Project submitted by:
Shibangshu Pal
B.Tech, Electronics and Communication Engineering
National Institute of Technology, Durgapur
Email : spaloct21002@gmail.com

Teacher: Mr. Ameen Manna
Email: mail.ameenmanna@gmail.com

Application of Regression model for predicting rents of flat across various metropolis in India

---

Introduction

With the emergence of the Corporate sector in India, there have been a rising demand for rented apartments across the country, especially in the metropolitan cities of India, which includes Delhi, Kolkata(erstwhile Calcutta), Mumbai(erstwhile Bombay) and Chennai(erstwhile Madras). Other than that, cities like Bangalore and Hyderabad has been a hub of several IT firms and thus, the quest for finding a perfect place to stay within the limits of the pocket has been of utmost concern for several millennials coming to these cities for their jobs. Moreover, buying an apartment in these cities have become an expensive affair and with the volatility of being transferred or job switching to various cities, rented apartments have become a hot market at present.

Our machine learning model, with the available data, shall try to predict the cost of rent of an apartment with certain specifications with the help of available data collected across various cities of India as mentioned above.

Understanding the Data

The following headings of the columns are present in the dataset whose meanings are presented as follows:

BHK: Number of Bedrooms, Hall, Kitchen.

Rent: Rent of the Houses/Apartments/Flats.

Size: Size of the Houses/Apartments/Flats in Square Feet.

Floor: Houses/Apartments/Flats situated in which Floor and Total Number of Floors [Example: Ground(0) , Upper Basement(0) and Lower Basement (0) other than that 1,2,.. out of 2, 3, 5, etc.]

Area Type: Size of the Houses/Apartments/Flats calculated on either Super Area or Carpet Area or Build area.

Area Locality: Locality of the Houses/Apartments/Flats**.**

City: City where the Houses/Apartments/Flats are Located.

Furnishing Status: Furnishing Status of the Houses/Apartments/Flats, either it is Furnished or Semi-Furnished or Unfurnished.

Tenant Preferred: Type of Tenant Preferred by the Owner or Agent(Bachelor/family)

Bathroom: Number of bathrooms in the flat.

Point of Contact: Whom should you contact for more information regarding the Houses/Apartments/Flats.

The Data:

The link to the data is contained in a CSV file which is uploaded to Github. Its link is enclosed as follows:

https://github.com/shibangshu-pal/DataScienceCourse-Major-Project-1/blob/main/House_Rent_Dataset.csv

The dataset was resourced from the following open source link. I would like to sincerely thank Mr. Sourav Banerjee, for uploading the dataset to an open source for academic and research purposes.

https://www.kaggle.com/datasets/iamsouravbanerjee/house-rent-prediction-dataset

Creation of the Dataset

```
import pandas as pd
df=pd.read_csv('https://raw.githubusercontent.com/shibangshu-pal/DataScienceCourse-Major-Project-1/main/House_Rent_Dataset.csv')
# accessing the csv dataset from GitHub
```

Working on the Data

Here, we have worked in extracting the data in the various columns of the original dataframe for the purpose of exploratory data analysis and use in the machine learning model. The various blocks of code are shown as follows, along with the modifications and conversion of certain values in certain columns to integer values for analysis. The detailed descriptions of the modifications made in the data are discussed at a later stage.

```
import numpy as np
df_floor = np.array(df.iloc[:4746,3].values)
floors = []
for i in df_floor:
    c=''
    for j in i:
      if j!=' ':
        c = c+j
      else:
        break
    floors.append(int(c))
#floors array stores the data of the floor in which the flat is situated in
```

```
furnishing = []
#0 for unfurnished
#1 for semi-furnished
#2 for fully furnished
fur = df.iloc[:4746,7].values
for i in fur:
  if i == 'Unfurnished':
    furnishing.append(0)
  elif i == 'Semi-Furnished':
    furnishing.append(1)
  else:
    furnishing.append(2)
#furnishing array grades the flats on type of furnishing
```

```
city = np.array(df.iloc[:,6].values)
#array containing cities
```

```
bhk = np.array(df.iloc[:,0].values)
b_room = np.array(df.iloc[:,9].values)
#bhk - stores the bhk of the apartment
#b_room - stores the number of bathrooms in the apartment
```

```
#storing rent
rent = np.array(df.iloc[:,1].values)
```

Exploratory Data Analysis

For the purpose of exploratory data analysis, we have considered the following:

1. Average rent in the various metropolitan cities
2. BHK wise average rent in various cities of India as in the dataset
3. Availability of flats for rent BHK wise in various cities
4. Furnishing type wise average rent in various cities

Now we shall explore them one by one.

1. **Average rent in the various metropolitan cities**

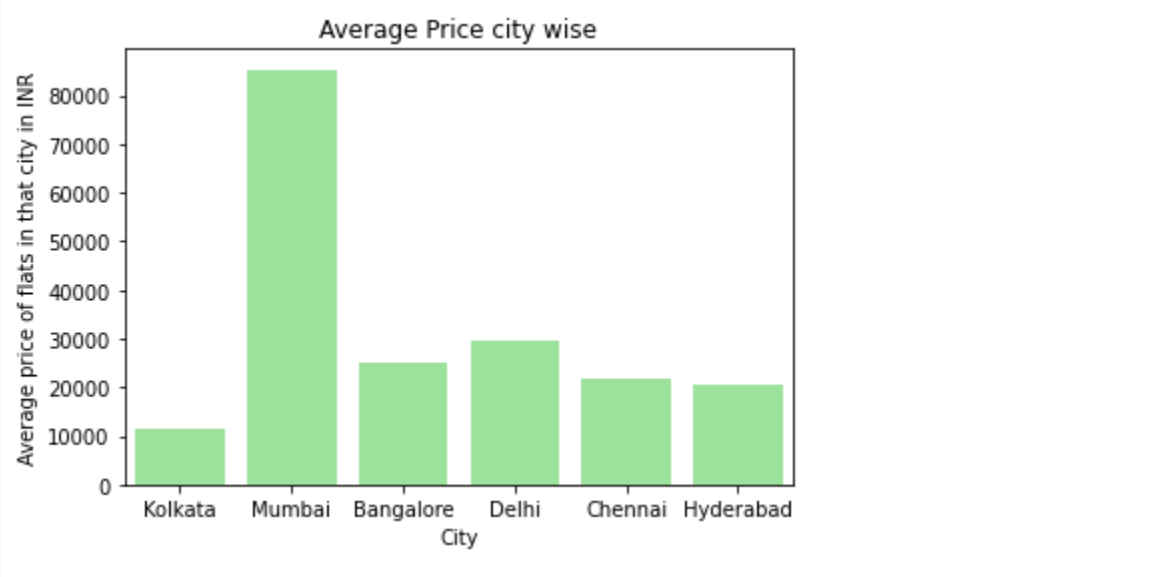
   Code:

```
#Exploratory Data Analysis
#1. We calculate the average rent of flats in the cities
import matplotlib.pyplot as plt
import seaborn as sns

avg = [0,0,0,0,0,0]
counter = [0,0,0,0,0,0]
#the locations respectively stand for Kolkata, Mumbai, Bangalore, Delhi,
Chennai and Hyderabad respectively
for i in range(0,4746):
  if city[i] == 'Kolkata':
        avg[0] += rent[i]
        counter[0] += 1
  elif city[i] == 'Mumbai':
        avg[1] += rent[i]
        counter[1] += 1
  elif city[i] == 'Bangalore':
        avg[2] += rent[i]
        counter[2] += 1
  elif city[i] == 'Delhi':
        avg[3] += rent[i]
```

```
        counter[3] += 1
  elif city[i] == 'Chennai':
        avg[4] += rent[i]
        counter[4] += 1
  else:
        avg[5] += rent[i]
        counter[5] += 1
for j in range(0,6):
  avg[j] = round(avg[j]/counter[j],2)

#print("The average in various cities are ",avg)

sns.barplot(p,avg, color= 'lightgreen')
plt.xlabel("City")
plt.ylabel("Average price of flats in that city in INR")
plt.title("Average Price city wise")
```

