MACHINE LEARNING HOUSE PRICE PREDICTION

CONTENT

LOVELY PROFESSIONAL UNIVERSITY

SUBMITTED BY

Name-Vishal Lazrus

Registration no-12004235

Section-KM096

Roll no-6

Submitted to Md.Imran Hussain

Introduction

- In current time, there are hundreds and thousands of houses being bought and sold every day. Now, there are two different kinds of buyers, one who is very much experienced and the other who has just begun buying a house or better to say, an absolute beginner. Now, it becomes very difficult for a newcomer to know the best price of a particular house due to lack of experience and because of this, many buyers are being cheated by sellers every day.

- In order to solve this problem, we came up with the idea of building a machine learning model which will help newcomers predict or know the best price of a particular house. Our model not only helps predict the price of a house but also helps the buyer to have an idea about the right time to buy the house.

LIBRARIES USED

- NUMPY

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

- At the core of the NumPy package, is the ndarray object. This encapsulates ndimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences: NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of a ndarray will create a new array and delete the original. The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different-sized elements. NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.

- A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today's scientific/mathematical Pythonbased software, just knowing how to use Python's built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

PANDAS

Pandas is a Python library used for working with data sets.

It has functions for analyzing, cleaning, exploring, and manipulating data.

The name "Pandas" has a reference to both "Panel Data", and "Python Data

Analysis" and was created by Wes McKinney in 2008.

Pandas allow us to analyze big data and make conclusions based on statistical theories.

Pandas can clean messy data sets, and make them readable and relevant.

Relevant data is very important in data science.

Python with Pandas is used in a wide range of fields including academic and

commercial domains including finance, economics, statistics, analytics, etc.

KEY FEATURES OF PANDAS

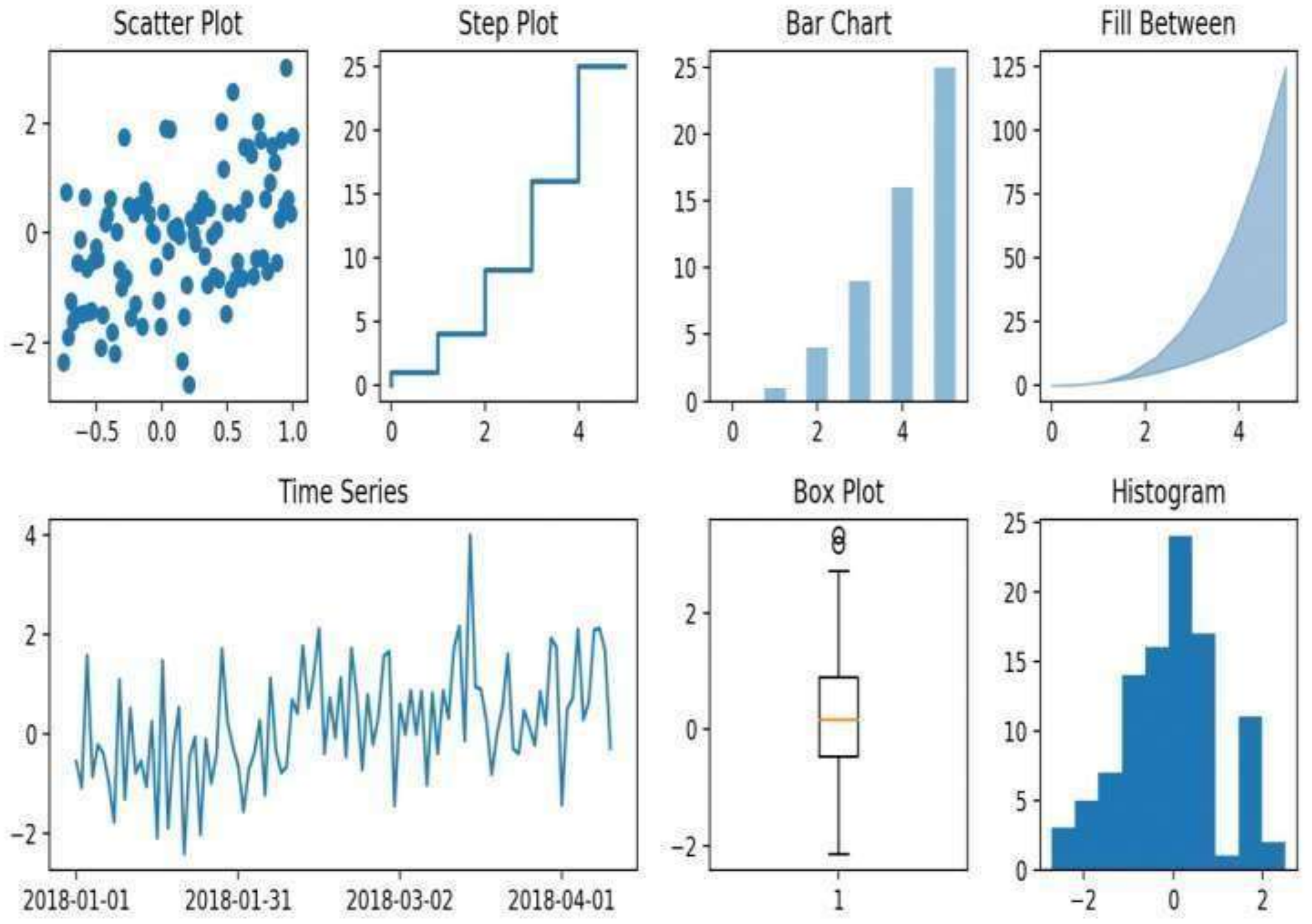
•Fast and efficient DataFrame object with default and customized indexing.

- Tools for loading data into in-memory data objects from different file
- formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, indexing, and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

MATPLOTLIB

- Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

- One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.



SEABORN

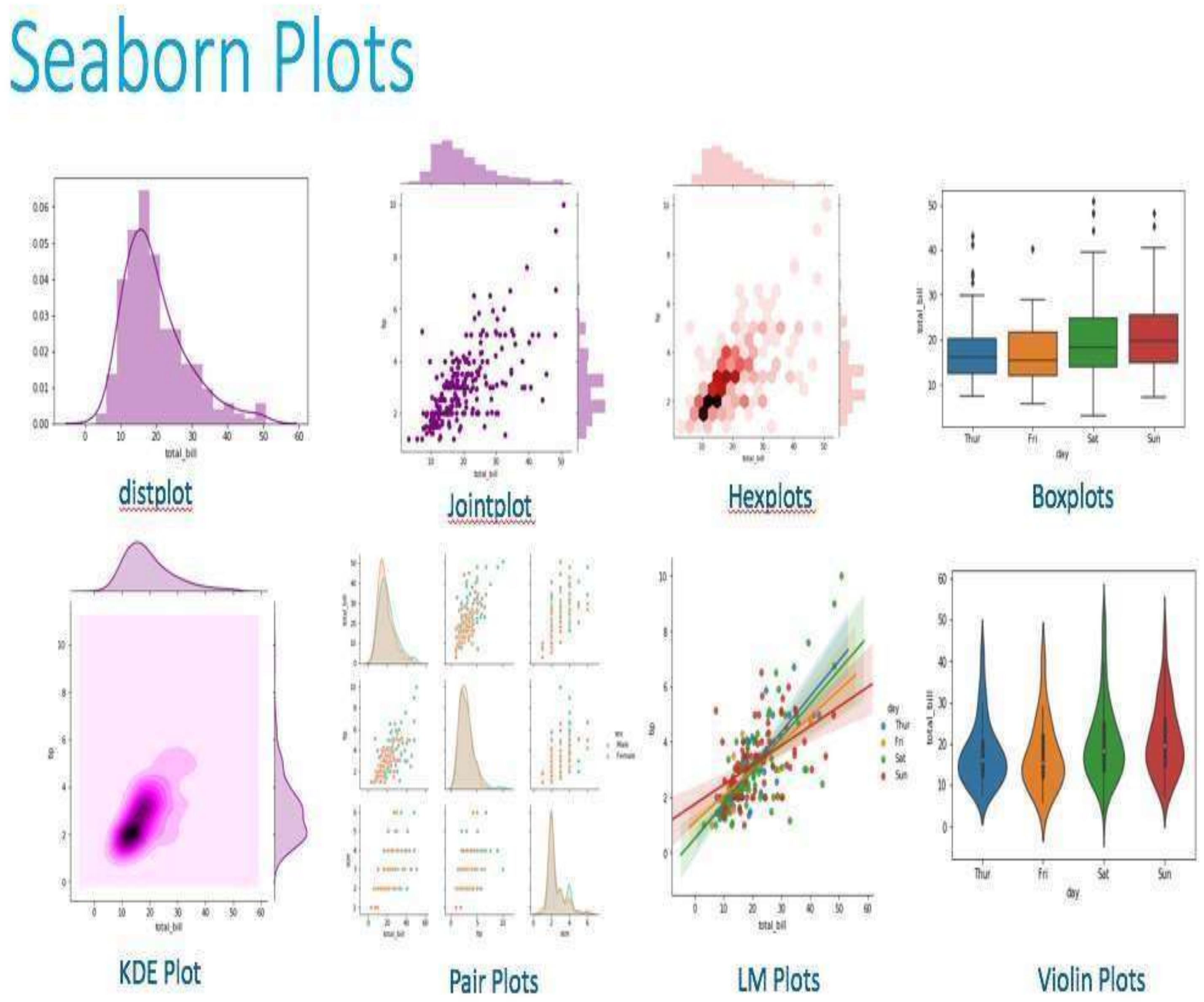
Seaborn is an amazing visualization library for statistical graphics plotting in Python.

