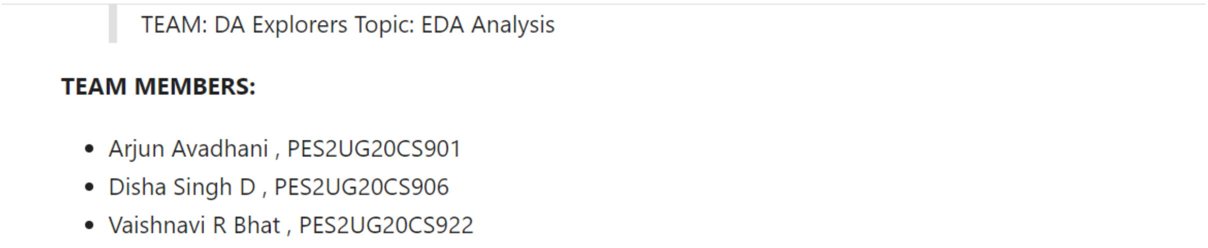
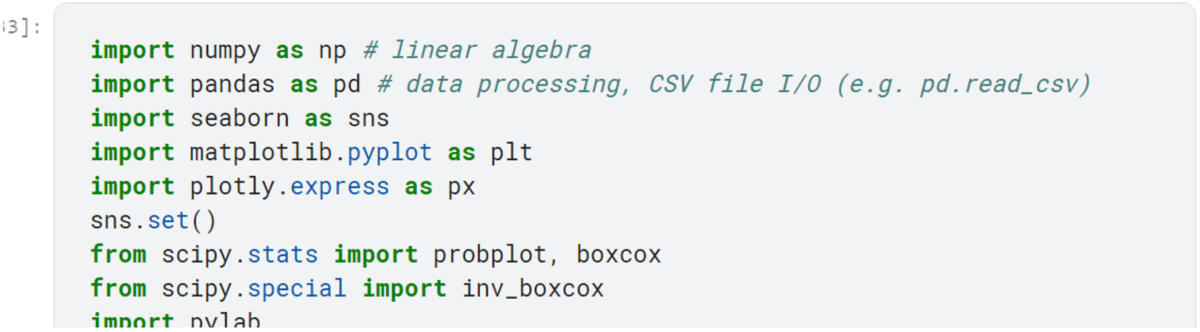
TOPIC: EDA ANALYSIS FOR HOUSE PREDICTION DATASET



> TEAM: DA Explorers Topic: EDA Analysis

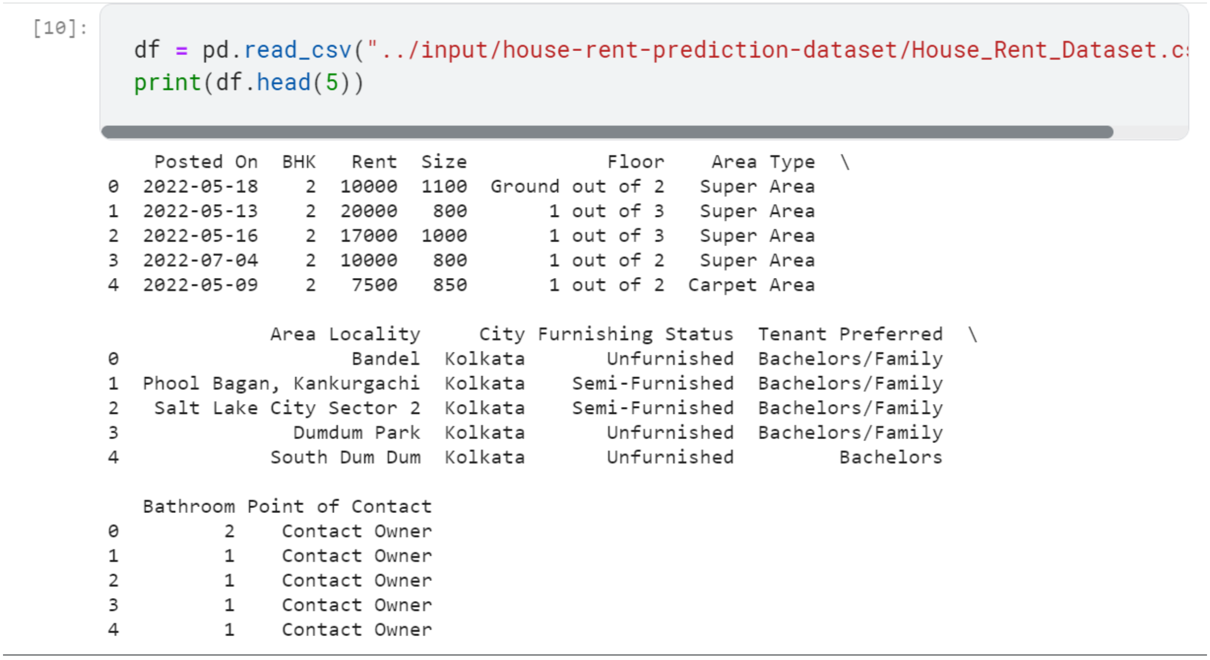
TEAM MEMBERS:

- Arjun Avadhani , PES2UG20CS901
- Disha Singh D , PES2UG20CS906
- Vaishnavi R Bhat , PES2UG20CS922



3]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
sns.set()
from scipy.stats import probplot, boxcox
from scipy.special import inv_boxcox
import pylab
```



[10]:

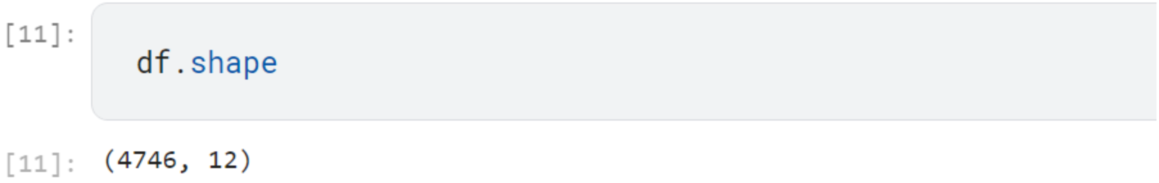
```
df = pd.read_csv("../input/house-rent-prediction-dataset/House_Rent_Dataset.c
print(df.head(5))
```

```
   Posted On  BHK   Rent  Size            Floor    Area Type  \
0  2022-05-18    2  10000  1100  Ground out of 2   Super Area   
1  2022-05-13    2  20000   800       1 out of 3   Super Area   
2  2022-05-16    2  17000  1000       1 out of 3   Super Area   
3  2022-07-04    2  10000   800       1 out of 2   Super Area   
4  2022-05-09    2   7500   850       1 out of 2  Carpet Area   

              Area Locality     City Furnishing Status  Tenant Preferred  \
0                    Bandel  Kolkata       Unfurnished  Bachelors/Family   
1  Phool Bagan, Kankurgachi  Kolkata    Semi-Furnished  Bachelors/Family   
2   Salt Lake City Sector 2  Kolkata    Semi-Furnished  Bachelors/Family   
3               Dumdum Park  Kolkata       Unfurnished  Bachelors/Family   
4             South Dum Dum  Kolkata       Unfurnished         Bachelors   

   Bathroom Point of Contact  
0         2    Contact Owner  
1         1    Contact Owner  
2         1    Contact Owner  
3         1    Contact Owner  
4         1    Contact Owner  
```

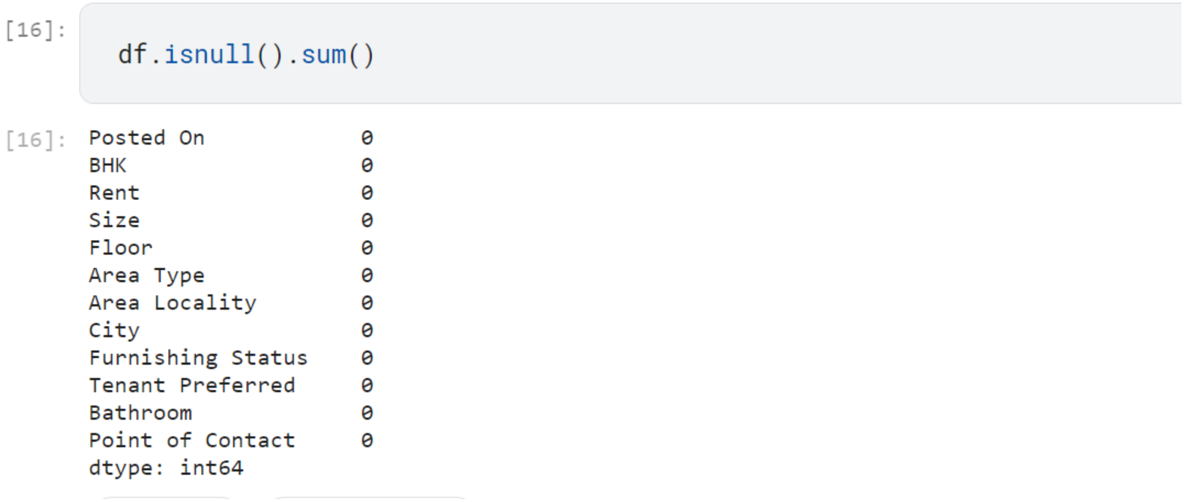
Gives the number of rows and attributes



