A PROJECT REPORT ON
"BANGALORE HOUSE PRICE"

SINCHANA C
TLS21A2415

INDEX

ABSTRACT

The real estate sector is an important industry with many stakeholders ranging from regulatory bodies to private companies and investors. Among these stakeholders, there is a high demand for a better understanding of the industry operational mechanism and driving factors. This project can be considered as a further step towards more evidence-based decision making for the benefit of these stakeholders. By conducting explanatory data analysis, we obtain a better understanding of our data. This yields insights that can be helpful later when building a model, as well as insights that are independently interesting.

Many sub steps are taken to get, clean and transform the data. The process presented is used that have been chosen according to their similarities in terms of presentation of the estates and if they give the same information about them. Using Machine learning techniques, we are then able to identify a subset of the original features that are in a sense sufficient to describe our data.

INTRODUCTION

The aim of this project is to predict the sale price of the houses in Bangalore. Input variables are area_type, availability, location, size, society, total_sqrt, bath, balcony. And the output variable is price. We are dealing with only the location, total_sqrt, bath and size. The Machine Learning part is about trying to find the best learning algorithm for a given problem even if it is highly conditioned by how well the data has been processed and tune some parameters to improve it.

During the development and evaluation of our model, we will show the code used for each step followed by its output. This will facilitate the reproducibility of our work. In this study, Python programming language with a number of Python packages will be used. To apply data pre-processing and preparation techniques in order to obtain clean data to build machine learning models able to predict house price based on house features to analyse and compare models' performance in order to choose the best model.

TASKS

- DATA ACQUISITION AND CLEANING
- DATA VISUALIZATION
- DATA MODELLING
- TESTING

DATA ACQUISITION AND CLEANING

The statistics were gathered from Bangalore home prices. The information includes many variables such as area type, availability, location, BHK, society, total square feet, bathrooms, and balconies.

The data is the most important aspect of a Data Science. the data will heavily affect the findings depending on how they are presented, if they are consistent, if there is an outlier, and so on. Many questions must be addressed at this stage to ensure that the learning algorithm is efficient and correct.

Input variables:

- Area_type
- availability
- location
- size
- society
- total_sqrt
- bath
- balcony

Output variable based on sensory data:

- Price

DATA VISUALIZATION

Perform an Exploratory Data Analysis. In EDA, Check the shape of the data set using the shape method. It displays the number of rows and number of columns. Then display the percentage of null values like how much percent it contains NULL values. Then check the value count of the area_type column.

In [4]:
```
data
```

Out[4]:

	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
...	...	...	...	...	...	...	...	...	...
13315	Built-up Area	Ready To Move	Whitefield	5 Bedroom	ArsiaEx	3453	4.0	0.0	231.00
13316	Super built-up Area	Ready To Move	Richards Town	4 BHK	NaN	3600	5.0	NaN	400.00
13317	Built-up Area	Ready To Move	Raja Rajeshwari Nagar	2 BHK	Mahla T	1141	2.0	1.0	60.00
13318	Super built-up Area	18-Jun	Padmanabhanagar	4 BHK	SollyCl	4689	4.0	1.0	488.00
13319	Super built-up Area	Ready To Move	Doddathoguru	1 BHK	NaN	550	1.0	1.0	17.00

13320 rows × 9 columns

Then drop some features (columns) which are of no use to train our model. The features which we are going to drop are availability, area_type, society, balcony. Now display the data set.

In [7]:
```
data=data.drop(['area_type','availability','balcony','society'],axis=1)
data
```

Out[7]:

	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
...	...	...	...	...	...
13315	Whitefield	5 Bedroom	3453	4.0	231.00
13316	Richards Town	4 BHK	3600	5.0	400.00
13317	Raja Rajeshwari Nagar	2 BHK	1141	2.0	60.00
13318	Padmanabhanagar	4 BHK	4689	4.0	488.00
13319	Doddathoguru	1 BHK	550	1.0	17.00

13320 rows × 5 columns

Then again check if there are Null values or not. So, you can see there are some null values. Then we drop all the rows which contain null values using the method dropna(). Then check the shape of the data set and display the top 5 rows of the data set.

