Lab Exercise 1

Predicting House prices

Submitted by: Nigel Haim N. Sebastian

Importing

- When Imported pandas it prompts me that "Pyarrow will become a required dependency of pandas in the next major release of pandas (pandas 3.0)"

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy

from sklearn.preprocessing import StandardScaler, minmax_scale
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
from sklearn import linear_model
```

Loading and checking the dataset

In [ ]:
```
data = pd.read_csv("House_Rent_Dataset.csv")
data.value_counts()
```

Out[ ]:
```
Posted On   BHK  Rent    Size  Floor           Area Type    Area Locality
City       Furnishing Status  Tenant Preferred  Bathroom  Point of Contact
2022-04-13  3    260000  1800  10 out of 11     Carpet Area  JVPD Scheme
Mumbai     Furnished          Family           4         Contact Agent       1
2022-06-22  3    17000   1200  Ground out of 2  Super Area   Selaiyur
Chennai    Unfurnished        Bachelors/Family  3         Contact Owner       1
                  15000   2200  1 out of 2       Super Area   Irumbuliyur
Chennai    Unfurnished        Bachelors         3         Contact Owner       1
                         1850  1 out of 1       Super Area   Tarapuri Colony, Secu
nderabad  Hyderabad  Semi-Furnished     Bachelors/Family  2         Contact Owner
1
                  13000   900   1 out of 4       Super Area   Perambur
Chennai    Unfurnished        Bachelors/Family  1         Contact Owner       1

..
2022-05-26  4    37000   2700  2 out of 3       Carpet Area  AGCR Enclave, Anand V
ihar      Delhi      Semi-Furnished     Bachelors/Family  4         Contact Agent
1
            3    80000   2000  3 out of 4       Carpet Area  Sarvodaya Enclave
Delhi      Unfurnished        Bachelors/Family  3         Contact Agent       1
                  69999   950   7 out of 12      Carpet Area  Chembur East
Mumbai     Unfurnished        Family           3         Contact Agent       1
                  37000   1800  1 out of 4       Carpet Area  Preet Vihar
Delhi      Semi-Furnished     Bachelors/Family  3         Contact Agent       1
2022-07-11  2    12000   550   Ground out of 1  Super Area   Choolaimedu
Chennai    Unfurnished        Bachelors/Family  2         Contact Owner       1
Name: count, Length: 4746, dtype: int64
```

In [ ]:
```
data.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
```

Check for any errors in the floor

In [ ]:
```
index = 0
```

```
def Floor_Score(row):
    r = str(row).split(' out of ')
    if len(r) == 1:
        return None
    try:
        if 'GROUND' in str(r).upper():
            r[0] = 1
            # r[1] = int(r[1]) + 1
            # print(r[0], r[1])

        elif 'UPPER BASEMENT' in str(r).upper():
            r[0] = 2
            # r[1] = int(r[1]) + 2
            # r[1] = int(r[1])
        elif 'LOWER BASEMENT' in str(r).upper():
            r[0] = 3
            # r[1] = int(r[1]) + 3
        else:
            r[0] = int(r[0])
            r[1] = int(r[1])
        if int(r[0]) > int(r[1]):
            return None

        return int(r[0]) / int(r[1])
    except:
        print("Error")
    index += 1


data['Floor'] = data['Floor'].apply(Floor_Score)
data.dropna(subset=['Floor'], inplace=True)
data.info()
data.head(20)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4732 entries, 0 to 4745
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Posted On          4732 non-null   object
 1   BHK                4732 non-null   int64
 2   Rent               4732 non-null   int64
 3   Size               4732 non-null   int64
 4   Floor              4732 non-null   float64
 5   Area Type          4732 non-null   object
 6   Area Locality      4732 non-null   object
 7   City               4732 non-null   object
 8   Furnishing Status  4732 non-null   object
 9   Tenant Preferred   4732 non-null   object
 10  Bathroom           4732 non-null   int64
 11  Point of Contact   4732 non-null   object
dtypes: float64(1), int64(4), object(7)
memory usage: 480.6+ KB
```

Out[ ]:

	Posted On	BHK	Rent	Size	Floor	Area Type	Area Locality	City	Furnishing Status	
0	2022-05-18	2	10000	1100	0.500000	Super Area	Bandel	Kolkata	Unfurnished	Bachelc
1	2022-05-13	2	20000	800	0.333333	Super Area	Phool Bagan, Kankurgachi	Kolkata	Semi-Furnished	Bachelc
2	2022-05-16	2	17000	1000	0.333333	Super Area	Salt Lake City Sector 2	Kolkata	Semi-Furnished	Bachelc
3	2022-07-04	2	10000	800	0.500000	Super Area	Dumdum Park	Kolkata	Unfurnished	Bachelc
4	2022-05-09	2	7500	850	0.500000	Carpet Area	South Dum Dum	Kolkata	Unfurnished	
5	2022-04-29	2	7000	600	1.000000	Super Area	Thakurpukur	Kolkata	Unfurnished	Bachelc
6	2022-06-21	2	10000	700	0.250000	Super Area	Malancha	Kolkata	Unfurnished	
7	2022-06-21	1	5000	250	0.500000	Super Area	Malancha	Kolkata	Unfurnished	
8	2022-06-07	2	26000	800	0.500000	Carpet Area	Palm Avenue Kolkata, Ballygunge	Kolkata	Unfurnished	
9	2022-06-20	2	10000	1000	0.333333	Carpet Area	Natunhat	Kolkata	Semi-Furnished	Bachelc
10	2022-05-23	3	25000	1200	0.250000	Carpet Area	Action Area 1, Rajarhat Newtown	Kolkata	Semi-Furnished	Bachelc
11	2022-06-07	1	5000	400	1.000000	Carpet Area	Keshtopur	Kolkata	Unfurnished	Bachelc
12	2022-05-14	1	6500	250	0.250000	Carpet Area	Tarulia, Keshtopur	Kolkata	Furnished	
13	2022-05-09	1	5500	375	0.500000	Carpet Area	Dum Dum Metro	Kolkata	Unfurnished	Bachelc
14	2022-05-05	3	8500	900	0.500000	Carpet Area	Paschim Barisha	Kolkata	Unfurnished	
15	2022-06-01	3	40000	1286	1.000000	Carpet Area	New Town Action Area 1	Kolkata	Furnished	Bachelc
16	2022-05-17	2	6000	600	0.500000	Super Area	Barasat	Kolkata	Semi-Furnished	Bachelc

	Posted On	BHK	Rent	Size	Floor	Area Type	Area Locality	City	Furnishing Status	
17	2022-06-20	2	10000	800	0.500000	Super Area	Behala	Kolkata	Unfurnished	Bachelc
18	2022-06-09	2	11000	2000	0.333333	Carpet Area	Behala Chowrasta	Kolkata	Unfurnished	Bachelc
19	2022-06-09	2	6000	660	0.500000	Super Area	Behala	Kolkata	Unfurnished	Bachelc

In [ ]:
```
CheckNegative = data['Rent'].where(data['Rent'] < 0)
CheckNegative.head()
```

Out[ ]:
```
0   NaN
1   NaN
2   NaN
3   NaN
4   NaN
Name: Rent, dtype: float64
```

In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4732 entries, 0 to 4745
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Posted On          4732 non-null   object 
 1   BHK                4732 non-null   int64  
 2   Rent               4732 non-null   int64  
 3   Size               4732 non-null   int64  
 4   Floor              4732 non-null   float64
 5   Area Type          4732 non-null   object 
 6   Area Locality      4732 non-null   object 
 7   City               4732 non-null   object 
 8   Furnishing Status  4732 non-null   object 
 9   Tenant Preferred   4732 non-null   object 
 10  Bathroom           4732 non-null   int64  
 11  Point of Contact   4732 non-null   object 
dtypes: float64(1), int64(4), object(7)
memory usage: 480.6+ KB
```

