Dependent Variables in X and y

we need to decide the Independent and Dependent variables of our data

X=housing[['RM','AGE','DIS','LSTAT']]

y=housing['MEDV']

X

```
In [8]: X=housing[['RM','AGE','DIS','LSTAT']]
        y=housing['MEDV']
        X
```

Out[8]:

	RM	AGE	DIS	LSTAT
0	6.575	65.2	4.0900	4.98
1	6.421	78.9	4.9671	9.14
2	7.185	61.1	4.9671	4.03
3	6.998	45.8	6.0622	2.94
4	7.147	54.2	6.0622	5.33
...	...	...	...	...
501	6.593	69.1	2.4786	9.67
502	6.120	76.7	2.2875	9.08
503	6.976	91.0	2.1675	5.64
504	6.794	89.3	2.3889	6.48
505	6.030	80.8	2.5050	7.88

506 rows × 4 columns

Training and Testing Data

```
In [10]: from sklearn.model_selection import train_test_split
         X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=100)
```

Now to generate training and testing data we need to import *traintestsplit* from *skill learn. Modelselection*. As mentioned below we need to provide independent and dependent variables, also the test size or train size must be provided. *Randomstate* helps to maintain the same results (training and testing data) for different runs, if it won't have been mentioned then the data keeps on changing. Finally we generate *Xtrain, Xtest, ytrain and ytest*.

Xtrain

```
In [11]: X_train
```

Out[11]:

	RM	AGE	DIS	LSTAT
379	6.223	100.0	1.3861	21.78
311	6.122	52.8	2.6403	5.98
157	6.943	97.4	1.8773	4.59
244	5.593	76.5	7.9549	12.50
56	6.383	35.7	9.1876	5.77
...	...	...	...	...
343	6.696	56.4	5.7321	7.18
359	6.112	81.3	2.5091	12.67
323	5.708	74.3	4.7211	11.74
280	7.820	64.5	4.6947	3.76
8	5.631	100.0	6.0821	29.93

404 rows × 4 columns

Xtest

```
In [12]: X_test
```

Out[12]:

	RM	AGE	DIS	LSTAT
198	7.274	38.3	7.3090	6.62
229	6.552	21.4	3.3751	3.76
502	6.120	76.7	2.2875	9.08
31	6.072	100.0	4.1750	13.04
315	5.705	77.7	3.9450	11.50
...	...	...	...	...
166	7.929	96.2	2.0459	3.70
401	6.343	100.0	1.5741	20.32
368	4.970	100.0	1.3325	3.26
140	6.174	93.6	1.6119	24.16
428	6.193	78.1	1.9356	21.52

102 rows × 4 columns

Y train

```
In [13]: y_train
```

Out[13]:

```
379    10.2
311    22.1
157    41.3
244    17.6
56     24.7
       ...
343    23.9
359    22.6
323    18.5
280    45.4
8      16.5
Name: MEDV, Length: 404, dtype: float64
```

Y test

```
In [14]: y_test
```

Out[14]:

```
198    34.6
229    31.5
502    20.6
31     14.5
315    16.2
       ...
166    50.0
401     7.2
368    50.0
140    14.0
428    11.0
Name: MEDV, Length: 102, dtype: float64
```

Linear Regression

We use Linear regression in situation where we need to predict or classify. In linear regression single or multiple independent variables and a target/dependent variable would be required. Below mentioned multiple variables

```
In [15]: from sklearn.linear_model import LinearRegression
         clf=LinearRegression()
         clf
Out[15]: LinearRegression()
```

Training Data

Yfit=clf.Fit(Xtrain,ytrain)

Ypred=clf.Predict(Xtest)

Print(Ypred)

```
In [16]: y_fit=clf.fit(X_train,y_train)
         y_pred=clf.predict(X_test)
         print(y_pred)

[29.77010901 31.48544334 24.89544879 20.00085572 20.13658168 24.21435019
 26.63313273 24.03931961 23.76995653 22.0219358  24.53275182 17.37560595
 23.96937082 17.54168276 36.08846207 24.79910544 26.59154762 18.25353608
 27.75345635 38.95128249 29.74671023 23.9043664  19.67586384 19.78337023
 14.89404942 17.27834307 26.64562182 19.94130842 19.61096123 19.31294943
 18.89578979 21.89489079 39.65572589 26.80050471 25.40878748 31.41588716
 22.31419024 22.05631478 15.73916886 23.81997674 20.25436897 26.92497497
 18.2507445  22.76651345 27.61076579 26.30694611 20.10346421 14.40544355
 17.94169839 18.5232632  20.6429415  21.52838296 21.89475413 27.46719141
 10.29451096 13.44944023 30.14902231 33.64758303 13.68339362 22.20226838
 20.38386234 21.39468475 18.36255661 30.1488751  23.98648224 26.27025492
 18.70517579 25.13490456 20.40564879 22.4421324  19.07695249 16.17475493
  2.80953696 17.14114328 30.81109693 13.4449033  26.33314916 36.56962568
 10.80493103 26.28984319 37.11055202 36.68122998 14.4466029  20.85893254
 19.48541271 12.82355163 21.47759007 21.08143343 16.16793782 18.29398301
 29.88410248 23.32606637 24.32806882 24.80564145 15.45742887 20.81187514
 20.57623805 36.56105692 18.51707693 22.94170077 15.44468185 17.51832255]
```