Output:

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  FutureWarning
Text(0.5, 1.0, 'Average Price city wise')
```



2. **BHK wise average rent in various cities of India as in the dataset**

Code and output

```
# 2. We calculate the average price of various BHK flats in the cities
bhk_uni = np.array(df['BHK'].unique())
bhk_uni = np.sort(bhk_uni)
#print(bhk_uni)
bhk_calc =
[[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0
,0,0]]
counter =
[[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0,0,0],[0,0,0,0
,0,0]]
for i in range(4746):
  if city[i] == 'Kolkata':
    if bhk[i] == 1:
      bhk_calc[0][0]+=rent[i]
      counter[0][0]+=1
    elif bhk[i] == 2:
      bhk_calc[0][1]+=rent[i]
      counter[0][1]+=1
    elif bhk[i] == 3:
      bhk_calc[0][2]+=rent[i]
      counter[0][2]+=1
    elif bhk[i] == 4:
      bhk_calc[0][3]+=rent[i]
      counter[0][3]+=1
    elif bhk[i] == 5:
      bhk_calc[0][4]+=rent[i]
      counter[0][4]+=1
    else:
      bhk_calc[0][5]+=rent[i]
      counter[0][5]+=1
  elif city[i] == 'Mumbai':
    if bhk[i] == 1:
      bhk_calc[1][0]+=rent[i]
      counter[1][0]+=1
    elif bhk[i] == 2:
      bhk_calc[1][1]+=rent[i]
      counter[1][1]+=1
    elif bhk[i] == 3:
      bhk_calc[1][2]+=rent[i]
      counter[1][2]+=1
    elif bhk[i] == 4:
      bhk_calc[1][3]+=rent[i]
```

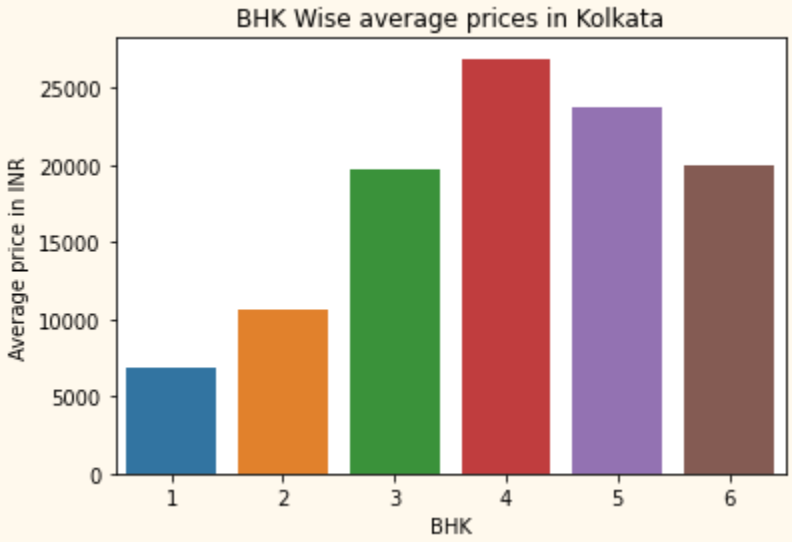
```
      counter[1][3]+=1
    elif bhk[i] == 5:
      bhk_calc[1][4]+=rent[i]
      counter[1][4]+=1
    else:
      bhk_calc[1][5]+=rent[i]
      counter[1][5]+=1
  elif city[i] == 'Bangalore':
    if bhk[i] == 1:
      bhk_calc[2][0]+=rent[i]
      counter[2][0]+=1
    elif bhk[i] == 2:
      bhk_calc[2][1]+=rent[i]
      counter[2][1]+=1
    elif bhk[i] == 3:
      bhk_calc[2][2]+=rent[i]
      counter[2][2]+=1
    elif bhk[i] == 4:
      bhk_calc[2][3]+=rent[i]
      counter[2][3]+=1
    elif bhk[i] == 5:
      bhk_calc[2][4]+=rent[i]
      counter[2][4]+=1
    else:
      bhk_calc[2][5]+=rent[i]
      counter[2][5]+=1
  elif city[i] == 'Delhi':
    if bhk[i] == 1:
      bhk_calc[3][0]+=rent[i]
      counter[3][0]+=1
    elif bhk[i] == 2:
      bhk_calc[3][1]+=rent[i]
      counter[3][1]+=1
    elif bhk[i] == 3:
      bhk_calc[3][2]+=rent[i]
      counter[3][2]+=1
    elif bhk[i] == 4:
      bhk_calc[3][3]+=rent[i]
      counter[3][3]+=1
    elif bhk[i] == 5:
      bhk_calc[3][4]+=rent[i]
      counter[3][4]+=1
    else:
      bhk_calc[3][5]+=rent[i]
      counter[3][5]+=1
  elif city[i] == 'Chennai':
```