This concludes that there are no null values in the dataset

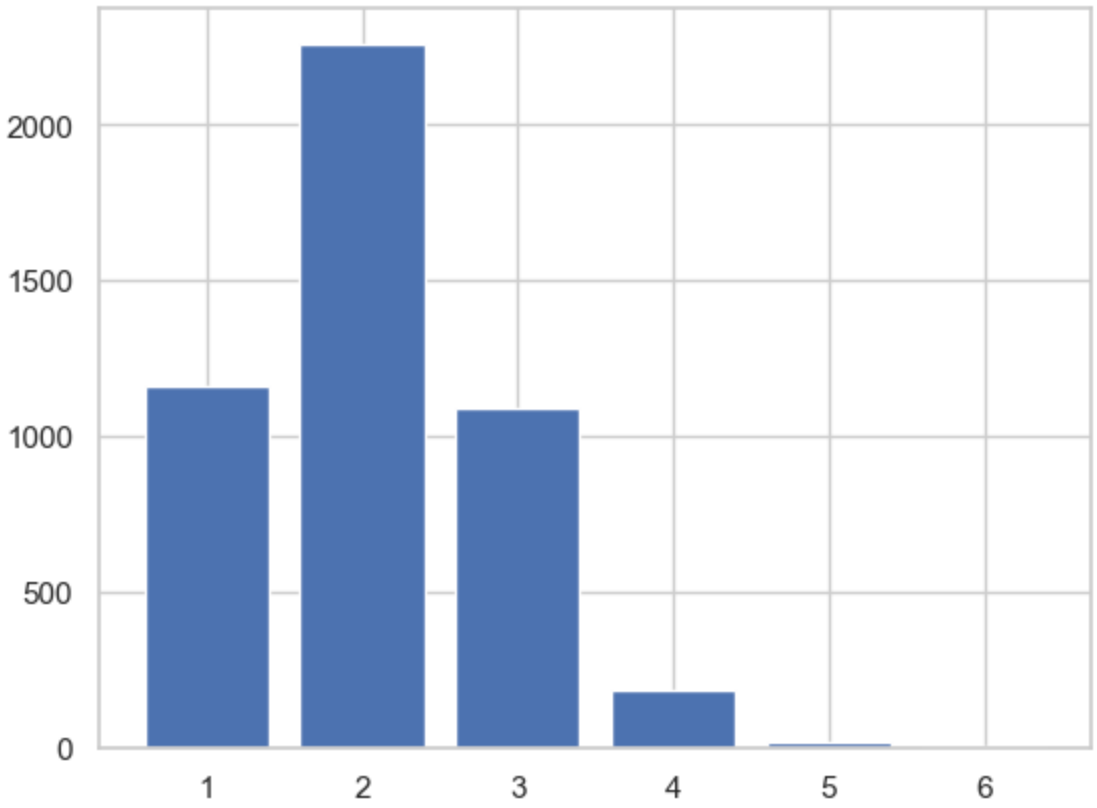
In [ ]:
```
data.isnull().sum()
```

Out[ ]:
```
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
```

Data Analysis

In [ ]:
```
plt.bar(data['BHK'].value_counts().index, data['BHK'].value_counts().values)
```

