Enhanced approach of house cost prediction using Machine learning

N.K SINGH
Department of computer science ,BIT Mesra, nksingh27@gmail.com
ANURAG SINHA
anuragsinha257@gmail.com
Department of computer science and IT, UG SCHOLAR Amity University Jharkhand, Ranchi, Jharkhand (India)
Shubham Singh
DepartmentofcomputerscienceandEngineering,GalgotiasUniversity,GreaterNoida,U.P.
shubham932singh@gmail.com
Yuvraj Singh Rajawat,
Yuvraj_singh2.scsebtech@galgotiasuniversity.edu.in, School of computer science engineering, Galgotias university ,greater noida(India
PIYUSH PUNIA ,
piyush_punia.scsebtech@galgotiasuniversity.edu.in, School of computer science engineering, Galgotias university ,greater noida(India)

ABSTRACT

The common dilemma that haunts the average home buyer is what is the right property in the right market and at the right price point? It is very difficult for a home buyer to choose and without scientific research they cannot make an informed decision. The decision made by the customers is based on their short- term want while they overlook their long-term wants or they are confused between the end-user necessities. Properties prices are the one side of the face of economy, well priced properties are lure for both sellers and buyers. Well priced Properties are the best investment of the individuals. With the development in a city, hundreds of property dealings happen every day due to which property prices in cities are keep varying, making it difficult to predict the most accurate and exact price of the property at a time. This also creating a bad competition among property dealers because they used to manually calculate the prices, which all always results in irrelevant prices. Also make either buyer or seller disappointed.

To overcome this situation, undoubtedly there is a need for a Machine Learning model that can do better research on the subject which can help both the developers and the home buyers. The ML model can predict the property prices with the help of present data. It also can analyze the home buyer's preferences for a location and the value that hold for him we have to make REGRESSSION model which can predict best outcome prices of the property.

For this we have to go through many phases like Data Extraction, Data Cleaning, Outlier Detection, Training and Testing of Model etc.

This model will satisfy the best need of Buyers and Sellers. And give a scientific reason for their properties prices.

KEYWORDS: Real Estate Property, Price Prediction, Machine Learning Model, Supervised Learning.

CHAPTER-1
Introduction

In Property project, we will develop a supervised Machine Learning Machine learning model, our project based on predictive power of the Machine learning Model, which will be trained and tested on the aggregated dataset of Property Prices of a particular city.
We will use different regression model to check good accuracy and once if we get good fit, we will use it to forecast the monetary value of the Properties. This Modelwill be very helpful to the both Property sellers and Buyers, also this will give a scientific idea for the Property Prices, which is more accurate than the manual prediction method. In general, real estate may be required to provide a valuation of the land. Many various players in the commercial center, such as land agents, appraisers, assessors, mortarboard lenders, and others, perform a quantitative assessment of profit. Brokers, developers, and gurus are all types of people who work in the real estate industry. Reserve managers, lenders, and others are also included. The value of your company will increase. That requisition will be used to evaluate you. In addition to claiming valuation systems, there are approaches that reflect the nature of the property and the circumstances in which it is delivered. The Property may well be on the road to exchange in the open market under a variety of conditions and circumstances; nevertheless, many people are uninformedof the current situation and begin to lose money. Changes in property values would affect both the regular people and the wealthy.
Price prediction is required by the government in order to avert certain situations. Many methods have been utilized in price prediction, such as linear regression, andin this paper, I am attempting to forecast the future real estate price using machine learning techniques and previous research. To anticipate the house price, I used linear forest, lasso regression, and other algorithms with various tools. As a result,it would be beneficial for people to be aware of both current and future conditions in order to prevent making mistakes.

Formulation of Problem

To investigate the traditional and manual way of the price prediction for Real Estate Properties, we conducted a survey, experience of buyer and developers, then evaluated and analyzed their reviews. From insights we found large amount of the property buyers are mostly unsatisfied with their property prices and even some of the developers are also struggling in completion due to their irrelevant property prices. Also, Real state property buyers facing difficulties in selecting the good and genuine property dealers while selecting property and also they were facing problems because of brokers and scammers. Also, if you are new to town and want to buy or sell a house but you don't know about pricing in that locality and how much does a certain bhk house will cost. If you don't know such things then you can sell or buy house at different price. For this our Product will utilize the data of the properties, having information of the properties like locality, rooms etc. And perform some Scientific Machine learning statistical algorithmic approach to predict the closest price of the property with very high accuracy and low error, this will be beneficial for both property buyers to get the will full price with satisfaction and property developers in their completion.

Tool and Technology Used

•***Jupyter Notebook***-

It is an open-source web application that allows creating and sharing documentsthat contains live codes, equations, visualizations and narrative text.

In a single document, notebooks integrate computer code (such as Python, SQL, or R), the results of the code's execution, and rich text features (such as formatting, tables, figures, equations, links, etc.). The ability to provide commentary with yourcode is the main advantage of using notebooks. The error-prone practise of
copying and pasting analysis results into a different report can therefore be avoided. In the notebook, you only combine your analysis with the report text.

Three essential parts make up a Jupyter Notebook: cells, a runtime environment, and a file system.

The notebook's individual units are called cells, and they can contain either text orcomputer code:

- ➢ Images, links, equations, and narrative text are all included in text cells.A simple markup language called Markdown is used to write text cells.
- ➢ Code is written in and run from code cells. Code cell output will be shown exactly beneath the code cell.
- ➢ The code in the notebook is executed by the runtime environment. There are many languages that can be supported by the runtime environment, including Python, R, and SQL.
- ➢ You can upload, store, and download data files, code files, and analytic outputs using the file system.

•***Pandas Library***

Pandas is an open-source data analysis and manipulation tool that is fast, powerful,flexible, and user-friendly. It is built on Python and enables working with data sets efficiently. By utilizing Pandas, large data sets can be examined and statistical principles can be applied to derive meaningful conclusions..

Pandas provides the capability to organize and reorganize data sets, enhancing their readability and usefulness. It is constructed on the foundation of the Numpy package, which implies that Numpy is essentialfor executing operations in Pandas.

•***Seaborn Library***

Seaborn is a Python library that specializes in creating statistical graphics. It isbuilt on top of Matplotlib and works seamlessly with Pandas data structures.
Seaborn aids in data exploration and comprehension. Its plotting functions are designed to work with entire datasets stored in dataframes and arrays, automatically performing semantic mapping

and statistical aggregation to generate insightful plots. By providing a dataset-oriented and declarative API, Seaborn allows users to focus on the interpretation of plot elements rather than the intricacies of their implementation.

•***Scikit -Learn***

It Scikit-learn is a Python machine learning library that offers a range of features, including classification, regression, and clustering algorithms. It is well-documentedand easy to learn and use, making it suitable for both beginners and advanced researchers. As a high-level library, it enables the creation of predictive data modelswith minimal code, which can be applied to fit data. It seamlessly integrates with other Python libraries like Matplotlib for visualizations, NumPy for array vectorization, and Pandas for dataframes.

•***Matplotlib Library***

Matplotlib is a comprehensive Python library for creating static, animated, and interactive visualizations. It provides various plot types such as line, bar, scatter, and histogram, allowing easy interpretation of data through graphical representations. Visualization facilitates the analysis of large data sets by presenting information in a digestible format, enabling efficient decision-making.Matplotlib produces high-quality figures suitable for publication and can be utilized in Python scripts, IPython shells, web application servers, and various graphical user interface toolkits.

Visualizing data helps facilitate understanding and enhances the analytical process. Representing data through graphs enables efficient data analysis and informed decision-making. Matplotlib excels in producing high-quality figures that can be utilized in various environments, including Python scripts, Python/IPython shells, web application servers, and graphical user interface toolkits. Its versatility ensures compatibility across different platforms, providing interactive and visually appealing visualizations in a range of print and digital formats.

CHAPTER-2
Literature Survey

The In the decision-making process for buyers and investors, accurate forecasting of property prices is crucial. It supports budget allocation, aids in finding suitable property funding, and informs policy decisions. However, stakeholders often lack knowledge about statistical methods for predicting property prices using various factors such as location, amenities, and property characteristics. This paper aims to provide a scientific and statistical approach to predict real estate property prices. The fluctuation of housing prices reflects the current economic situation, raising concerns for both buyers and sellers. Factors like the number of bedrooms, bathrooms, and proximity to highways, schools, malls, and offices significantly impact housing prices. Manual prediction of property prices is challenging and lacks accuracy. Previous research by Sifei Lu (2017) proposed an advanced property prediction system using linear regression models. Aayush Varma (2018) suggested using neural networks in conjunction with linear algorithms, resulting in improved prediction accuracy. Aayush utilized linear regression, forest regression, and boosted regression algorithms, with the dataset tested on neural networks to determine the most appropriate prediction. Adyan Nur Alfiyatin (2017) introduced practical swarm optimization and regression analysis in their predictive model, incorporating factors such as land area, land price, and building price to predict property prices. The practical swarm optimization technique was employed to select influential variables.

Project Design

A. **Linear regression**:

Linear regression analysis is a statistical method utilized to forecast the value of a dependent variable by considering the value of one or more independent variables.The dependent variable is the one we aim to predict, while the independent variable(s) are used to make the prediction. By estimating the coefficients of a linear equation, the analysis identifies the most suitable relationship between the variables. It aims to minimize the disparities between the predicted values and the actual outputs, typically by fitting a straight line or surface. Utilizing the "least squares" approach, linear regression enables us to estimate the value of the dependent variable (X) based on the independent variable (Y) in the equation.

The linear regression model represents the relationship between the variables througha straight line with a specific slope. This model captures the association between the variables by minimizing the discrepancies between the predicted and actual values.

To illustrate this concept, refer to the accompanying image:

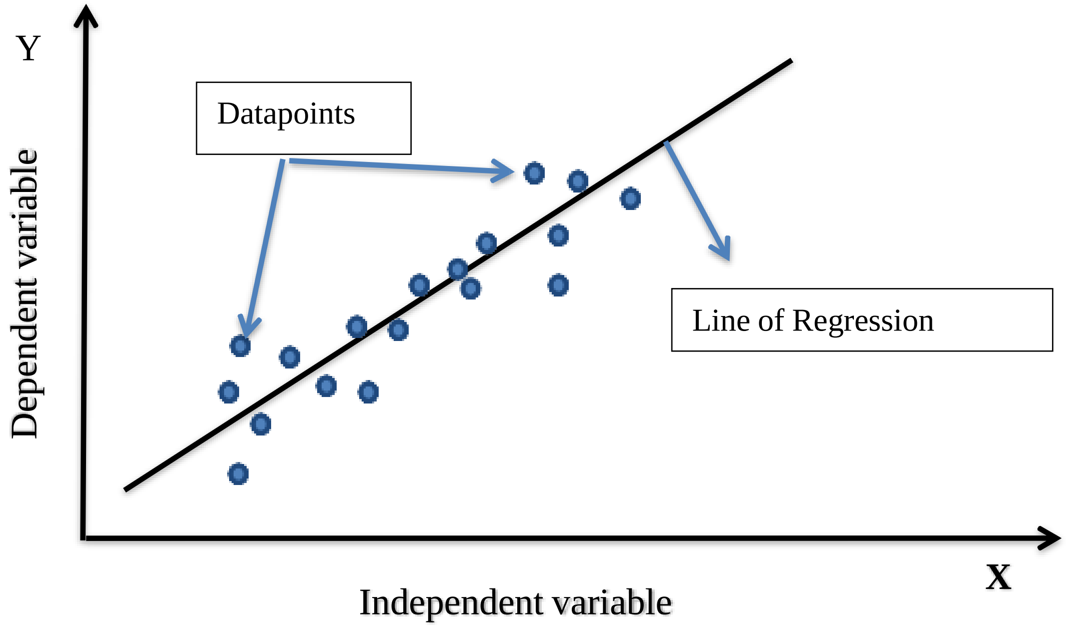


Fig: Linear Regression

B. **Lasso regression**:

The acronym "LASSO" denotes "Least Absolute Shrinkage and Selection Operator." It is a statistical method employed to regularize data models and perform feature selection. Lasso regression, a type of regularization technique, is utilized for enhanced prediction accuracy compared to traditional regression methods. This approach incorporates shrinkage, which involves shrinking data values towards a central point, typically the mean. By promoting simplicity and sparsity in models with fewer parameters, the lasso procedure proves advantageous, particularly for models exhibiting high levels of multicollinearity or when automating aspects of model selection, such as variable selection and parameter elimination.

