Delhi Property Price Prediction Project

Made by- Atul
Mail- 4uatulsharma@gmail.com
Phone- 9990220006

Importing Modules

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

Importing Data

In [2]:
```
dp1 = pd.read_csv("Delhi_v2.csv")
dp1.drop(columns='Unnamed: 0',inplace=True)
```

In [3]:
```
dp1.head()
```

Out[3]:

	price	Address	area	latitude	longitude	Bedrooms	Bathrooms	Balcony	Status
0	5600000.0	Noida Extension, Noida, Delhi NCR	1350.0	28.608850	77.460560	3.0	3.0	NaN	Under Construction
1	8800000.0	Sector 79, Gurgaon, Delhi NCR	1490.0	28.374236	76.952416	3.0	3.0	NaN	Ready to Move
2	16500000.0	Vaishali, Ghaziabad, Delhi NCR	2385.0	28.645769	77.385110	4.0	5.0	NaN	Ready to Move
3	3810000.0	Link Road, F Block, Sector 50, Noida, Uttar Pr...	1050.0	28.566914	77.436434	2.0	2.0	3.0	NaN
4	6200000.0	Jaypee Pavilion Court Sector 128, Noida, Secto...	1350.0	28.520732	77.356491	2.0	2.0	3.0	Ready to Move

In [4]:
```
dp1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7738 entries, 0 to 7737
Data columns (total 17 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   price             7738 non-null   float64
 1   Address           7738 non-null   object 
 2   area              7738 non-null   float64
 3   latitude          7738 non-null   float64
 4   longitude         7738 non-null   float64
 5   Bedrooms          7738 non-null   float64
 6   Bathrooms         7738 non-null   float64
 7   Balcony           5166 non-null   float64
 8   Status            7164 non-null   object 
 9   neworold          7738 non-null   object 
 10  parking           2612 non-null   float64
 11  Furnished_status  4124 non-null   object 
 12  Lift              1733 non-null   float64
 13  Landmarks         2759 non-null   object 
 14  type_of_building  7738 non-null   object 
 15  desc              7738 non-null   object 
 16  Price_sqft        7738 non-null   float64
dtypes: float64(10), object(7)
memory usage: 1.0+ MB
```

In [5]:
```
dp1.describe()
```

Out[5]:

	price	area	latitude	longitude	Bedrooms	Bathrooms	Balcony
count	7.738000e+03	7738.000000	7738.000000	7738.000000	7738.000000	7738.000000	5166.000000
mean	8.320635e+06	1409.506591	28.552092	77.273476	2.708193	2.501163	2.426442
std	7.223197e+06	718.929581	0.107420	0.180606	0.877026	0.867050	1.083677
min	1.700000e+06	501.000000	28.240023	76.884101	2.000000	2.000000	1.000000
25%	4.200000e+06	990.000000	28.455539	77.078590	2.000000	2.000000	2.000000
50%	6.000000e+06	1250.000000	28.574637	77.345320	3.000000	2.000000	2.000000
75%	9.500000e+06	1650.000000	28.642520	77.421054	3.000000	3.000000	3.000000
max	8.500000e+07	9500.000000	28.799748	77.688028	10.000000	10.000000	10.000000