Out[ ]:
```
<BarContainer object of 6 artists>
```

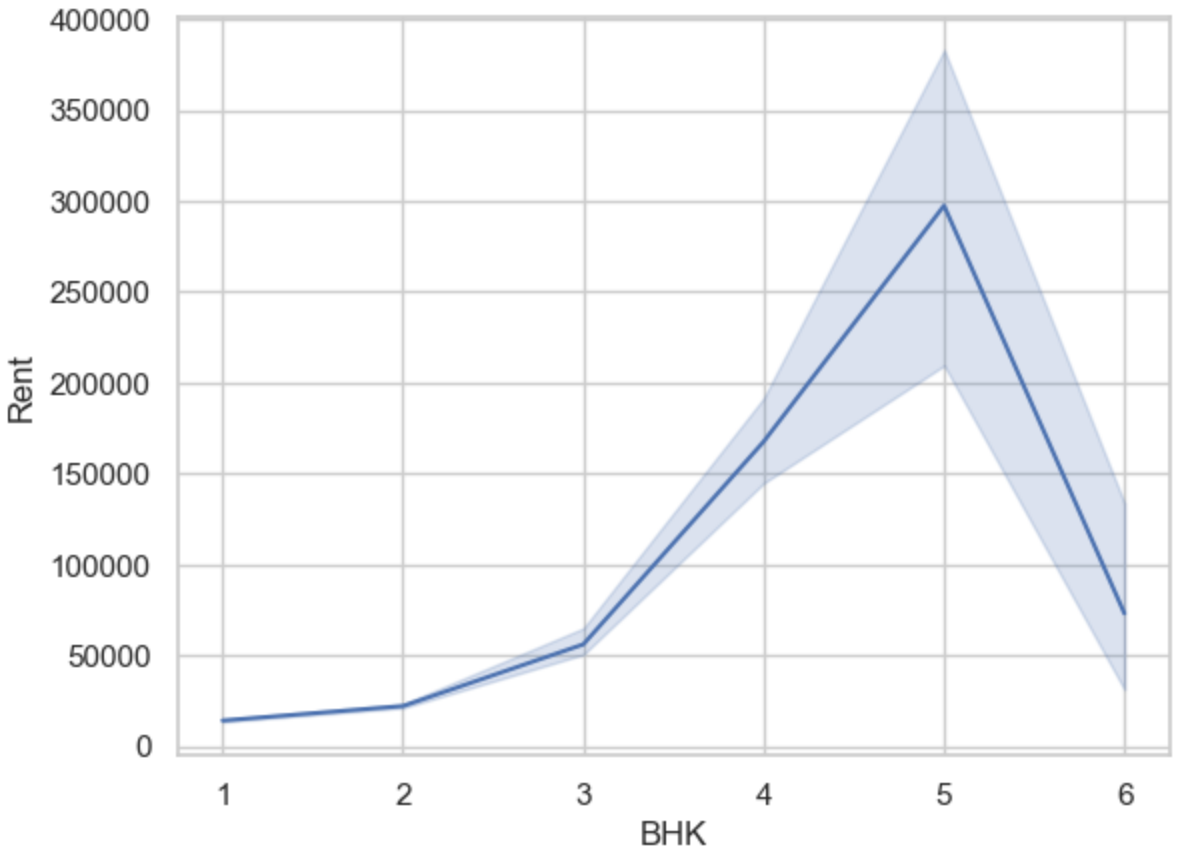


In [ ]:
```
data['Area Locality'].value_counts()
```

Out[ ]:
```
Area Locality
Bandra West                                                  37
Gachibowli                                                   29
Electronic City                                              24
Miyapur, NH 9                                                22
Velachery                                                    22
                                                             ..
Irla, Vile Parle West                                         1
Serenity Complex                                              1
Charkop Sector 6                                              1
Hiranandani Gardens Odyssey I, Hiranandani Gardens            1
Manikonda, Hyderabad                                          1
Name: count, Length: 2231, dtype: int64
```

In [ ]:
```
sns.lineplot(x=data['BHK'], y=data['Rent'], data=data)
```

Out[ ]:
```
<Axes: xlabel='BHK', ylabel='Rent'>
```

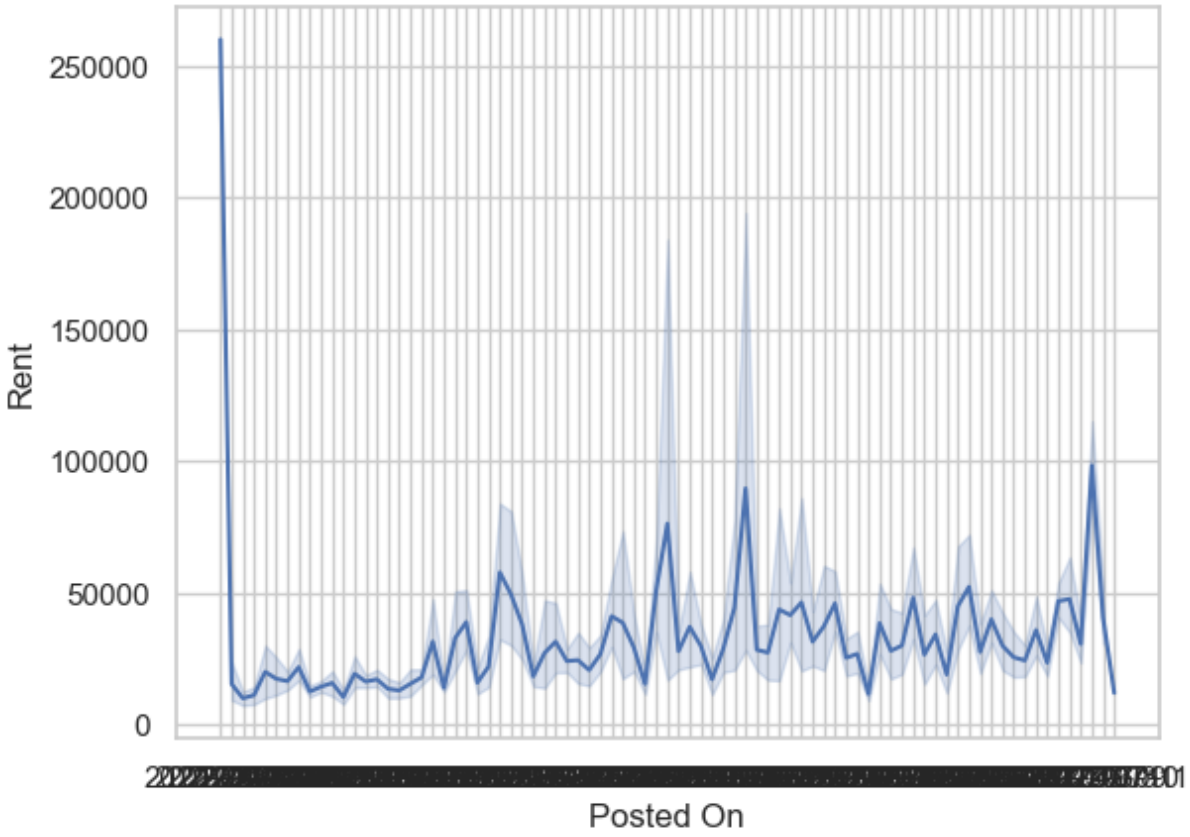


In [ ]:
```
sorted_date = data.sort_values(by='Posted On')

# sorted_date.head(20)
```