In [8]:
```
data.isna().sum()
```

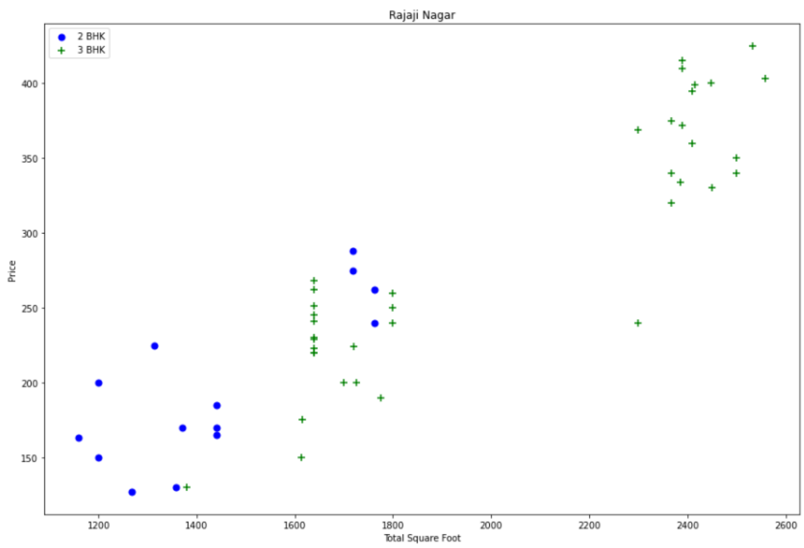
Out[8]:
```
location       1
size          16
total_sqft     0
bath          73
price          0
dtype: int64
```

Now check the unique values of size feature and you can see there are different types of values like in BHK, bedrooms etc. So, we write a function to extract only the starting integer values from the size feature and store it into a new bhk feature. And now you can see the size feature of the data set. Now drop the size feature which is of no use now.

```
In [10]: data.shape
Out[10]: (13246, 5)

In [11]: data['size'].unique()
Out[11]: 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)
```

Now visualize the “Rajaji Nagar” location with 2 bhk and 3 bhk. 2 bhk is in blue color and 3 bhk is in green colour. So, you can see in the below graph that the 3 bhk house price is less than the 2 bhk house price.

```
import matplotlib.pyplot as plt
def plot_scatter_chart(df,location):
    bhk2=df[(df.location==location)&(df.BHK==2)]
    bhk3=df[(df.location==location)&(df.BHK==3)]
    plt.rcParams['figure.figsize']=(15,10)
    plt.scatter(bhk2.total_sqft,bhk2.price,color='Blue',label='2 BHK',s=50)
    plt.scatter(bhk3.total_sqft,bhk3.price,color='green',marker='+',label='3 BHK',s=50)
    plt.xlabel('Total Square Foot')
    plt.ylabel('Price')
    plt.title(location)
    plt.legend()
plot_scatter_chart(data3,"Rajaji Nagar")
```



DATA MODELLING

Data set is split into the independent and dependent features and stored into the "x" and "y" data set. And check the shape of "x" and "y" as you can see below.

Then split the data set into the training and testing using the train_test_split() method which returns 4 data sets as you can see in the below image. Then check the shape of all four data sets.

Now define our linear regression model and train the model using the training data set and check the score of the model using the validation data sets.

```
In [169]: x = df8.drop('price',axis=1)
          y = df8['price']

In [170]: x.shape
Out[170]: (7325, 244)

In [171]: y.shape
Out[171]: (7325,)

In [179]: 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=101)

In [180]: X_train.shape, X_test.shape, y_train.shape, y_test.shape
Out[180]: ((5860, 244), (1465, 244), (5860,), (1465,))
```

TESTING

We are keeping 20% of our dataset to treat it as unseen data and be able and test the performance of our models. We are splitting our dataset in a way such that all of the qualities are represented proportionally equally in both training and testing dataset.

```
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=101)
```

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((5860, 244), (1465, 244), (5860,), (1465,))
```

```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X_train,y_train)
lr.score(X_test,y_test)
```

```
0.8629898728935371
```

```
pred = lr.predict(X_test)
pred
```

```
array([ 32.66700357, 291.55286051,  69.36556057, ..., 112.8263403 ,
        43.43288776, 135.77405539])
```

```
y_test
```

```
7892     41.745
3357    380.000
126      75.000
3767    175.000
4871     80.000
         ...
9870    120.000
9802     87.000
2955    113.000
917      65.000
748      59.520
Name: price, Length: 1465, dtype: float64
```

Create a function to test the model on a custom data set which takes the location, sqft, bath, bhk, etc. So, I tested a model on 3 custom data sets as you can see in the below image.

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

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