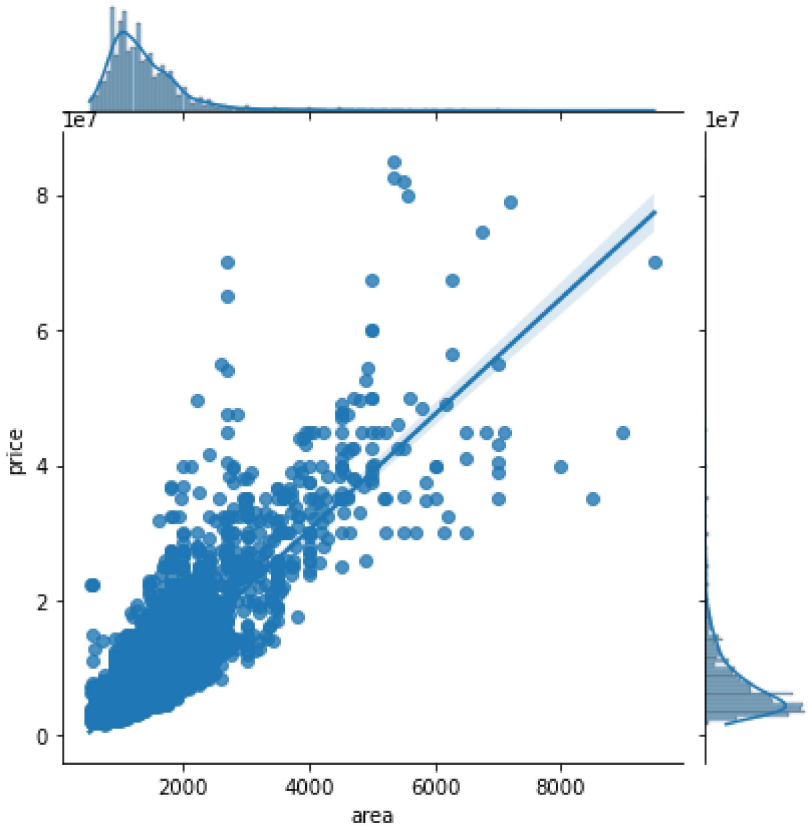
In [6]:
```
dp1.shape
```

Out[6]:
```
(7738, 17)
```

Analyzing data and drawing informational insights

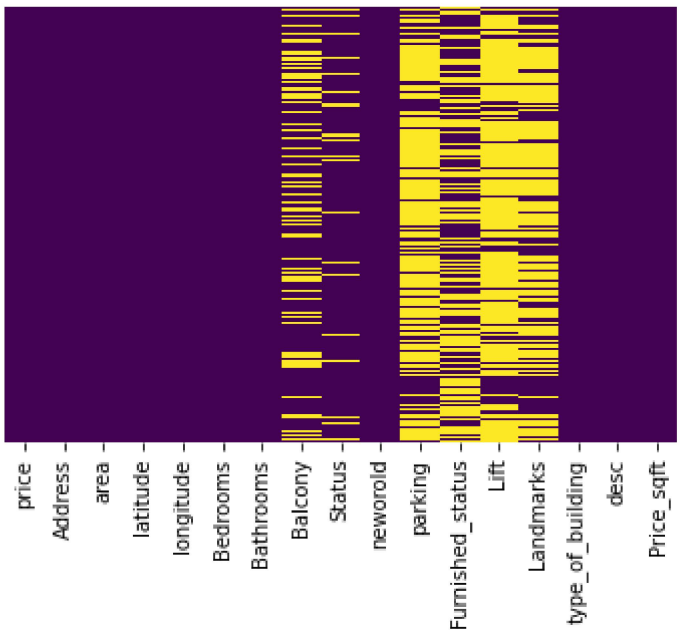
In [7]:
```
sns.jointplot(data=dp1,x=dp1["area"],y=dp1["price"],kind='reg')
```