```
[11]:  df.shape

[11]:  (4746, 12)
```

To find out if there is any null values, use isnull(). There is no null values in the dataset.



```
[16]:  df.isnull().sum()

[16]:  Posted On            0
       BHK                  0
       Rent                 0
       Size                 0
       Floor                0
       Area Type            0
       Area Locality        0
       City                 0
       Furnishing Status    0
       Tenant Preferred     0
       Bathroom             0
       Point of Contact     0
       dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4746 entries, 0 to 4745
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Posted On          4746 non-null   object
 1   BHK                4746 non-null   int64 
 2   Rent               4746 non-null   int64 
 3   Size               4746 non-null   int64 
 4   Floor              4746 non-null   object
 5   Area Type          4746 non-null   object
 6   Area Locality      4746 non-null   object
 7   City               4746 non-null   object
 8   Furnishing Status  4746 non-null   object
 9   Tenant Preferred   4746 non-null   object
 10  Bathroom           4746 non-null   int64 
 11  Point of Contact   4746 non-null   object
dtypes: int64(4), object(8)
memory usage: 445.1+ KB
```

Posted On column has an object datatype while its a date, so lets convert it into right datatype

[27]:

```
df['Posted On'] = pd.to_datetime(df['Posted On'])
```

[29]:

```
df.describe()
```

[29]:

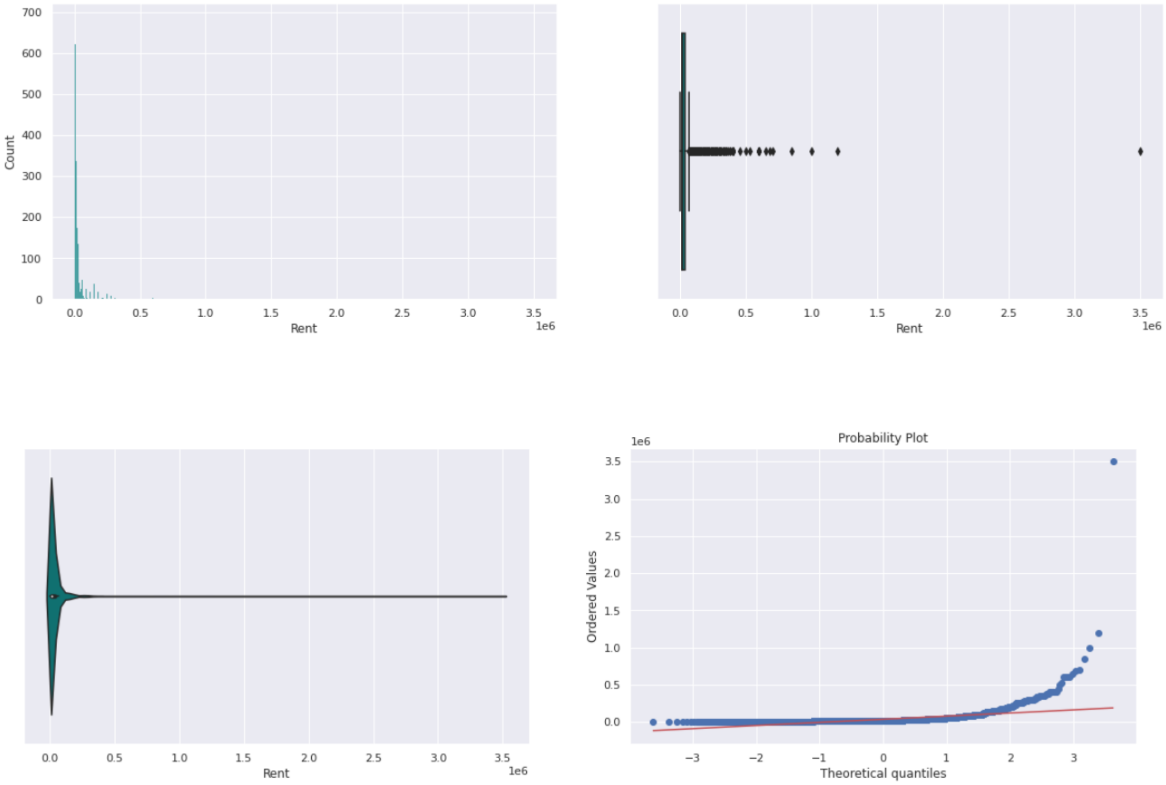
	BHK	Rent	Size	Bathroom
count	4746.000000	4.746000e+03	4746.000000	4746.000000
mean	2.083860	3.499345e+04	967.490729	1.965866
std	0.832256	7.810641e+04	634.202328	0.884532
min	1.000000	1.200000e+03	10.000000	1.000000
25%	2.000000	1.000000e+04	550.000000	1.000000
50%	2.000000	1.600000e+04	850.000000	2.000000
75%	3.000000	3.300000e+04	1200.000000	2.000000
max	6.000000	3.500000e+06	8000.000000	10.000000

Mean Rent is greater than twice of Median rent, so there are definitely some outliers in this column

Checking the Distribution of Rent

[35]:

```
fig, ax = plt.subplots(2, 2, figsize=(20, 12))
ax1 = sns.histplot(x = df['Rent'], color='teal', ax= ax[0, 0])
ax2 = sns.boxplot(x = df['Rent'], ax= ax[0, 1], color= 'teal')
ax3 = sns.violinplot(x = df['Rent'], ax= ax[1, 0], color= 'teal')
ax4 = probplot(df['Rent'], plot=pylab)
pylab.show()
```



There are definitely some outliers present here which might cause problems later during modelling, therefore, we will apply boxcox transformation to it and we will also remove some of the extreme outliers

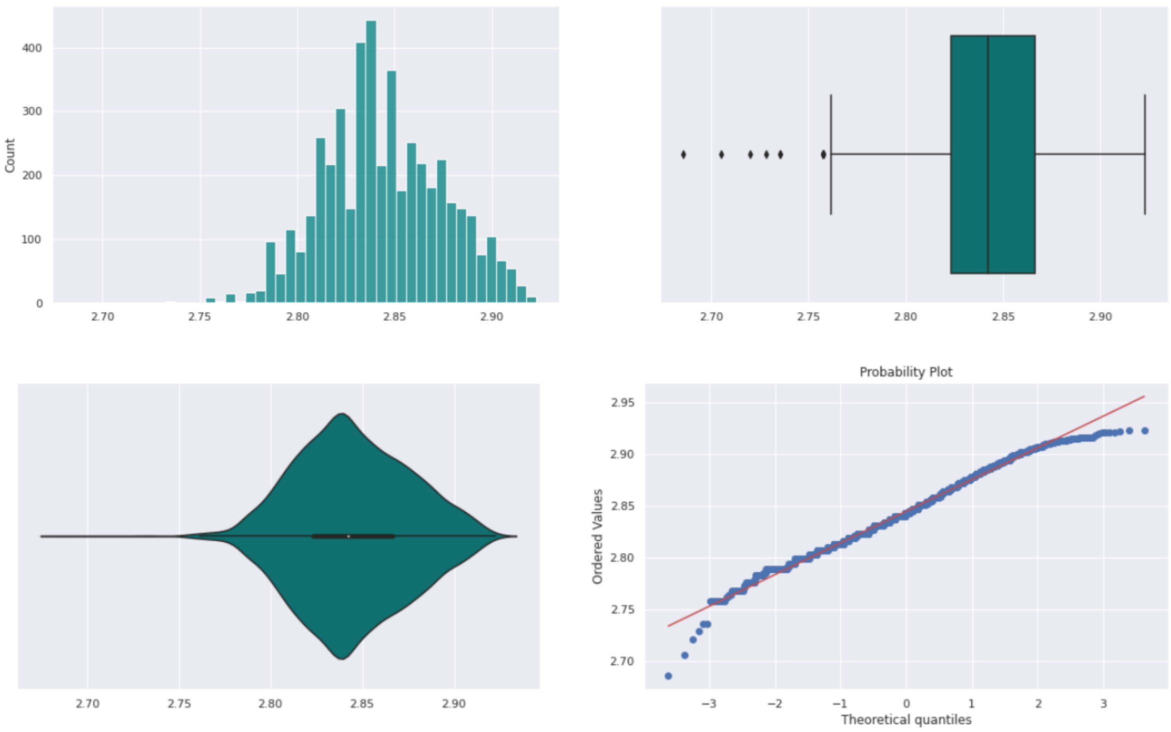
[40]:

```
max_rent = df['Rent'].max()
index_max_rent = df[df['Rent'] == max_rent].index
df = df.drop(index_max_rent)

bc_result = boxcox(df['Rent'])
boxcox_y = bc_result[0]
lam = bc_result[1]
```

[41]:

```
fig, ax = plt.subplots(2, 2, figsize=(20, 12))
ax1 = sns.histplot(x = boxcox_y, color='teal', ax= ax[0, 0])
ax2 = sns.boxplot(x = boxcox_y, ax= ax[0, 1], color= 'teal')
ax3 = sns.violinplot(x = boxcox_y, ax= ax[1, 0], color= 'teal')
ax4 = probplot(boxcox_y, plot=pylab)
pylab.show()
```

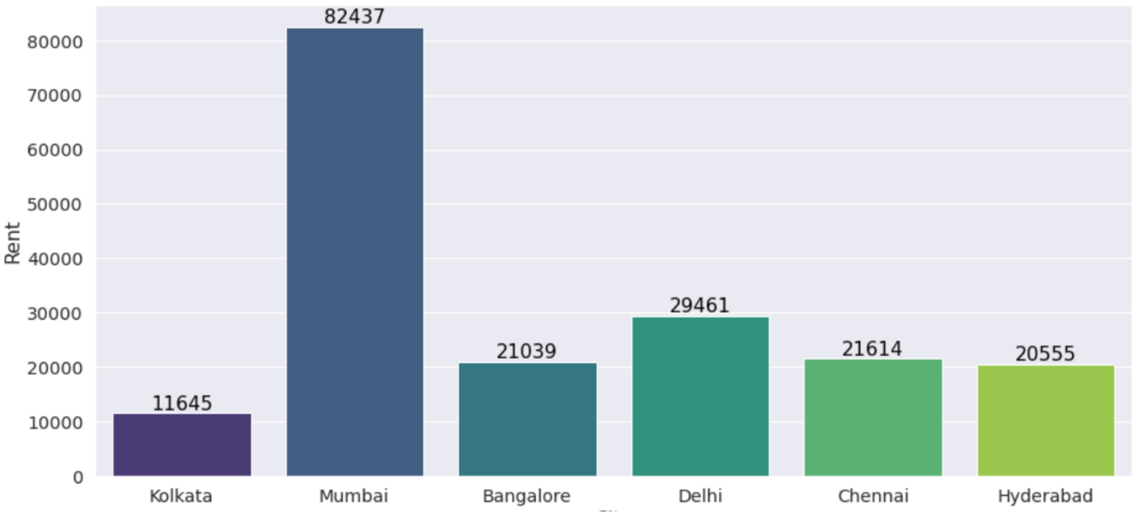


Plotting categorical variables vs Rent

[67]:

```
sns.set_context('notebook', font_scale = 1.3)
plt.figure(figsize=(15, 7))
ax = sns.barplot(x=df['City'],
                 y=df['Rent'],
                 palette='viridis',
                 ci = None)
plt.ylabel('Rent');

for p in ax.patches:
    ax.annotate(int(p.get_height()), (p.get_x() + 0.4, p.get_height() + 1), ha = 'center',
```



Mumbai has the highest Rent followed by Delhi