```
85.2974569797724
```

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

```
81.70512816315643
```

```
predict_price('Indira Nagar',1400, 2, 3)
```

```
217.40708528156847
```

PYTHON CODE

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


df = pd.read_csv('Bengaluru_House_Dataset.csv')

df.head()
```

out:

	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

```
df.shape
```

out:

(13320, 9)

```
df.isnull().mean()*100
```

out:

```
area_type        0.000000
availability     0.000000
location         0.007508
size             0.120120
society         41.306306
total_sqft       0.000000
bath             0.548048
balcony          4.572072
price            0.000000
dtype: float64
```

```
df['area_type'].value_counts()
```

out:

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

```
df.drop(columns=["availability","area_type","society","balcony"],axis=1,inplace=True)

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

out:

```
location       1
size          16
total_sqft     0
bath          73
price          0
dtype: int64
```

df.dropna(inplace=True)

df.isnull().sum()

out:

```
location      0
size          0
total_sqft    0
bath          0
price         0
dtype: int64
```

df['size'].unique()

out:

```
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)
```

df['bhk'] = df['size'].apply(lambda x: int(x.split(' ')[0]))

df.drop(columns=["size"],axis=1,inplace=True)

df[df.bhk>22]

out:

	location	total_sqft	bath	price	bhk
1718	2Electronic City Phase II	8000	27.0	230.0	27
4684	Munnekollal	2400	40.0	660.0	43

```
df.total_sqft.unique()

def is_float(x):
    try:
        float(x)
    except:
        return False
    return True

df[~df['total_sqft'].apply(is_float)].head(10)
```

out:

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

```
def convert_sqft_into_number(x):
    token = x.split('-')
    if len(token) == 2:
        return (float(token[0]) + float(token[1])) / 2
    try:
        return float(x)
    except:
        return None
df1 = df.copy()
df1['total_sqft'] = df1['total_sqft'].apply(convert_sqft_into_number)
df2 = df1.copy()
df2['price_per_sqft'] = df2['price']*100000 / df2['total_sqft']
df2.head()
```

out:

	location	total_sqft	bath	price	bhk	price_per_sqft
0	Electronic City Phase II	1056.0	2.0	39.07	2	3699.810606
1	Chikka Tirupathi	2600.0	5.0	120.00	4	4615.384615
2	Uttarahalli	1440.0	2.0	62.00	3	4305.555556
3	Lingadheeranahalli	1521.0	3.0	95.00	3	6245.890861
4	Kothanur	1200.0	2.0	51.00	2	4250.000000

```
df2['location'] = df2['location'].apply(lambda x: x.strip())
```

```
df2.location.value_counts()
```

out:

```
Whitefield                                534
Sarjapur  Road                            392
Electronic City                           302
Kanakpura Road                            266
Thanisandra                               233
                                         ...
Escorts Colony                              1
Nagarbhavi  BDA Complex                     1
Bande Nallasandra                           1
RMV extension stage 2, rmv extension        1
MEI layout, Bagalgunte                      1
Name: location, Length: 1304, dtype: int64
```

```
loc_stats[loc_stats<=10]

loc_less_than_10 = loc_stats[loc_stats<=10]

loc_less_than_10

df2.location = df2.location.apply(lambda x: 'other' if x in loc_less_than_10 else x)