It provides beautiful default styles and color palettes to make statistical plots more attractive.

It is built on the top of matplotlib library and also closely integrated to the data structures from pandas.

Seaborn aims to make visualization the central part of exploring and understanding data.

It provides dataset-oriented APIs, so that we can switch between different visual representations for same variables for better understanding of dataset.



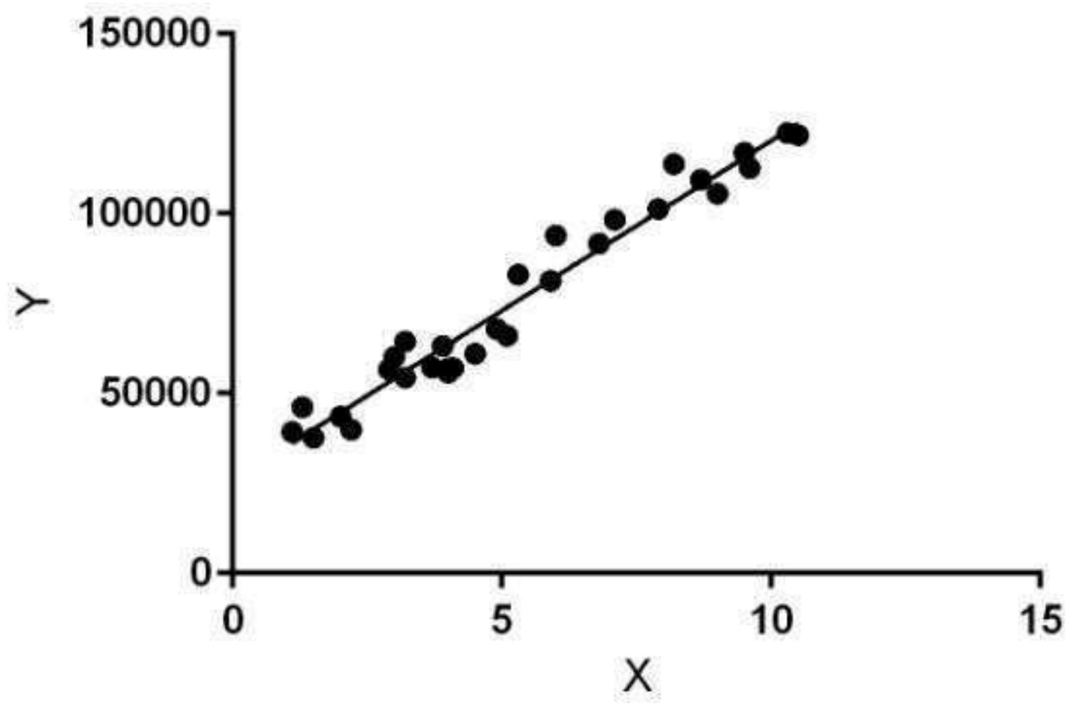
SCIKIT-LEARN

Scikit-learn is an open-source machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data preprocessing, model selection and evaluation, and many other utilities. Scikit-learn provides dozens of built-in machine learning algorithms and models, called estimators.

- Rather than focusing on loading, manipulating, and summarizing data, the ScikitLearn library is focused on modeling the data.
- Some of the most popular groups of models provided by Sklearn are as follows − Supervised Learning algorithms − Almost all the popular supervised learning algorithms, like Linear Regression, Support Vector Machine (SVM), Decision Tree, etc., are part of scikit-learn.
- Unsupervised Learning algorithms − On the other hand, it also has all the popular unsupervised learning algorithms from clustering, factor analysis, PCA (Principal Component Analysis) to unsupervised neural networks.
- Clustering − This model is used for grouping unlabeled data. Cross-Validation − It is used to check the accuracy of supervised models on unseen data. Dimensionality Reduction − It is used for reducing the number of attributes in data which can be further used for summarization, visualization, and feature selection.
- Ensemble methods − As the name suggests, it is used for combining the predictions of multiple supervised models. Feature extraction − It is used to extract the features from data to define the attributes in image and text data. Feature selection It is used to identify useful attributes to create supervised models.

LINEAR REGRESSION

- Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on the kind of relationship between dependent and independent variables they are considering, and the number of independent variables getting used.



Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

Hypothesis function for Linear Regression :

$$y = \theta_1 + \theta_2 . x$$

While training the model we are given :

x: input training data (univariate – one input variable(parameter))

y: labels to data (supervised learning)

When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ1 and θ2 values.

θ1: intercept

θ2: coefficient of x

Once we find the best θ1 and θ2 values, we get the best fit line. So when we are finally using our model for prediction, it will predict the value of y for the input value of x.

How to update θ1 and θ2 values to get the best fit line?

Cost Function (J):

By achieving the best-fit regression line, the model aims to predict y value such that the error difference between predicted value and true value is minimum. So, it is very important to update the θ1 and θ2 values, to reach the best value that minimize the error between predicted y value (pred) and true y value (y).

$$minimize \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2 \qquad J = \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

Cost function(J) of Linear Regression is the Root Mean Squared Error (RMSE) between predicted y value (pred) and true y value (y).