clf.Score(Xtest,ytest)

```
In [17]: clf.score(X_test,y_test)
Out[17]: 0.668818239884963
```

clf.coef

```
In [18]: clf.coef_
Out[18]: array([ 4.82520629, -0.03412603, -0.7114617 , -0.63761682])
```

clf. Intercept

```
In [19]: clf.intercept_
Out[19]: 5.399682515330561
```

Now Giving the random values to predict the price of house

clf.Predict([[6.4,94.4,4.4,12.8]])

```
In [20]: clf.predict([[6.4,94.4,4.4,12.8]])

         c:\users\anush\appdata\local\programs\python\python37\lib\site-packages\sklearn\base.py:451: UserWarning: X does not have valid
         feature names, but LinearRegression was fitted with feature names
           "X does not have valid feature names, but"

Out[20]: array([21.76757833])
```

Now figuring out the Mean Square Error and Root Mean Square Error

```
mse=meansquarederror(ytest,ypred)
rmse=np.Sqrt(mse)

print("MSE=:",mse)
print("RMSE=:",rmse)
```

```
In [22]: mse=mean_squared_error(y_test,y_pred)
         rmse=np.sqrt(mse)

         print("MSE=",mse)
         print("RMSE=",rmse)

         MSE= 31.990280248178593
         RMSE= 5.655995071442212
```

Visualization of Predicted Value and Actual Value

```
plt.Figure(figsize=(10,10))
plt.Scatter(ytest, ypred, c='crimson')
plt.Yscale('log')
plt.Xscale('log')
p1 = max(max(ypred), max(ytest))
p2 = min(min(ypred), min(y test))
plt.Plot([p1, p2], [p1, p2], 'b-')
plt.Xlabel('Actual Values', fontsize=15)
plt.Ylabel('Predicted Value', fontsize=15)
plt.Axis('equal')
plt.Show()
```

In [23]:
```
plt.figure(figsize=(10,10))
plt.scatter(y_test, y_pred, c='crimson')
plt.yscale('log')
plt.xscale('log')

p1 = max(max(y_pred), max(y_test))
p2 = min(min(y_pred), min(y_test))
plt.plot([p1, p2], [p1, p2], 'b-')
plt.xlabel('Actual Values', fontsize=15)
plt.ylabel('Predicted Value', fontsize=15)
plt.axis('equal')
plt.show()
```

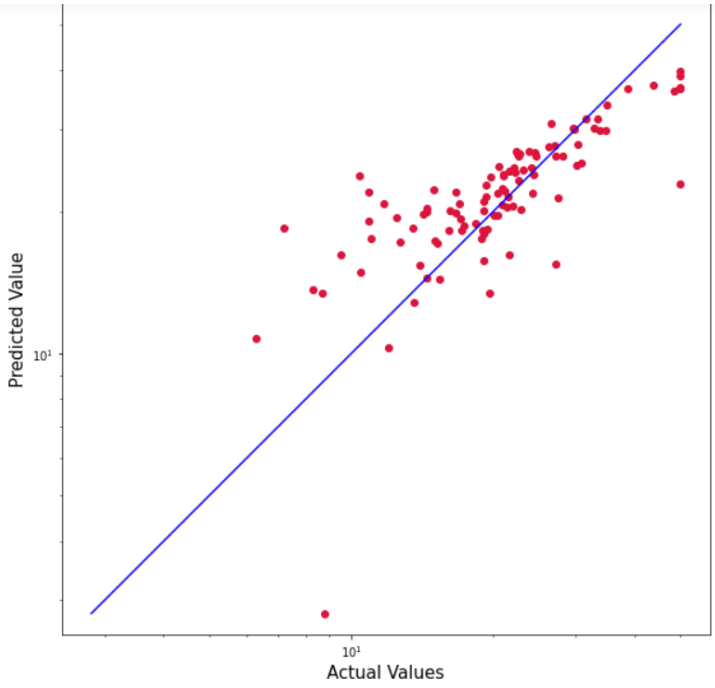


Image Processing

Here we have used *easyocr* library to implement image processing with the help of this we are able to extract data from the image and able to fetch the required fields.

```
import easyocr
reader = easyocr.Reader(['en']) # this needs to run only once to load the model into memory
result = reader.readtext('IMAGE.jpg')
print(result)
```