Lasso Regression employs the L1 regularization technique, which will be discussedfurther in this article. It is particularly useful when dealing with datasets that contain a large number of features, as it automatically performs feature selection. Lasso Regression is an extension of linear regression, introducing a penalty term based on the sum of absolute values of the weights. This penalty leads to a reduction in the absolute values of the weights, often resulting in many weights becoming zeros. By applying regularization, Lasso Regression helps prevent overfitting and improves themodel's performance on diverse datasets. This regression technique is particularly suited for datasets exhibiting high levels of multicollinearity and offers the advantageof automating variable elimination and feature selection.

C. **Decision Tree**

In the context of classification and regression, Decision Trees (DTs) serve as non-parametric supervised learning algorithms. Their purpose is to extract fundamental rules from the features of a dataset in order to construct a model capable of predicting the value of a target variable. Decision Trees are versatile in that they can be employed for both classification and regression tasks, although they are predominantly utilized for solving classification problems.

These tree-structured classifiers consist of internal nodes representing dataset features, branches representing decision rules, and leaf nodes representing the final outcomes or predictions.

Within a Decision Tree, two types of nodes can be identified: the Decision Nodeand the Leaf Node. Decision nodes play a pivotal role in making decisions and possess multiple branches that lead to subsequent nodes. Conversely, leaf nodes represent the ultimate output or outcome resulting from those decisions and do not contain any additional branches for further exploration or decision-making.

In a Decision Tree, decisions or tests are executed based on the features present in

the given dataset. This graphical representation aims to provide a comprehensive view of all possible solutions or decisions by considering specific conditions. Theterm "decision tree" is derived from its resemblance to a tree structure, where the root node serves as the starting point and branches further out, forming a tree-like structure to encompass various decision paths.