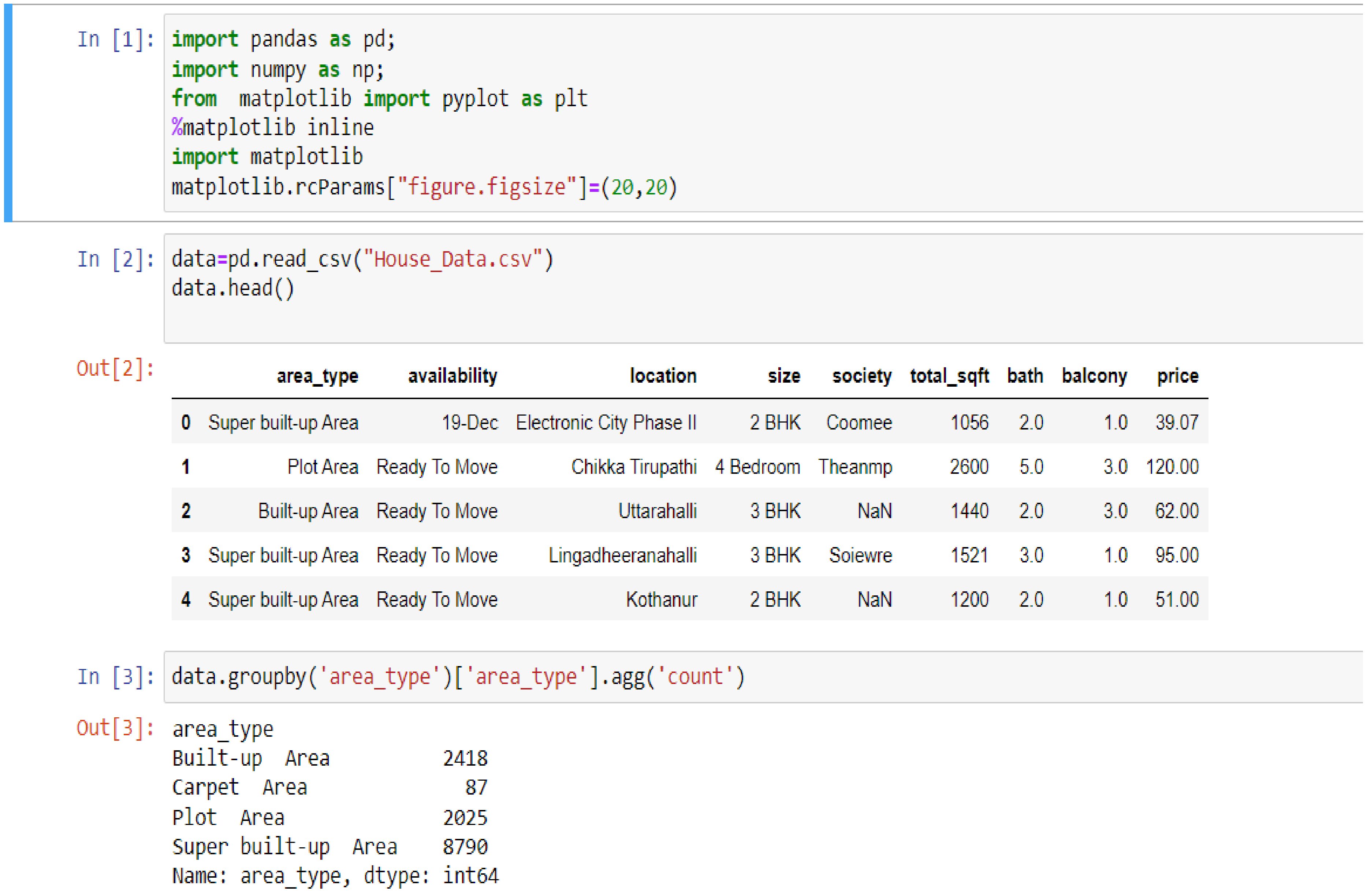
Gradient Descent:

To update θ1 and θ2 values in order to reduce Cost function (minimizing RMSE value) and achieving the best fit line the model uses Gradient Descent. The idea is to start with random θ1 and θ2 values and then iteratively updating the values, reaching minimum cost.

PYTHON LIBRARIES USED

```
import pandas as pd;

import numpy as np;

from  matplotlib import pyplot as plt

%matplotlib inline

import matplotlib

matplotlib.rcParams["figure.figsize"]=(20,20)

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.model_selection import GridSearchCV
```

DATA SAMPLE



In [1]:
```
import pandas as pd;
import numpy as np;
from  matplotlib import pyplot as plt
%matplotlib inline
import matplotlib
matplotlib.rcParams["figure.figsize"]=(20,20)
```

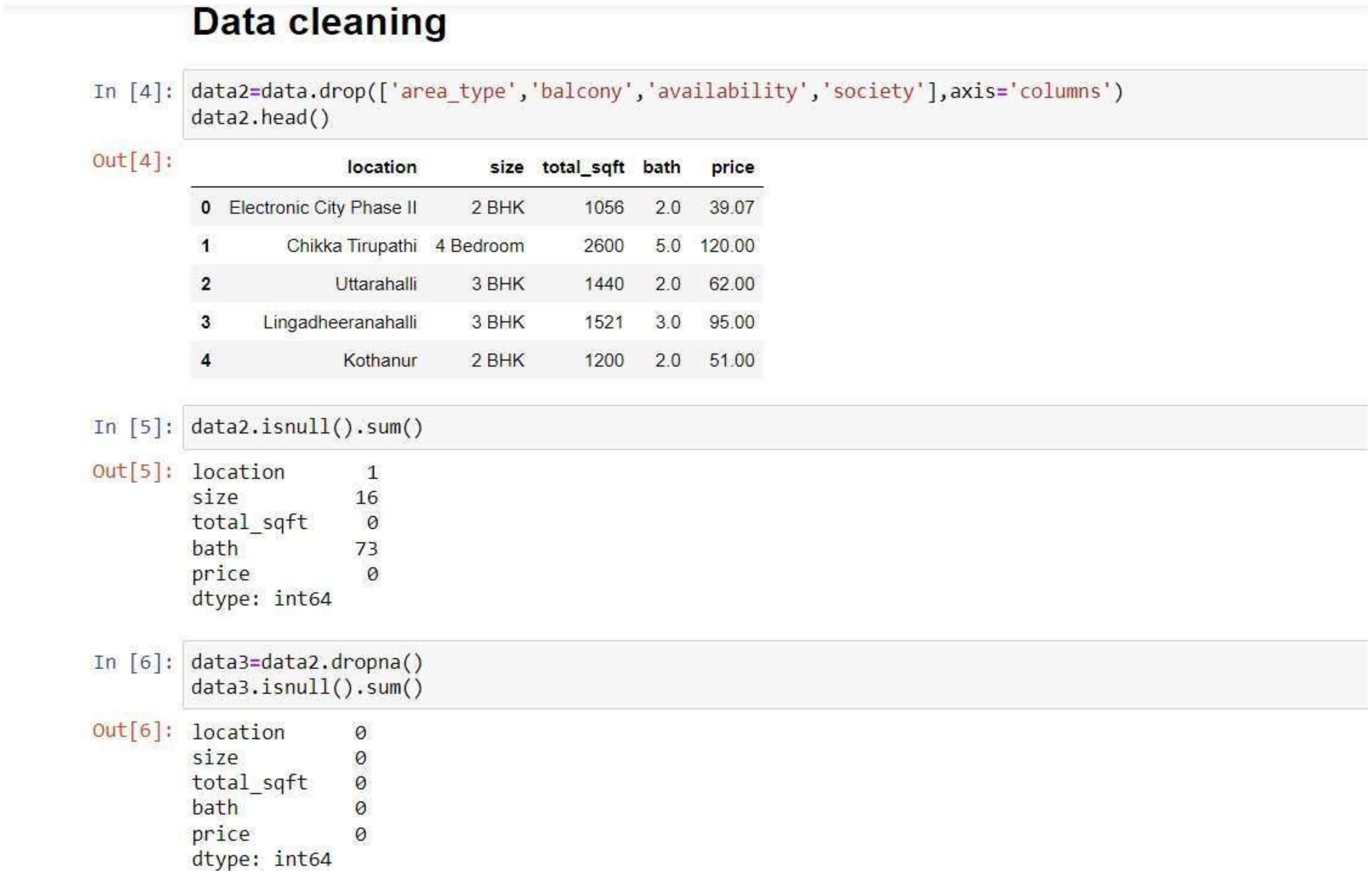
In [2]:
```
data=pd.read_csv("House_Data.csv")
data.head()
```

Out[2]:

	area_type	availability	location	size	society	total_sqft	bath	balcony	price
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056	2.0	1.0	39.07
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600	5.0	3.0	120.00
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440	2.0	3.0	62.00
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521	3.0	1.0	95.00
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200	2.0	1.0	51.00

In [3]:
```
data.groupby('area_type')['area_type'].agg('count')
```

Out[3]:
```
area_type
Built-up  Area          2418
Carpet  Area              87
Plot  Area              2025
Super built-up  Area    8790
Name: area_type, dtype: int64
```



Data cleaning

In [4]:
```python
data2=data.drop(['area_type','balcony','availability','society'],axis='columns')
data2.head()
```

Out[4]:

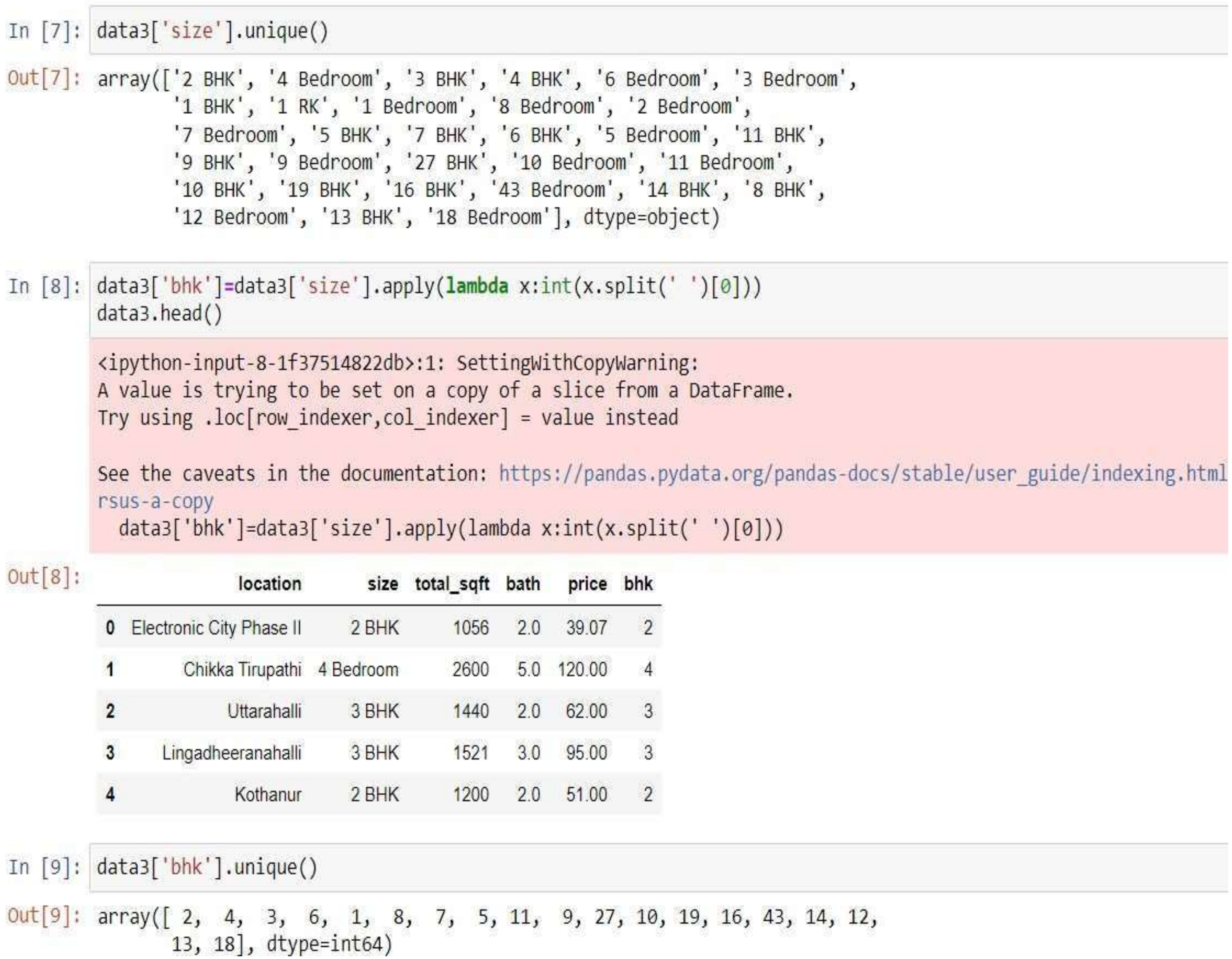
	location	size	total_sqft	bath	price
0	Electronic City Phase II	2 BHK	1056	2.0	39.07
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00
2	Uttarahalli	3 BHK	1440	2.0	62.00
3	Lingadheeranahalli	3 BHK	1521	3.0	95.00
4	Kothanur	2 BHK	1200	2.0	51.00

In [5]:
```python
data2.isnull().sum()
```

Out[5]:
```
location       1
size          16
total_sqft     0
bath          73
price          0
dtype: int64
```

In [6]:
```python
data3=data2.dropna()
data3.isnull().sum()
```

Out[6]:
```
location      0
size          0
total_sqft    0
bath          0
price         0
dtype: int64
```



```
In [7]: data3['size'].unique()
```

```
Out[7]: array(['2 BHK', '4 Bedroom', '3 BHK', '4 BHK', '6 Bedroom', '3 Bedroom',
               '1 BHK', '1 RK', '1 Bedroom', '8 Bedroom', '2 Bedroom',
               '7 Bedroom', '5 BHK', '7 BHK', '6 BHK', '5 Bedroom', '11 BHK',
               '9 BHK', '9 Bedroom', '27 BHK', '10 Bedroom', '11 Bedroom',
               '10 BHK', '19 BHK', '16 BHK', '43 Bedroom', '14 BHK', '8 BHK',
               '12 Bedroom', '13 BHK', '18 Bedroom'], dtype=object)
```

```
In [8]: data3['bhk']=data3['size'].apply(lambda x:int(x.split(' ')[0]))
        data3.head()
```

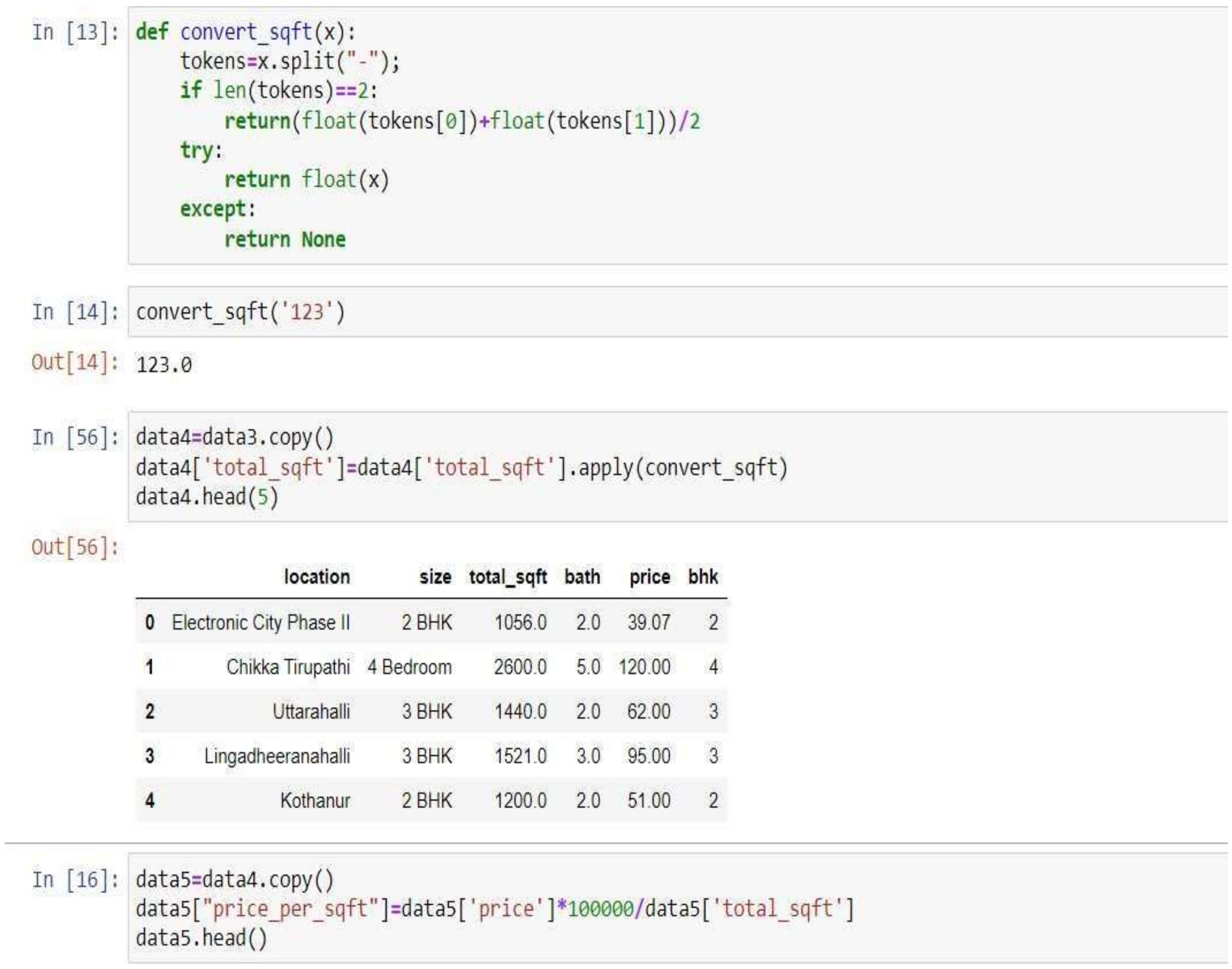
```
<ipython-input-8-1f37514822db>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
rsus-a-copy
  data3['bhk']=data3['size'].apply(lambda x:int(x.split(' ')[0]))
```