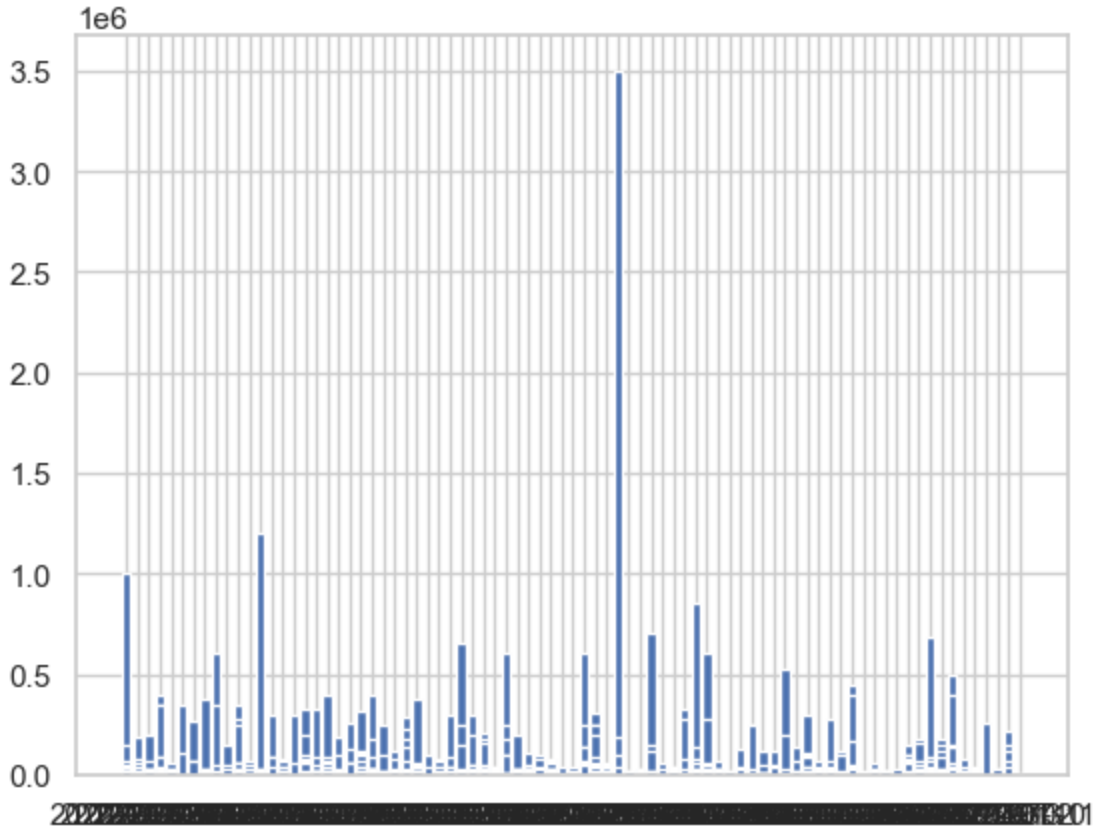
In [ ]:
```
sns.set_theme(style="whitegrid")
sns.lineplot(x=sorted_date['Posted On'], y=sorted_date['Rent'], data=data)
```

Out[ ]:
```
<Axes: xlabel='Posted On', ylabel='Rent'>
```



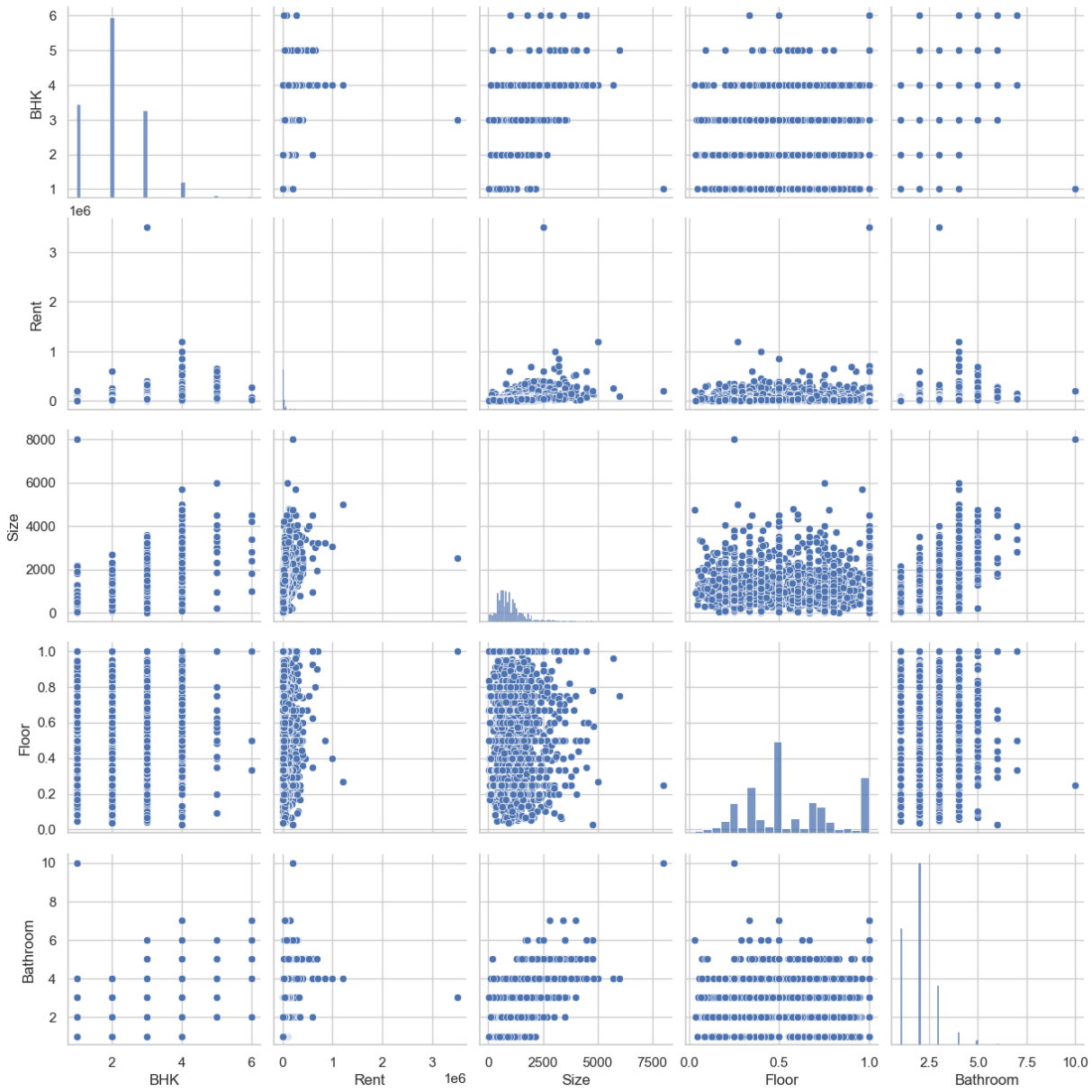
In [ ]:
```
plt.bar(data['Posted On'], data['Rent'])
```

Out[ ]: <BarContainer object of 4732 artists>



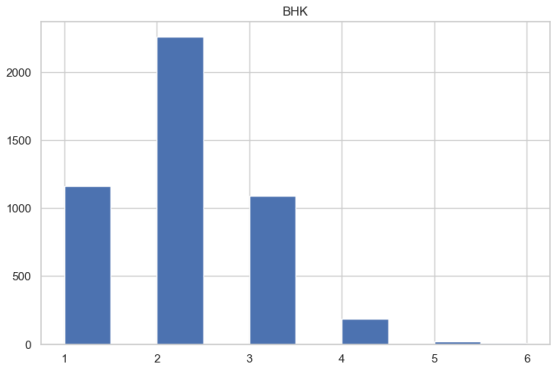
In [ ]:
```
sns.pairplot(data)
```

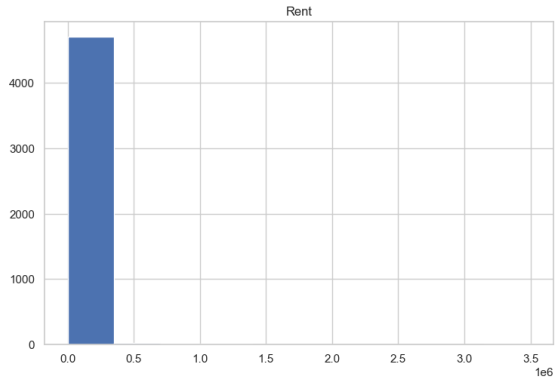
Out[ ]:
```
<seaborn.axisgrid.PairGrid at 0x1e74a254ce0>
```