Out[7]:
```
<seaborn.axisgrid.JointGrid at 0x26438702130>
```



In [8]:
```
sns.heatmap(dp1.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[8]: <AxesSubplot:>



In [9]:
```
dp1["parking"].count()
```

Out[9]: 2612

In [10]:
```
dp1.value_counts("Address")
```

Out[10]:

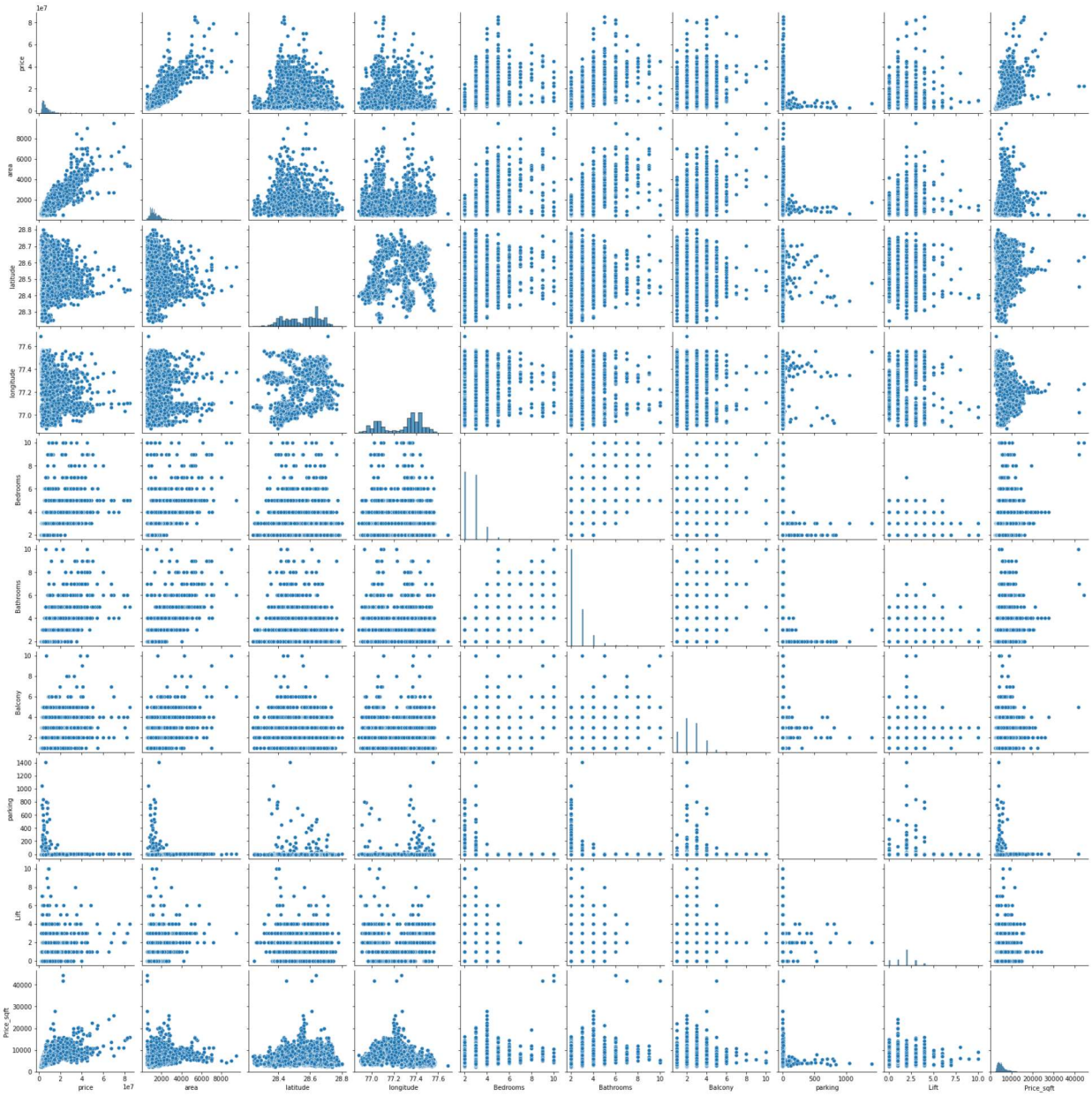
```
Address
Raj Nagar Extension, Ghaziabad, Delhi NCR                                163
Indirapuram, Ghaziabad, Delhi NCR                                        152
Greater Noida West, Greater Noida, Delhi NCR                             100
Noida Extension, Noida, Delhi NCR                                         63
Crossings Republik, Ghaziabad, Delhi NCR                                  61
                                                                         ...
Godrej woods sector 43, Sector 43, Noida, Delhi NCR                        1
Golf Course Road, Gurgaon, Delhi NCR                                       1
Golf Course Road, Gurgaon, South City 1, Gurgaon, Delhi NCR                1
Golf course Ex-road sec 68 gurgaon, Sector 68, Gurgaon, Delhi NCR          1
zeta 1, Zeta 1, Greater Noida, Delhi NCR                                   1
Length: 4145, dtype: int64
```

In [11]:

```
sns.pairplot(dp1)
```

Out[11]:

```
<seaborn.axisgrid.PairGrid at 0x26431ed8850>
```

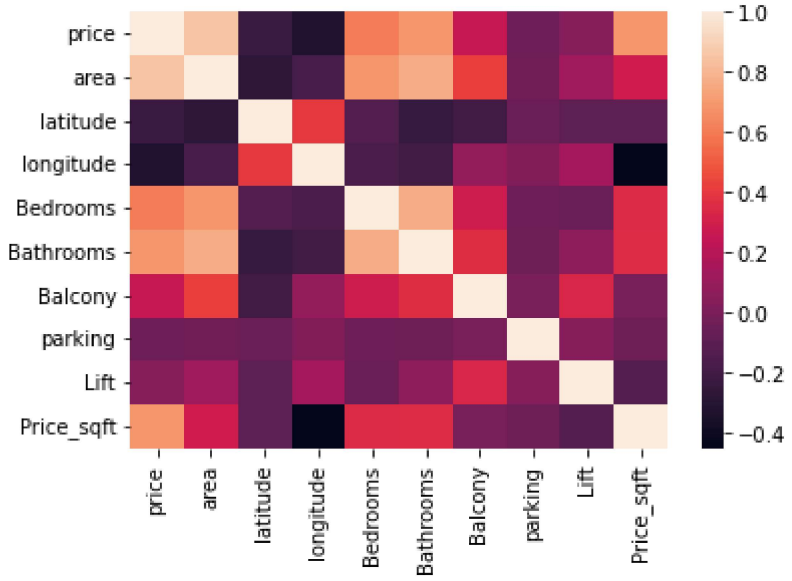


In [12]:

```
sns.heatmap(dp1.corr())
```

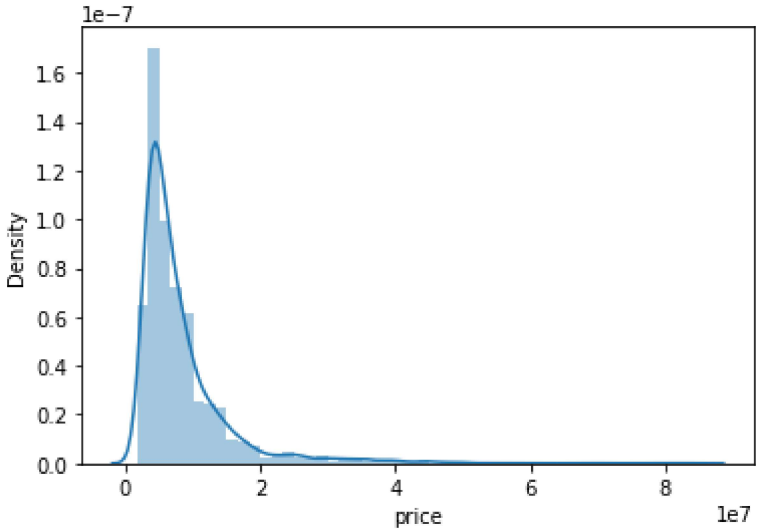
Out[12]:

```
<AxesSubplot:>
```



In [13]:
```
sns.distplot(dp1['price'])
```

Out[13]:
```
<AxesSubplot:xlabel='price', ylabel='Density'>
```



In [14]:
```
dp1.head(1)
```

Out[14]:

	price	Address	area	latitude	longitude	Bedrooms	Bathrooms	Balcony	Status	r
0	5600000.0	Noida Extension, Noida, Delhi NCR	1350.0	28.60885	77.46056	3.0	3.0	NaN	Under Construction	

In [15]:
```
dp1.groupby("neworold")["neworold"].agg("count")
```

Out[15]:
```
neworold
New Property     833
Resale          6905
Name: neworold, dtype: int64
```

In [16]:
```
dp1.groupby("Bedrooms")["Bedrooms"].agg("count")
```

Out[16]:
```
Bedrooms
2.0     3525
3.0     3373
4.0      642
5.0       98
6.0       46
7.0       17
8.0       10
9.0       16
10.0      11
Name: Bedrooms, dtype: int64
```

In [17]:
```
dp1.groupby("type_of_building")["type_of_building"].agg("count")
```

Out[17]:
```
type_of_building
Flat                6226
Individual House    1512
Name: type_of_building, dtype: int64
```

In [18]:
```
dp1.groupby("Furnished_status")["Furnished_status"].agg("count")
```

Out[18]:
```
Furnished_status
Furnished          695
Semi-Furnished    2199
Unfurnished       1230
Name: Furnished_status, dtype: int64
```

In [19]:
```
dp1.groupby("parking")["parking"].agg("count")
```

Out[19]:
```
parking
1.0       1715
2.0        640
3.0        110
4.0         45
5.0         12
          ...
787.0        1
804.0        1
835.0        1
1050.0       1
1406.0       1
Name: parking, Length: 65, dtype: int64
```

In [20]:
```
dp1["Bedrooms"].unique()
```

Out[20]:
```
array([ 3.,  4.,  2.,  5.,  6., 10.,  7.,  8.,  9.])
```

In [21]:
```
dp1["parking"].unique()
```

Out[21]:
```
array([      nan, 1.000e+00, 2.000e+00, 4.000e+00, 3.000e+00, 1.230e+02,
       1.000e+01, 7.000e+00, 7.020e+02, 6.000e+00, 1.100e+01, 5.000e+00,
       2.030e+02, 1.240e+02, 7.870e+02, 9.200e+01, 5.020e+02, 8.040e+02,
       8.000e+00, 3.200e+01, 3.300e+01, 5.500e+01, 1.610e+02, 5.120e+02,
       4.500e+02, 9.000e+00, 1.000e+02, 8.350e+02, 4.300e+01, 5.320e+02,
       1.940e+02, 2.580e+02, 1.800e+01, 2.500e+01, 6.020e+02, 1.600e+01,
       6.240e+02, 8.700e+01, 9.100e+01, 1.200e+01, 1.010e+02, 2.650e+02,
       1.630e+02, 4.450e+02, 1.406e+03, 4.020e+02, 7.500e+02, 1.050e+03,
       5.400e+01, 1.300e+02, 7.030e+02, 3.740e+02, 6.600e+01, 2.300e+01,
       2.200e+02, 1.600e+02, 2.300e+02, 3.400e+01, 1.190e+02, 3.380e+02,
       1.030e+02, 2.000e+01, 1.450e+02, 9.900e+01, 3.030e+02, 1.790e+02])
```

In [22]:
```
dp1["area"].nunique()
```

Out[22]:
```
1102
```

In [23]:
```
dp1.isnull().sum()
```

Out[23]:
```
price                 0
Address               0
area                  0
latitude              0
longitude             0
Bedrooms              0
Bathrooms             0
Balcony            2572
Status              574
neworold              0
parking            5126
Furnished_status   3614
Lift               6005
Landmarks          4979
type_of_building      0
desc                  0
Price_sqft            0
dtype: int64
```

Droping some Columns that are not essential in Price Prediction

In [24]:
```
dp2=dp1.drop(["Address","Status","Balcony","Landmarks","desc"],axis="columns")
```

In [25]:
```
dp2.head()
```

Out[25]:

	price	area	latitude	longitude	Bedrooms	Bathrooms	neworold	parking	Furnished_st
0	5600000.0	1350.0	28.608850	77.460560	3.0	3.0	New Property	NaN	
1	8800000.0	1490.0	28.374236	76.952416	3.0	3.0	New Property	NaN	Semi-Furnis
2	16500000.0	2385.0	28.645769	77.385110	4.0	5.0	New Property	1.0	Unfurnis
3	3810000.0	1050.0	28.566914	77.436434	2.0	2.0	New Property	1.0	Unfurnis
4	6200000.0	1350.0	28.520732	77.356491	2.0	2.0	Resale	1.0	

In [26]:
```
print(dp2["price"].min())
print(dp2["price"].max())
```