Out[8]:

	location	size	total_sqft	bath	price	bhk
0	Electronic City Phase II	2 BHK	1056	2.0	39.07	2
1	Chikka Tirupathi	4 Bedroom	2600	5.0	120.00	4
2	Uttarahalli	3 BHK	1440	2.0	62.00	3
3	Lingadheeranahalli	3 BHK	1521	3.0	95.00	3
4	Kothanur	2 BHK	1200	2.0	51.00	2

```
In [9]: data3['bhk'].unique()
```

```
Out[9]: array([ 2,  4,  3,  6,  1,  8,  7,  5, 11,  9, 27, 10, 19, 16, 43, 14, 12,
               13, 18], dtype=int64)
```



In [13]:
```python
def convert_sqft(x):
    tokens=x.split("-");
    if len(tokens)==2:
        return(float(tokens[0])+float(tokens[1]))/2
    try:
        return float(x)
    except:
        return None
```

In [14]:
```python
convert_sqft('123')
```

Out[14]:
```
123.0
```

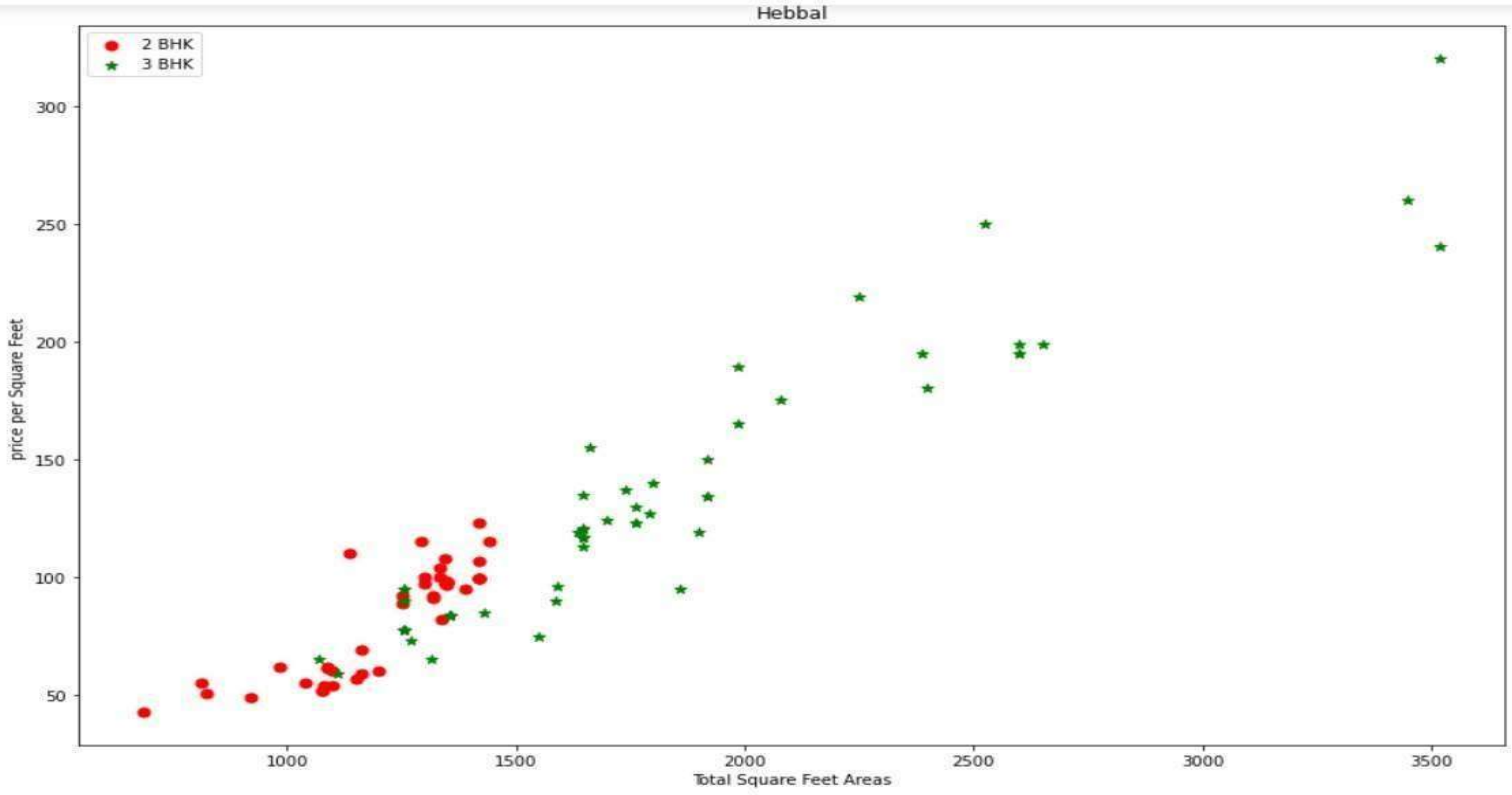
In [56]:
```python
data4=data3.copy()
data4['total_sqft']=data4['total_sqft'].apply(convert_sqft)
data4.head(5)
```

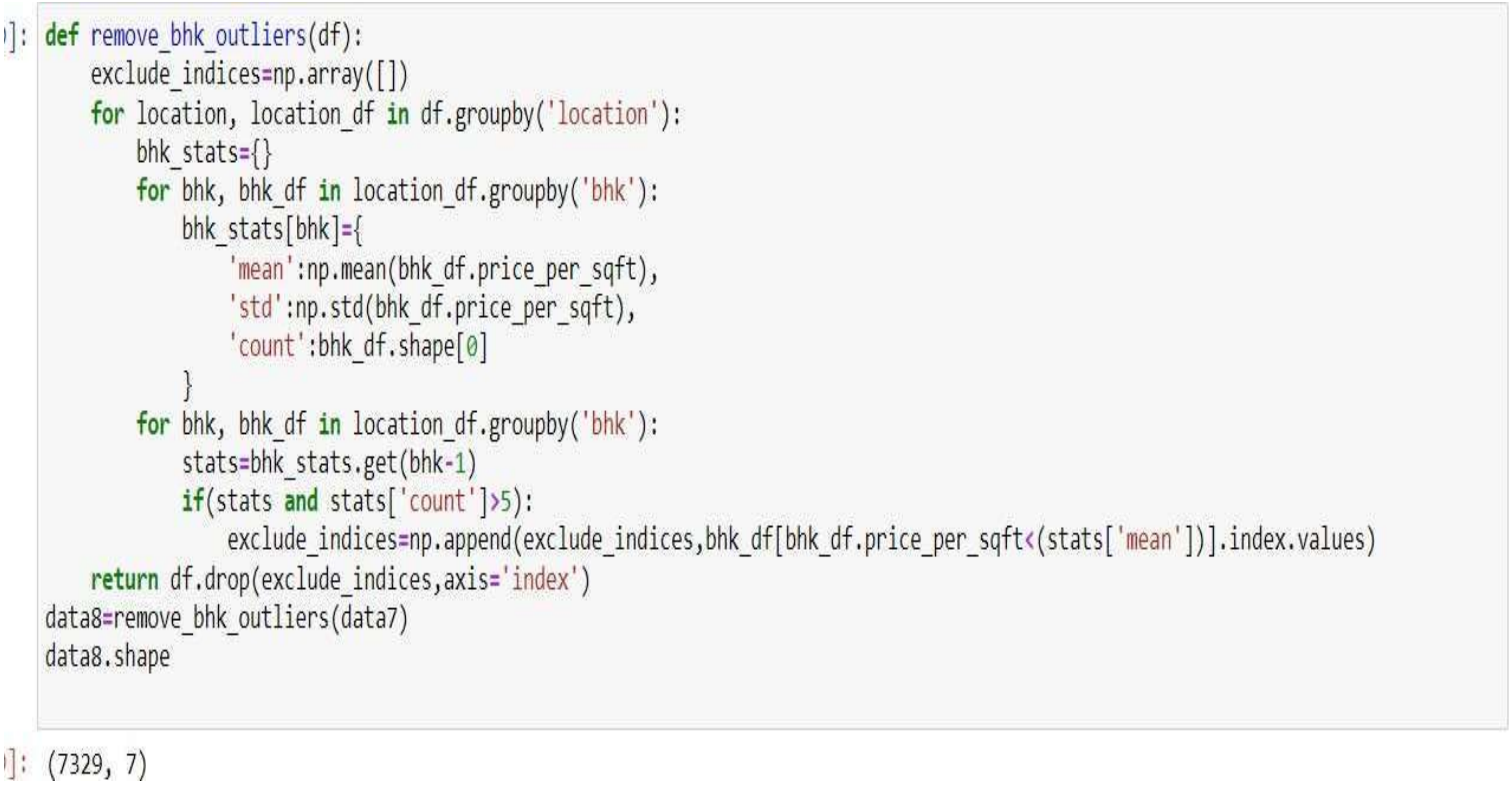
Out[56]:

	location	size	total_sqft	bath	price	bhk
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3
4	Kothanur	2 BHK	1200.0	2.0	51.00	2