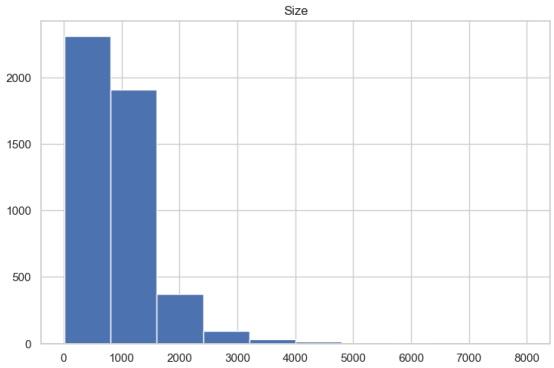


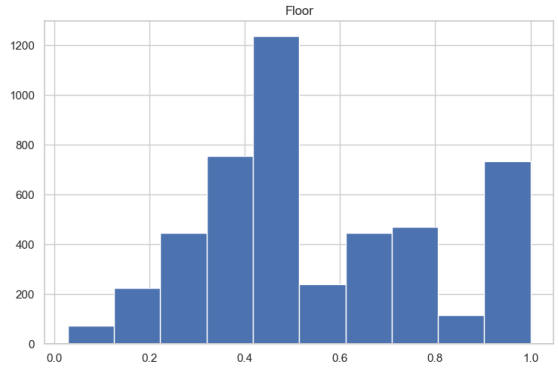
In [ ]:
```
data.hist(figsize=(20, 20))
```

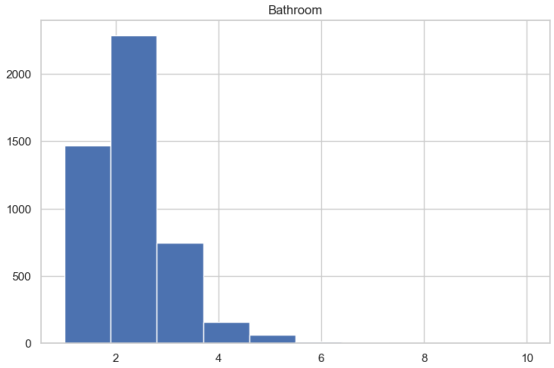
Out[ ]:
```
array([[<Axes: title={'center': 'BHK'}>,
        <Axes: title={'center': 'Rent'}>],
       [<Axes: title={'center': 'Size'}>,
        <Axes: title={'center': 'Floor'}>],
       [<Axes: title={'center': 'Bathroom'}>, <Axes: >]], dtype=object)
```





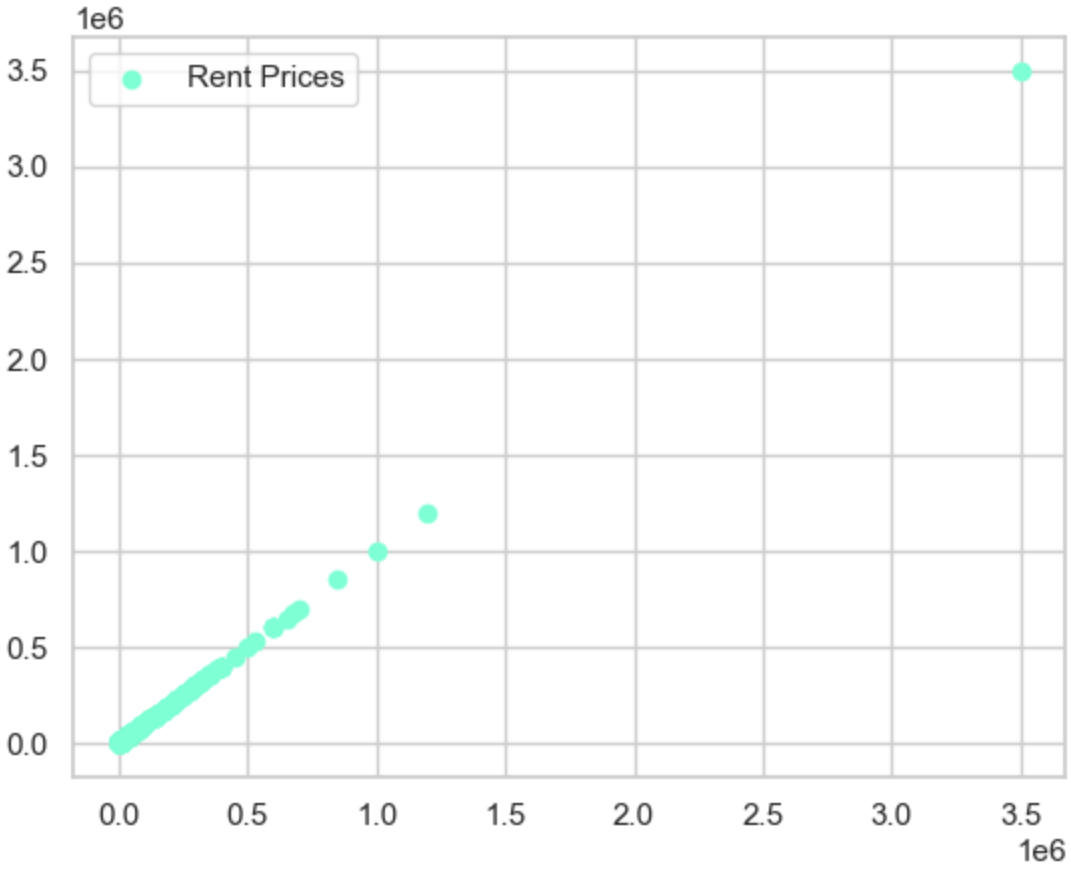






In [ ]:
```
plt.figure()
plt.scatter(data['Rent'], data['Rent'], color='aquamarine', label='Rent Prices' )
plt.legend()
```