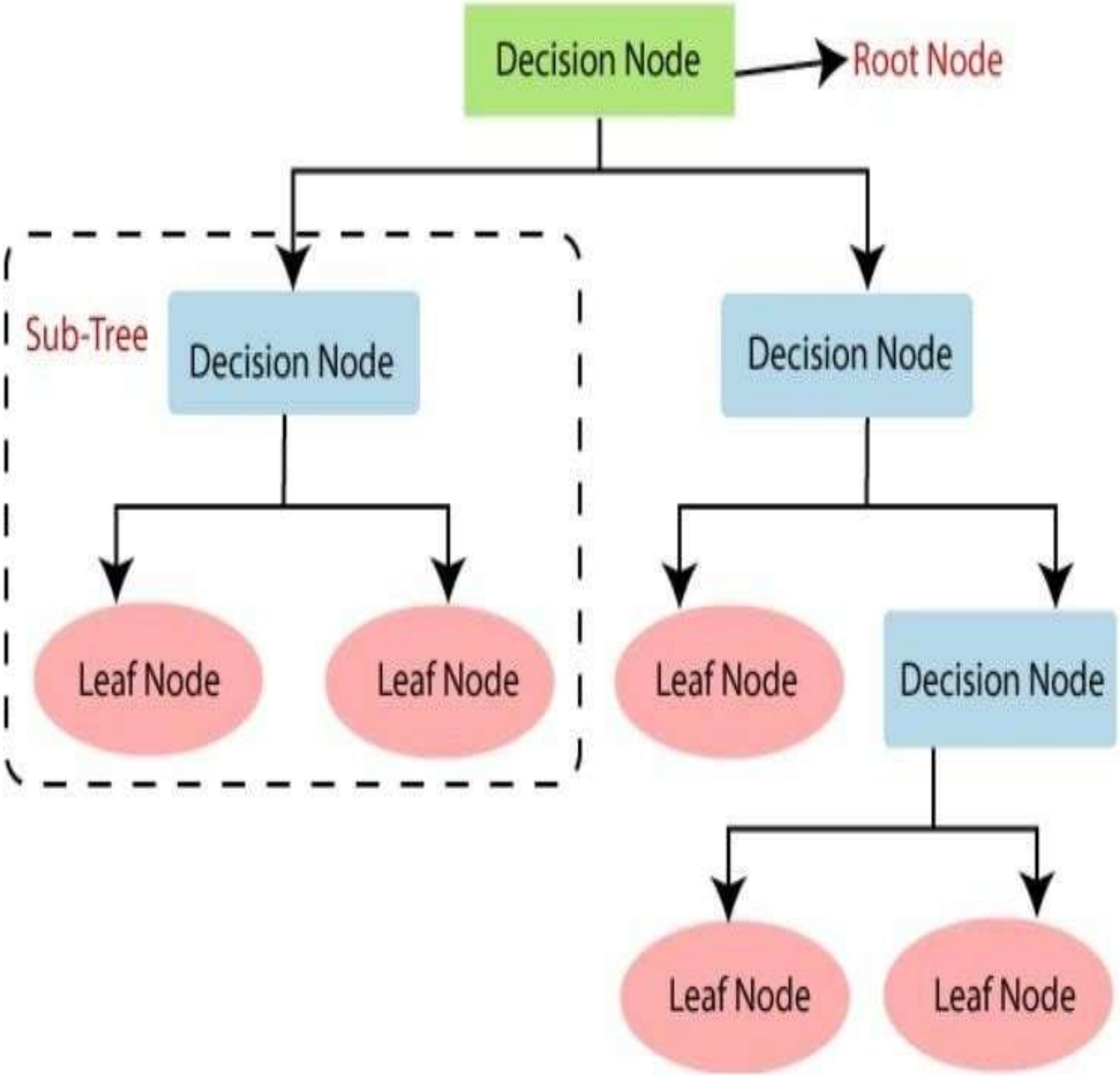


Fig: Decision Tree

Flow Diagram

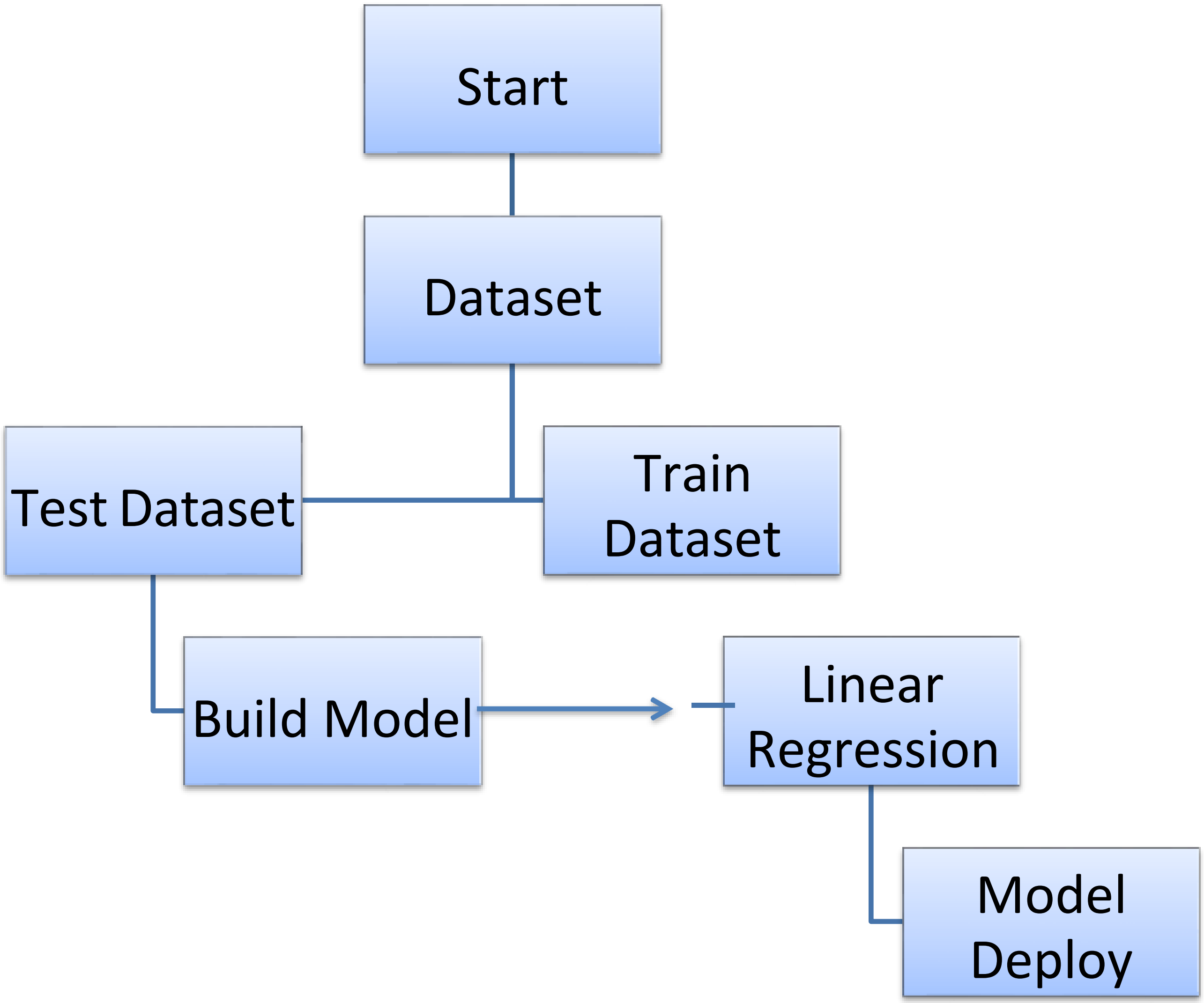


Fig: Flow Diagram of project

Chapter-3 Working of Project

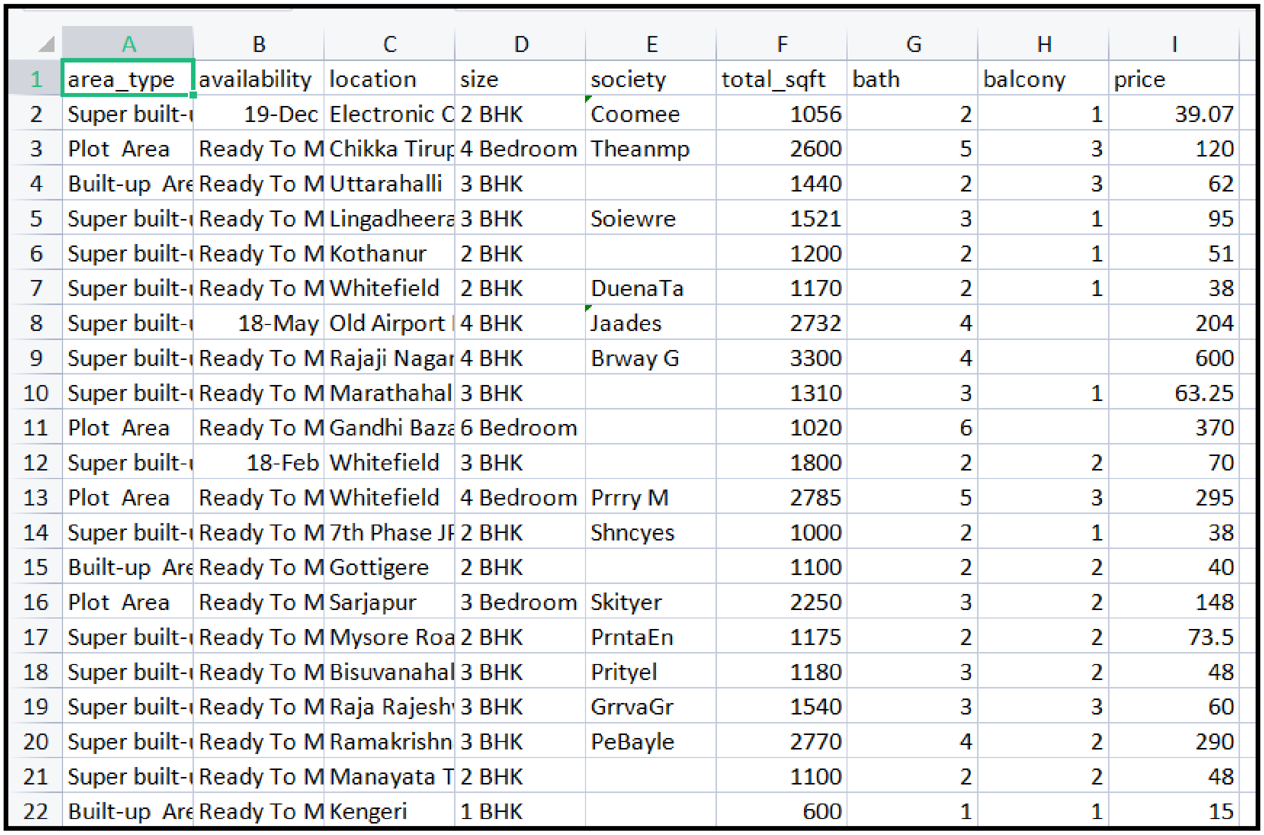
There are several steps in working of project.

1) Data Collection- Collecting the data from different data sources.
2) Pre processing- Processing the data and working it into a form from where it would be easy to analyse the data.
3) Data Analysis- Analyzing the data.
4) Application of Algorithms- Applying different algorithms on the dataset and building a model.
5) Evaluating the model - The different models will be evaluated and the best willwork.

Data Collection
Pre-processing
Data Analysis
Application of Algorithms
Evaluating the model

Fig: Working of project

STEP1- DATA EXTRACTING - Here we gathered the data from the different part of cities, where we need to perform house predictions.
In the context of the paper, data extraction refers to the procedure of gathering or obtaining diverse data from multiple sources, which often exhibit poor organization or lack structure.
In this step we imported dataset downloaded from kaggle in form of csv file ofBengaluru house prices in jupyter notebook.



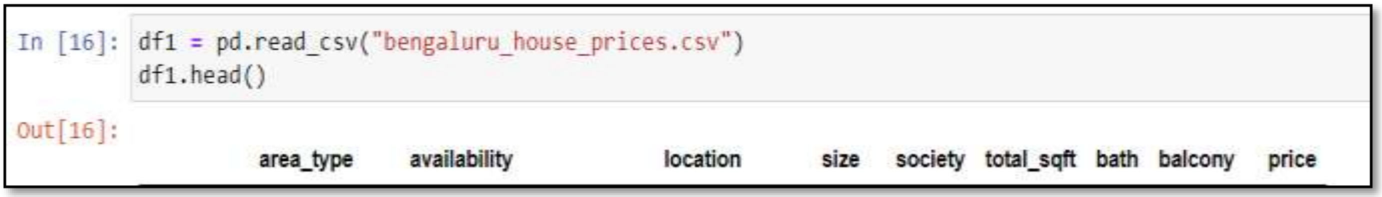
	A	B	C	D	E	F	G	H	I
1	area_type	availability	location	size	society	total_sqft	bath	balcony	price
2	Super built-	19-Dec	Electronic C	2 BHK	Coomee	1056	2	1	39.07
3	Plot Area	Ready To M	Chikka Tirup	4 Bedroom	Theanmp	2600	5	3	120
4	Built-up Are	Ready To M	Uttarahalli	3 BHK		1440	2	3	62
5	Super built-	Ready To M	Lingadheera	3 BHK	Soiewre	1521	3	1	95
6	Super built-	Ready To M	Kothanur	2 BHK		1200	2	1	51
7	Super built-	Ready To M	Whitefield	2 BHK	DuenaTa	1170	2	1	38
8	Super built-	18-May	Old Airport	4 BHK	Jaades	2732	4		204
9	Super built-	Ready To M	Rajaji Nagar	4 BHK	Brway G	3300	4		600
10	Super built-	Ready To M	Marathahal	3 BHK		1310	3	1	63.25
11	Plot Area	Ready To M	Gandhi Baza	6 Bedroom		1020	6		370
12	Super built-	18-Feb	Whitefield	3 BHK		1800	2	2	70
13	Plot Area	Ready To M	Whitefield	4 Bedroom	Prrry M	2785	5	3	295
14	Super built-	Ready To M	7th Phase JF	2 BHK	Shncyes	1000	2	1	38
15	Built-up Are	Ready To M	Gottigere	2 BHK		1100	2	2	40
16	Plot Area	Ready To M	Sarjapur	3 Bedroom	Skityer	2250	3	2	148
17	Super built-	Ready To M	Mysore Roa	2 BHK	PrntaEn	1175	2	2	73.5
18	Super built-	Ready To M	Bisuvanahal	3 BHK	Prityel	1180	3	2	48
19	Super built-	Ready To M	Raja Rajesh	3 BHK	GrrvaGr	1540	3	3	60
20	Super built-	Ready To M	Ramakrishn	3 BHK	PeBayle	2770	4	2	290
21	Super built-	Ready To M	Manayata T	2 BHK		1100	2	2	48
22	Built-up Are	Ready To M	Kengeri	1 BHK		600	1	1	15

Fig: Dataset from kaggle

STEP2-DATA PREPROCESSING – As a preliminary step towards reducing model complexity, we initially extracted the relevant schema from the dataset.

Data preprocessing, which involves preparing raw data to be suitable for a machine learning model, is a crucial and primary stage in model development.

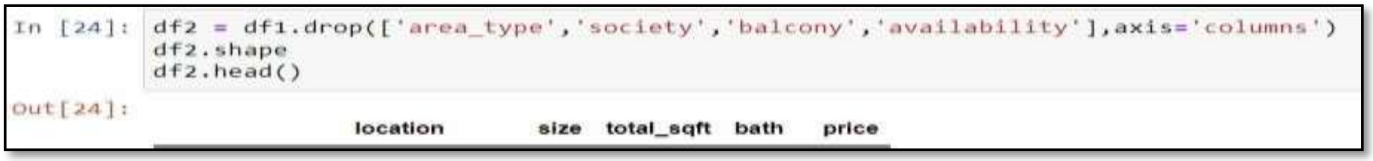
It is common to encounter datasets that are not clean or properly prepared when working on machine learning projects. Therefore, data preprocessing is necessaryto clean and format the data for further analysis and modeling.



```
In [16]: df1 = pd.read_csv("bengaluru_house_prices.csv")
         df1.head()

Out[16]:
              area_type    availability          location      size   society  total_sqft  bath  balcony   price
```

Fig: Reading csv file in jupyter noterbook

Then, we drop some of the columns that are not required and are not significant inbuilding of ML model. So, we dropped features like 'area_type','society','balcony','availability'.
After dropping the columns.

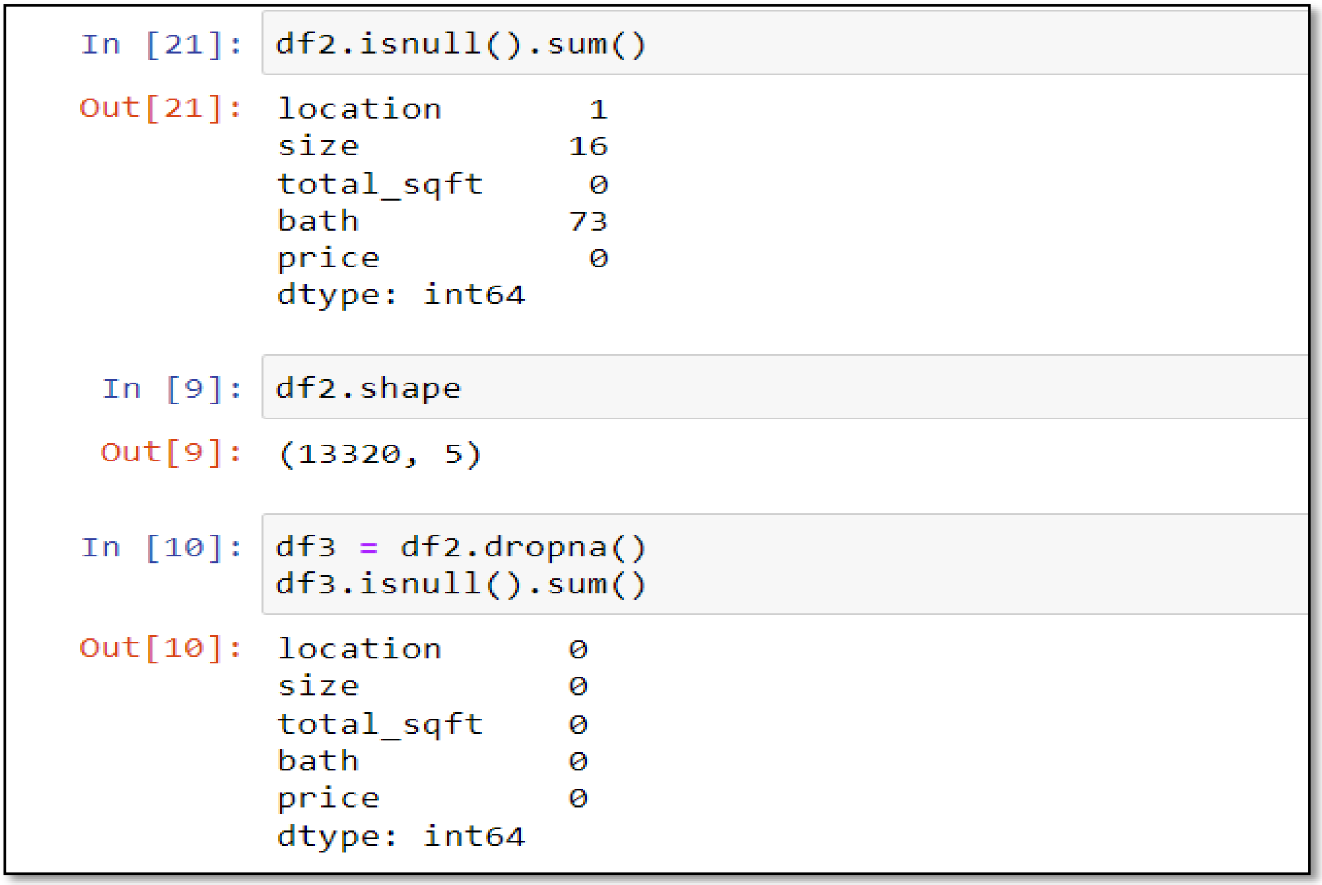


```
In [24]: df2 = df1.drop(['area_type','society','balcony','availability'],axis='columns')
         df2.shape
         df2.head()

Out[24]:
                    location          size   total_sqft   bath     price
```

STEP 3- DATA CLEANING:

Data cleaning refers to the process of rectifying or removing inaccurate, corrupted, improperly formatted, duplicated, or incomplete data within a dataset. When merging multiple data sources, there are various ways in which data can become duplicated or incorrectly labeled.