In [16]:
```python
data5=data4.copy()
data5["price_per_sqft"]=data5['price']*100000/data5['total_sqft']
data5.head()
```

```
def plot(df,location):
    bhk2=df[(df.location==location) & (df.bhk==2)]
    bhk3=df[(df.location==location) & (df.bhk==3)]
    matplotlib.rcParams['figure.figsize']=(15,10)
    plt.scatter(bhk2.total_sqft,bhk2.price,color='red',label='2 BHK',s=50)
    plt.scatter(bhk3.total_sqft,bhk3.price,marker='*',color='green',label='3 BHK',s=50)
    plt.xlabel("Total Square Feet Areas")
    plt.ylabel("price per Square Feet ")
    plt.title(location)
    plt.legend()
plot(data7,"Hebbal")
```

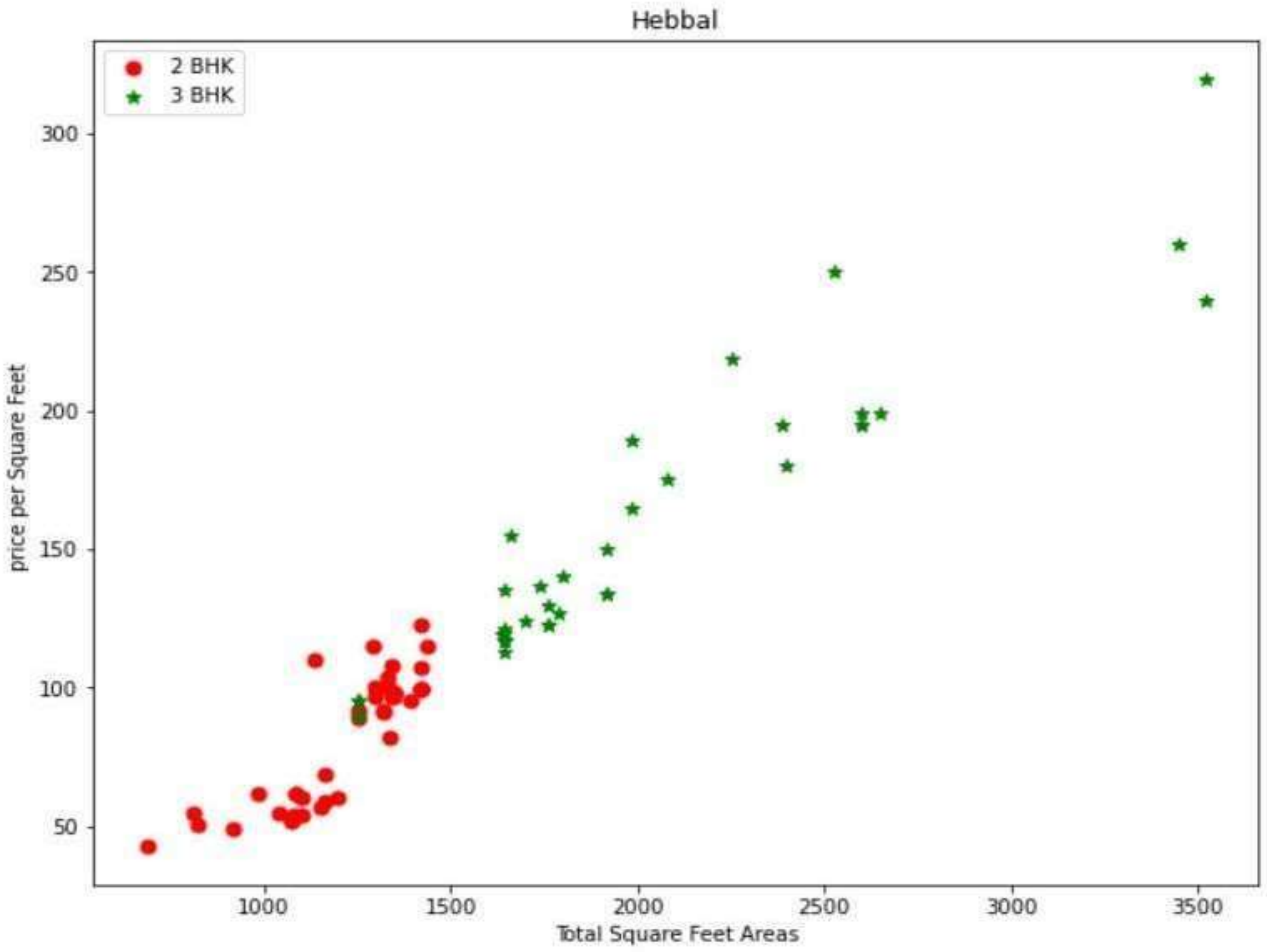


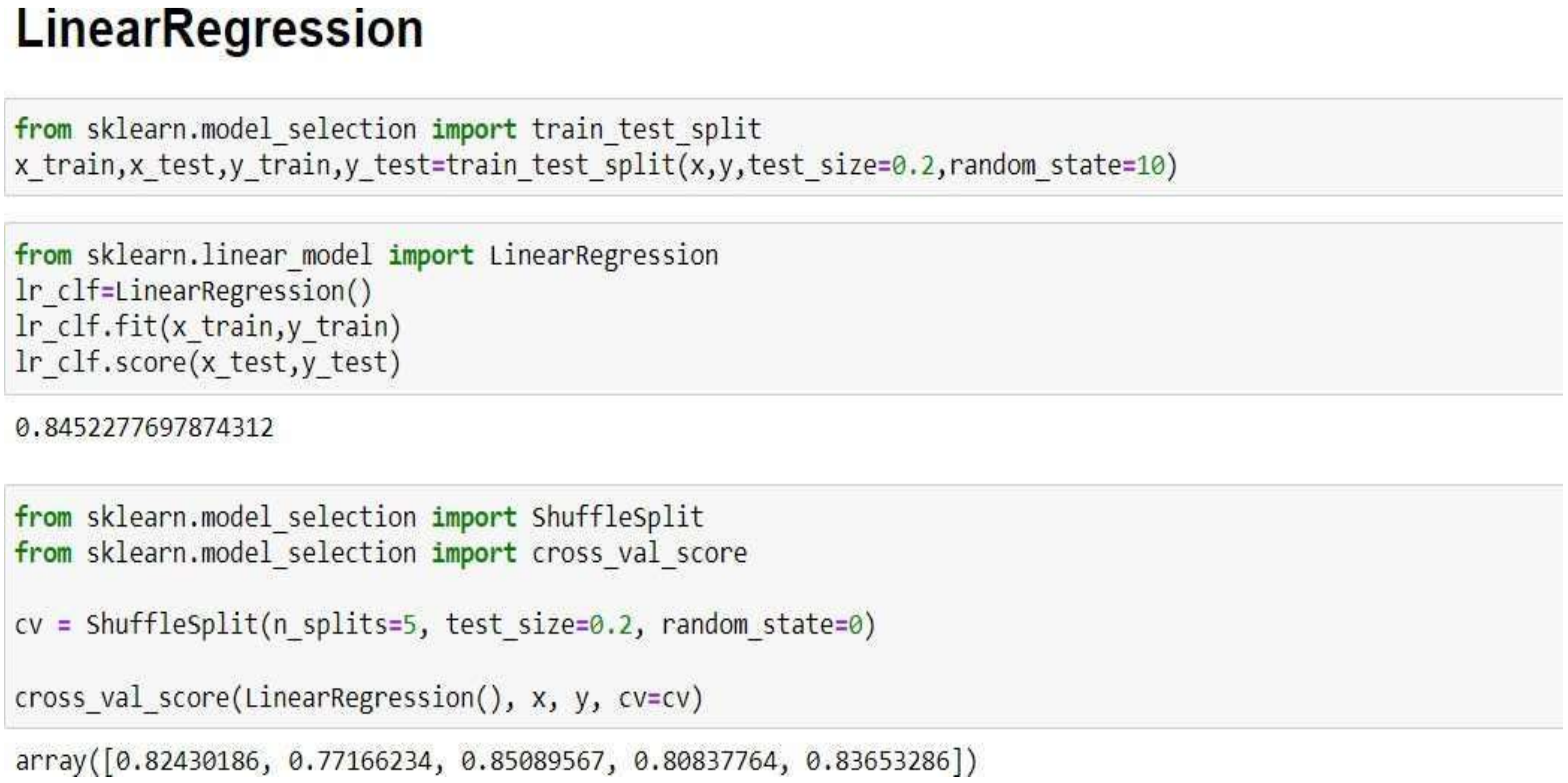


```
]: def remove_bhk_outliers(df):
       exclude_indices=np.array([])
       for location, location_df in df.groupby('location'):
           bhk_stats={}
           for bhk, bhk_df in location_df.groupby('bhk'):
               bhk_stats[bhk]={
                   'mean':np.mean(bhk_df.price_per_sqft),
                   'std':np.std(bhk_df.price_per_sqft),
                   'count':bhk_df.shape[0]
               }
           for bhk, bhk_df in location_df.groupby('bhk'):
               stats=bhk_stats.get(bhk-1)