```
    if bhk[i] == 1:
      bhk_calc[4][0]+=rent[i]
      counter[4][0]+=1
    elif bhk[i] == 2:
      bhk_calc[4][1]+=rent[i]
      counter[4][1]+=1
    elif bhk[i] == 3:
      bhk_calc[4][2]+=rent[i]
      counter[4][2]+=1
    elif bhk[i] == 4:
      bhk_calc[4][3]+=rent[i]
      counter[4][3]+=1
    elif bhk[i] == 5:
      bhk_calc[4][4]+=rent[i]
      counter[4][4]+=1
    else:
      bhk_calc[4][5]+=rent[i]
      counter[4][5]+=1
  else:
    if bhk[i] == 1:
      bhk_calc[5][0]+=rent[i]
      counter[5][0]+=1
    elif bhk[i] == 2:
      bhk_calc[5][1]+=rent[i]
      counter[5][1]+=1
    elif bhk[i] == 3:
      bhk_calc[5][2]+=rent[i]
      counter[5][2]+=1
    elif bhk[i] == 4:
      bhk_calc[5][3]+=rent[i]
      counter[5][3]+=1
    elif bhk[i] == 5:
      bhk_calc[5][4]+=rent[i]
      counter[5][4]+=1
    else:
      bhk_calc[5][5]+=rent[i]
      counter[5][5]+=1
for i in range(6):
  for j in range(6):
    if counter[i][j]!=0:
      bhk_calc[i][j] = bhk_calc[i][j]/counter[i][j]
```

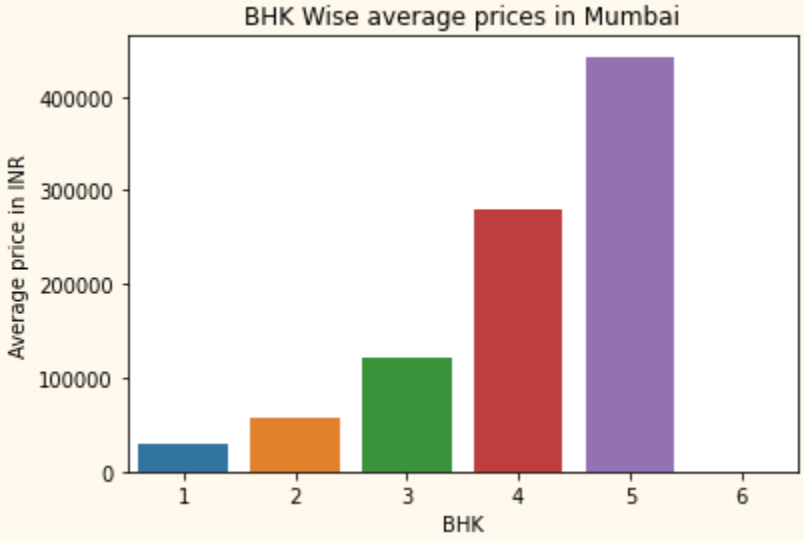
```
p1 = np.array(df['Furnishing Status'].unique())
```

Plotting the graphs,

```
sns.barplot(bhk_uni,bhk_calc[0])
plt.title('BHK Wise average prices in Kolkata')
plt.xlabel('BHK')
plt.ylabel('Average price in INR')
```

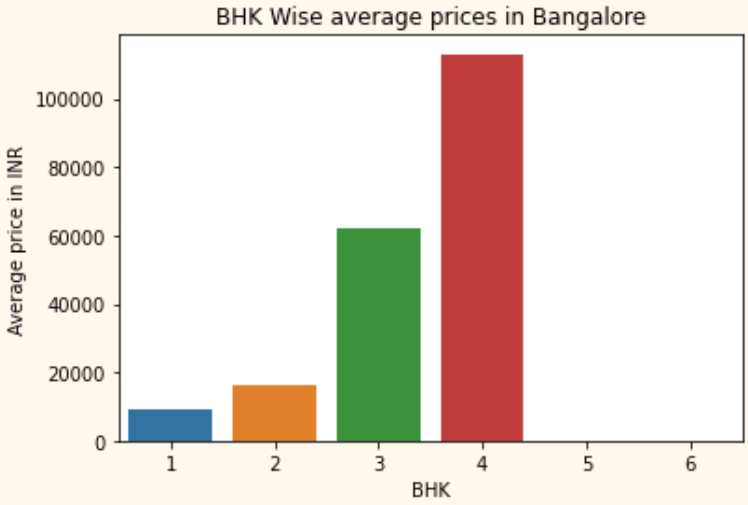



```
sns.barplot(bhk_uni,bhk_calc[1])
plt.title('BHK Wise average prices in Mumbai')
plt.xlabel('BHK')
plt.ylabel('Average price in INR')
```

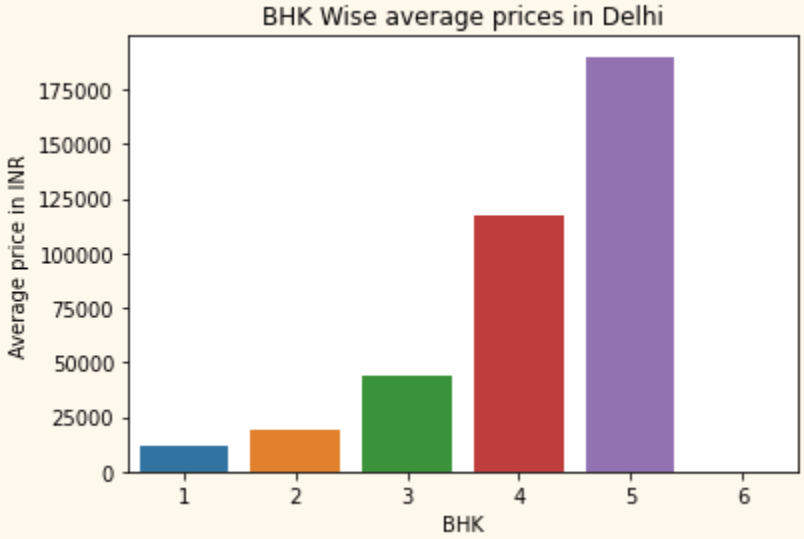