Then in next step we cleaned the raw data by removing the Null values, we found that column(location, size, bath has 1,16,73) having the Null values respectively. So, eliminated their rows



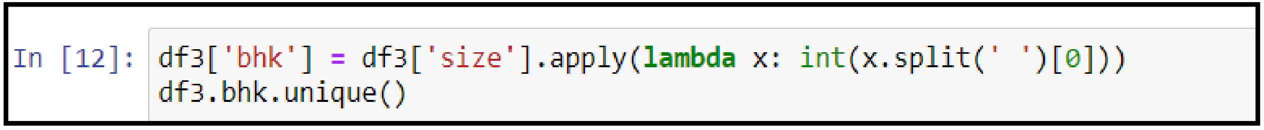
```
  In [21]: df2.isnull().sum()

 Out[21]: location       1
          size          16
          total_sqft     0
          bath          73
          price          0
          dtype: int64

   In [9]: df2.shape

  Out[9]: (13320, 5)

  In [10]: df3 = df2.dropna()
           df3.isnull().sum()

 Out[10]: location      0
          size          0
          total_sqft    0
          bath          0
          price         0
          dtype: int64
```

Fig: Removing null values

Also, we resolved improper data values with Exception handling. STEP 4-

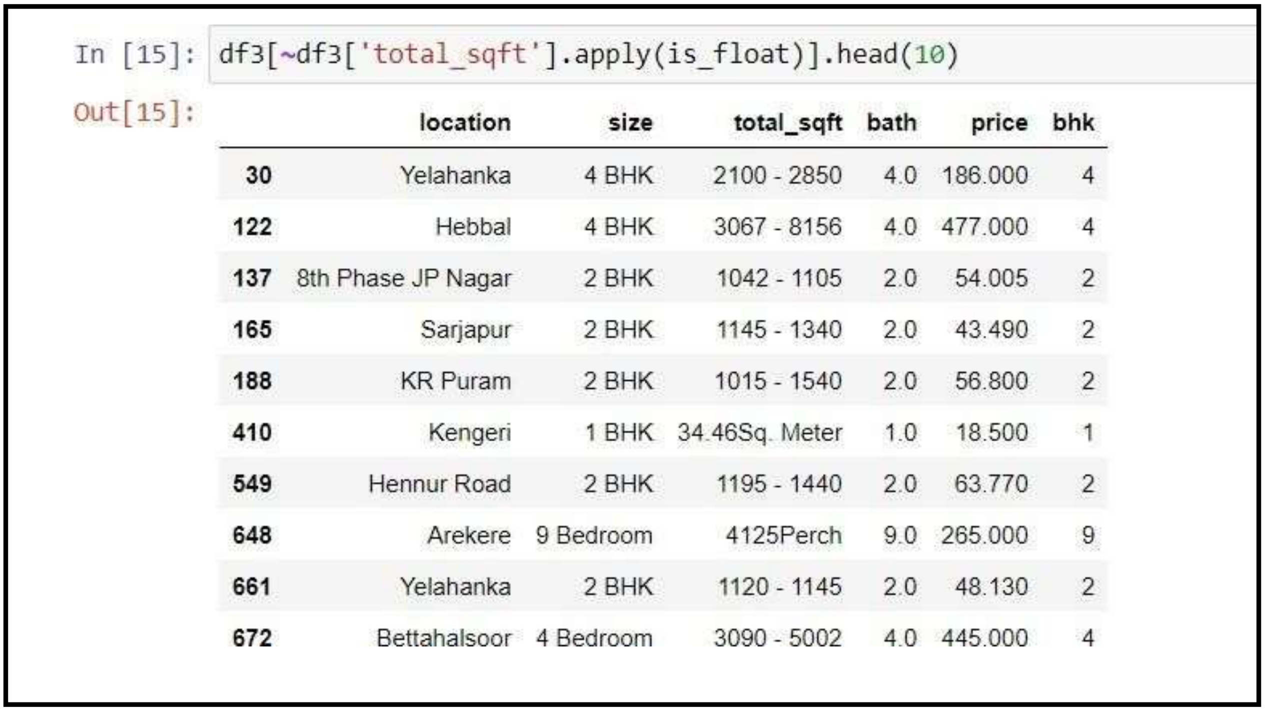
FEATURE ENGINEERING:

Feature engineering is the process of selecting, modifying, and transforming raw data into meaningful features that can be utilized in supervised learning. It is often necessary to create and refine features to enhance the effectiveness of machine learning in handling new tasks. Features, as quantifiable inputs, play a crucial role in predictive models, encompassing attributes such as object color or voice characteristics. In essence, feature engineering involves applying statistical or machine learning techniques to convert unprocessed observations into desired features.



```
In [12]: df3['bhk'] = df3['size'].apply(lambda x: int(x.split(' ')[0]))
         df3.bhk.unique()
```

Fig: Adding new feature(integer) for bhk (Bedrooms Hall Kitchen)

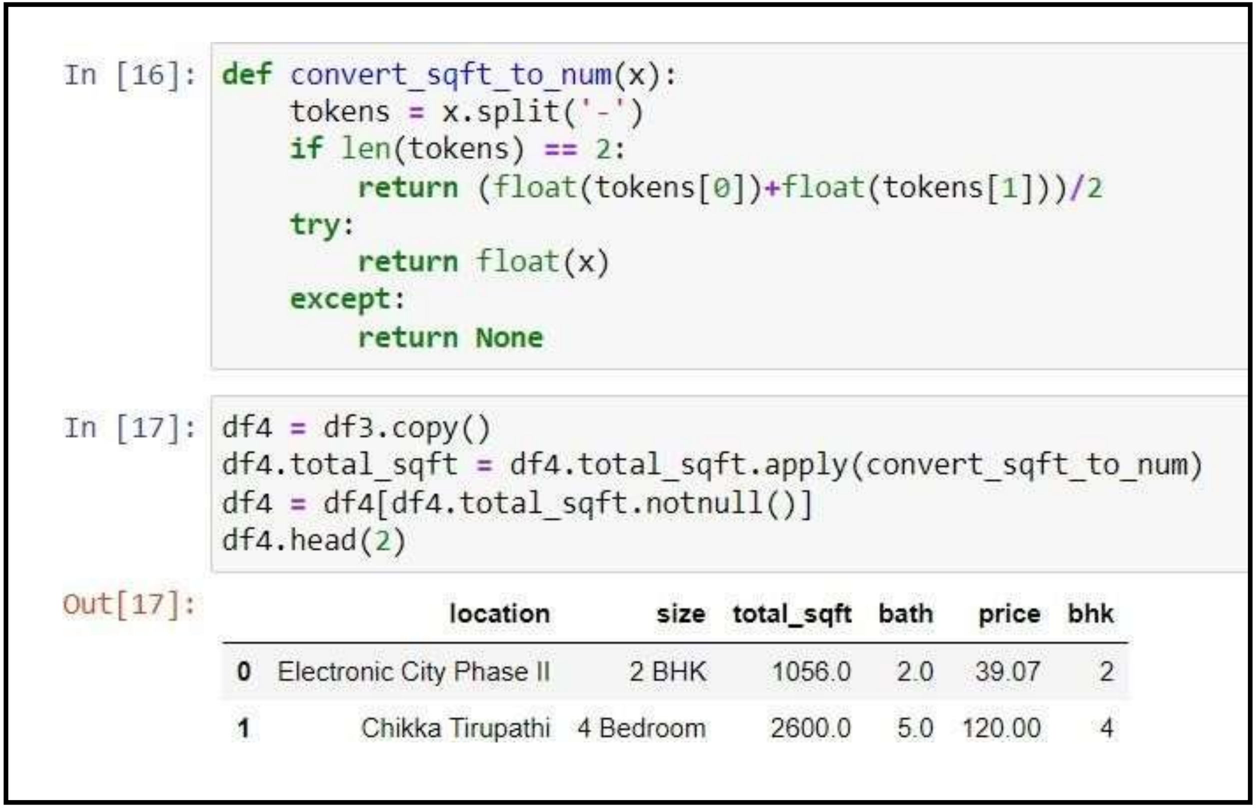


```
In [15]: df3[~df3['total_sqft'].apply(is_float)].head(10)
Out[15]:
```

	location	size	total_sqft	bath	price	bhk
30	Yelahanka	4 BHK	2100 - 2850	4.0	186.000	4
122	Hebbal	4 BHK	3067 - 8156	4.0	477.000	4
137	8th Phase JP Nagar	2 BHK	1042 - 1105	2.0	54.005	2
165	Sarjapur	2 BHK	1145 - 1340	2.0	43.490	2
188	KR Puram	2 BHK	1015 - 1540	2.0	56.800	2
410	Kengeri	1 BHK	34.46Sq. Meter	1.0	18.500	1
549	Hennur Road	2 BHK	1195 - 1440	2.0	63.770	2
648	Arekere	9 Bedroom	4125Perch	9.0	265.000	9
661	Yelahanka	2 BHK	1120 - 1145	2.0	48.130	2
672	Bettahalsoor	4 Bedroom	3090 - 5002	4.0	445.000	4

Fig: New BHK feature

As depicted above, the feature "total_sqft" can sometimes be represented as arange, such as 2100-2850. In such cases, it is appropriate to compute the average of the minimum and maximum values within the range. Additionally, there are instances where the measurement is given in a different unit, such as 34.46 Sq. Meter, which can be converted to square feet using unit conversion. However, to maintain simplicity, these specific corner cases will be excluded or dropped from the analysis.



```
In [16]: def convert_sqft_to_num(x):
             tokens = x.split('-')
             if len(tokens) == 2:
                 return (float(tokens[0])+float(tokens[1]))/2
             try:
                 return float(x)
             except:
                 return None
```

```
In [17]: df4 = df3.copy()
         df4.total_sqft = df4.total_sqft.apply(convert_sqft_to_num)
         df4 = df4[df4.total_sqft.notnull()]
         df4.head(2)
```

```
Out[17]:
                  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
```

Fig: Average of sqft

Also, some new features added such as ‘price per square feet’, which will help in model building.

STEP 4- Dimensionality Reduction:

Dimensionality reduction techniques refer to methods that transform a dataset with high dimensions into a lower-dimensional representation while preserving relevant information. They are commonly employed in machine learning to improve the performance of predictive models for classification and regression tasks. In such tasks, dealing with a large number of variables, also known as features, can be challenging. The curse of dimensionality arises when the number of features increases, making modeling and analysis more complex. The upcoming section will delve into a detailed exploration of this concept.