Out[ ]: <matplotlib.legend.Legend at 0x1e78a338860>



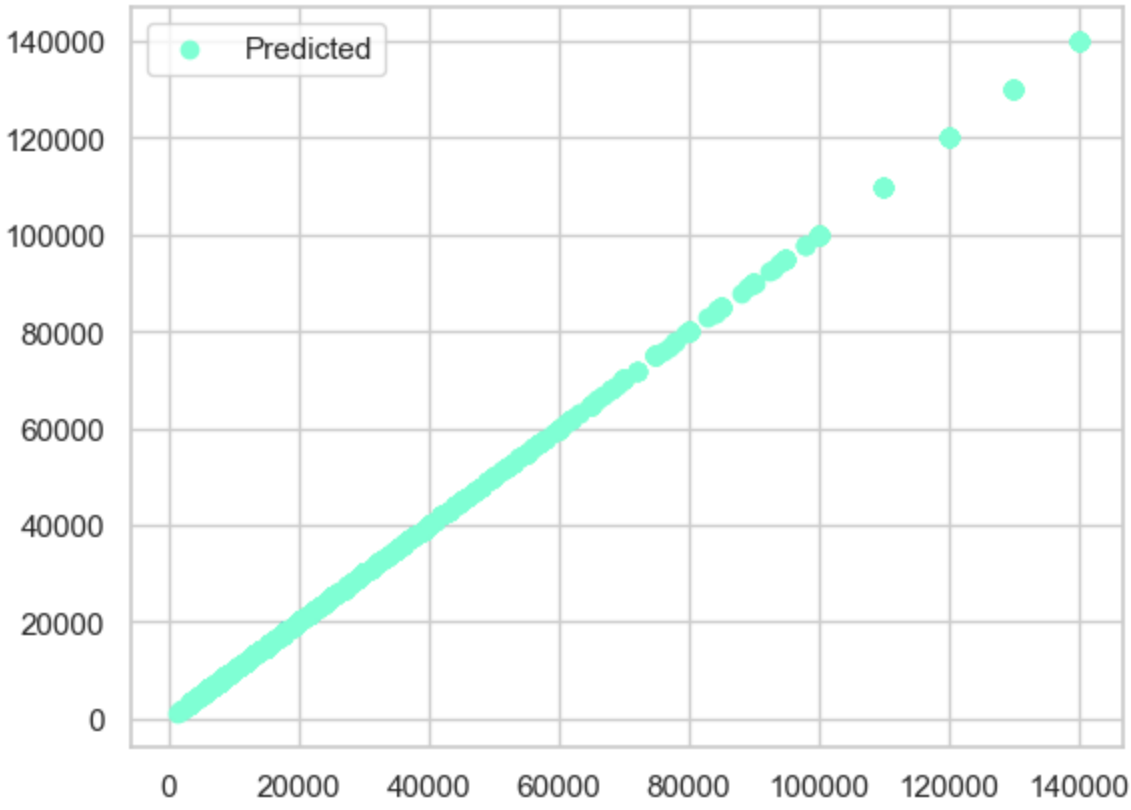
In [ ]:

In [ ]:
```
data = data[data['Rent'] < 150000]
plt.figure()
plt.scatter(data['Rent'], data['Rent'], color='aquamarine', label='Predicted' )
plt.legend()

# data = data[data['Rent'] < 120000]
# plt.figure()
# plt.scatter(data['Rent'], data['Rent'], color='aquamarine', label='Predicted' )
# plt.legend()

# data = data[data['Rent'] < 100000]
# plt.figure()
# plt.scatter(data['Rent'], data['Rent'], color='aquamarine', label='Predicted' )
# plt.legend()
```

Out[ ]: <matplotlib.legend.Legend at 0x1e78bfde5a0>



In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4534 entries, 0 to 4745
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Posted On          4534 non-null   object 
 1   BHK                4534 non-null   int64  
 2   Rent               4534 non-null   int64  
 3   Size               4534 non-null   int64  
 4   Floor              4534 non-null   float64
 5   Area Type          4534 non-null   object 
 6   Area Locality      4534 non-null   object 
 7   City               4534 non-null   object 
 8   Furnishing Status  4534 non-null   object 
 9   Tenant Preferred   4534 non-null   object 
 10  Bathroom           4534 non-null   int64  
 11  Point of Contact   4534 non-null   object 
dtypes: float64(1), int64(4), object(7)
memory usage: 460.5+ KB
```

Preprocessing

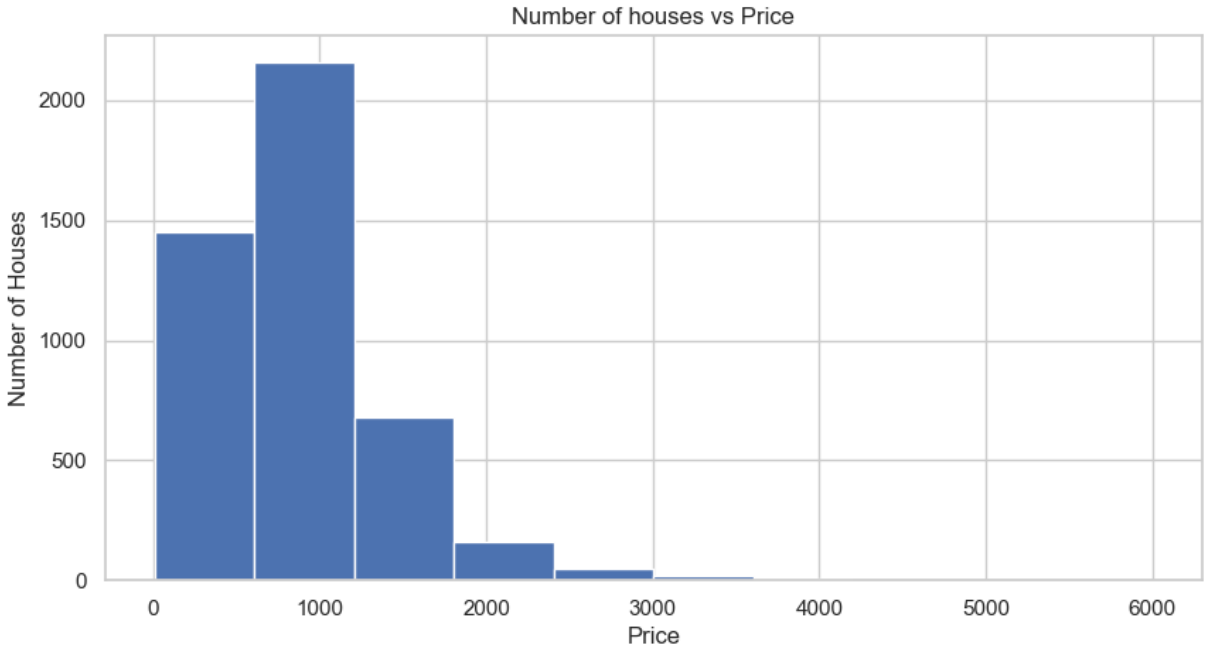
Logarithmic Transformation in Linear Regression Models

https://dev.to/rokaandy/logarithmic-transformation-in-linear-regression-models-why-when-3a7c

https://stats.stackexchange.com/questions/145383/getting-negative-predicted-values-after-linear-regression

In [ ]:
```
data.hist('Size', figsize=(10, 5))
plt.title('Number of houses vs Price')
plt.ylabel('Number of Houses')
plt.xlabel("Price")
```

Out[ ]:
```
Text(0.5, 0, 'Price')
```



After

In [ ]:
```
# data['Rent'] = np.log(data['Rent'] + 1)
# data.hist('Rent', figsize=(10, 5))
# plt.title('Number of houses vs Price')
# plt.ylabel('Number of Houses')
# plt.xlabel("Price")