```
1700000.0
85000000.0
```

In [27]:
```
dp2.isnull().sum()
```

Out[27]:
```
price                 0
area                  0
latitude              0
longitude             0
Bedrooms              0
Bathrooms             0
neworold              0
parking            5126
Furnished_status   3614
Lift               6005
type_of_building      0
Price_sqft            0
dtype: int64
```

In [28]:
```
dp2.head(1)
```

Out[28]:

	price	area	latitude	longitude	Bedrooms	Bathrooms	neworold	parking	Furnished_statu
0	5600000.0	1350.0	28.60885	77.46056	3.0	3.0	New Property	NaN	Na

In [29]:
```
dp2c=dp2.copy()
```

Cleaning Data

In [30]:
```
def cleaning(info):
    info.neworold.replace(["Resale","New Property"],[0,1],inplace=True)
    info.type_of_building.replace(["Flat","Individual House"],[0,1],inplace=True)
    x=[]
    for i in info.parking:
        if i>0:
            x.append(1)
        else:
            x.append(0)
    info.parking=x
    x=[]
    for i in info.Furnished_status:
        if(i=="Semi-Furnished"):
            x.append(1)
        elif(i=="Furnished"):
            x.append(2)
        else:
            x.append(0)
    info.Furnished_status=x
    x=[]
    for i in info.Lift:
```