To minimize the number of categories, locations with fewer than 10 data points areclassified as "other" locations. This approach significantly reduces the overall number of categories. Consequently, during the subsequent step of one-hot encoding, having fewer dummy columns proves advantageous and simplifies the data representation.

location_stats_less_than_10 = location_stats[location_stats<=10]

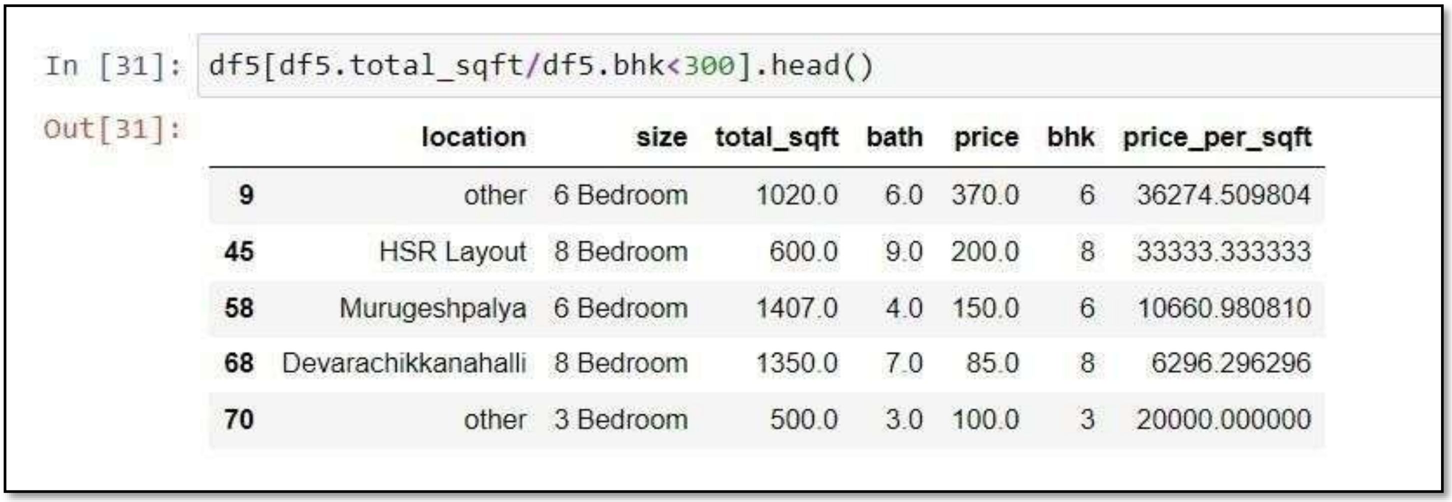
df5.location=df5.location.apply(lambda x: 'other' if x in location_stats_less_than_10 else x)

This will mark 'others' to locations with less than 10 data points.

STEP 5- OUTLIER REMOVAL:

An observation that differs from the other observations is called an outlier. Data points that stand out from the rest of the dataset are known as outliers. The data distribution is frequently skewed by these anomalous observations, which are frequently the result of inaccurate observations or incorrect data entry.

It's crucial to find and eliminate outliers in order to guarantee that the training model generalises adequately to the acceptable range of test inputs.



```
In [31]: df5[df5.total_sqft/df5.bhk<300].head()
```

Out[31]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
9	other	6 Bedroom	1020.0	6.0	370.0	6	36274.509804
45	HSR Layout	8 Bedroom	600.0	9.0	200.0	8	33333.333333
58	Murugeshpalya	6 Bedroom	1407.0	4.0	150.0	6	10660.980810
68	Devarachikkanahalli	8 Bedroom	1350.0	7.0	85.0	8	6296.296296
70	other	3 Bedroom	500.0	3.0	100.0	3	20000.000000

Fig: Outliers

Upon reviewing the data points mentioned above, it is evident that there are some clear data errors. For instance, there is an entry for a 6 BHK apartment with a total area of 1020 sqft, and another entry for an 8 BHK apartment with a total area of 600 sqft. These errors can be safely removed from the dataset.

Also must ensure the maximum amounts of toilets should equal to the amount of Rooms or some casesmax 2 more. But we found an outlier.So we removed the rows (having bathroom>BHK+2).

STEP6- EXPLORATORY DATA ANALYSIS -

This will help us in to get the insights about data in an effective visual manner. MATPLOTLIB and SEABORN are the most used packages for this.
This also helps in Outlier Detection which makes model more accurate. Also we find the correlation between features.

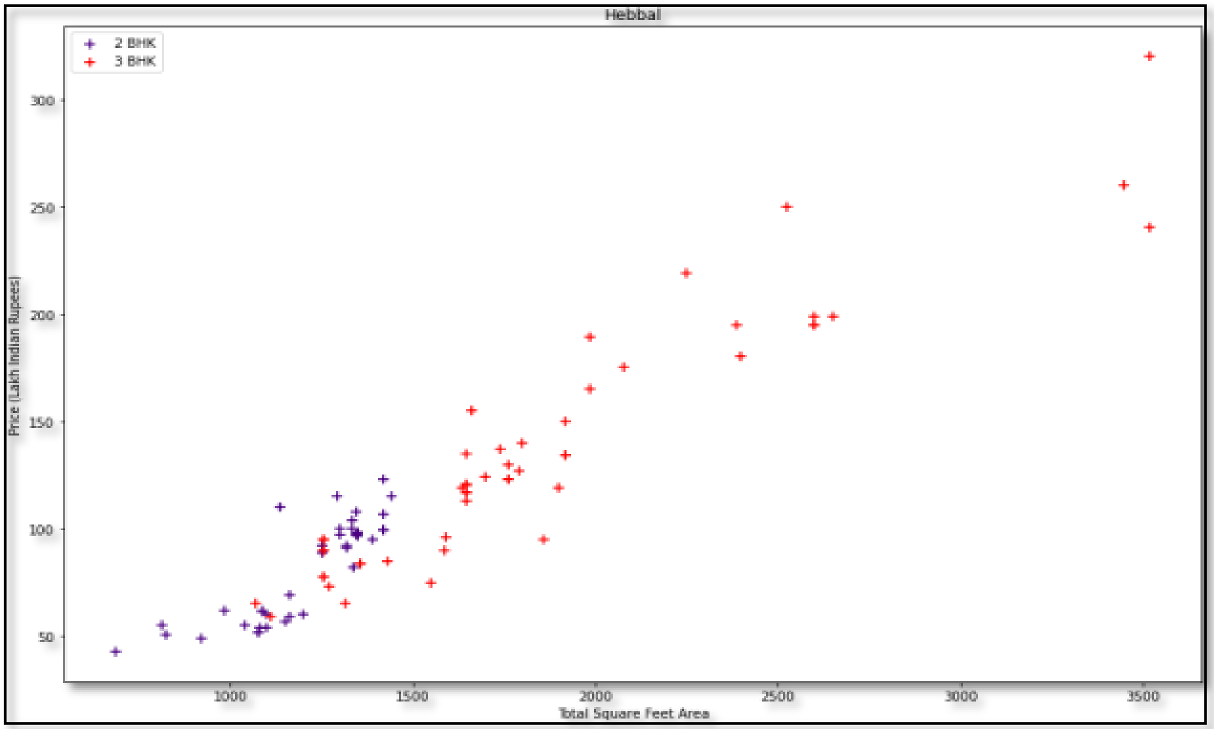
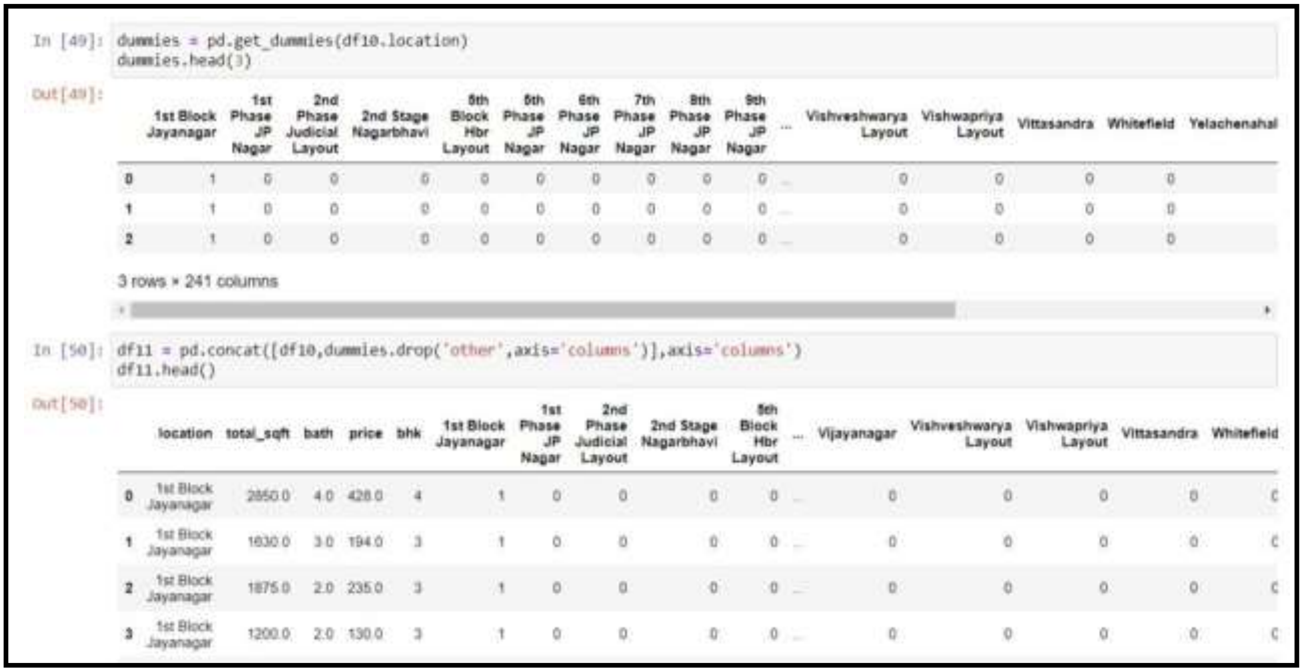


Fig: Visualization of total sqft with price

Above Visualization we can see that there are some outliers (i.e price of 3BHK are equivalent to the priceof 2BHK)

STEP 7-ONE HOT ENCODING:

In a machine learning model, we encode categorical variables as numerical valuesusing one hot encoding.
Here we will convert our Categorical feature LocationInto some numerical value tofeed it to ML algorithms.



```
In [49]: dummies = pd.get_dummies(df10.location)
         dummies.head(3)
Out[49]:
```

	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	5th Phase JP Nagar	6th Phase JP Nagar	7th Phase JP Nagar	8th Phase JP Nagar	9th Phase JP Nagar	...	Vishveshwarya Layout	Vishwapriya Layout	Vittasandra	Whitefield	Yelachenahal
0	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	
1	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	
2	1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	

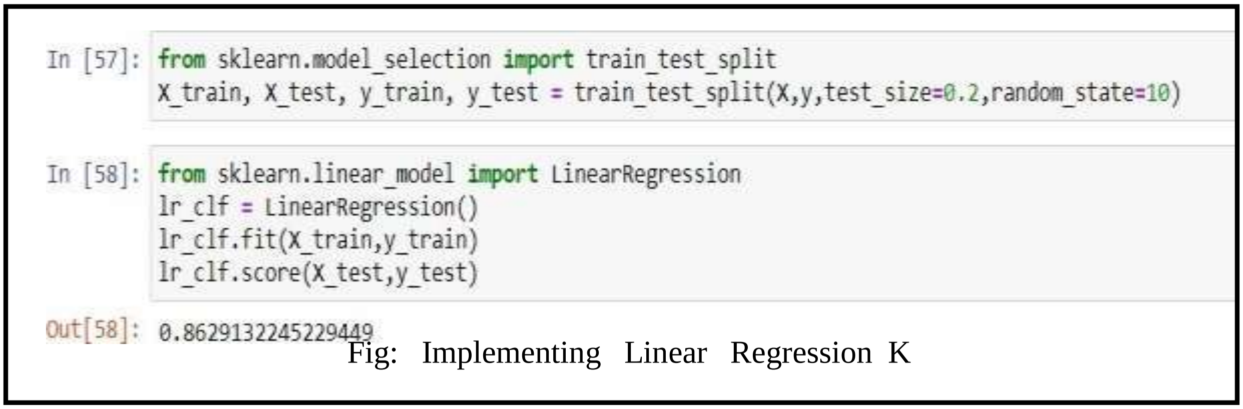
3 rows × 241 columns

```
In [50]: df11 = pd.concat([df10,dummies.drop('other',axis='columns')],axis='columns')
         df11.head()
Out[50]:
```

	location	total_sqft	bath	price	bhk	1st Block Jayanagar	1st Phase JP Nagar	2nd Phase Judicial Layout	2nd Stage Nagarbhavi	5th Block Hbr Layout	...	Vijayanagar	Vishveshwarya Layout	Vishwapriya Layout	Vittasandra	Whitefield
0	1st Block Jayanagar	2850.0	4.0	428.0	4	1	0	0	0	0	...	0	0	0	0	C
1	1st Block Jayanagar	1630.0	3.0	194.0	3	1	0	0	0	0	...	0	0	0	0	C
2	1st Block Jayanagar	1875.0	2.0	235.0	3	1	0	0	0	0	...	0	0	0	0	C
3	1st Block Jayanagar	1200.0	2.0	130.0	3	1	0	0	0	0	...	0	0	0	0	C

