Rent_Prediction

June 22, 2025

[1]:
```
import os
import math
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
import statistics

import matplotlib.pyplot as plt
import seaborn as sns

import keras
import tensorflow as tf
from sklearn import metrics
from scipy.stats import stats
from sklearn.model_selection import train_test_split, cross_val_score,␣
  ↪GridSearchCV, RandomizedSearchCV
from sklearn.svm import SVR
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import LabelEncoder, MinMaxScaler, StandardScaler,␣
  ↪RobustScaler
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

[2]:
```
!pip install keras
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: keras in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (3.9.2)
Requirement already satisfied: absl-py in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from keras)
(2.2.2)
Requirement already satisfied: numpy in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from keras)
(1.26.4)
Requirement already satisfied: rich in c:\programdata\anaconda3\lib\site-
packages (from keras) (13.7.1)
Requirement already satisfied: namex in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from keras)
```

```
(0.0.8)
Requirement already satisfied: h5py in c:\programdata\anaconda3\lib\site-
packages (from keras) (3.11.0)
Requirement already satisfied: optree in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from keras)
(0.15.0)
Requirement already satisfied: ml-dtypes in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from keras)
(0.5.1)
Requirement already satisfied: packaging in c:\programdata\anaconda3\lib\site-
packages (from keras) (24.1)
Requirement already satisfied: typing-extensions>=4.5.0 in
c:\programdata\anaconda3\lib\site-packages (from optree->keras) (4.11.0)
Requirement already satisfied: markdown-it-py>=2.2.0 in
c:\programdata\anaconda3\lib\site-packages (from rich->keras) (2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in
c:\programdata\anaconda3\lib\site-packages (from rich->keras) (2.15.1)
Requirement already satisfied: mdurl~=0.1 in c:\programdata\anaconda3\lib\site-
packages (from markdown-it-py>=2.2.0->rich->keras) (0.1.0)
```

[3]:

```
pip install tensorflow
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: tensorflow in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (2.19.0)
Requirement already satisfied: absl-py>=1.0.0 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(2.2.2)
Requirement already satisfied: astunparse>=1.6.0 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(25.2.10)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(0.2.0)
Requirement already satisfied: libclang>=13.0.0 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(18.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(3.4.0)
Requirement already satisfied: packaging in c:\programdata\anaconda3\lib\site-
packages (from tensorflow) (24.1)
```

```
Requirement already satisfied:
protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3
in c:\programdata\anaconda3\lib\site-packages (from tensorflow) (4.25.3)
Requirement already satisfied: requests<3,>=2.21.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (2.32.3)
Requirement already satisfied: setuptools in c:\programdata\anaconda3\lib\site-
packages (from tensorflow) (75.1.0)
Requirement already satisfied: six>=1.12.0 in c:\programdata\anaconda3\lib\site-
packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(3.0.1)
Requirement already satisfied: typing-extensions>=3.6.6 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (4.11.0)
Requirement already satisfied: wrapt>=1.11.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (1.14.1)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(1.71.0)
Requirement already satisfied: tensorboard~=2.19.0 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(2.19.0)
Requirement already satisfied: keras>=3.5.0 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(3.9.2)
Requirement already satisfied: numpy<2.2.0,>=1.26.0 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(1.26.4)
Requirement already satisfied: h5py>=3.11.0 in
c:\programdata\anaconda3\lib\site-packages (from tensorflow) (3.11.0)
Requirement already satisfied: ml-dtypes<1.0.0,>=0.5.1 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from tensorflow)
(0.5.1)
Requirement already satisfied: wheel<1.0,>=0.23.0 in
c:\programdata\anaconda3\lib\site-packages (from astunparse>=1.6.0->tensorflow)
(0.44.0)
Requirement already satisfied: rich in c:\programdata\anaconda3\lib\site-
packages (from keras>=3.5.0->tensorflow) (13.7.1)
Requirement already satisfied: namex in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from
keras>=3.5.0->tensorflow) (0.0.8)
Requirement already satisfied: optree in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from
keras>=3.5.0->tensorflow) (0.15.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\programdata\anaconda3\lib\site-packages (from
requests<3,>=2.21.0->tensorflow) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in
```

```
c:\programdata\anaconda3\lib\site-packages (from
requests<3,>=2.21.0->tensorflow) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\programdata\anaconda3\lib\site-packages (from
requests<3,>=2.21.0->tensorflow) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in
c:\programdata\anaconda3\lib\site-packages (from
requests<3,>=2.21.0->tensorflow) (2024.8.30)
Requirement already satisfied: markdown>=2.6.8 in
c:\programdata\anaconda3\lib\site-packages (from
tensorboard~=2.19.0->tensorflow) (3.4.1)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in
c:\users\mrunal\appdata\roaming\python\python312\site-packages (from
tensorboard~=2.19.0->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in
c:\programdata\anaconda3\lib\site-packages (from
tensorboard~=2.19.0->tensorflow) (3.0.3)
Requirement already satisfied: MarkupSafe>=2.1.1 in
c:\programdata\anaconda3\lib\site-packages (from
werkzeug>=1.0.1->tensorboard~=2.19.0->tensorflow) (2.1.3)
Requirement already satisfied: markdown-it-py>=2.2.0 in
c:\programdata\anaconda3\lib\site-packages (from rich->keras>=3.5.0->tensorflow)
(2.2.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in
c:\programdata\anaconda3\lib\site-packages (from rich->keras>=3.5.0->tensorflow)
(2.15.1)
Requirement already satisfied: mdurl~=0.1 in c:\programdata\anaconda3\lib\site-
packages (from markdown-it-py>=2.2.0->rich->keras>=3.5.0->tensorflow) (0.1.0)
Note: you may need to restart the kernel to use updated packages.
```

[4]:
```
rent_df = pd.read_csv('House_Rent_Dataset.csv')
rent_df
```

[4]:
```
       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
…            …   …     …    …               …           …
4741  2022-05-18    2  15000  1000       3 out of 5  Carpet Area
4742  2022-05-15    3  29000  2000       1 out of 4   Super Area
4743  2022-07-10    3  35000  1750       3 out of 5  Carpet Area
4744  2022-07-06    3  45000  1500     23 out of 34  Carpet Area
4745  2022-05-04    2  15000  1000       4 out of 5  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
…                        …        …               …               …
4741              Bandam Kommu  Hyderabad   Semi-Furnished  Bachelors/Family
4742      Manikonda, Hyderabad  Hyderabad   Semi-Furnished  Bachelors/Family
4743      Himayath Nagar, NH 7  Hyderabad   Semi-Furnished  Bachelors/Family
4744                Gachibowli  Hyderabad   Semi-Furnished            Family
4745           Suchitra Circle  Hyderabad      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
…        …              …
4741         2    Contact Owner
4742         3    Contact Owner
4743         3    Contact Agent
4744         2    Contact Agent
4745         2    Contact Owner

[4746 rows x 12 columns]
```

```
[5]: rent_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
```

[6]:
```
rent_df.tail()
```

[6]:
```
      Posted On  BHK   Rent  Size         Floor    Area Type  \
4741  2022-05-18    2  15000  1000    3 out of 5  Carpet Area
4742  2022-05-15    3  29000  2000    1 out of 4   Super Area
4743  2022-07-10    3  35000  1750    3 out of 5  Carpet Area
4744  2022-07-06    3  45000  1500  23 out of 34  Carpet Area
4745  2022-05-04    2  15000  1000    4 out of 5  Carpet Area

             Area Locality       City Furnishing Status  Tenant Preferred  \
4741           Bandam Kommu  Hyderabad    Semi-Furnished  Bachelors/Family
4742  Manikonda, Hyderabad  Hyderabad    Semi-Furnished  Bachelors/Family
4743  Himayath Nagar, NH 7  Hyderabad    Semi-Furnished  Bachelors/Family
4744            Gachibowli  Hyderabad    Semi-Furnished            Family
4745        Suchitra Circle  Hyderabad       Unfurnished         Bachelors

      Bathroom Point of Contact
4741         2    Contact Owner
4742         3    Contact Owner
4743         3    Contact Agent
4744         2    Contact Agent
4745         2    Contact Owner
```

[7]:
```
rent_df.describe()
```

[7]:
```
               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
```

[8]:
```
rent_df.shape
```

[8]:
```
(4746, 12)
```

[9]:
```
rent_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
```

[10]:

```
# Outliers
```

[11]:

```
def detect_outliers(data_series):
    # Using the  Mean ± 2 * Standard Deviation as threshold
    mean_val = data_series.mean()
    std_dev = data_series.std()

    upper_limit = mean_val + 2 * std_dev
    lower_limit = mean_val - 2 * std_dev

    #  outlier
    has_outlier = any((data_series < lower_limit) | (data_series > upper_limit))

    outlier_indicator = 1 if has_outlier else 0


    summary = pd.Series([
        data_series.count(),                # Non-null count
        data_series.isnull().sum(),        # Null values
        data_series.sum(),                 # Sum of values
        data_series.mean(),                # Mean
        data_series.median(),              # Median
        data_series.std(),                 # Standard deviation
        data_series.var(),                 # Variance
        data_series.min(),                 # Minimum
        data_series.quantile(0.01),        # 1st percentile
        data_series.quantile(0.05),        # 5th percentile
        data_series.quantile(0.10),
        data_series.quantile(0.25),        # 25th percentile
        data_series.quantile(0.50),        # 50th percentile
        data_series.quantile(0.75),        # 75th percentile
        data_series.quantile(0.90),
        data_series.quantile(0.95),
        data_series.quantile(0.99),
        data_series.max(),                 # Maximum
        lower_limit,                       # Lower threshold
        upper_limit,                       # Upper threshold
        outlier_indicator                  # Outlier flag
    ], index=[
        'Count', 'Missing', 'Sum', 'Mean', 'Median', 'Std_Dev', 'Variance',␣
↪'Min',
        'Q1', 'Q5', 'Q10', 'Q25', 'Q50', 'Q75', 'Q90', 'Q95', 'Q99', 'Max',
        'Lower_Limit', 'Upper_Limit', 'Outlier_Flag'
    ])

    return summary
```

[12]:

```
numeric_columns= []
for cols in rent_df:
    if rent_df[cols].dtypes == 'int64':
        numeric_columns.append(cols)
print(numeric_columns)
```

```
['BHK', 'Rent', 'Size', 'Bathroom']
```

[13]:

```
summary_stats = rent_df[numeric_columns].apply(detect_outliers)
summary_stats
```

[13]:

```
                      BHK          Rent          Size     Bathroom
Count         4746.000000  4.746000e+03  4.746000e+03  4746.000000
Missing          0.000000  0.000000e+00  0.000000e+00     0.000000
Sum           9890.000000  1.660789e+08  4.591711e+06  9330.000000
Mean             2.083860  3.499345e+04  9.674907e+02     1.965866
Median           2.000000  1.600000e+04  8.500000e+02     2.000000
Std_Dev          0.832256  7.810641e+04  6.342023e+02     0.884532
Variance         0.692650  6.100612e+09  4.022126e+05     0.782396
Min              1.000000  1.200000e+03  1.000000e+01     1.000000
Q1               1.000000  4.000000e+03  7.000000e+01     1.000000
Q5               1.000000  6.000000e+03  2.000000e+02     1.000000
Q10              1.000000  7.000000e+03  4.000000e+02     1.000000
Q25              2.000000  1.000000e+04  5.500000e+02     1.000000
Q50              2.000000  1.600000e+04  8.500000e+02     2.000000
Q75              3.000000  3.300000e+04  1.200000e+03     2.000000
Q90              3.000000  7.200000e+04  1.700000e+03     3.000000
Q95              3.000000  1.300000e+05  2.000000e+03     3.000000
Q99              4.000000  3.000000e+05  3.289200e+03     5.000000
Max              6.000000  3.500000e+06  8.000000e+03    10.000000
Lower_Limit      0.419348 -1.212194e+05 -3.009139e+02     0.196803
Upper_Limit      3.748372  1.912063e+05  2.235895e+03     3.734929
```

```
Outlier_Flag     1.000000  1.000000e+00  1.000000e+00     1.000000
```

[ ]:

```

```

[14]:

```
def detect_outliers(col):
    mean = col.mean()
    std = col.std()
    upper = mean + 2 * std
    lower = mean - 2 * std
    outlier_flag = int(any((col < lower) | (col > upper)))

    return pd.Series([
        col.count(),
        col.isnull().sum(),
        col.sum(),
        mean,
        col.median(),
        std,
        col.var(),
        col.min(),
        col.quantile(0.01),
        col.quantile(0.05),
        col.quantile(0.10),
        col.quantile(0.25),
        col.quantile(0.50),
        col.quantile(0.75),
        col.quantile(0.90),
        col.quantile(0.95),
        col.quantile(0.99),
        col.max(),
        lower,
        upper,
        outlier_flag
    ], index=[
        'N', 'NMISS', 'SUM', 'MEAN', 'MEDIAN', 'STD', 'VAR', 'MIN',
        'P1', 'P5', 'P10', 'P25', 'P50', 'P75', 'P90', 'P95', 'P99', 'MAX',
        'LC', 'UC', 'Outlier_Flag'
    ])
```

[15]:

```
summary_df = rent_df[numeric_columns].apply(lambda x: detect_outliers(x))
summary_df
```

[15]:

```
                      BHK          Rent          Size     Bathroom
N             4746.000000  4.746000e+03  4.746000e+03  4746.000000
NMISS            0.000000  0.000000e+00  0.000000e+00     0.000000
SUM           9890.000000  1.660789e+08  4.591711e+06  9330.000000
MEAN             2.083860  3.499345e+04  9.674907e+02     1.965866
```

```
MEDIAN            2.000000  1.600000e+04  8.500000e+02     2.000000
STD               0.832256  7.810641e+04  6.342023e+02     0.884532
VAR               0.692650  6.100612e+09  4.022126e+05     0.782396
MIN               1.000000  1.200000e+03  1.000000e+01     1.000000
P1                1.000000  4.000000e+03  7.000000e+01     1.000000
P5                1.000000  6.000000e+03  2.000000e+02     1.000000
P10               1.000000  7.000000e+03  4.000000e+02     1.000000
P25               2.000000  1.000000e+04  5.500000e+02     1.000000
P50               2.000000  1.600000e+04  8.500000e+02     2.000000
P75               3.000000  3.300000e+04  1.200000e+03     2.000000
P90               3.000000  7.200000e+04  1.700000e+03     3.000000
P95               3.000000  1.300000e+05  2.000000e+03     3.000000
P99               4.000000  3.000000e+05  3.289200e+03     5.000000
MAX               6.000000  3.500000e+06  8.000000e+03    10.000000
LC                0.419348 -1.212194e+05 -3.009139e+02     0.196803
UC                3.748372  1.912063e+05  2.235895e+03     3.734929
Outlier_Flag      1.000000  1.000000e+00  1.000000e+00     1.000000
```

[16]:

```python
print(rent_df['Rent'].dtype)
```

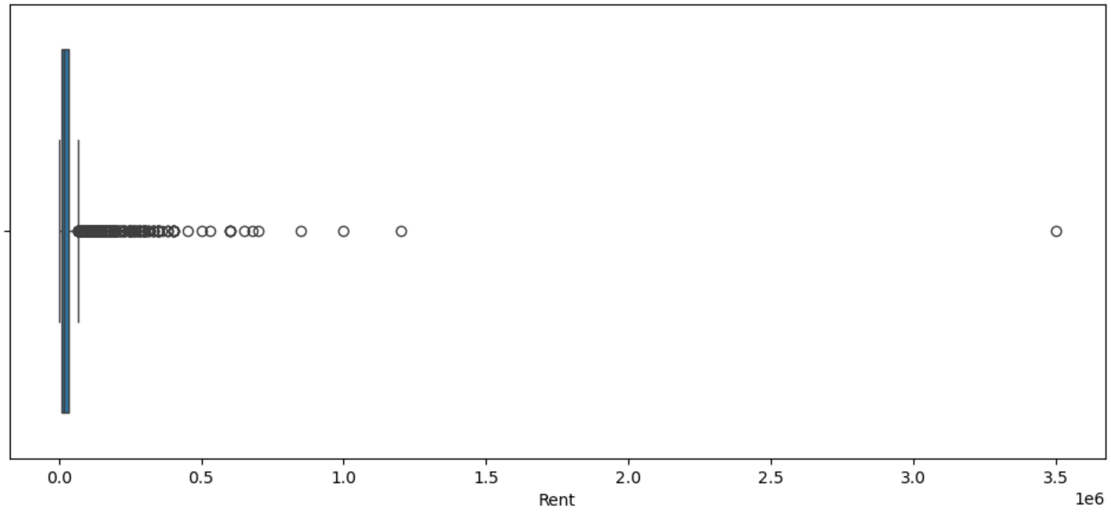
```
int64
```

[17]:

```python
plt.figure(figsize=(12,5))
sns.boxplot(data=rent_df, x='Rent')
```

[17]:

```
<Axes: xlabel='Rent'>
```



[18]:

```python
rent_df = rent_df[rent_df['Rent'] <= 200000]
```

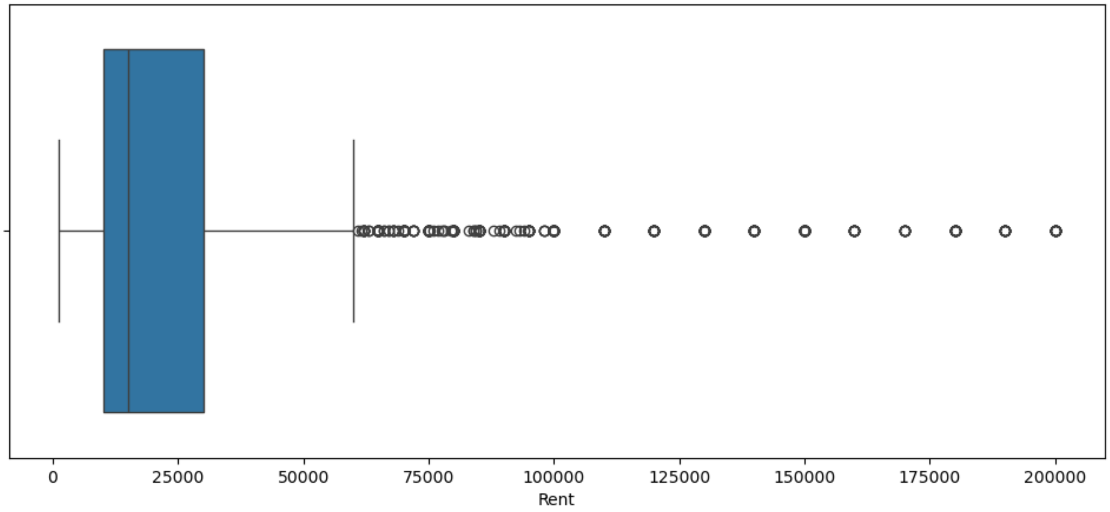
[19]:

```python
rent_df
```

[19]:
```
      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
…          …  …    …   …              …          …
4741  2022-05-18    2  15000  1000       3 out of 5  Carpet Area
4742  2022-05-15    3  29000  2000       1 out of 4   Super Area
4743  2022-07-10    3  35000  1750       3 out of 5  Carpet Area
4744  2022-07-06    3  45000  1500     23 out of 34  Carpet Area
4745  2022-05-04    2  15000  1000       4 out of 5  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
…                         …        …               …                …
4741             Bandam Kommu  Hyderabad    Semi-Furnished  Bachelors/Family
4742      Manikonda, Hyderabad  Hyderabad    Semi-Furnished  Bachelors/Family
4743      Himayath Nagar, NH 7  Hyderabad    Semi-Furnished  Bachelors/Family
4744               Gachibowli  Hyderabad    Semi-Furnished            Family
4745           Suchitra Circle  Hyderabad       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
…        …              …
4741         2    Contact Owner
4742         3    Contact Owner
4743         3    Contact Agent
4744         2    Contact Agent
4745         2    Contact Owner

[4647 rows x 12 columns]
```

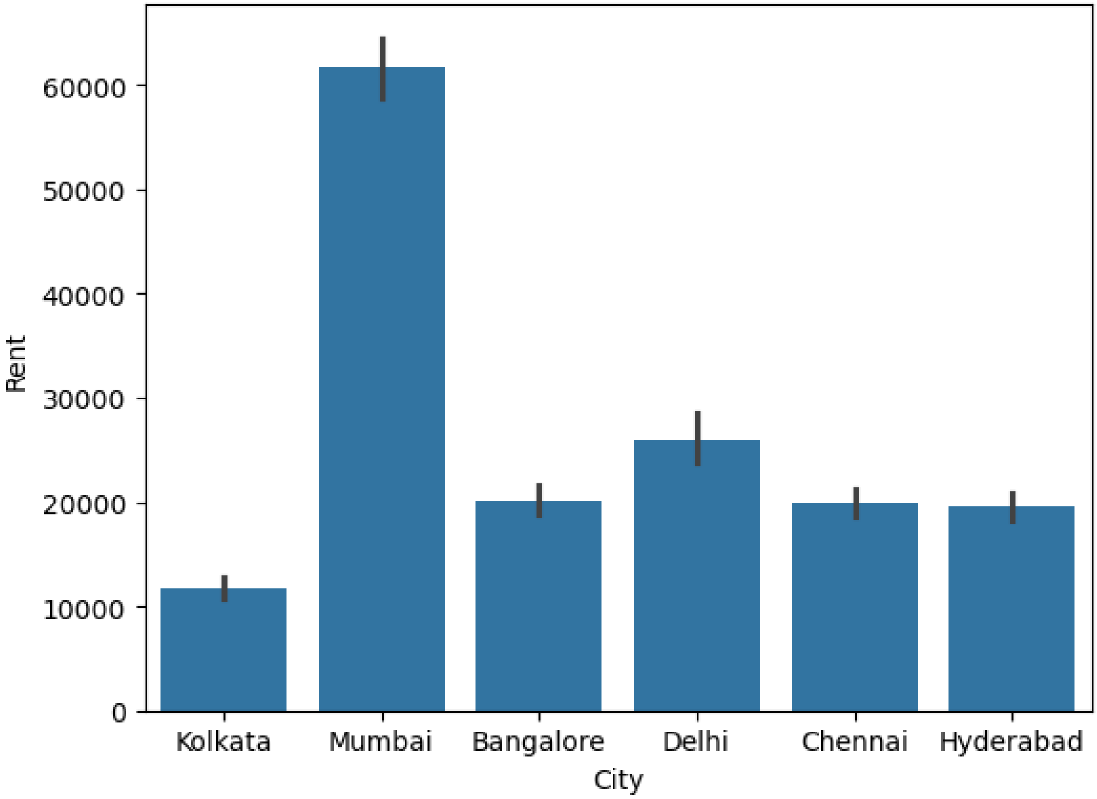
[20]:
```
plt.figure(figsize=(12,5))
sns.boxplot(data=rent_df, x='Rent')
```

[20]:
```
<Axes: xlabel='Rent'>
```