               if(stats and stats['count']>5):
                   exclude_indices=np.append(exclude_indices,bhk_df[bhk_df.price_per_sqft<(stats['mean'])].index.values)
       return df.drop(exclude_indices,axis='index')
   data8=remove_bhk_outliers(data7)
   data8.shape
```

```
]: (7329, 7)
```

```
: plot(data8,"Hebbal")
```





LinearRegression

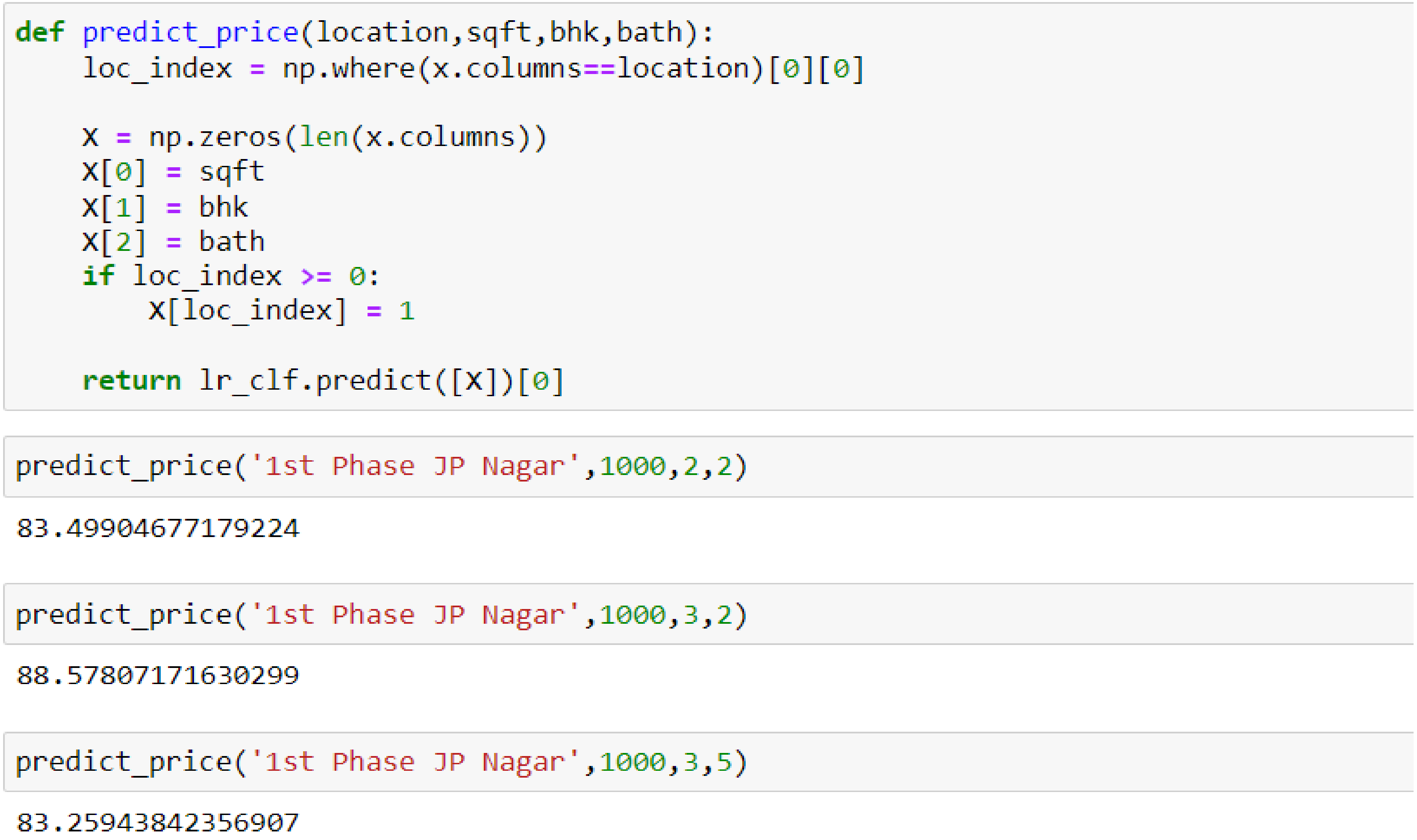
```python
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=10)
```

```python
from sklearn.linear_model import LinearRegression
lr_clf=LinearRegression()
lr_clf.fit(x_train,y_train)
lr_clf.score(x_test,y_test)
```

```
0.8452277697874312
```

```python
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score

cv = ShuffleSplit(n_splits=5, test_size=0.2, random_state=0)

cross_val_score(LinearRegression(), x, y, cv=cv)
```

```
array([0.82430186, 0.77166234, 0.85089567, 0.80837764, 0.83653286])
```



```python
def predict_price(location,sqft,bhk,bath):
    loc_index = np.where(x.columns==location)[0][0]

    X = np.zeros(len(x.columns))
    X[0] = sqft
    X[1] = bhk
    X[2] = bath
    if loc_index >= 0:
        X[loc_index] = 1

    return lr_clf.predict([X])[0]
```

```python
predict_price('1st Phase JP Nagar',1000,2,2)
```