Fig: One hot encoding

STEP 8- BUILDING MODEL:



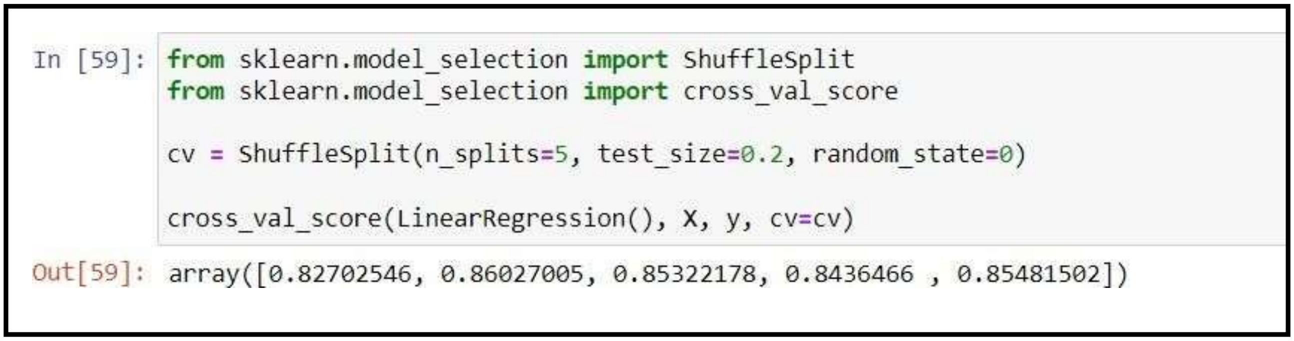
```
In [57]: from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=10)

In [58]: from sklearn.linear_model import LinearRegression
         lr_clf = LinearRegression()
         lr_clf.fit(X_train,y_train)
         lr_clf.score(X_test,y_test)

Out[58]: 0.8629132245229449
```

Fig: Implementing Linear Regression K

FOLD VALIDATION-

By training the model on a subset of the input data and testing it on a subset of the input data that hasn't been used before, you may validate the model's effectiveness. It is also a method for determining how well a statistical model generalises to a different dataset.

In this we will divide the entire data in K parts(5-10),then we train model with(k-1)part and test it with the remaining K part.(.generally training set=80% and testing set=20% data).



```
In [59]: 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)

Out[59]: array([0.82702546, 0.86027005, 0.85322178, 0.8436466 , 0.85481502])
```

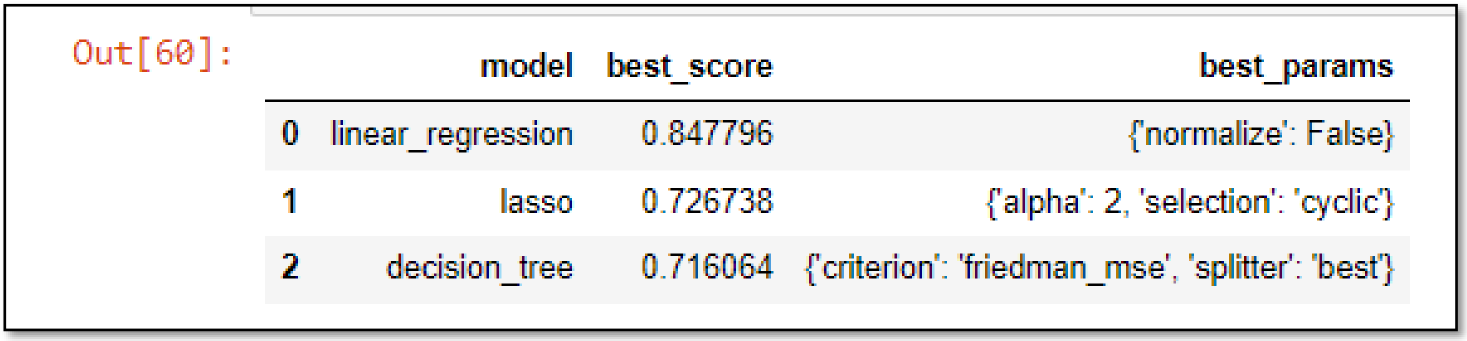
Fig:Employ K Fold cross-validation to assess the accuracy of our LinearRegression model.

During 5 iterations, we consistently achieve a score above 80%, which is commendable. However, we aim to explore other regression algorithms to potentially achieve an even higher score. To accomplish this, we will utilize GridSearchCV.

STEP6-HYPERPARAMETER TUNNING-

GridSearchCV:GridSearchCV is a technique used to optimize the hyperparameters of a model by systematically searching through a grid of parameter combinations. Hyperparameters play a crucial role in determining the performance of a model,and GridSearchCV helps in finding the best hyperparameter values for improved model performance.

Here we have used the Grid Search CV to Compare the accuracy of the different ML Regression algorithms (Linear Regression, Lasso Regression and Decision Tree) on different parameters.



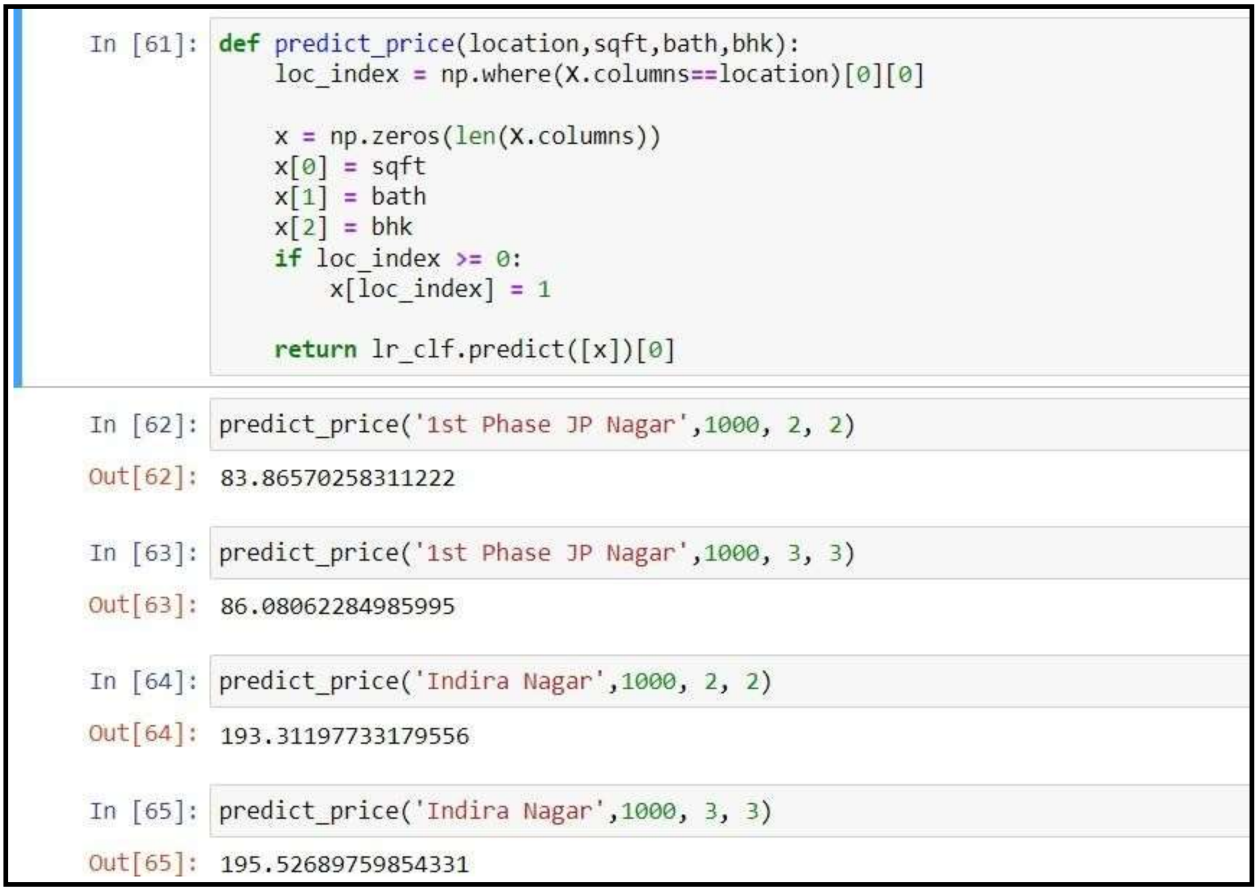
Out[60]:

	model	best_score	best_params
0	linear_regression	0.847796	{'normalize': False}
1	lasso	0.726738	{'alpha': 2, 'selection': 'cyclic'}
2	decision_tree	0.716064	{'criterion': 'friedman_mse', 'splitter': 'best'}

Fig:Performance of several ML models

Based on above results we can say that LinearRegression gives the best score. Hence we will use that.

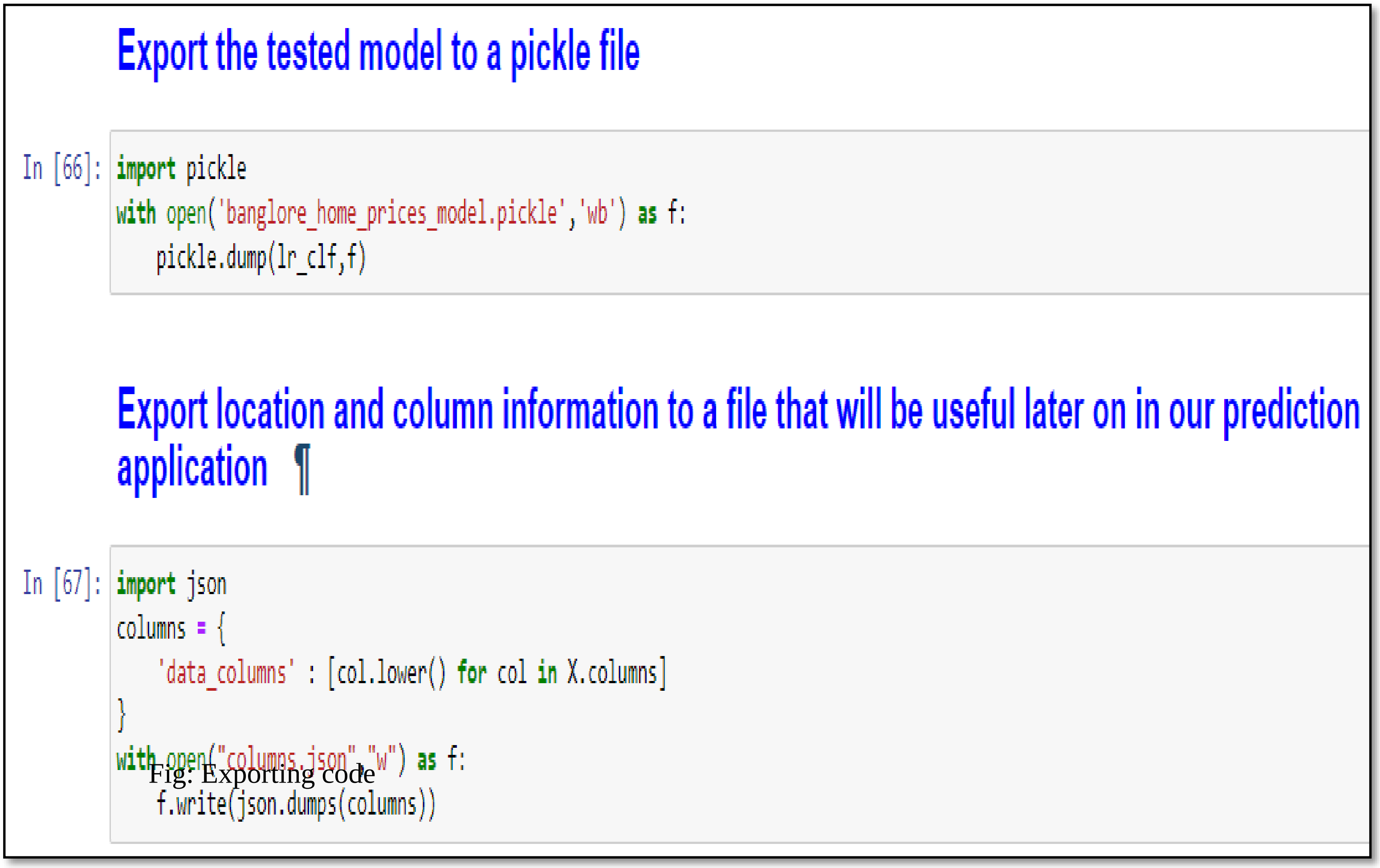
TEST MODEL:



```
In [61]: def predict_price(location,sqft,bath,bhk):
             loc_index = np.where(X.columns==location)[0][0]

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

             return lr_clf.predict([x])[0]

In [62]: predict_price('1st Phase JP Nagar',1000, 2, 2)

Out[62]: 83.86570258311222

In [63]: predict_price('1st Phase JP Nagar',1000, 3, 3)

Out[63]: 86.08062284985995

In [64]: predict_price('Indira Nagar',1000, 2, 2)

Out[64]: 193.31197733179556

In [65]: predict_price('Indira Nagar',1000, 3, 3)

Out[65]: 195.52689759854331
```