```
        if i>0:
            x.append(1)
        else:
            x.append(0)
    info.Lift=x
    info.price=info.price
    return info
dp3=cleaning(dp2c)
```

In [31]:
```
dp3.head()
```

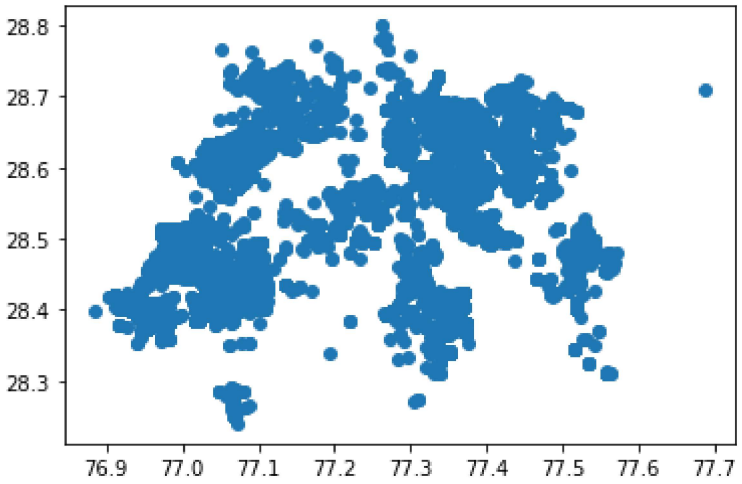
Out[31]:

	price	area	latitude	longitude	Bedrooms	Bathrooms	neworold	parking	Furnished_st
0	5600000.0	1350.0	28.608850	77.460560	3.0	3.0	1	0	
1	8800000.0	1490.0	28.374236	76.952416	3.0	3.0	1	0	
2	16500000.0	2385.0	28.645769	77.385110	4.0	5.0	1	1	
3	3810000.0	1050.0	28.566914	77.436434	2.0	2.0	1	1	
4	6200000.0	1350.0	28.520732	77.356491	2.0	2.0	0	1	

In [32]:
```
dp3.value_counts("type_of_building")
```

Out[32]:
```
type_of_building
0    6226
1    1512
dtype: int64
```

In [33]:
```
plt.scatter(x=dp3['longitude'], y=dp3['latitude'])
plt.show()
```

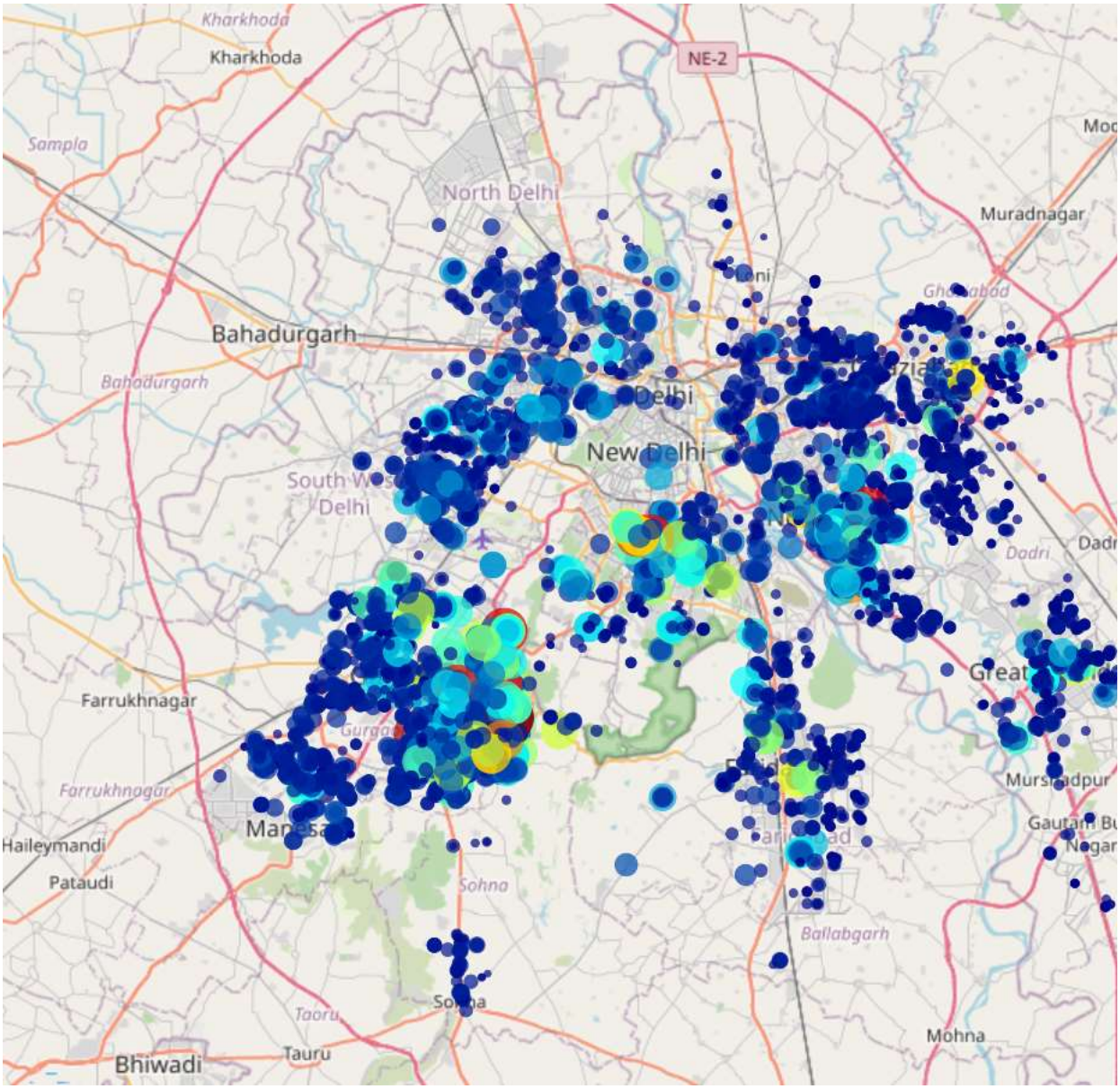


Data and information visualization

In [34]:
```
fig = px.scatter_mapbox(dp3,
                        lat="latitude",
                        lon="longitude",
                        hover_name="price",
                        hover_data=["price","area"],
```