# data.hist('BHK', figsize=(10, 5))
```

In [ ]:
```
# data['BHK'] = np.log(data['BHK'] + 1)
# plt.title('Number of houses vs Price')
# plt.ylabel('Number of Houses')
# plt.xlabel("Price")

# data['Bathroom'] = np.log(data['Bathroom'] + 1)
# data.hist('Bathroom', figsize=(10, 5))
# plt.title('Number of houses vs Price')
# plt.ylabel('Number of Houses')
# plt.xlabel("Price")
```

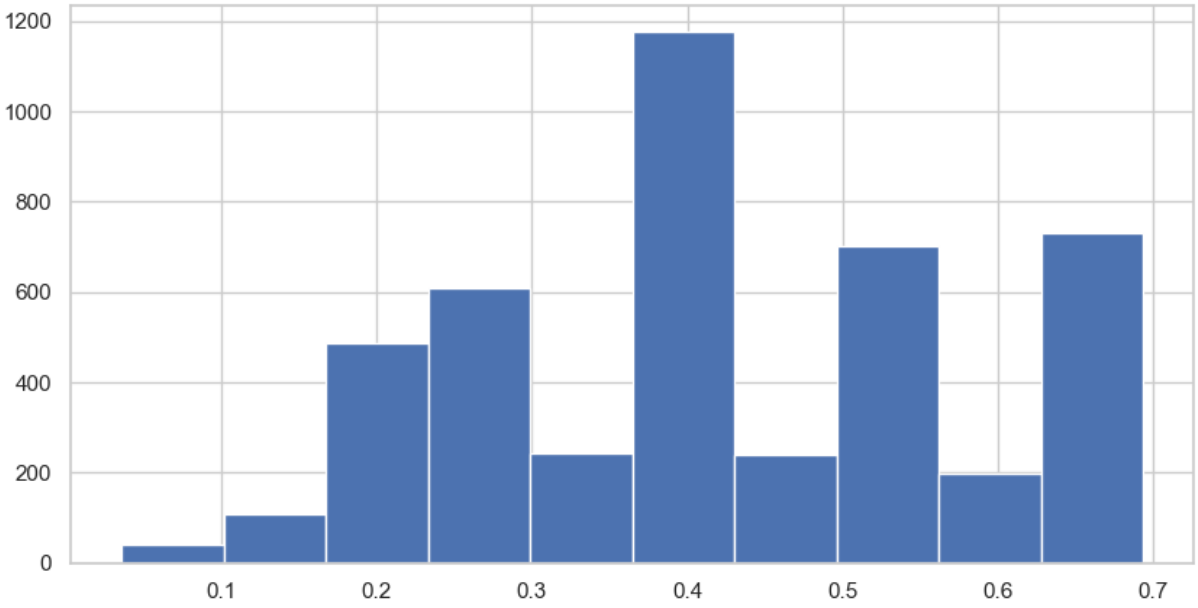
```
# data['Size'] = np.log(data['Size'] + 1)
# plt.title('Number of houses vs Price')
# plt.ylabel('Number of Houses')
# plt.xlabel("Price")

data['Floor'] = np.log(data['Floor'] + 1)
data['Floor'].hist(figsize=(10, 5))
```

```
C:\Users\nigel\AppData\Local\Temp\ipykernel_22436\1318459247.py:17: SettingWithCopyW
arning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  data['Floor'] = np.log(data['Floor'] + 1)
```

Out[ ]: <Axes: >



Fixing some data for modelling

Converting all white spaces to undersdcore (_)

In [ ]:
```
# sorted_date['Area Locality'] = sorted_date['Area Locality'].replace(' ', '_')
# sorted_date.head(20)
```

One hot encoding

In [ ]:
```
def one_hot_encode(data, column):
 encoded = pd.get_dummies(data[column], drop_first= True)
 data = data.drop(column, axis = 1)
 data = data.join(encoded)
 return data
```

Adding values of Area Type and City

In [ ]:
```
data = data[['BHK', 'Bathroom', 'Furnishing Status', 'Rent', 'City', 'Size', 'Tenan
data = one_hot_encode(data, 'City')
data = one_hot_encode(data, 'Point of Contact')
data = one_hot_encode(data, 'Tenant Preferred')
```

In [ ]:
```
from sklearn.preprocessing import OrdinalEncoder
enc = OrdinalEncoder()
data[['Furnishing Status']] = enc.fit_transform(data[['Furnishing Status']])
data
```

Out[ ]:

	BHK	Bathroom	Furnishing Status	Rent	Size	Floor	Chennai	Delhi	Hyderabad	Ko
0	2	2	2.0	10000	1100	0.405465	False	False	False	
1	2	1	1.0	20000	800	0.287682	False	False	False	
2	2	1	1.0	17000	1000	0.287682	False	False	False	
3	2	1	2.0	10000	800	0.405465	False	False	False	
4	2	1	2.0	7500	850	0.405465	False	False	False	
...	...	...	...	...	...	...	...	...	...	
4741	2	2	1.0	15000	1000	0.470004	False	False	True	
4742	3	3	1.0	29000	2000	0.223144	False	False	True	
4743	3	3	1.0	35000	1750	0.470004	False	False	True	
4744	3	2	1.0	45000	1500	0.516691	False	False	True	
4745	2	2	2.0	15000	1000	0.587787	False	False	True	

4534 rows × 15 columns

Clustering Area Locality (Unfinished Attempt)

In [ ]:
```
# from Levenshtein import distance
# from sklearn.cluster import dbscan
```

```
# def lev_metric(x,y):
#     i, j = int(x[0]), int(y[0])     # extract indices
#     return distance(data['Area Locality'].iloc[i], data['Area Locality'].iloc[j])