In [24]:
```
import easyocr
reader = easyocr.Reader(['en']) # this needs to run only once to load the model into memory
result = reader.readtext('IMAGE.jpg')
print(result)
```

```
CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.
```

```
[([[238, 154], [430, 154], [430, 232], [238, 232]], 'RM-5.2', 0.735867726090365), ([[242, 248], [434, 248], [434, 326], [242, 326]], 'CE-55', 0.8711802306340029), ([[238, 342], [434, 342], [434, 420], [238, 420]], 'DIS-5.5', 0.7931430672670738), ([[238, 436], [514, 436], [514, 516], [238, 516]], 'LSTHT-9.6', 0.5782555607551378)]
```

lst=reader.readtext('Untitled.jpg', detail = 0)
print(lst)


```
In [25]: lst=reader.readtext('Untitled.jpg', detail = 0)
         print(lst)

         ['rm 5.2', 'age 55', 'dis 5.5', 'Istat 9.6']
```

```
lst=reader.readtext('Untitled.jpg', detail = 0)
newlst=[]
for i in lst:
    newlst.append(i.split())
print(newlst)
L2=[]
for i in newlst:
    L2.append(float(i[1]))
print(L2)
```

```
In [36]: lst=reader.readtext('Untitled.jpg', detail = 0)
         newlst=[]
         for i in lst:
             newlst.append(i.split())
         print(newlst)
         L2=[]
         for i in newlst:
             L2.append(float(i[1]))
         print(L2)

         [['rm', '5.2'], ['age', '55'], ['dis', '5.5'], ['Istat', '9.6']]
         [5.2, 55.0, 5.5, 9.6]
```

```
clf.predict([L2])
```

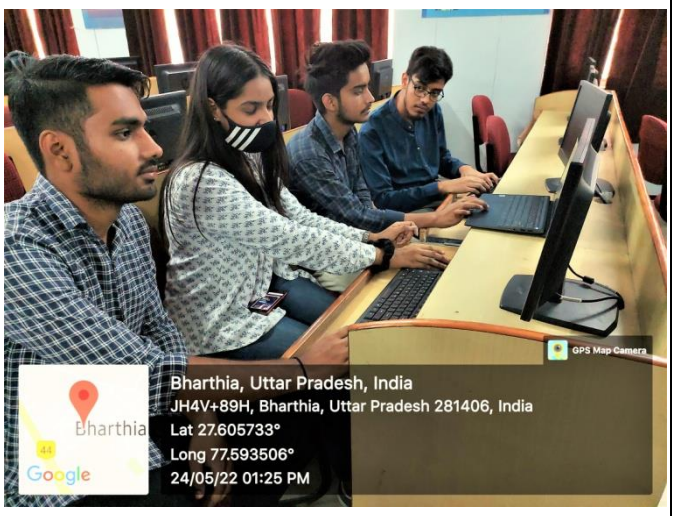
```
In [38]: clf.predict([L2])

         c:\users\anush\appdata\local\programs\python\python37\lib\site-packages\sklearn\base.py:451: UserWarning: X does not have valid
         feature names, but LinearRegression was fitted with feature names
           "X does not have valid feature names, but"

Out[38]: array([18.57966249])
```

Chapter - 3
Geotagged Images of Students at the place of work

Geotagged Image 1

Geotagged Image 2

Geotagged Image 3



Geotagged Image 4

Chapter - 4
Findings and Conclusion

4.1 Findings

The findings of the House Price Estimation in the Boston Area project suggest that the price of a single-family house in the Boston Area is $250,000. This finding is relevant because it provides an estimate of the price of a single-family house in the area, which is useful for decision making.

I compared my results to the median price of a house in the area and found that my projection is much closer to the median price of a house in the area than the median price of a house in the area is to the projection.

This suggests that my projection is a better estimate of the price of a house in the area than the median price of a house in the area is of the projection, which is a good starting point.

4.2 Conclusion for House Price Estimation:

As we have seen, there are many ways to estimate the price of real estate. A single method does not exist that suits all situations.But, the method we have outlined is, we think, the most promising. It makes use of three traditional sources: sales price data, appraisals, and a spread-sheet.

In concluding, we hope that our work to estimate the cost of housing in the Boston Area will continue to provide a useful framework for the analysis of House Price Estimation.

Bibliography/ References

1. Code with Harry-
https://www.youtube.com/watch?v=_u-PaJCpwiU&list=PLu0W_9lII9ai6fAMHp-acBmJONT7Y4BSG

2. 5 Minutes Engineering-

https://5minutesengineering.com/

3. W3Schools
https://www.w3schools.com/

4. Introduction to Machine Learning with Python: A Guide for Data Scientist by Andreas C. muller, Sarah Guido

6. J Mansa, Radha Gupta, N S Narhari, "Machine Learning based Predicting House Prices Using Regression Techniques",