```
[21]: sns.barplot(data = rent_df , x = 'City', y = 'Rent')
```

```
[21]: <Axes: xlabel='City', ylabel='Rent'>
```



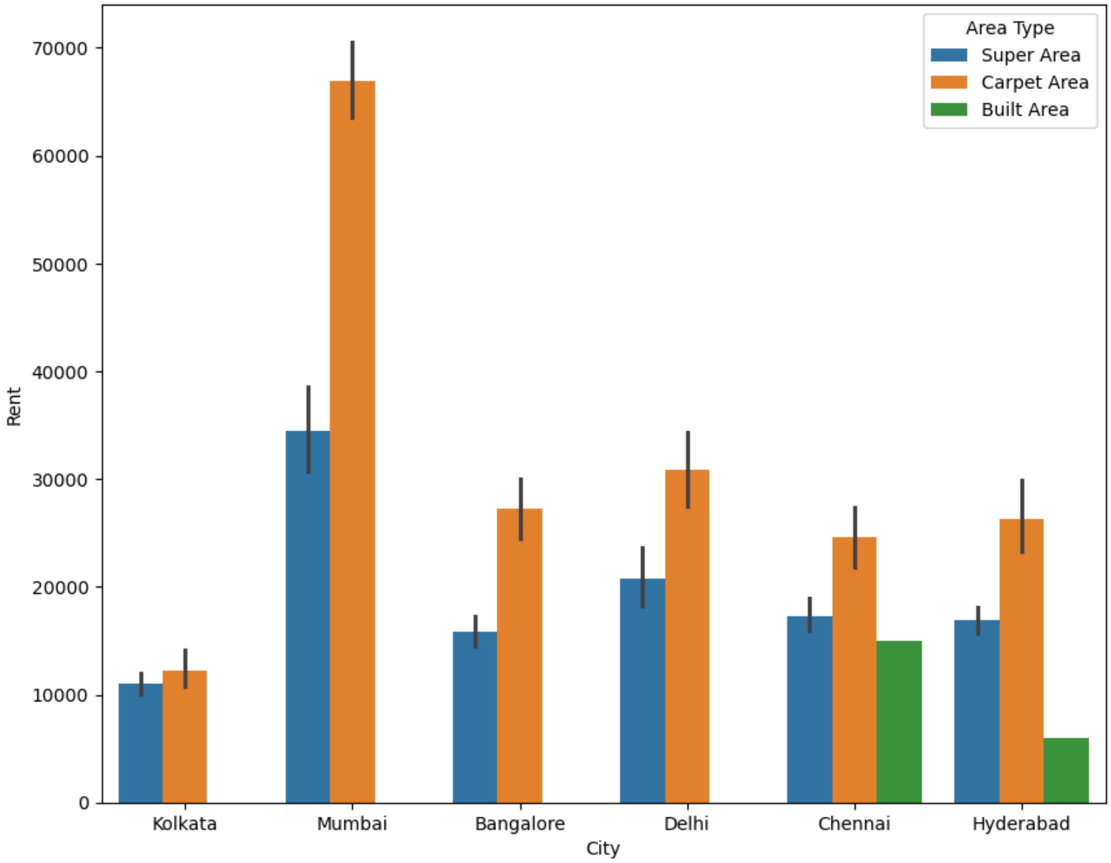
As We can See That Mumbai Has Highest Rent amongest all

[22]:
```
rent_df['Area Type'].value_counts()
```

[22]:
```
Area Type
Super Area     2436
Carpet Area    2209
Built Area        2
Name: count, dtype: int64
```

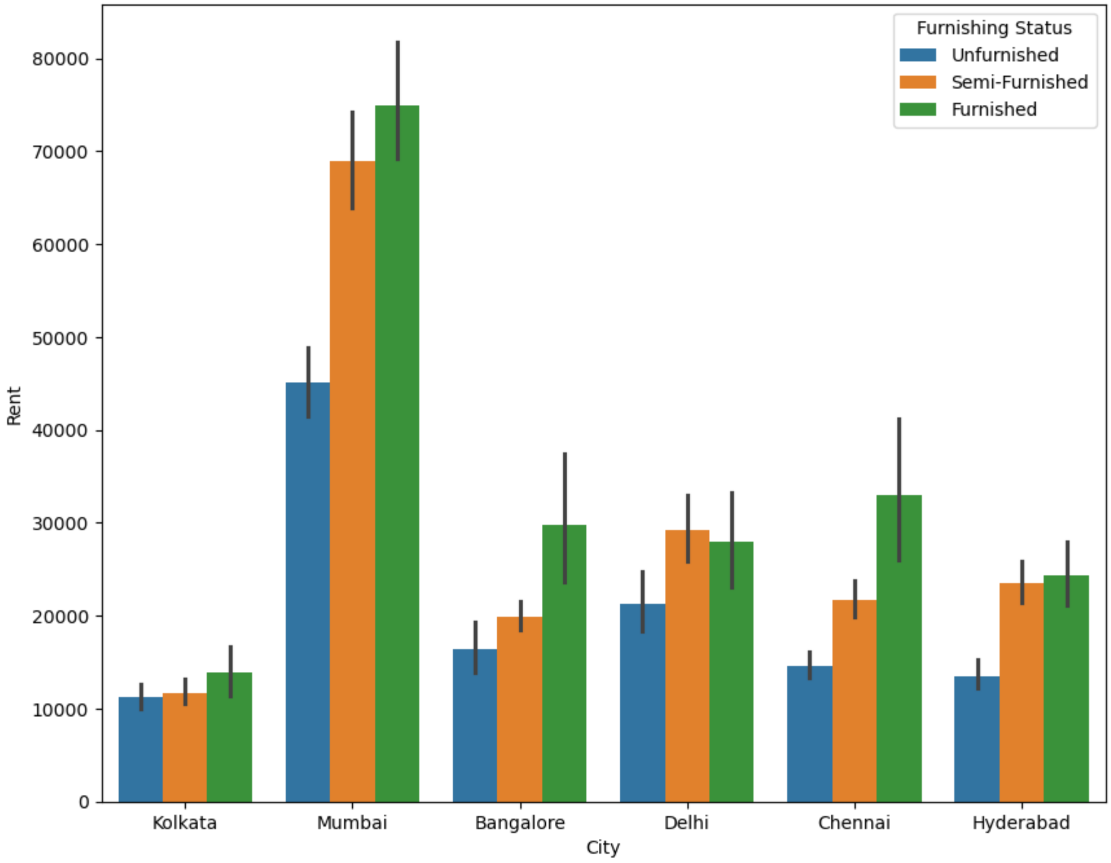
[23]:
```
plt.figure(figsize = (10,8))
sns.barplot(data = rent_df , x = 'City',  y = 'Rent' , hue = 'Area Type')
```

[23]:
```
<Axes: xlabel='City', ylabel='Rent'>
```



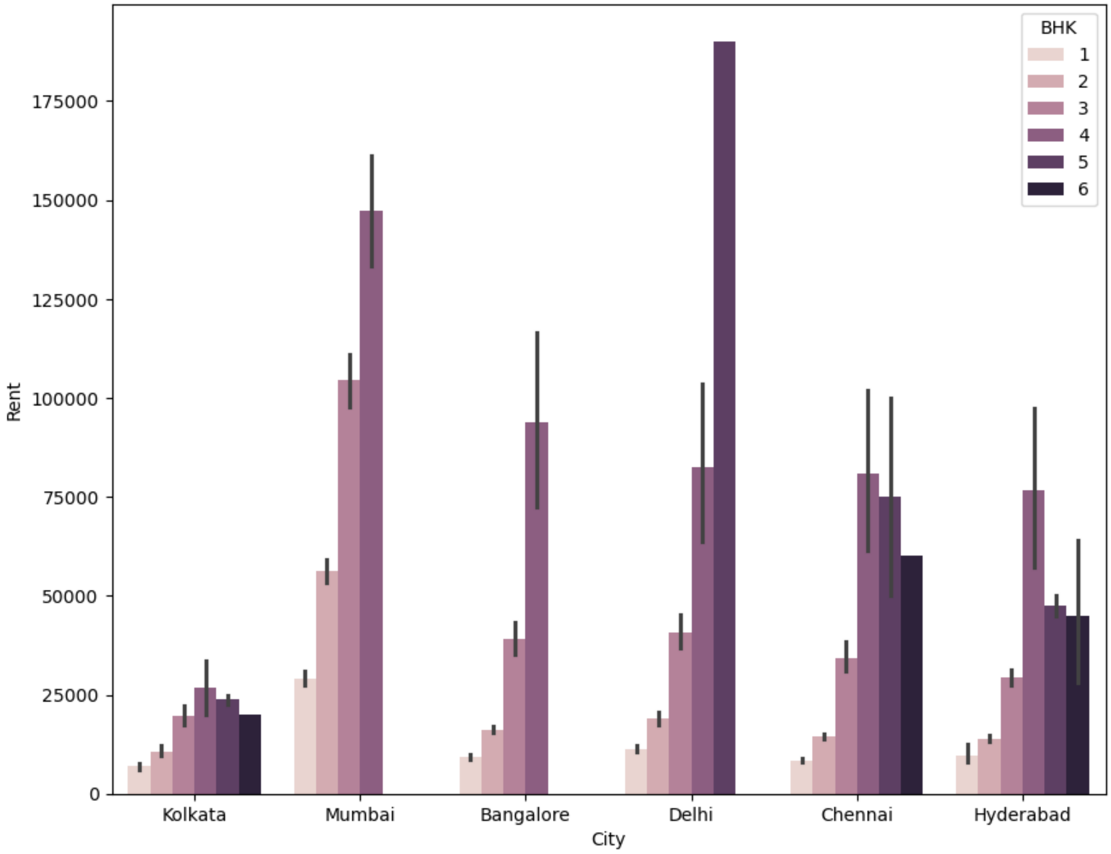
[24]:
```
plt.figure(figsize = (10,8))
sns.barplot(data = rent_df , x = 'City',  y = 'Rent' , hue = 'Furnishing␣
↪Status')
```

[24]:
```
<Axes: xlabel='City', ylabel='Rent'>
```



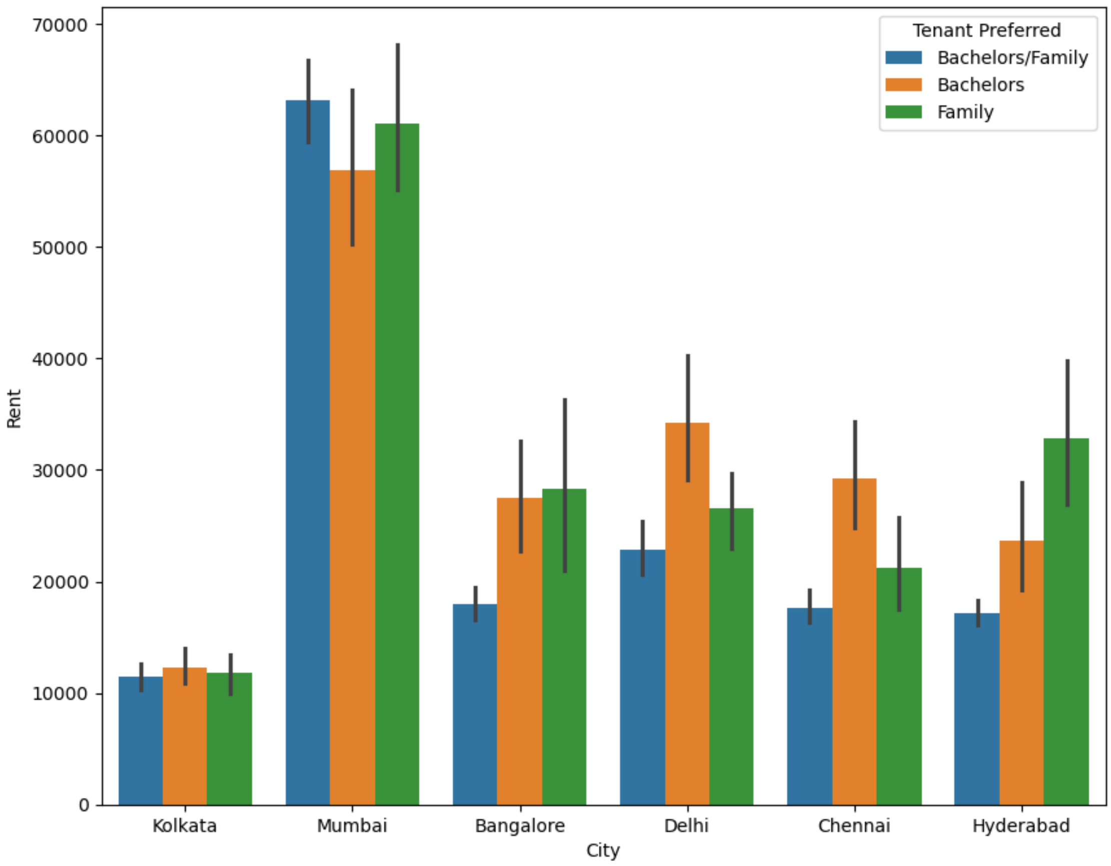
```
[25]: plt.figure(figsize = (10,8))
      sns.barplot(data = rent_df , x = 'City',  y = 'Rent' , hue = 'BHK')
```