Fig: Testing the model

Exporting the tested model to a pickle file &Location and Columns inJSON file



Export the tested model to a pickle file

```
In [66]: import pickle
         with open('banglore_home_prices_model.pickle','wb') as f:
             pickle.dump(lr_clf,f)
```

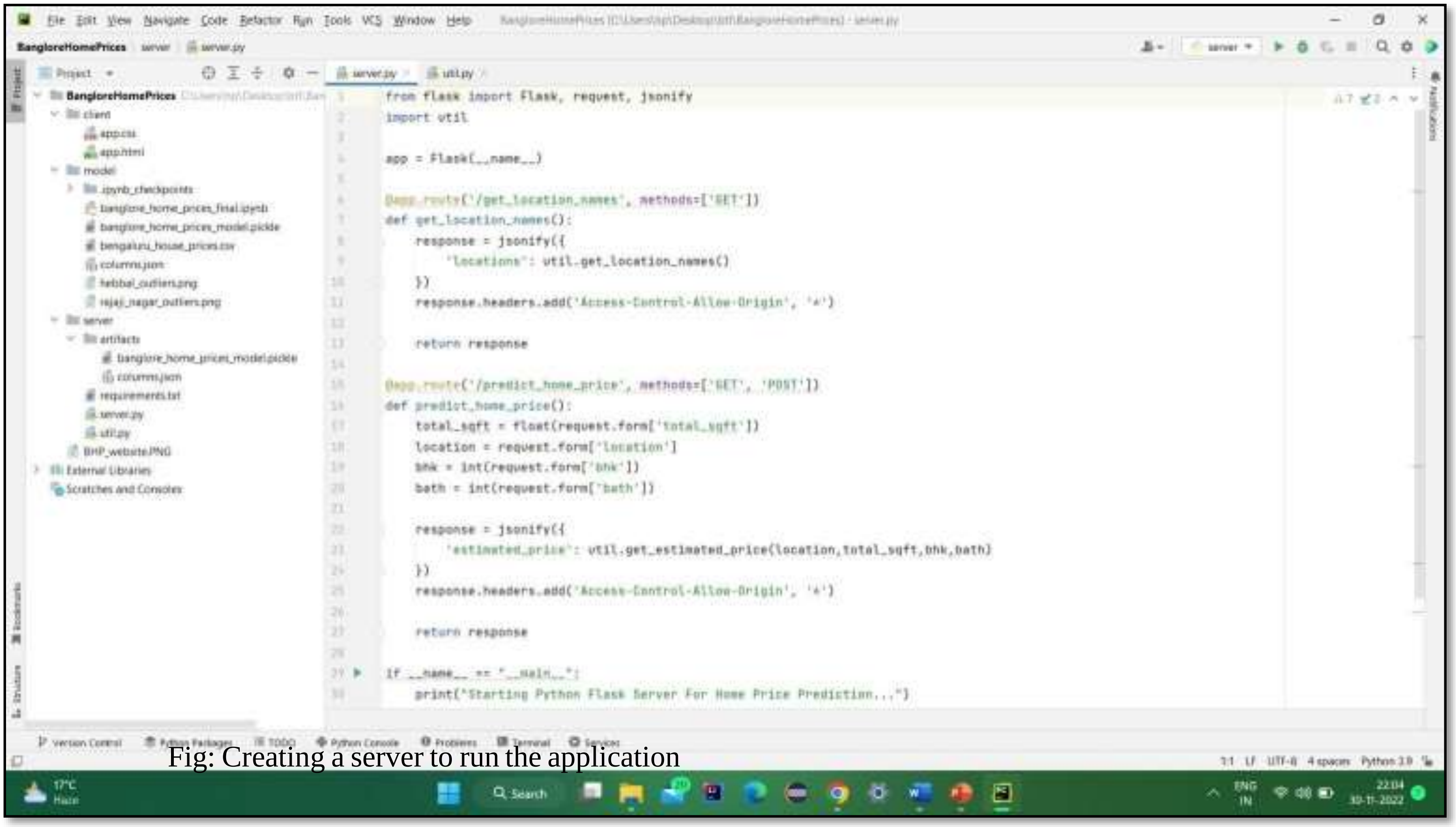
Export location and column information to a file that will be useful later on in our prediction application ¶

```
In [67]: import json
         columns = {
             'data_columns' : [col.lower() for col in X.columns]
         }
         with open("columns.json","w") as f:
             f.write(json.dumps(columns))
```

Fig: Exporting code

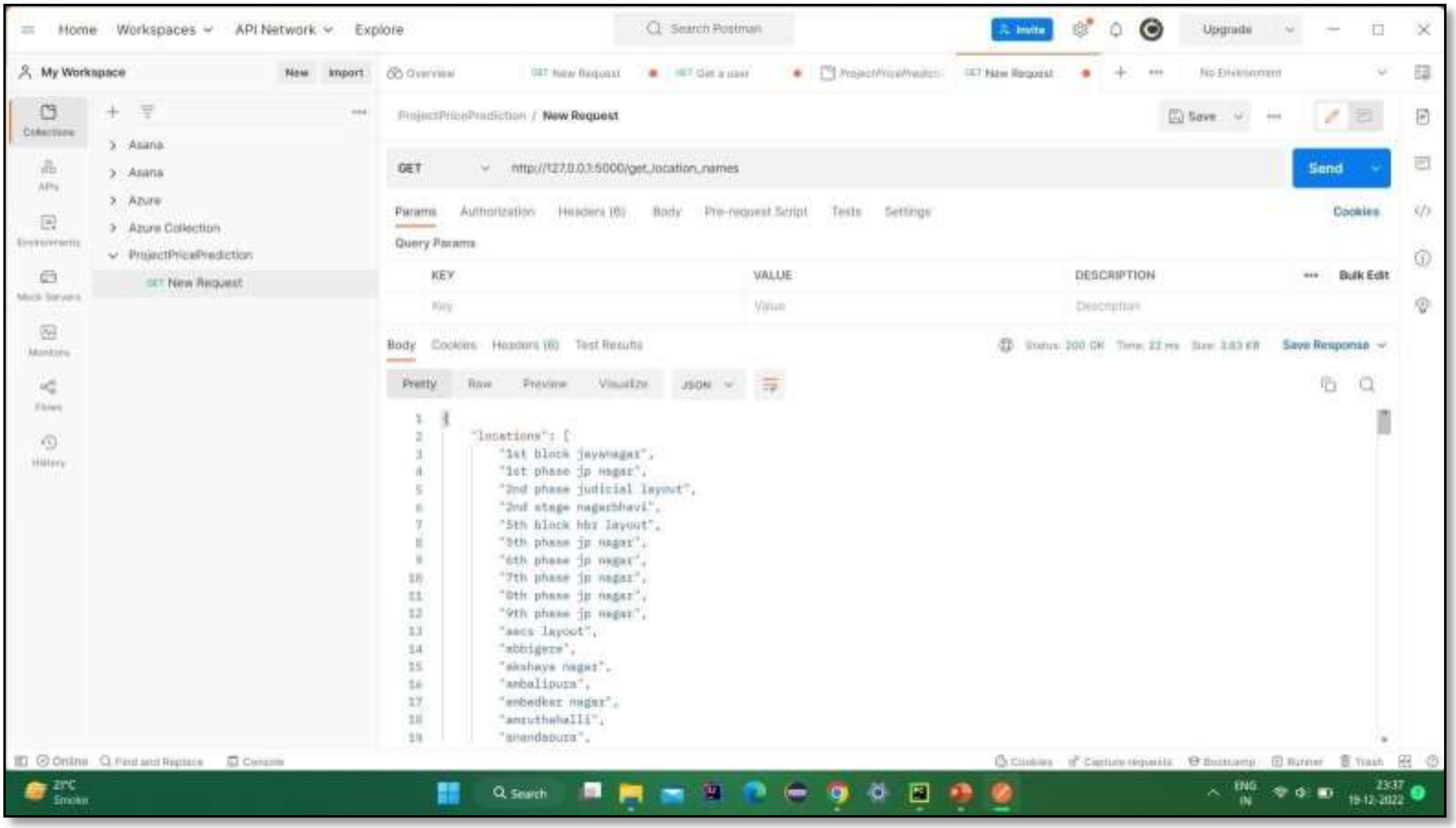
Creating Server and using Postman

Here, a server is created using python Flask for creating endpoints or APIs to connect front-end and back-end and to run our ML model.



```
from flask import Flask, request, jsonify
import util

app = Flask(__name__)

@app.route('/get_location_names', methods=['GET'])
def get_location_names():
    response = jsonify({
        'locations': util.get_location_names()
    })
    response.headers.add('Access-Control-Allow-Origin', '*')

    return response

@app.route('/predict_home_price', methods=['GET', 'POST'])
def predict_home_price():
    total_sqft = float(request.form['total_sqft'])
    location = request.form['location']
    bhk = int(request.form['bhk'])
    bath = int(request.form['bath'])

    response = jsonify({
        'estimated_price': util.get_estimated_price(location,total_sqft,bhk,bath)
    })
    response.headers.add('Access-Control-Allow-Origin', '*')

    return response

if __name__ == "__main__":
    print("Starting Python Flask Server For Home Price Prediction...")
```