```
sns.barplot(bhk_uni,bhk_calc[2])
plt.title('BHK Wise average prices in Bangalore')
plt.xlabel('BHK')
```

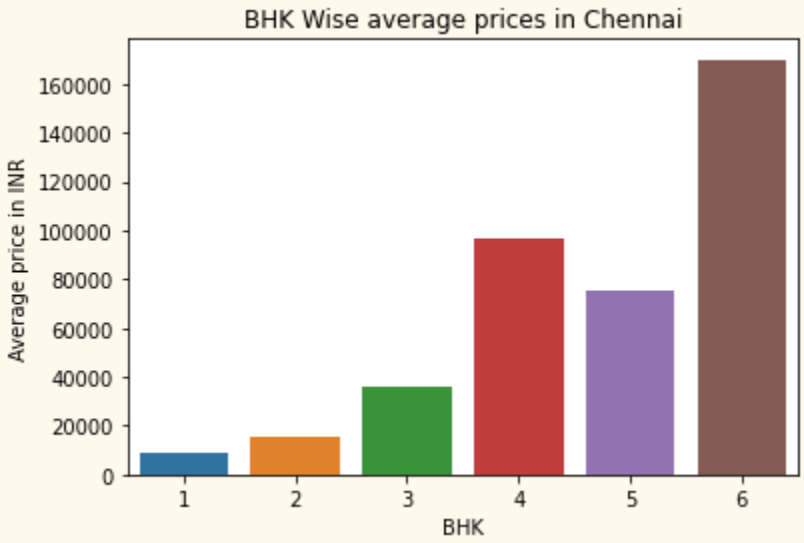
```
plt.ylabel('Average price in INR')
```



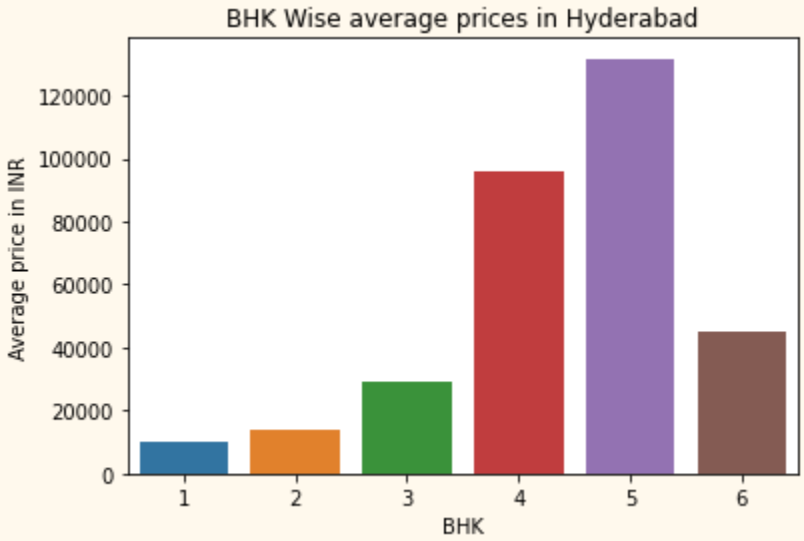

```
sns.barplot(bhk_uni,bhk_calc[3])
plt.title('BHK Wise average prices in Delhi')
plt.xlabel('BHK')
plt.ylabel('Average price in INR')
```




```
sns.barplot(bhk_uni,bhk_calc[4])
plt.title('BHK Wise average prices in Chennai')
plt.xlabel('BHK')
plt.ylabel('Average price in INR')
```



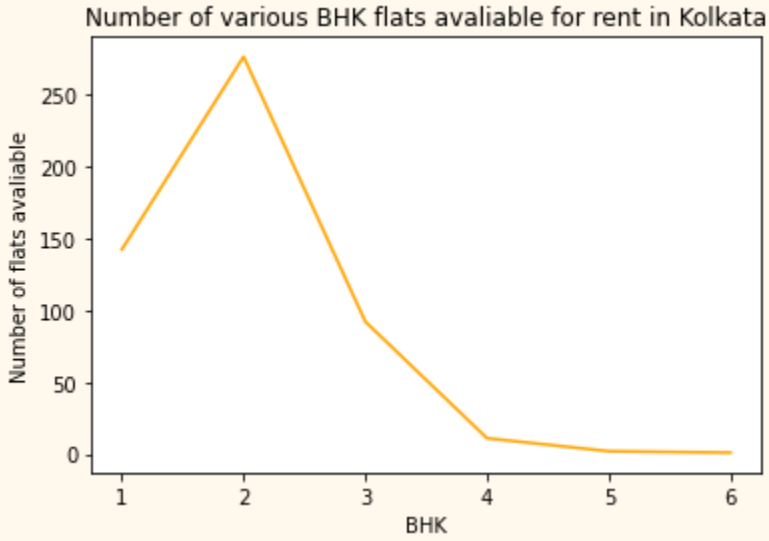