```
                        color="price",
                        color_continuous_scale="jet",
                        size="price",
                        zoom=9,
                        height=600,
                        width=900)

fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



In [35]:
```
dp3.shape
```

Out[35]:
```
(7738, 12)
```

Preparing Data for Machine Learning

```
```

In [36]:
```
Xdp=dp3.drop(["price","Price_sqft"],axis="columns")
Xdp.head()
```

Out[36]:

	area	latitude	longitude	Bedrooms	Bathrooms	neworold	parking	Furnished_status	Lift	ty
0	1350.0	28.608850	77.460560	3.0	3.0	1	0	0	1	
1	1490.0	28.374236	76.952416	3.0	3.0	1	0	1	1	
2	2385.0	28.645769	77.385110	4.0	5.0	1	1	0	0	
3	1050.0	28.566914	77.436434	2.0	2.0	1	1	0	1	
4	1350.0	28.520732	77.356491	2.0	2.0	0	1	0	1	

In [37]:
```
Ydp=dp3.price
Ydp.head()
```

Out[37]:
```
0     5600000.0
1     8800000.0
2    16500000.0
3     3810000.0
4     6200000.0
Name: price, dtype: float64
```

Splitting Data

In [38]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(Xdp, Ydp, test_size = 0.2, random_
```

Linear Regression Model

In [39]:
```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(X_train, Y_train)
```

Out[39]:
```
LinearRegression()
```

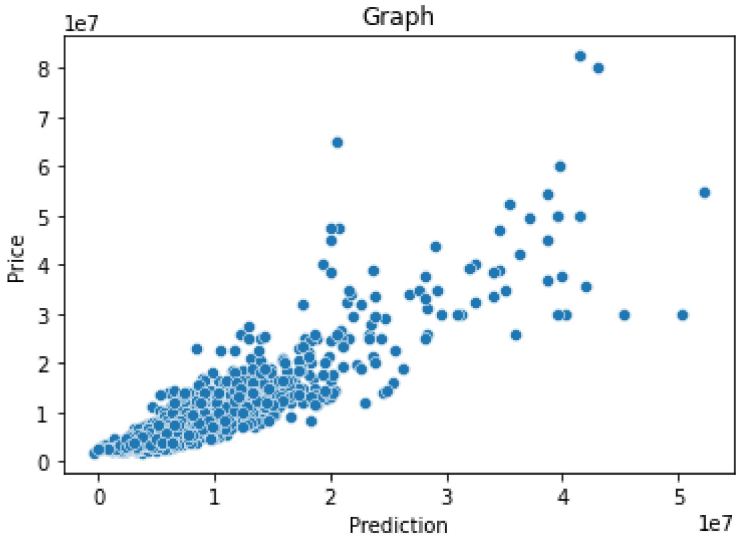
In [40]:
```
lm.score(X_test, Y_test)
```

Out[40]:
```
0.7658990836964648
```

Prediction Visulization

In [41]:
```
graph=sns.scatterplot(x=lm.predict(X_test),y=Y_test)
graph.set(xlabel ="Prediction", ylabel = "Price", title ="Graph")
```

Out[41]:
```
[Text(0.5, 0, 'Prediction'), Text(0, 0.5, 'Price'), Text(0.5, 1.0, 'Graph')]
```



Predicting Price

In [42]:
```
def prediction_price(area,latitude,longitude,Bedrooms,Bathrooms,new_or_old,parking,fur
    pp=np.zeros(len(Xdp.columns))
    pp[0]=area
    pp[1]=latitude
    pp[2]=longitude
    pp[3]=Bedrooms
    pp[4]=Bathrooms
    pp[5]=new_or_old
    pp[6]=parking
    pp[7]=furnished_status
    pp[8]=lift
    pp[9]=t_o_b

    return lm.predict([pp])[0]
```

In [43]:
```
prediction_price(1350,28.608850,77.460560,2,1,1,0,0,1,1)
```

Out[43]: 5835739.17798382

In [44]:
```
prediction_price(1500,28.608850,77.460560,4,5,1,0,0,1,1)
```

Out[44]: 9650015.19281292

In [45]:
```
prediction_price(1800,28.520733,77.356491,3,2,1,1,0,0,1)
```

Out[45]: 10389729.143565059

In [46]:
```
prediction_price(1300,28.520733,77.356491,3,2,1,0,1,0,0)
```

Out[46]: 6515597.931874871

Thank you