Fig: Creating a server to run the application



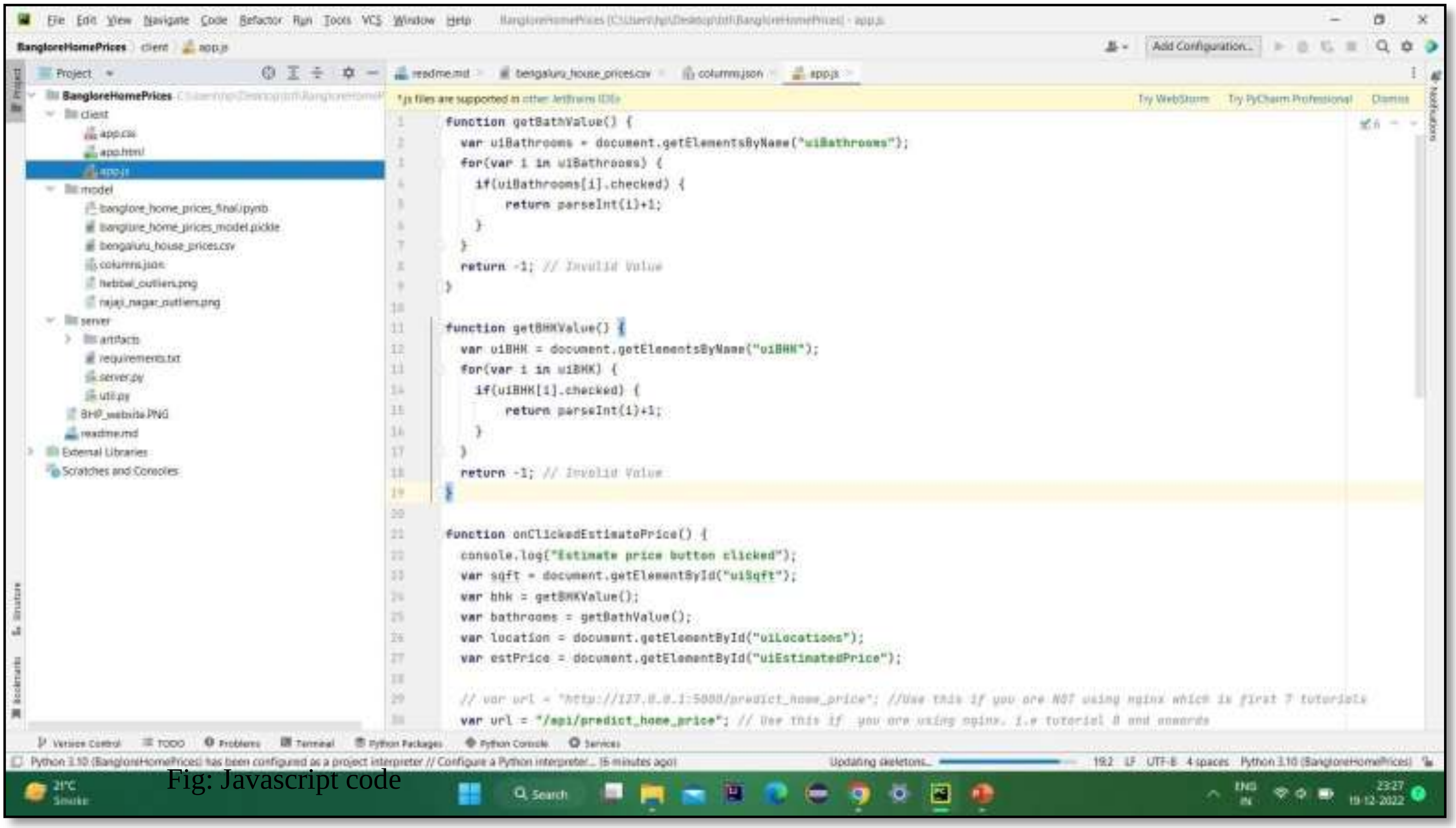
GET http://127.0.0.1:5000/get_location_names

Status: 200 OK Time: 22 ms Size: 3.83 KB

```
{
    "locations": [
        "1st block jayanagar",
        "1st phase jp nagar",
        "2nd phase judicial layout",
        "2nd stage nagarbhavi",
        "5th block hbr layout",
        "5th phase jp nagar",
        "6th phase jp nagar",
        "7th phase jp nagar",
        "8th phase jp nagar",
        "9th phase jp nagar",
        "aecs layout",
        "abbigere",
        "akshaya nagar",
        "ambalipura",
        "ambedkar nagar",
        "amruthahalli",
        "anandapura",
```

Fig: Running the endpoint on port location 127.0.0.1:5000 to get the location oflocalities

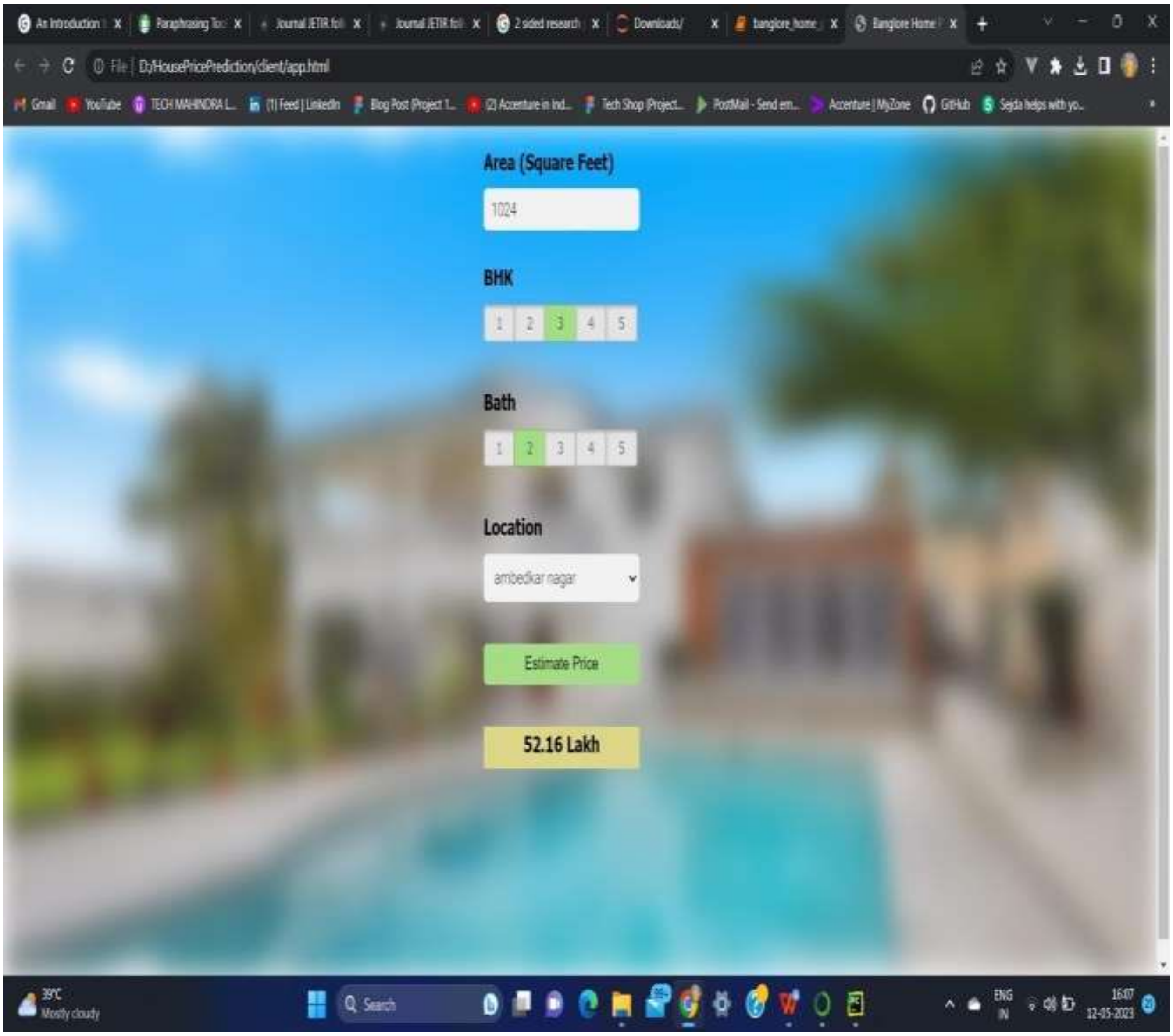
Connecting front-end with back-end



```
function getBathValue() {
  var uiBathrooms = document.getElementsByName("uiBathrooms");
  for(var i in uiBathrooms) {
    if(uiBathrooms[i].checked) {
        return parseInt(i)+1;
    }
  }
  return -1; // Invalid Value
}

function getBHKValue() {
  var uiBHK = document.getElementsByName("uiBHK");
  for(var i in uiBHK) {
    if(uiBHK[i].checked) {
        return parseInt(i)+1;
    }
  }
  return -1; // Invalid Value
}

function onClickedEstimatePrice() {
  console.log("Estimate price button clicked");
  var sqft = document.getElementById("uiSqft");
  var bhk = getBHKValue();
  var bathrooms = getBathValue();
  var location = document.getElementById("uiLocations");
  var estPrice = document.getElementById("uiEstimatedPrice");

  // var url = "http://127.0.0.1:5000/predict_home_price"; //Use this if you are NOT using nginx which is first 7 tutorials
  var url = "/api/predict_home_price"; // Use this if  you are using nginx. i.e tutorial 8 and onwards
```

Fig: Javascript code

Chapter-4RESULT



Area (Square Feet)

1024

BHK

1 2 [✓ 3] 4 5

Bath

1 [✓ 2] 3 4 5

Location

ambedkar nagar

52.16 Lakh

Fig: Estimating price on wepage.

Chapter-5 CONCLUSION

In our study, we focused on the proactive pricing of houses in the Indian Real Estate market. Leveraging advanced machine learning techniques, we developed an algorithm capable of accurately predicting housing prices using specific input features. The practical application of this algorithm is that classified websites can directly utilize it to estimate prices for newly listed properties. By taking relevant input variables into account, the algorithm can provide justified and accurate price predictions, eliminating the need for customers to input their own prices. This approach promotes transparency within the system.

Machine Learning Algorithms we found out that Linear Regression is giving us the best and High Accuracy results with (84%accuracy). Also we would believe this research will be helpful for both Property sellers and buyers.

REFERENCES

1. Chen, S., Wang, T., & Xi, L. (2018). "House Price Prediction: An Empirical Study on the Impact of Location and House Features." IEEE Transactions on Big Data, 4(4), 517-527.

2. Zhang, Z., Zheng, L., & Zeng, X. (2020). "House Price Prediction Using Machine Learning: A Comparative Study." Expert Systems with Applications, 139, 112855.

3. Jahanbakhsh, F., & Barzegar, A. (2017). "House Price Prediction Using Support Vector Regression." Procedia Computer Science, 116, 101-107.

4. Patil, S., & Patil, P. (2018). "A Review on House Price Prediction Using Data Mining Techniques." International Journal of Computer Applications, 180(19), 37-42.

5. Vasilev, G., Marinov, P., & Ivanov, S. (2019). "Machine Learning Techniques for House Price Prediction: A Comparative Analysis." Proceedings of the International Conference on Machine Learning, 87-96.

6. Doan, H. T., & Nguyen, N. T. (2020). "House Price Prediction Using Time Series Analysis and Deep Learning." Journal of Ambient Intelligence and Humanized Computing, 11(12), 5945-5955.

7. Yoo, Y., & Suh, K. (2016). "House Price Prediction Based on Feature Selection and Multiple Regression Analysis." Journal of Real Estate Literature, 24(2), 285-298.

8. Amiri, N., & Dakhane, M. (2017). "House Price Prediction Using Machine Learning Algorithms." International Journal of Computer Applications, 176(5), 1-5.

9. Cao, X., & Xie, X. (2018). "Predicting House Prices with LSTM Networks." Proceedings of the IEEE International Conference on Data Mining Workshops, 689-695.

10. Kannan, D., & Krishnamurthy, R. (2019). "A Comparative Study of House Price Prediction Using Regression Techniques." Journal of Computational and Theoretical Nanoscience, 16(8), 3284-3289.