```
sns.barplot(bhk_uni,bhk_calc[5])
plt.title('BHK Wise average prices in Hyderabad')
plt.xlabel('BHK')
plt.ylabel('Average price in INR')
```



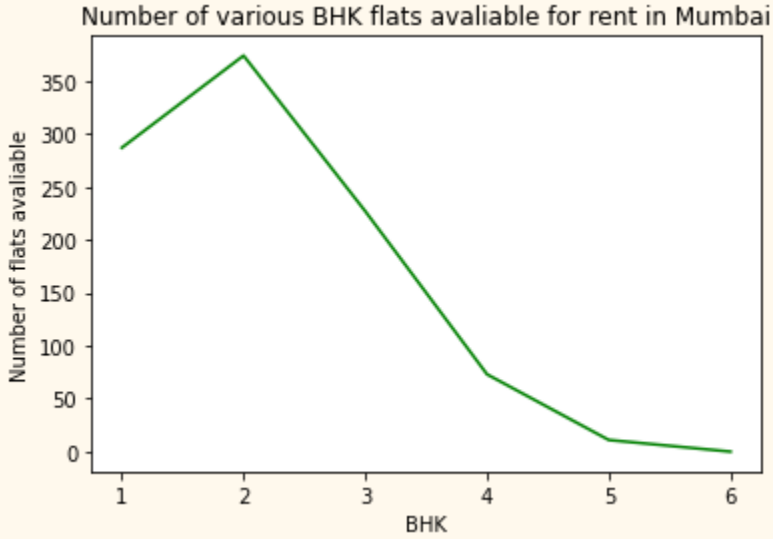
3. Availability of flats for rent BHK wise in various cities

Code and outputs

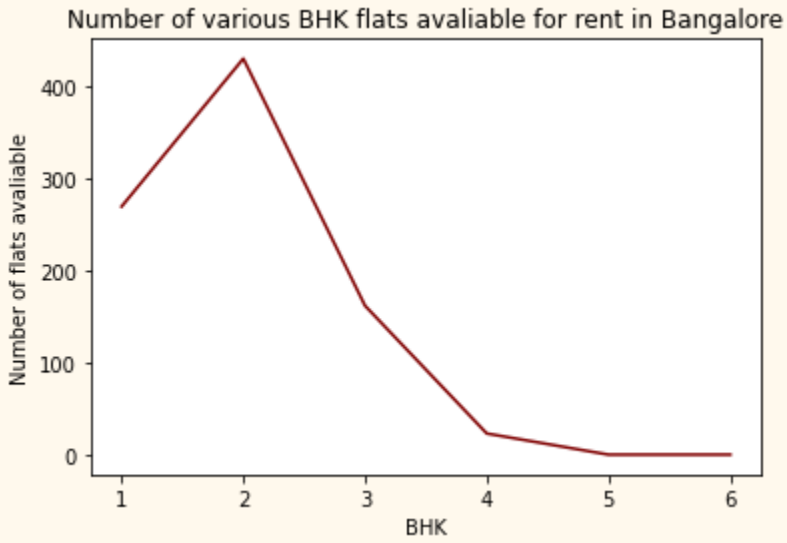
```
#2. BHK Wise distribution of flats in various cities
plt.xlabel('BHK')
plt.ylabel('Number of flats avaliable')
sns.lineplot(bhk_uni,counter[0],color ='orange')
plt.title('Number of various BHK flats avaliable for rent in Kolkata')
```



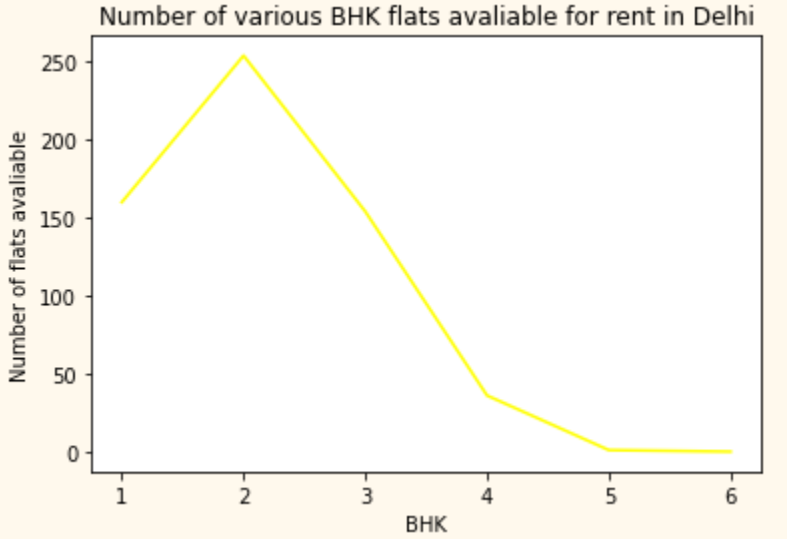

```
plt.xlabel('BHK')
plt.ylabel('Number of flats avaliable')
sns.lineplot(bhk_uni,counter[1],color ='green')
plt.title('Number of various BHK flats avaliable for rent in Mumbai')
```



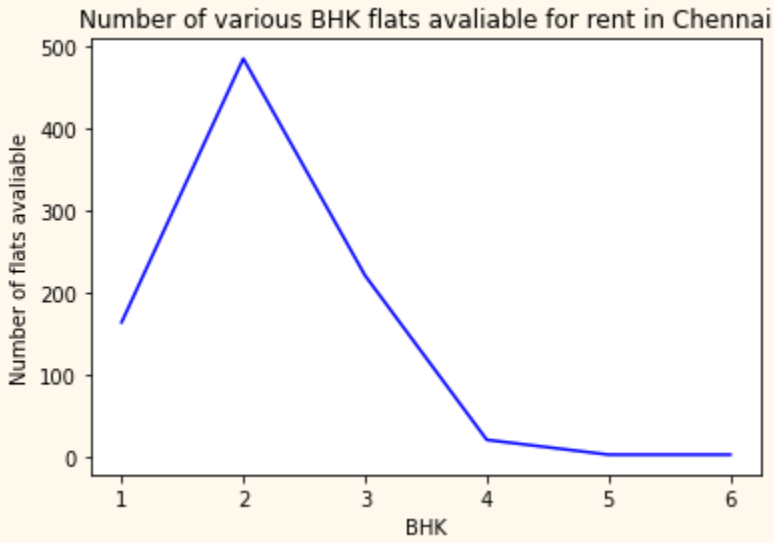

```
plt.xlabel('BHK')
plt.ylabel('Number of flats avaliable')
sns.lineplot(bhk_uni,counter[2],color ='maroon')
plt.title('Number of various BHK flats avaliable for rent in Bangalore')
```



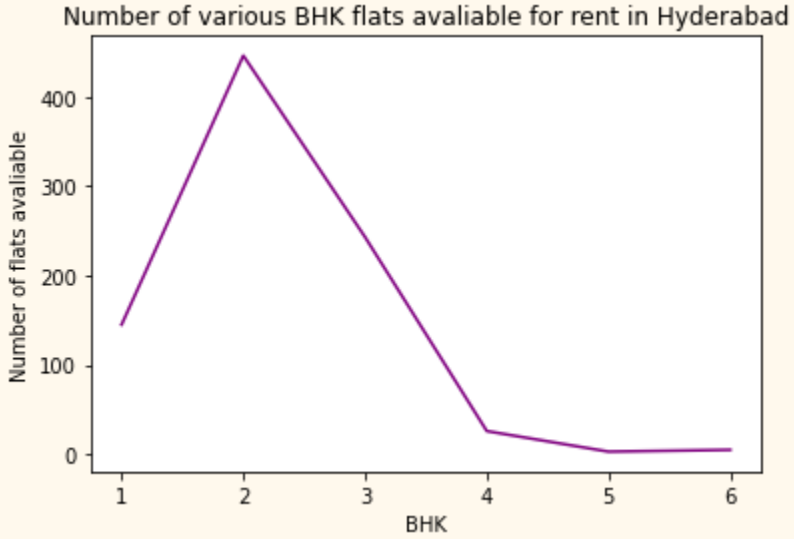

```
plt.xlabel('BHK')
plt.ylabel('Number of flats avaliable')
sns.lineplot(bhk_uni,counter[3],color ='yellow')
plt.title('Number of various BHK flats avaliable for rent in Delhi')
```




```
plt.xlabel('BHK')
plt.ylabel('Number of flats avaliable')
sns.lineplot(bhk_uni,counter[4],color ='blue')
plt.title('Number of various BHK flats avaliable for rent in Chennai')
```