df2.head()
```

	location	total_sqft	bath	price	bhk	price_per_sqft
0	Electronic City Phase II	1056.0	2.0	39.07	2	3699.810606
1	Chikka Tirupathi	2600.0	5.0	120.00	4	4615.384615
2	Uttarahalli	1440.0	2.0	62.00	3	4305.555556
3	Lingadheeranahalli	1521.0	3.0	95.00	3	6245.890861
4	Kothanur	1200.0	2.0	51.00	2	4250.000000

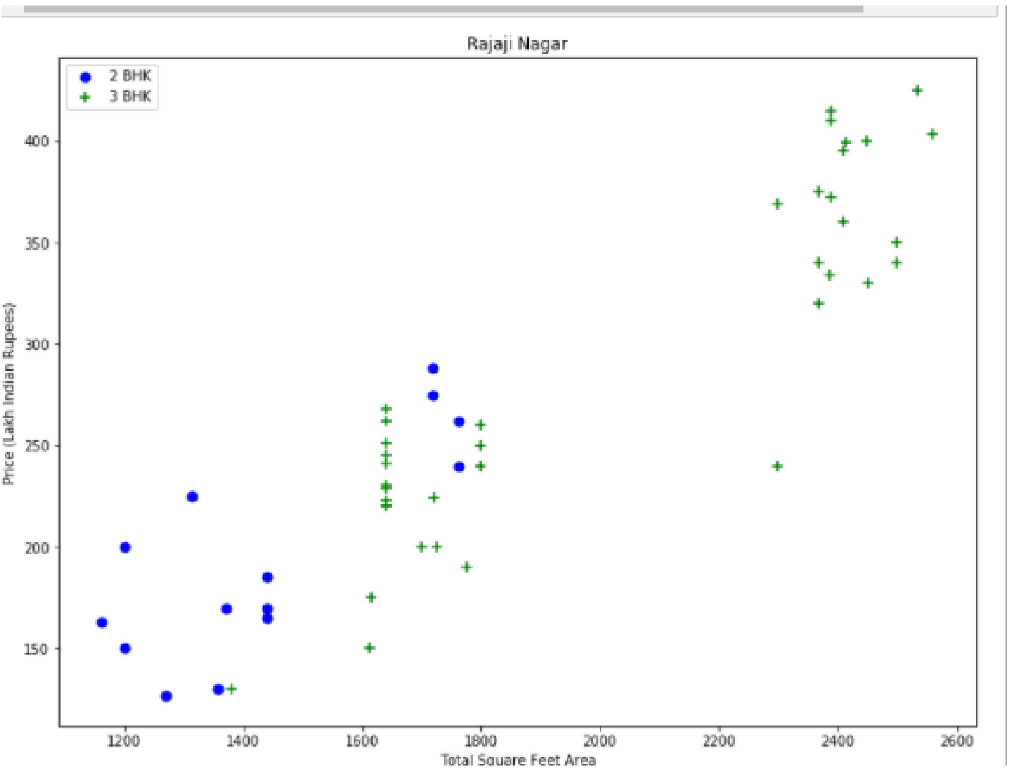
```
len(df2.location.unique())

df2[ (df2.total_sqft / df2.bhk < 300) ].head()

df3 = df2[ ~(df2.total_sqft / df2.bhk < 300) ]

def remove_outlier_from_price_per_sqft(df):

    df_out = pd.DataFrame()

    for key,sub in df.groupby('location'):

        m = np.mean( sub.price_per_sqft )

        st = np.std( sub.price_per_sqft )

        reduce_df = sub[( sub.price_per_sqft>(m-st) ) & ( sub.price_per_sqft<=(m+st) ) ]

        df_out = pd.concat( [df_out, reduce_df],ignore_index=True )

    return df_out

df4 = remove_outlier_from_price_per_sqft(df3)



df4.shape
```

out:

(10241, 6)

```
def plot_scatter_chart(df,location):
    bhk2 = df[(df.location==location) & (df.bhk==2)]
    bhk3 = df[(df.location==location) & (df.bhk==3)]
    plt.rcParams['figure.figsize'] = (12,9)
    plt.scatter(bhk2.total_sqft,bhk2.price,color='blue',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 Area")
    plt.ylabel("Price (Lakh Indian Rupees)")
    plt.title(location)
    plt.legend()
plot_scatter_chart(df4,"Rajaji Nagar")
```

out:

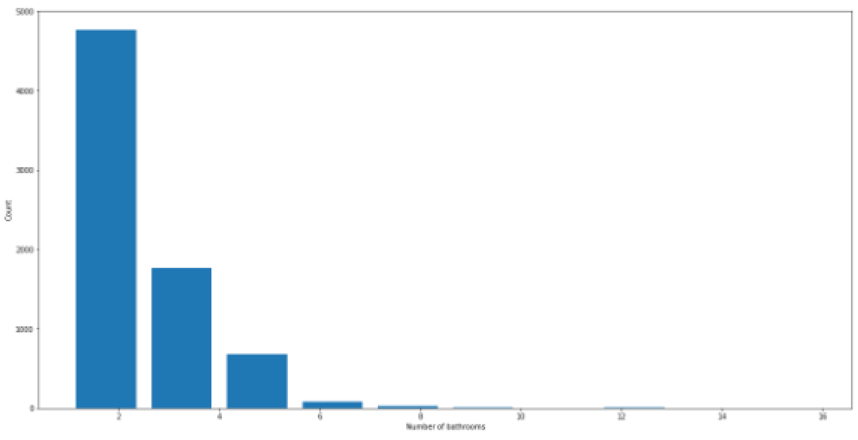