```
[25]: <Axes: xlabel='City', ylabel='Rent'>
```



```
[26]: plt.figure(figsize = (10,8))
      sns.barplot(data = rent_df , x = 'City',  y = 'Rent' , hue = 'Tenant Preferred')
```

```
[26]: <Axes: xlabel='City', ylabel='Rent'>
```

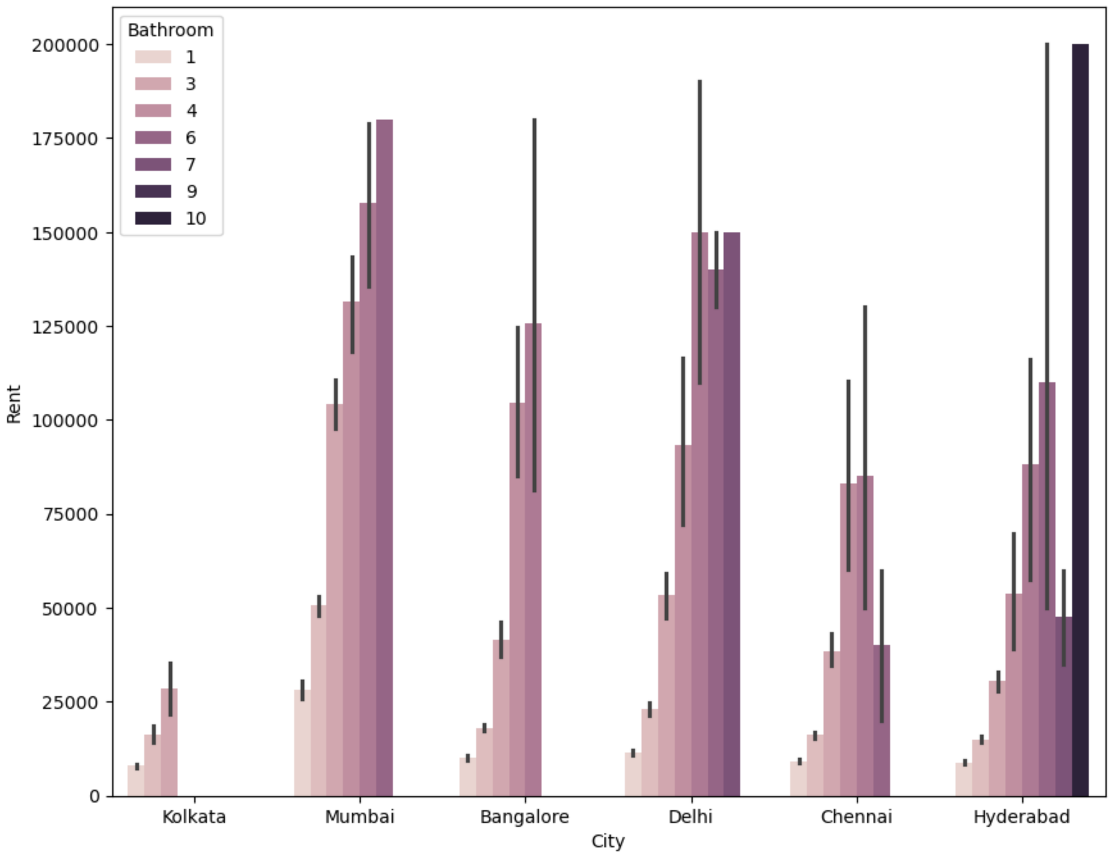


```
[28]: rent_df['Bathroom'].value_counts()
```

```
[28]: Bathroom
      2     2285
      1     1474
      3      732
      4      116
      5       28
      6        8
      7        3
      10       1
      Name: count, dtype: int64
```

```
[29]: plt.figure(figsize = (10,8))
      sns.barplot(data = rent_df , x = 'City',  y = 'Rent' , hue = 'Bathroom')
```

```
[29]: <Axes: xlabel='City', ylabel='Rent'>
```



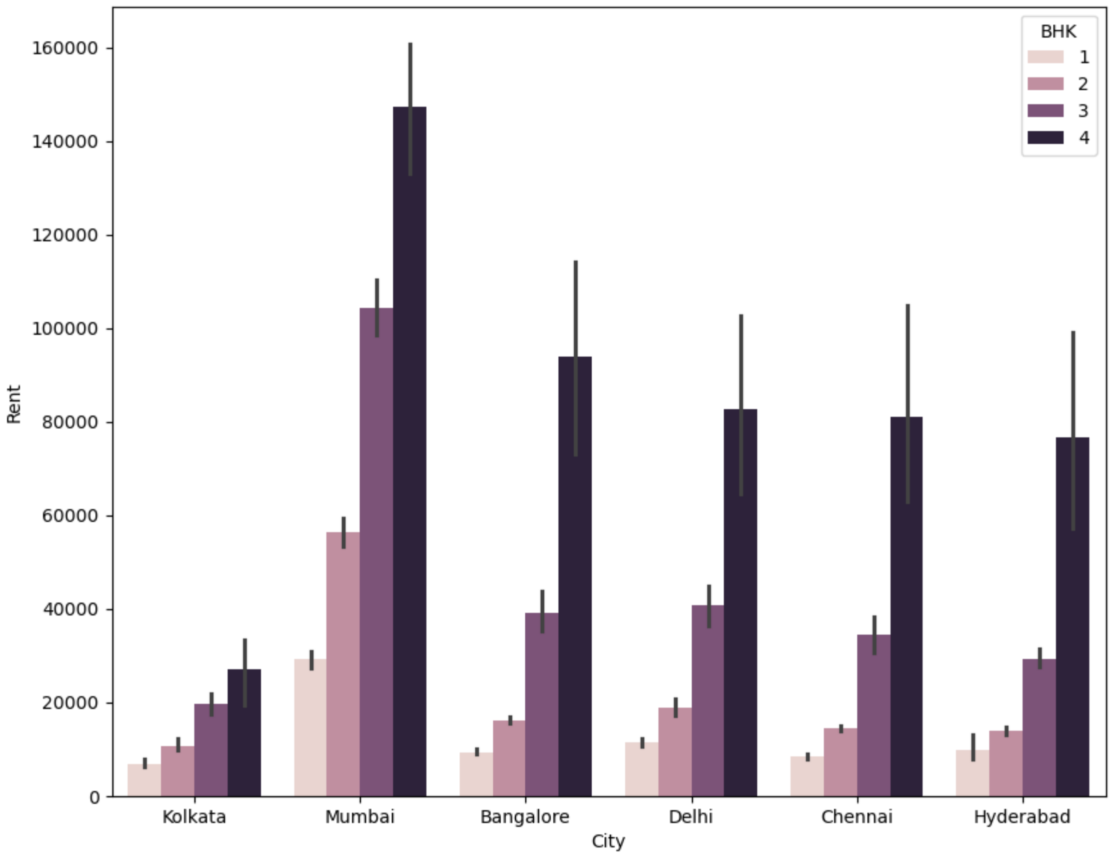
```
[31]: valid_rooms = [1, 2, 3, 4]
      rent_df = rent_df[rent_df['BHK'].isin(valid_rooms)]
      rent_df
```

```
[31]:       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
      …           …  …    …   …              …          …
      4741 2022-05-18    2  15000  1000       3 out of 5  Carpet Area
      4742 2022-05-15    3  29000  2000       1 out of 4   Super Area
      4743 2022-07-10    3  35000  1750       3 out of 5  Carpet Area
      4744 2022-07-06    3  45000  1500      23 out of 34  Carpet Area
      4745 2022-05-04    2  15000  1000       4 out of 5  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
…                        …         …               …               …
4741               Bandam Kommu  Hyderabad    Semi-Furnished  Bachelors/Family
4742       Manikonda, Hyderabad  Hyderabad    Semi-Furnished  Bachelors/Family
4743        Himayath Nagar, NH 7  Hyderabad    Semi-Furnished  Bachelors/Family
4744                 Gachibowli  Hyderabad    Semi-Furnished            Family
4745            Suchitra Circle  Hyderabad       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
…        …              …
4741         2    Contact Owner
4742         3    Contact Owner
4743         3    Contact Agent
4744         2    Contact Agent
4745         2    Contact Owner