```
plt.xlabel('BHK')
plt.ylabel('Number of flats avaliable')
sns.lineplot(bhk_uni,counter[5],color ='purple')
plt.title('Number of various BHK flats avaliable for rent in Hyderabad')
```



4. **Furnishing type wise average rent in various cities**

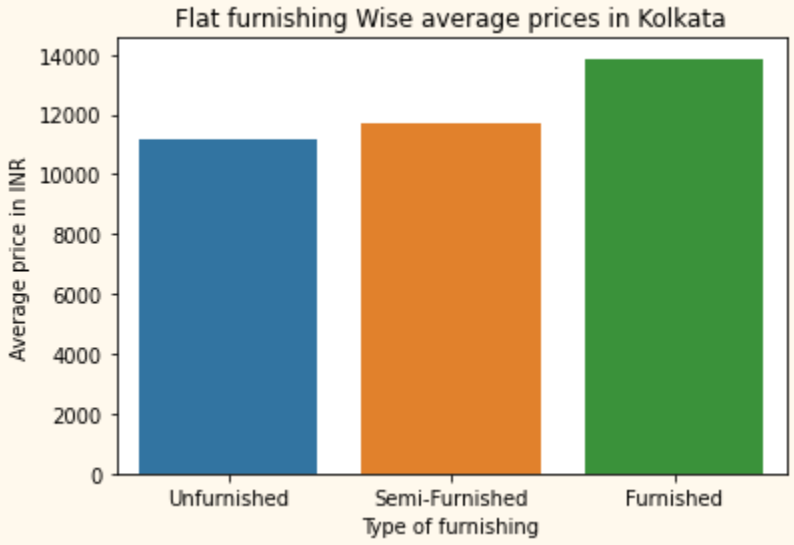
Code and Outputs

```
#4. We calculate the average pricing of various types of furnished apartments
across the various cities
bhk_calc = [[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0]]
counter = [[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0]]
for i in range(4746):
  if city[i] == 'Kolkata':
```

```
    if furnishing[i] == 0:
      bhk_calc[0][0]+=rent[i]
      counter[0][0]+=1
    elif furnishing[i] == 1:
      bhk_calc[0][1]+=rent[i]
      counter[0][1]+=1
    else:
      bhk_calc[0][2]+=rent[i]
      counter[0][2]+=1

  elif city[i] == 'Mumbai':
    if furnishing[i] == 0:
      bhk_calc[1][0]+=rent[i]
      counter[1][0]+=1
    elif furnishing[i] == 1:
      bhk_calc[1][1]+=rent[i]
      counter[1][1]+=1
    else:
      bhk_calc[1][2]+=rent[i]
      counter[1][2]+=1

  elif city[i] == 'Bangalore':
    if furnishing[i] == 0:
      bhk_calc[2][0]+=rent[i]
      counter[2][0]+=1
    elif furnishing[i] == 1:
      bhk_calc[2][1]+=rent[i]
      counter[2][1]+=1
    else:
      bhk_calc[2][2]+=rent[i]
      counter[2][2]+=1

  elif city[i] == 'Delhi':
    if furnishing[i] == 0:
      bhk_calc[3][0]+=rent[i]
      counter[3][0]+=1
    elif furnishing[i] == 1:
      bhk_calc[3][1]+=rent[i]
      counter[3][1]+=1
    else:
      bhk_calc[3][2]+=rent[i]
      counter[3][2]+=1
  elif city[i] == 'Chennai':
    if furnishing[i] == 0:
      bhk_calc[4][0]+=rent[i]
      counter[4][0]+=1
```

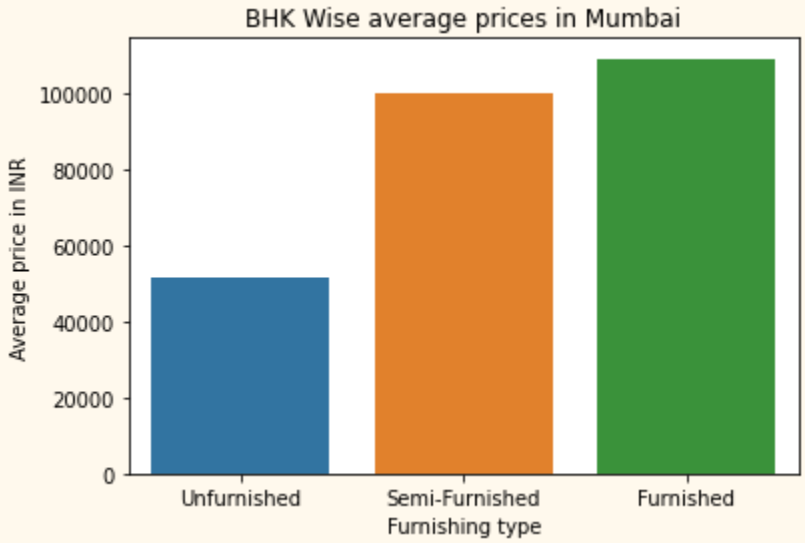
```
    elif furnishing[i] == 1:
      bhk_calc[4][1]+=rent[i]
      counter[4][1]+=1
    else:
      bhk_calc[4][2]+=rent[i]
      counter[4][2]+=1
  else:
    if furnishing[i] == 0:
      bhk_calc[5][0]+=rent[i]
      counter[5][0]+=1
    elif furnishing[i] == 1:
      bhk_calc[5][1]+=rent[i]
      counter[5][1]+=1
    else:
      bhk_calc[5][2]+=rent[i]
      counter[5][2]+=1

for i in range(3):
  for j in range(3):
    if counter[i][j]!=0:
      bhk_calc[i][j] = bhk_calc[i][j]/counter[i][j]
```