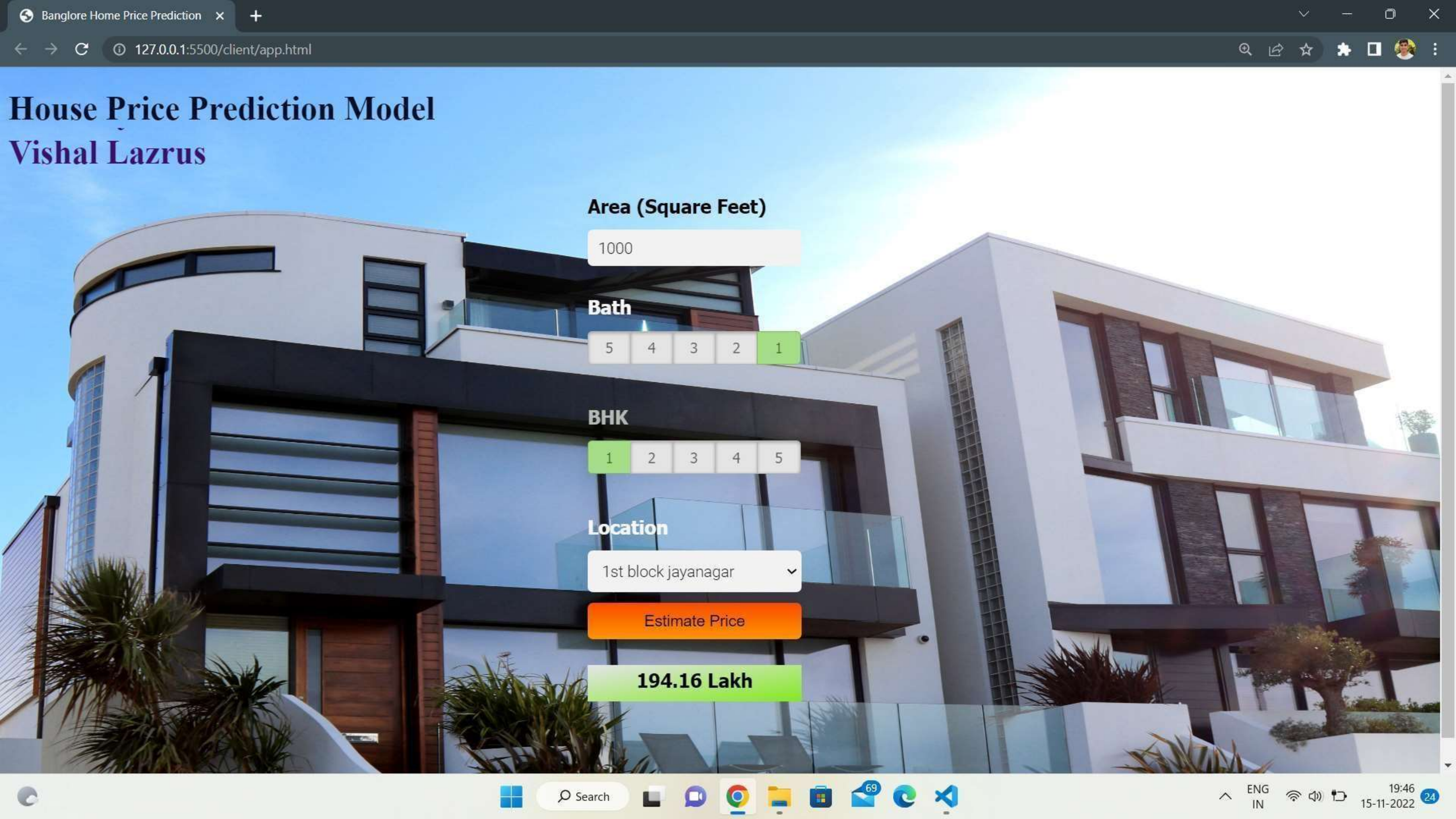
```
83.49904677179224
```

```python
predict_price('1st Phase JP Nagar',1000,3,2)
```

```
88.57807171630299
```

```python
predict_price('1st Phase JP Nagar',1000,3,5)
```

```
83.25943842356907
```



Banglore Home Price Prediction

127.0.0.1:5500/client/app.html

House Price Prediction Model

Vishal Lazrus

Area (Square Feet)

1000

Bath

5 4 3 2 [✓ 1]

BHK

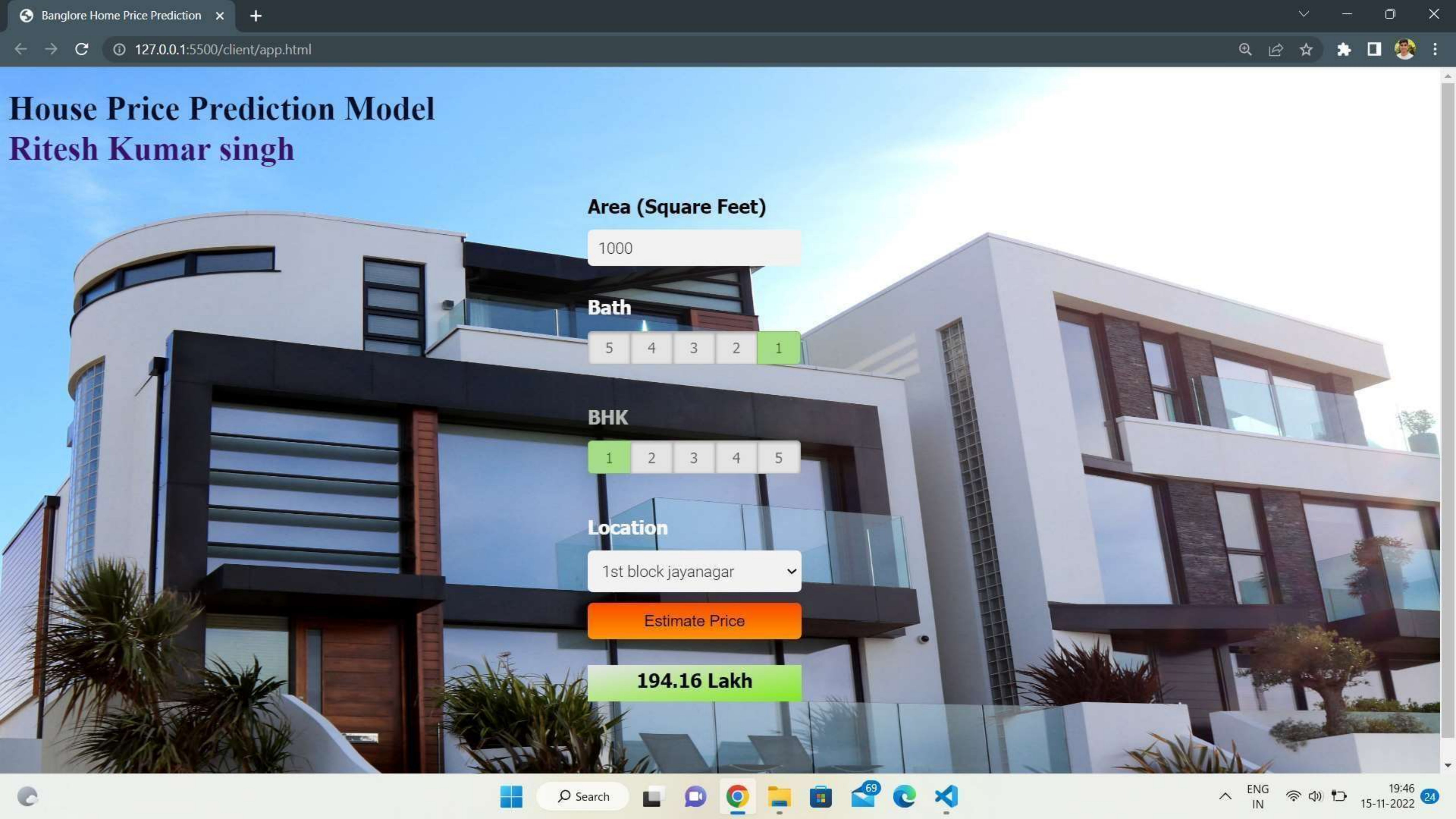
[✓ 1] 2 3 4 5

Location

1st block jayanagar

Estimate Price

194.16 Lakh



Banglore Home Price Prediction

127.0.0.1:5500/client/app.html

House Price Prediction Model

Ritesh Kumar singh

Area (Square Feet)

1000

Bath

5 4 3 2 [✓ 1]

BHK

[✓ 1] 2 3 4 5

Location

1st block jayanagar

Estimate Price

194.16 Lakh

REFERENCES AND LINKS

- Numpy documentation link: - https://numpy.org/doc/
- Pandas documentation link: - https://pandas.pydata.org/docs/
- Streamlit documentation link: - https://docs.streamlit.io/
- Scikit-learn documentation link: - https://scikitlearn.org/stable/tutorial/index.html

CONCLUSION

- Machine learning is a great field in modern time. We can rarely imagine any application made in present time without the usage of machine learning. With the help of machine learning, we can perform several activities a human being performs and also activities which is very difficult for a human being to perform. In recent years, we have seen those devices, computers trained with machine learning models beating human beings in several fields and in several games. At last, we can surely say that machine learning is the new face of future in the field of software development.

THANK YOU!