[4633 rows x 12 columns]
```

```
[32]: plt.figure(figsize = (10,8))
      sns.barplot(data = rent_df , x = 'City',  y = 'Rent' , hue = 'BHK')
```

```
[32]: <Axes: xlabel='City', ylabel='Rent'>
```



[34]:

```
rent_df['BHK'].value_counts()
```

[34]:

```
BHK
2    2261
1    1167
3    1071
4     134
Name: count, dtype: int64
```

[35]:

```
rent_df.sort_values(by = 'Rent' , ascending = False)
```

[35]:

```
       Posted On  BHK    Rent  Size           Floor    Area Type  \
1392  2022-06-04    3  200000  1375    15 out of 60  Carpet Area
1238  2022-07-09    4  200000  2200    11 out of 20  Carpet Area
2990  2022-07-10    3  200000  3000      1 out of 1   Super Area
788   2022-05-14    3  200000  1208     5 out of 14  Carpet Area
3639  2022-06-14    4  200000  2280      2 out of 3  Carpet Area
…           …  …     …   …              …          …
506   2022-06-20    1    2200   700      1 out of 3   Super Area
2475  2022-06-22    2    2000    60      1 out of 1   Super Area
```

```
471   2022-05-12    1    1800   500  Ground out of 1   Super Area
285   2022-05-24    1    1500   200  Ground out of 2   Super Area
4076  2022-05-31    3    1200  2100       1 out of 3  Carpet Area

              Area Locality       City Furnishing Status  Tenant Preferred  \
1392  Raheja Imperia, Worli     Mumbai    Semi-Furnished            Family
1238        Seven Bungalows     Mumbai    Semi-Furnished  Bachelors/Family
2990  Madras Boat Club Road    Chennai         Furnished            Family
788               Khar West     Mumbai       Unfurnished         Bachelors
3639               Mylapore    Chennai    Semi-Furnished  Bachelors/Family
…                     …        …               …               …
506               Baranagar    Kolkata       Unfurnished  Bachelors/Family
2475              Ram Nagar      Delhi       Unfurnished  Bachelors/Family
471             Shyam Bazar    Kolkata    Semi-Furnished  Bachelors/Family
285              Santoshpur    Kolkata    Semi-Furnished  Bachelors/Family
4076          Uppal, NH 2 2  Hyderabad         Furnished  Bachelors/Family

      Bathroom Point of Contact
1392         3     Contact Agent
1238         5     Contact Agent
2990         4     Contact Agent
788          3     Contact Agent
3639         4     Contact Agent
…        …              …
506          1     Contact Owner
2475         1     Contact Owner
471          1     Contact Owner
285          1     Contact Owner
4076         3     Contact Owner

[4633 rows x 12 columns]
```

[36]:

```
(rent_df['Rent'] == 200000).value_counts()
```

[36]:

```
Rent
False    4618
True       15
Name: count, dtype: int64
```

[41]:

```
def clean_rent_data(data, numeric_cols):
    mask_rent = data['Rent'] <= 150000
    mask_furnishing = data['Furnishing Status'] != 'Unfurnished'
    data = data[mask_rent | mask_furnishing]

    data = data[~(data[numeric_cols] < 0).any(axis=1)]
    columns_to_remove = ['Posted On', 'Floor', 'Area Locality', 'Point of␣
↪Contact']
    data = data.drop(columns=columns_to_remove)

    data = data[data['Area Type'] != 'Built Area']

    return data

# Example usage
numeric_cols = ['Rent', 'Size', 'Bathroom']
```

[42]:

```
rent_df = clean_rent_data(rent_df, numeric_cols)
```

[43]:

```
rent_df
```

[43]:

```
      BHK   Rent  Size    Area Type       City Furnishing Status  \
0       2  10000  1100   Super Area    Kolkata       Unfurnished
1       2  20000   800   Super Area    Kolkata    Semi-Furnished
2       2  17000  1000   Super Area    Kolkata    Semi-Furnished
3       2  10000   800   Super Area    Kolkata       Unfurnished
4       2   7500   850  Carpet Area    Kolkata       Unfurnished
…     …     …    …          …        …               …
4741    2  15000  1000  Carpet Area  Hyderabad    Semi-Furnished
4742    3  29000  2000   Super Area  Hyderabad    Semi-Furnished
4743    3  35000  1750  Carpet Area  Hyderabad    Semi-Furnished
4744    3  45000  1500  Carpet Area  Hyderabad    Semi-Furnished
4745    2  15000  1000  Carpet Area  Hyderabad       Unfurnished

      Tenant Preferred  Bathroom
0     Bachelors/Family         2
1     Bachelors/Family         1
2     Bachelors/Family         1
3     Bachelors/Family         1
4            Bachelors         1
…                  …        …
4741  Bachelors/Family         2
4742  Bachelors/Family         3
4743  Bachelors/Family         3
4744            Family         2
4745         Bachelors         2

[4622 rows x 8 columns]
```

[44]:

```
 from sklearn.preprocessing import LabelEncoder

def encode_data(rent_df):
    # Columns to encode
    cat_cols = ['City', 'Area Type', 'Furnishing Status', 'Tenant Preferred']

    # Apply label encoding to each column
    for col in cat_cols:
        rent_df[col] = rent_df[col].astype('category')
        le = LabelEncoder()
        rent_df[col] = le.fit_transform(rent_df[col])

    return rent_df
```

[45]:
```
rent_df =  encode_data(rent_df)
```

[58]:
```
from sklearn.model_selection import train_test_split

X = rent_df.drop(columns=['Rent'])  # Features
y = rent_df['Rent']                 # Target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[59]:
```
X_train
rent_df
```

[59]:
```
      BHK   Rent  Area Type  City  Furnishing Status  Tenant Preferred  \
0       2  10000          1     4                  2                 1
1       2  20000          1     4                  1                 1
2       2  17000          1     4                  1                 1
3       2  10000          1     4                  2                 1
4       2   7500          0     4                  2                 0
…     …    …          …   …                 …                …
4741    2  15000          0     3                  1                 1
4742    3  29000          1     3                  1                 1
4743    3  35000          0     3                  1                 1
4744    3  45000          0     3                  1                 2
4745    2  15000          0     3                  2                 0

      Bathroom  Size_scaled
0            2     0.227557
1            1     0.164927
2            1     0.206681
3            1     0.164927
4            1     0.175365
…          …          …
4741         2     0.206681
4742         3     0.415449
4743         3     0.363257
4744         2     0.311065
```

```
4745         2     0.206681

[4622 rows x 8 columns]
```

[50]:
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler

def scale_data(rent_df):
    """
    Scales the 'Size' column of a DataFrame using MinMaxScaler,
    adds the scaled size as a new column 'Size_scaled',
    drops the original 'Size' column, and
    separates the DataFrame into features (X) and target (y).

    Args:
        rent_df (pd.DataFrame): The input DataFrame containing at least
                                'Size' and 'Rent' columns.

    Returns:
        tuple: A tuple containing:
               - X (pd.DataFrame): DataFrame of features (excluding 'Rent').
               - y (pd.Series): Series of the target variable 'Rent'.
               - rent_df (pd.DataFrame): The modified DataFrame with␣
↪'Size_scaled'
                                         and without the original 'Size' column.
    """
    scaler = MinMaxScaler()

    # Scale the 'Size' column. Reshape is needed for single feature scaling.
    scaled = scaler.fit_transform(rent_df[['Size']].values.reshape(-1, 1))

    # Add the scaled 'Size' as a new column
    rent_df['Size_scaled'] = scaled

    # Drop the original 'Size' column
    rent_df.drop(columns=['Size'], inplace=True)

    # Separate features (X) and target (y)
    X = rent_df.drop(columns='Rent')
    y = rent_df['Rent']

    return X, y, rent_df
```

[51]:
```
X, y, rent_df = scale_data(rent_df)
```

[54]:
```
X, y, rent_df
```

```
[54]: (      BHK  Area Type  City  Furnishing Status  Tenant Preferred  Bathroom  \
       0       2          1     4                  2                 1         2
       1       2          1     4                  1                 1         1
       2       2          1     4                  1                 1         1
       3       2          1     4                  2                 1         1
       4       2          0     4                  2                 0         1
       …   …        …   …                …               …       …
       4741    2          0     3                  1                 1         2
       4742    3          1     3                  1                 1         3
       4743    3          0     3                  1                 1         3
       4744    3          0     3                  1                 2         2
       4745    2          0     3                  2                 0         2