```
sns.barplot(p1,bhk_calc[0])
plt.title('Flat furnishing Wise average prices in Kolkata')
plt.xlabel('Type of furnishing')
plt.ylabel('Average price in INR')
```

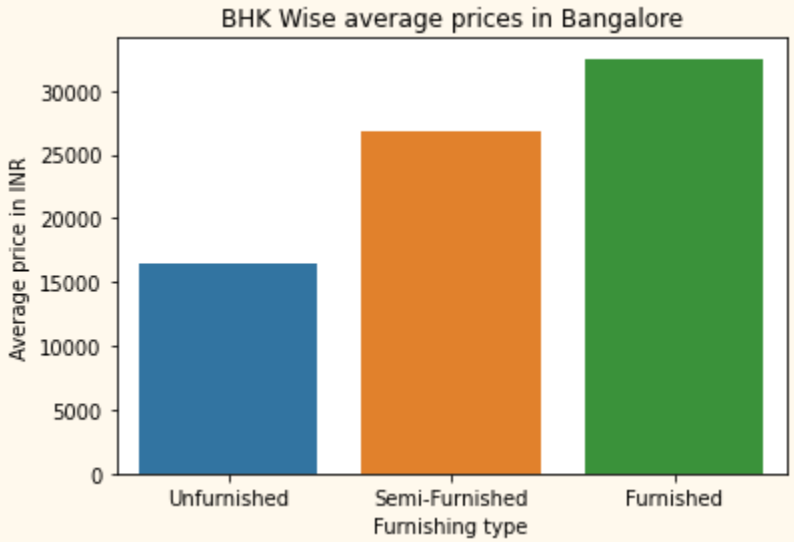



```
sns.barplot(p1,bhk_calc[1])
plt.title('BHK Wise average prices in Mumbai')
plt.xlabel('Furnishing type')
```

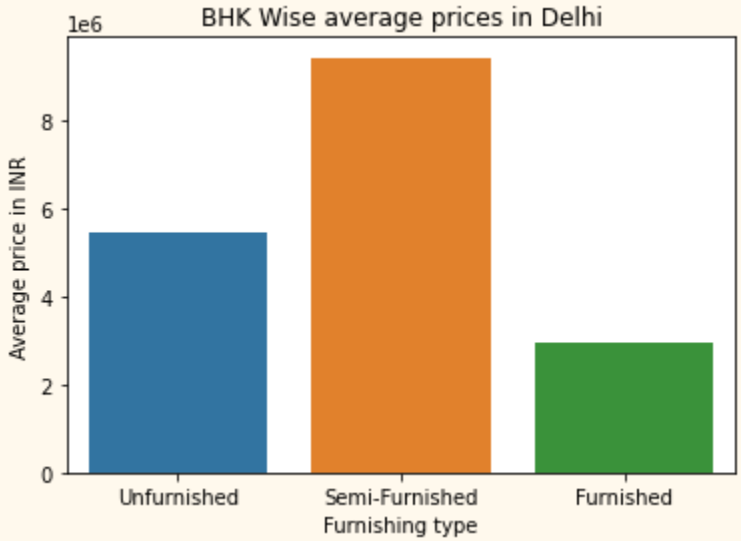
```
plt.ylabel('Average price in INR')
```



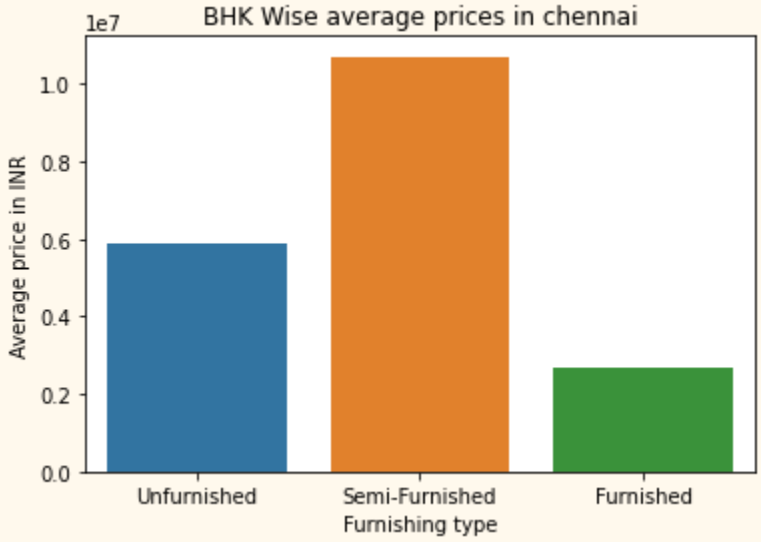

```
sns.barplot(p1,bhk_calc[2])
plt.title('BHK Wise average prices in Bangalore')
plt.xlabel('Furnishing type')
plt.ylabel('Average price in INR')
```



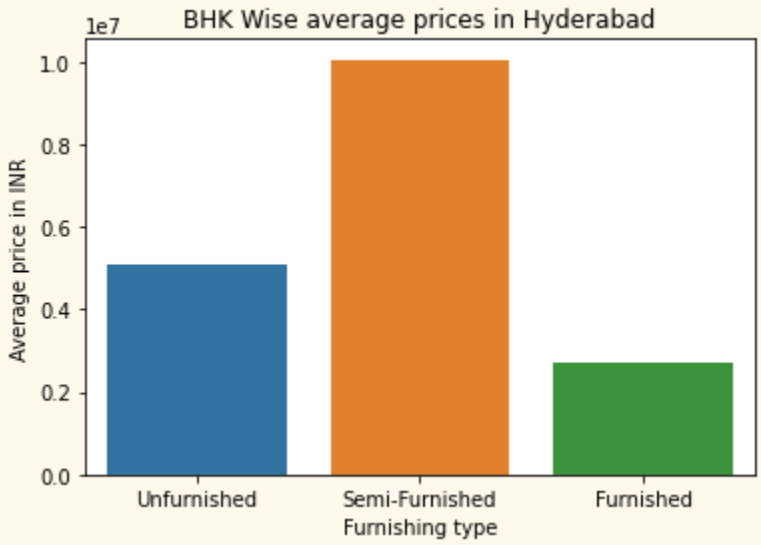

```
sns.barplot(p1,bhk_calc[3])
plt.title('BHK Wise average prices in Delhi')
plt.xlabel('Furnishing type')
plt.ylabel('Average price in INR')
```




```
sns.barplot(p1,bhk_calc[4])
plt.title('BHK Wise average prices in chennai')
plt.xlabel('Furnishing type')
plt.ylabel('Average price in INR')
```