```
def remove_bhk_outliers(df):
    exclude_indices = np.array([])
    for location, location_df in df.groupby('location'):
        bhk_stats = {}
        for bhk, bhk_df in location_df.groupby('bhk'):
            bhk_stats[bhk] = {
```

```
                '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')
df5 = remove_bhk_outliers(df4)
df5.bath.unique()
df5[df5.bath>10]
plt.hist(df5.bath,rwidth=0.8)
plt.xlabel("Number of bathrooms")
plt.ylabel("Count")
```

out:

```
Text(0, 0.5, 'Count')
```




```
df5[(df5.bath > df5.bhk+2)]
df6 = df5[~(df5.bath > df5.bhk+2)]
df6.head()
df7 = df6.drop(['price_per_sqft'],axis='columns')
```

df7.head()

out:

	location	total_sqft	bath	price	bhk
0	1st Block Jayanagar	2850.0	4.0	428.0	4
1	1st Block Jayanagar	1630.0	3.0	194.0	3
2	1st Block Jayanagar	1875.0	2.0	235.0	3
3	1st Block Jayanagar	1200.0	2.0	130.0	3
4	1st Block Jayanagar	1235.0	2.0	148.0	2

dummies = pd.get_dummies(df7.location)

dummies.head()

df8 = pd.concat([df7,dummies.drop('other',axis='columns')],axis='columns')

df8.drop('location',axis='columns',inplace=True)

df8.head()

out:

	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	Whitefielc
0	1st Block Jayanagar	2850.0	4.0	428.0	4	1	0	0	0	0	...	0	0	0	0	(
1	1st Block Jayanagar	1630.0	3.0	194.0	3	1	0	0	0	0	...	0	0	0	0	(
2	1st Block Jayanagar	1875.0	2.0	235.0	3	1	0	0	0	0	...	0	0	0	0	(
3	1st Block Jayanagar	1200.0	2.0	130.0	3	1	0	0	0	0	...	0	0	0	0	(
4	1st Block Jayanagar	1235.0	2.0	148.0	2	1	0	0	0	0	...	0	0	0	0	(

5 rows × 246 columns

x = df8.drop('price',axis=1)

y = df8['price']

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=101)

X_train.shape, X_test.shape, y_train.shape, y_test.shape

out:

((5860, 244), (1465, 244), (5860,), (1465,))

from sklearn.linear_model import LinearRegression

lr = LinearRegression()

lr.fit(X_train,y_train)

lr.score(X_test,y_test)

out:

0.8629898728935371

```
pred = lr.predict(X_test)
pred
```

out:

```
array([ 32.66700357, 291.55286051,  69.36556057, ..., 112.8263403 ,
        43.43288776, 135.77405539])
```

```
y_test
```

out:

```
7892     41.745
3357    380.000
126      75.000
3767    175.000
4871     80.000
         ...
9870    120.000
9802     87.000
2955    113.000
917      65.000
748      59.520
Name: price, Length: 1465, dtype: float64
```

```
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.predict([X])[0]
predict_price('1st Phase JP Nagar',1000, 2, 2)
```

out:

85.2974569797724

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

out:

81.70512816315643

```
predict_price('Indira Nagar',1400, 2, 3)
```

out:

217.40708528156847

Conclusion

The process presented is used that have been chosen according to their similarities in terms of presentation of the estates and if they give the same information about them.

Linear regression is one of the most well- known algorithms in statistics and machine learning. The objective of a linear regression model is to find a relationship between one or more features (independent/explanatory/predictor variables) and a continuous target variable (dependent/response) variable. If there is only one feature, the model is simple linear regression and if there are multiple features, the model is multiple linear regression

With the help of box plots, we can check for outliers. If present, we can remove outliers and check the model's performance for improvement. This technique helps in developing a model that have less variance and more stability.

We can build models through advanced techniques namely random forests, neural networks, and particle swarm optimization to improve the accuracy of predictions. In simple linear regression we attempt to minimize the error.

Data collected from a big urban city like Bengaluru would not be applicable in a rural city, as for equal value of feature prices, which will be comparatively higher in the urban area.