             Size_scaled
       0        0.227557
       1        0.164927
       2        0.206681
       3        0.164927
       4        0.175365
       …           …
       4741     0.206681
       4742     0.415449
       4743     0.363257
       4744     0.311065
       4745     0.206681

       [4622 rows x 7 columns],
       0       10000
       1       20000
       2       17000
       3       10000
       4        7500
               …
       4741    15000
       4742    29000
       4743    35000
       4744    45000
       4745    15000
       Name: Rent, Length: 4622, dtype: int64,
             BHK   Rent  Area Type  City  Furnishing Status  Tenant Preferred  \
       0       2  10000          1     4                  2                 1
       1       2  20000          1     4                  1                 1
       2       2  17000          1     4                  1                 1
       3       2  10000          1     4                  2                 1
       4       2   7500          0     4                  2                 0
       …   …    …        …   …                …                …
       4741    2  15000          0     3                  1                 1
       4742     3  29000          1     3                  1                 1
       4743     3  35000          0     3                  1                 1
       4744     3  45000          0     3                  1                 2
       4745     2  15000          0     3                  2                 0

             Bathroom  Size_scaled
       0            2     0.227557
       1            1     0.164927
       2            1     0.206681
       3            1     0.164927
       4            1     0.175365
       …          …            …
       4741         2     0.206681
       4742         3     0.415449
       4743         3     0.363257
       4744         2     0.311065
       4745         2     0.206681

       [4622 rows x 8 columns])
```

[95]:

```
from sklearn.model_selection import train_test_split
import pandas as pd

best_random_state = 42

# Train Test Split the Data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=best_random_state)

print("Data split successfully!")
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

```
Data split successfully!
X_train shape: (3697, 7)
X_test shape: (925, 7)
y_train shape: (3697,)
y_test shape: (925,)
```

[63]:

```
X_train
```

[63]:

```
      BHK  Area Type  City  Furnishing Status  Tenant Preferred  Bathroom  \
2814    2          1     2                  2                 1         1
3094    2          1     1                  1                 1         2
4237    2          1     3                  2                 1         2
4661    1          0     3                  1                 1         1
```

```
4560    3          0     3                  0                 2         2
…    …        …    …                …               …       …
4548    1          0     3                  1                 1         2
469     2          1     4                  2                 1         1
3197    1          0     1                  1                 2         1
3885    2          1     3                  1                 1         2
895     3          0     5                  0                 2         3

      Size_scaled
2814     0.013570
3094     0.133612
4237     0.248434
4661     0.123173
4560     0.263048
…             …
4548     0.144050
469      0.144050
3197     0.144050
3885     0.192067
895      0.212944

[3697 rows x 7 columns]
```

[65]:

```
model = LinearRegression()
model.fit(X_train, y_train)
```

[65]:

```
LinearRegression()
```

[66]:

```
y_pred = model.predict(X_test)
```

[67]:

```
import math
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Calculate Mean Squared Error (MSE)
MSE = mean_squared_error(y_test, y_pred)

# Calculate Mean Absolute Error (MAE)
MAE = mean_absolute_error(y_test, y_pred)

# Calculate R-squared (R2S)
R2S = r2_score(y_test, y_pred)


RMSE = math.sqrt(MSE)

n = len(y_test)
p = X_train.shape[1]
```

```
ADJ_R2S = 1 - (1 - R2S) * (n - 1) / (n - p - 1)

# Print the calculated metrics, rounded to specified decimal places
print("The MSE :", round(MSE, 3))
print("The RMSE :", round(RMSE, 3))
print("The MAE :", round(MAE, 3))
print("The R^2 :", round(R2S, 2))
print("The ADJ_R^2 :", round(ADJ_R2S, 2))
```

```
The MSE : 428258309.935
The RMSE : 20694.403
The MAE : 14501.248
The R^2 : 0.5
The ADJ_R^2 : 0.5
```

[83]:

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import cross_val_score

models = {
    "Linear Regression":LinearRegression(),
    "Gradient Boosting":GradientBoostingRegressor()
}

for model_name, current_model in models.items():
    scores = cross_val_score(current_model, X_train, y_train, cv=5,␣
 ↪scoring='neg_mean_squared_error')
    rmse_score = np.sqrt(-scores) # CONVERT -VE MSE TO RMSE
    mean_rmse = rmse_score.mean()
    std_rmse = rmse_score.std()
    print(f"[{model_name}] RMSE: {mean_rmse:.2f} (+/- {std_rmse:.2f})")
```

```
[Linear Regression] RMSE: 21876.21 (+/- 1318.68)
[Gradient Boosting] RMSE: 14976.51 (+/- 687.60)
```

[84]:

```
residuals = y_test - y_pred
```

[85]:

```
residuals
```

[85]:

```
3043     -3648.297280
4197    -36874.612695
1823     -9795.767151
4103    -63981.166102
3372     17434.820675
             …
2895     -3477.338522
3741     -6947.005345
```

```
1041     -6242.886228
1873    -31321.484458
1526      4509.714135
Name: Rent, Length: 925, dtype: float64
```

[87]:

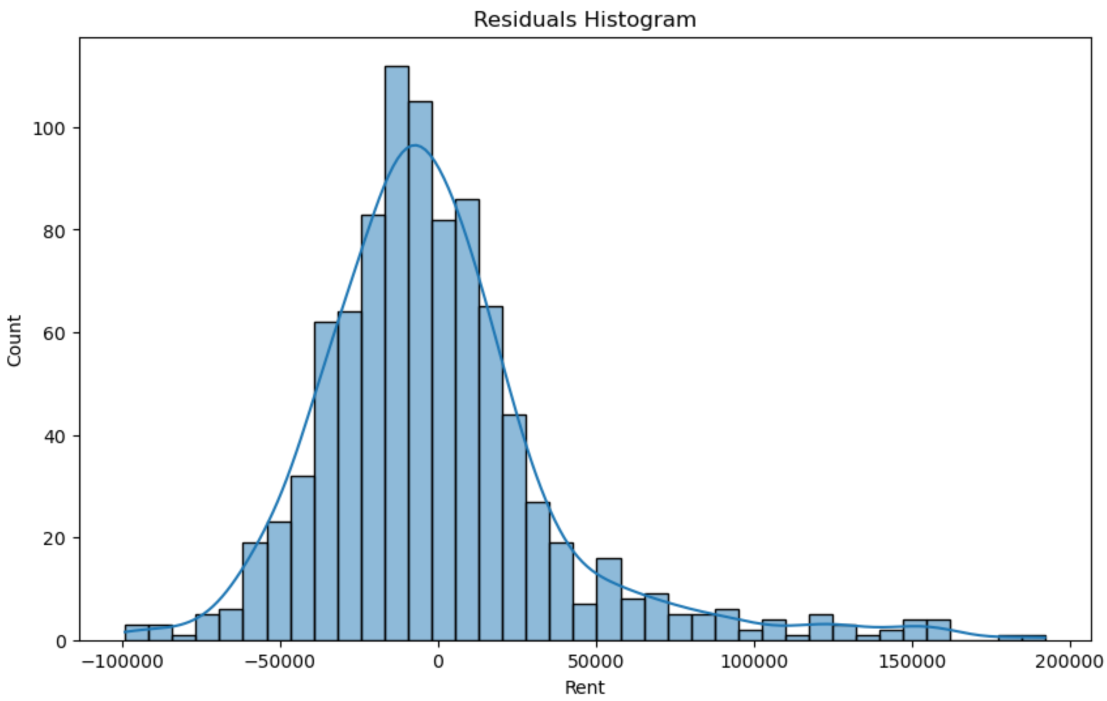
```
# Create a dictionary to hold the predicted values and residuals
data = {
    'Predicted': y_pred,
    'Residuals': residuals
}

# Create a Pandas DataFrame from the dictionary for analysis
analysis = pd.DataFrame(data)
analysis
```

[87]:

```
         Predicted     Residuals
3043  18648.297280  -3648.297280
4197  44374.612695 -36874.612695
1823  17295.767151  -9795.767151
4103  75981.166102 -63981.166102
3372  -5434.820675  17434.820675
…              …             …
2895  12977.338522  -3477.338522
3741  16947.005345  -6947.005345
1041  36242.886228  -6242.886228
1873  46321.484458 -31321.484458
1526   5990.285865   4509.714135