```
sns.barplot(p1,bhk_calc[5])
plt.title('BHK Wise average prices in Hyderabad')
plt.xlabel('Furnishing type')
plt.ylabel('Average price in INR')
```



Such exploratory data analysis was done before further proceeding with the machine learning model.

Normalization

With the type of data points available for the various columns, normalization of the data was not required.

Application of the Logistic Regression model

Columns considered for input**: BHK, size, Floor, furnishing, City**

Columns considered as output **: Rent**

Preliminary modifications made in the dataset

The values contained in the various columns considered for input were modified to give mathematical numeation to them as follows. It is to be also noted that we created a dataframe df_res containing the first five columns as input columns and last column as output column for easily applying our regression model. The removal was done for the fact that we considered only five columns that might be more significant in prediction of the apartment rent, the columns being mentioned above.

The modifications done are described as follows in the next page

Sl. No	Column name	Modifications in Data values made
1	BHK	None
2	Bathroom	None
3	Floor	Only the floor in which the flat is located is considered and not the total number of floors in that apartment
4	Furnishing	0 for Unfurnished 1 for Semi-Furnished 2 for Furnished
5	City	0 for Kolkata 1 for Mumbai 2 for Bangalore 3 for Delhi 4 for Chennai 5 for Hyderabad

Methodology followed for developing Machine Learning model

We created x_train and y_train lists containing approximately 75% of the data points for input and output part of the data, using the method train_test_split() of sklearn.model_selection which was used to train our logistic regression model. The rest 25% of the data was in x_test and y_test for the purpose of testing the data. Apart from that, I have also considered individual prediction for the machine learning model and calculated the accuracy of the model.

Code for the Machine learning Model

```
df2 = df
del df2['Area Locality']
del df2['Area Type']
del df2['Tenant Preferred']
del df2['Point of Contact']
del df2['Size']
#deleting unnecessary columns
```

```
furnishing = []
#0 for unfurnished
#1 for semi-furnished
#2 for fully furnished
fur = df.iloc[:4746,7].values
```

```
for i in fur:
  if i == 'Unfurnished':
    furnishing.append(0)
  elif i == 'Semi-Furnished':
    furnishing.append(1)
  else:
    furnishing.append(2)
#furnishing array grades the flats on type of furnishing
```

```
arr= []
#Modifying cities column and creating array for the final dataframe to work on
for i in range(4746):
  if(city[i]=='Kolkata'):
    arr.append([bhk[i],floors[i],furnishing[i],0,b_room[i],rent[i]])
  elif(city[i]=='Mumbai'):
    arr.append([bhk[i],floors[i],furnishing[i],1,b_room[i],rent[i]])
  elif(city[i]=='Bangalore'):
    arr.append([bhk[i],floors[i],furnishing[i],2,b_room[i],rent[i]])
  elif(city[i]=='Delhi'):
    arr.append([bhk[i],floors[i],furnishing[i],3,b_room[i],rent[i]])
  elif(city[i]=='Chennai'):
    arr.append([bhk[i],floors[i],furnishing[i],4,b_room[i],rent[i]])
  else:
    arr.append([bhk[i],floors[i],furnishing[i],5,b_room[i],rent[i]])
```

```
# We will use df_res for our final ML model as we are not considering the rest
of the columns
df_res = pd.DataFrame(data = arr, columns = ['BHK','Floor
No.','Furnishing','City','Bathroom','Rent'])
df_res
```

```
#Preparation for application of ML Model
#input - BHK, floor location, city, number of bathrooms
#output - rent
x = df_res.iloc[:,:5].values    #input
y = df_res.iloc[:,5].values          #output
#training and testing variables
from sklearn.model_selection import train_test_split
x_train,x_test, y_train,y_test = train_test_split(x,y,random_state = 0)

#Normalization
#Normalization here may reduce the accuracy even further leading to worse
output so avoided
```

```
#Application of Logistic Regression
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train,y_train)
```

Yielding Output from the model

1. Bulk prediction

```
#Getting output from model
y_pred = model.predict(x_test)
y_pred
```

The output generated by this code was like

```
array([18000, 15000, 15000, ..., 15000, 15000, 15000])
```

2. Individual prediction

```
#individual prediction
#BHK,floor,furnishing,city,bathroom
inp =[]
print('Input values for the following fields')
a1 = (int)(input('Enter the required number of BHK\n'))
inp.append(a1)
a2 =(int)(input('Enter the floor number in which you would like to have your
flat in, eg. 0 for ground,1 for first,2 for second and so on..\n'))
inp.append(a2)
a3 = (int)(input('Enter the type of furnishing: \n 0 for unfurnished\n 1 for
semi-furnished\n 2 for furnished\n'))
inp.append(a3)
a4 = (int)(input('Enter the City location:\n0 for Kolkata\n1 for Mumbai\n2 for
Bangalore\n3 for Delhi\n4 for Chennai\n5 for Hyderabad\n'))
inp.append(a4)
a5 = (int)(input('Enter the required number of bathrooms\n'))
inp.append(a5)
y = model.predict([inp]) #input must be 2 dimensional
print('The possible rent as predicted by our model is INR ',y[0])
```

The output(s) generated by this code was like

```
Input values for the following fields
Enter the required number of BHK
2
Enter the floor number in which you would like to have your flat in, eg. 0 for
ground,1 for first,2 for second and so on..
2
Enter the type of furnishing:
 0 for unfurnished
 1 for semi-furnished
 2 for furnished
2
Enter the City location:
0 for Kolkata
1 for Mumbai
2 for Bangalore
3 for Delhi
4 for Chennai
5 for Hyderabad
3
Enter the required number of bathrooms
1
The possible rent as predicted by our model is INR  15000
```

```
Input values for the following fields
Enter the required number of BHK
1
Enter the floor number in which you would like to have your flat in, eg. 0 for
ground,1 for first,2 for second and so on..
3
Enter the type of furnishing:
 0 for unfurnished
 1 for semi-furnished
 2 for furnished
2
Enter the City location:
0 for Kolkata
1 for Mumbai
2 for Bangalore
3 for Delhi
4 for Chennai
5 for Hyderabad
1
Enter the required number of bathrooms
2
The possible rent as predicted by our model is INR  20000
```

Finding the Accuracy of the model

```
#finding the accuracy percentage
from sklearn.metrics import accuracy_score
print('The accuracy of the model in percentage
was',accuracy_score(y_pred,y_test)*100)
```

The accuracy percentage yielded by the code was

```
The accuracy of the model in percentage was 6.571187868576242
```

Link to my GitHub account for referring to the code of this project

https://github.com/shibangshu-pal/DataScienceCourse-Major-Project-1

Inferences

Our dataset was fairly accurate in predicting the rent of flats of various specifications across the most important commercial and residential metropolis in India. However, it was observed that the accuracy of the model as calculated by accuracy_score() method was quite less, approximately 6.57%. Such a low accuracy score may be attributed to the factor that the original dataset had only 4746 rows, which may be considered quite low for real life Machine Learning models. However, this dataset and project and model built may serve as a small scale model for future real world projects that involve such a model, since this model is exhaustive in its size but not its applicability. Thus, my model is presented as a sample model for reference to future real life machine learning models of large scale and as we all know, small steps integrated can achieve a giant goal.