[925 rows x 2 columns]
```

[88]:

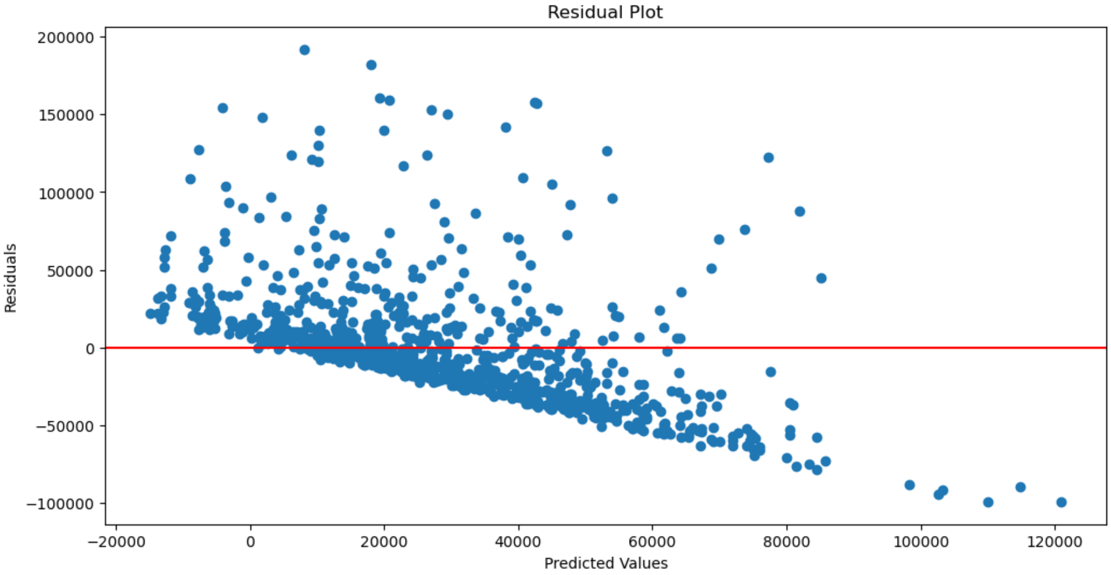
```
import matplotlib.pyplot as plt
import seaborn as sns
# Create a figure and a set of subplots with a specified size
plt.figure(figsize=(10, 6))

# Plot a histogram of the residuals
# kde=True adds a Kernel Density Estimate (KDE) line, which shows the␣
 ↪probability
# density of the data, helping to visualize the distribution's shape.
sns.histplot(residuals, kde=True)

# Set the title of the histogram
plt.title("Residuals Histogram")

# Display the plot
plt.show()
```



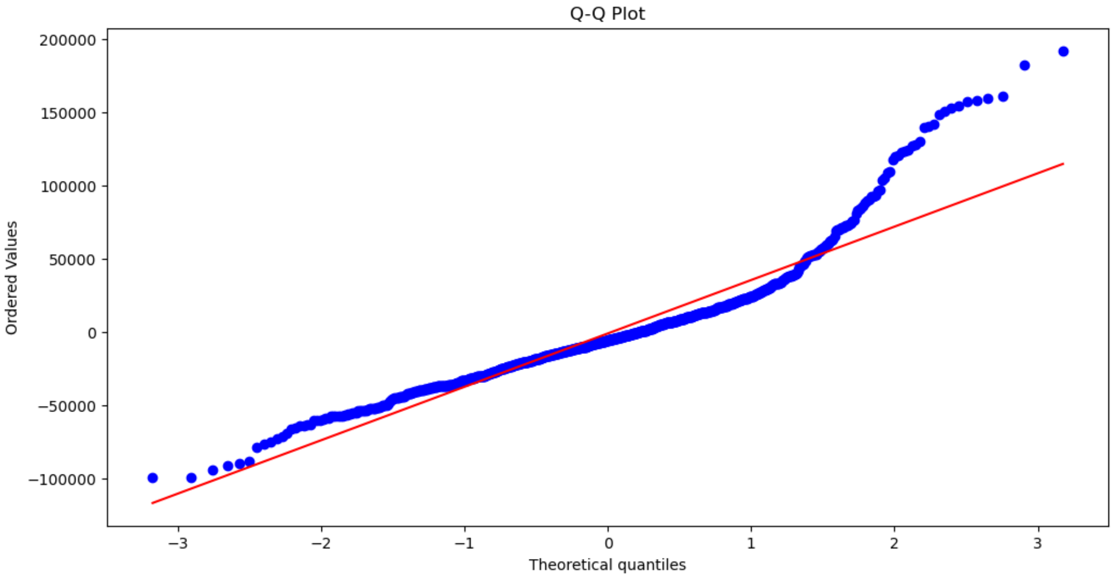
[96]:

```
import matplotlib.pyplot as plt

# Create a new figure with a specified size for the plot
plt.figure(figsize=(12, 6))

# Create a scatter plot of predicted values vs. residuals
# This plot helps to identify patterns in the errors (e.g., heteroscedasticity)
plt.scatter(y_pred, residuals)

# Add a horizontal line at y=0 (red, solid line)
# This line serves as a reference to easily see if residuals are centered␣
 ↪around zero
plt.axhline(y=0, color='r', linestyle='-')

# Set the title of the plot
plt.title("Residual Plot")

# Set the label for the x-axis
plt.xlabel("Predicted Values")

# Set the label for the y-axis
plt.ylabel("Residuals")

# Display the plot
```

```
plt.show()
```

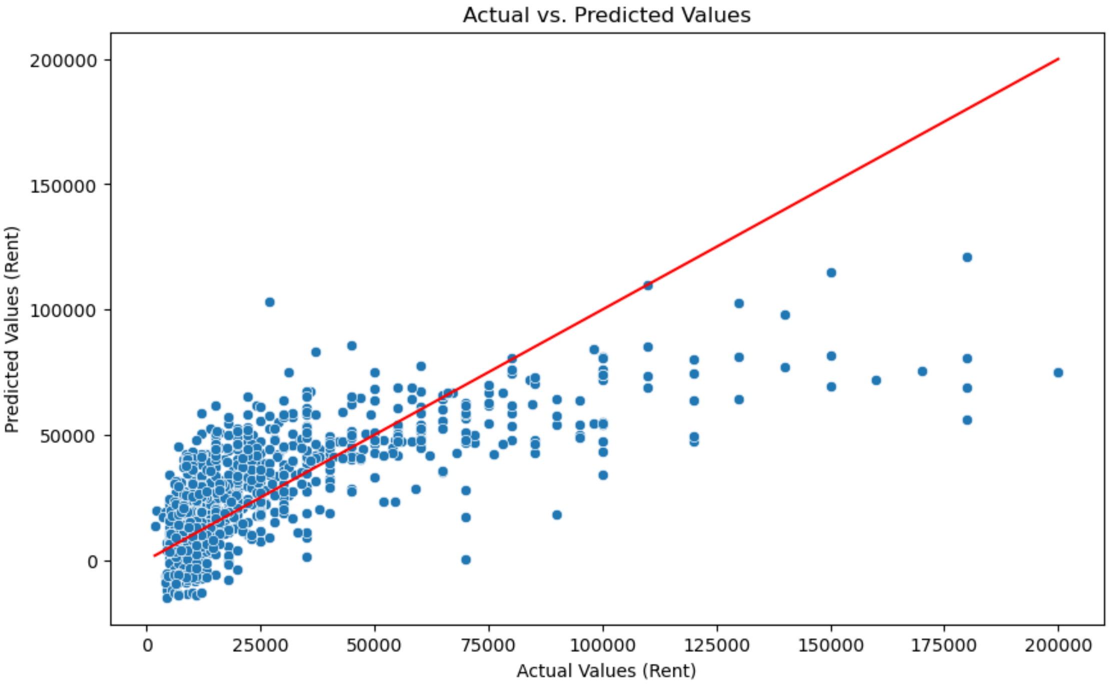


[97]:
```
import scipy.stats as stats
import matplotlib.pyplot as plt # Import matplotlib for figure and show
import pylab # Often imported with scipy.stats.probplot for direct plotting

# Assuming 'residuals' is a pandas Series or a numpy array containing the␣
  ↪residuals
# calculated from y_test - y_pred.

# Create a new figure with a specified size for the plot
plt.figure(figsize=(12, 6))


stats.probplot(residuals, dist="norm", plot=pylab)

# Set the title of the Q-Q plot
pylab.title("Q-Q Plot")

# Display the plot
pylab.show()
```



[102]:

```
def plot_predictions(y_test, y_pred):
    actual_values = y_test
    predicted_values = y_pred

    plt.figure(figsize=(10, 6))
    sns.scatterplot(x=actual_values, y=predicted_values)

    plt.xlabel("Actual Values (Rent)")
    plt.ylabel("Predicted Values (Rent)")
    plt.title("Actual vs. Predicted Values")

    plt.plot([min(actual_values), max(actual_values)],
             [min(actual_values), max(actual_values)], color='red')

    plt.show()

plot_predictions(y_test, y_pred)
```



```
[103]: results = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
```

```
[104]: results
```

```
[104]:       Actual      Predicted
       744    25000   18648.297280
       504    40000   44374.612695
       1674   15000   17295.767151
       906   100000   75981.166102
       2238    6000   -5434.820675
       …        …            …
       2832    9300   12977.338522
       2815   15000   16947.005345
       4064   25000   36242.886228
       576    40000   46321.484458
       2330   11000    5990.285865

       [925 rows x 2 columns]
```

```
[ ]:
```