# x = np.arange(len(data)).reshape(-1, 1)
# f_AL = (dbscan(x, metric=lev_metric, eps=5, min_samples=2) )
```

In [ ]:
```
# for i in range(len(f_AL[1])):
#     # print(f_AL[1][i])
#     data[['Area Locality']].replace(data[['Area Locality']].iloc[i], f_AL[1][i])


# data.head(10)
```

Training and Test Spill

In [ ]:
```
X = data.drop('Rent', axis= 1)
y = data['Rent']
```

In [ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size= 0.2, random_st
```

Attempt to fix the distribution on the training data set

Standardize

In [ ]:
```
# from sklearn.preprocessing import PolynomialFeatures -11k
# X_test = sc.fit_transform(X_test)
# sc= MinMaxScaler()
# sc = RobustScaler()
# from sklearn.preprocessing import MinMaxScaler
# 0.57
# from sklearn.preprocessing import RobustScaler
# 0.57
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Modeling

In [ ]:
```
model = linear_model.LinearRegression()
model.fit(X_train, y_train)
model.coef_
```

Out[ ]:
```
array([ 3179.93921755,  4231.33143619, -1696.50924328,  6839.41157078,
         125.90743057,  -603.67873485,  1347.46666257, -1641.57863413,
        -903.28931482, 10644.93193236,   -47.26380969, -5724.03875343,
         210.06523484,  -499.78493903])
```

Quantitative Evaluation

In [ ]:
```
y_preds = model.predict(X_test)
print(y_preds)
```

```
[ 3.05925621e+04  3.22935510e+04  2.19679558e+04  1.99201651e+04
  8.85400794e+03  4.28434797e+04 -8.54878911e+03  2.73267913e+04
  1.21317481e+04 -8.28340549e+03  1.94162950e+04  5.17613533e+04
  8.62995886e+03  3.74224727e+04 -3.50242692e+02  1.63877166e+04
  3.54375085e+04  1.03547211e+02  1.42803291e+04  1.56586049e+04
  1.85069905e+04  5.47468825e+04  2.34956597e+04  1.16119899e+04
  3.26302817e+04  1.28806903e+04  1.03981932e+04  3.03418207e+04
  1.69033341e+04  3.43747778e+04  2.04264020e+04  3.62788042e+04
  3.44591604e+04  1.11440269e+04  3.02354861e+04  3.20809789e+04
  4.91663086e+02  3.10340451e+04  2.06996496e+04  2.55374692e+04
  4.60475554e+04  7.21100964e+02  1.69790634e+04  4.91794075e+02
  2.11550622e+04  2.88804921e+04  1.54202302e+04  5.59326575e+04
  1.45623532e+04  3.19383365e+04  6.74542635e+04  1.00049275e+04
  1.46748051e+04  2.31918672e+04  2.84264390e+03  3.26239854e+04
  6.31096707e+04  1.25961594e+04  4.24456192e+04  1.22916160e+04
  1.19639185e+04  1.20868448e+04  5.84192109e+04  1.35265911e+04
  5.38740459e+04  1.46272557e+04  2.97804058e+04  2.04581602e+04
  1.65915192e+04  2.02769107e+04  2.28984684e+04  1.94254242e+04
  5.84904364e+04  4.64395432e+03  3.72814639e+03  1.51563211e+04
  5.08074655e+04  1.75621044e+04  1.10352491e+04  1.96625707e+04
  1.36996535e+04  4.10314379e+04  5.47134620e+04  3.86268062e+04
  1.92029520e+04  2.63571497e+04  1.03153807e+04  8.07000402e+03
  4.29053940e+03  1.19408402e+04  3.70901416e+03  1.44015575e+04
  4.88624085e+03  2.85193631e+04  1.99806744e+04  2.27058554e+03
 -2.32140771e+02  1.30653283e+04  3.24411515e+04  5.47844027e+04
  2.06347962e+04  1.73677770e+04 -7.21532544e+00  6.90655720e+04
  9.21645693e+02  3.12142309e+04  4.87091374e+04  2.26117198e+04
  1.69693950e+04  4.34847641e+04  1.49859778e+04  2.06876338e+04
  5.85509643e+04  4.33044703e+04  6.09238683e+03  7.80476973e+03
  5.88485810e+04  2.35668125e+03  3.66295770e+04 -2.96426237e+03
  1.13733191e+04 -3.09598419e+03  3.14837271e+04  4.89850604e+04
  2.22382282e+04  5.18948806e+04  1.01472222e+03  5.57924678e+03
  2.38063786e+04  1.89531795e+04  6.10411042e+03  4.41508250e+04
  5.06099304e+04  5.40374405e+04  1.22241536e+04  1.52918403e+04
  5.00624567e+03  6.52286232e+03  1.12697892e+04  8.28758635e+02
  1.32049001e+04  2.17299101e+04  2.27790685e+03  2.26601177e+04
  2.94344900e+04  8.10627210e+03  3.55208356e+04  2.61761242e+04
  1.60936939e+04  1.91523189e+04  4.31007483e+04 -4.75609408e+03
  2.29657076e+04  6.65147805e+04  3.64521029e+03  2.48389512e+04
  3.75519638e+03  1.79972329e+04  2.46362927e+04  5.12739937e+04
  8.28758635e+02  3.07984042e+04 -4.93281315e+02  4.73367988e+03
  8.07534003e+03  1.39807505e+04  9.58876188e+03  2.22603294e+04
  5.49533615e+04  1.19909000e+04  4.27836122e+04  4.54687786e+04
  4.08396046e+03  3.09662631e+04  6.08774957e+03  5.24540888e+04
 -2.64615862e+03  1.62685473e+03  1.22332153e+04  8.65186142e+03
  1.86212737e+04  2.09031800e+04  4.88415898e+04  1.71310852e+04
  4.43033271e+04  1.73649381e+04  5.58733523e+04  7.45314569e+03
  6.14144907e+03 -2.47483291e+03  5.63816556e+03  1.52394353e+04
  3.75519638e+03  7.28657156e+03  5.73226849e+04  3.83785596e+03
  1.29148728e+04  1.18207212e+04  2.28677986e+03  1.75866400e+04
  1.30567952e+04  1.77145198e+04  3.57220042e+04  1.09517166e+04
  1.21317481e+04  1.27622807e+04  1.05264302e+04  1.40593286e+04
  5.31478180e+04 -5.46022463e+03  5.94940694e+04  1.44842171e+04
  6.26525599e+04  2.92698034e+04  5.24792981e+03  2.32950503e+04
  3.38716992e+04  2.73121843e+04  2.46364656e+03  4.93621324e+04
  8.42616322e+03  5.77560852e+04  1.61971866e+04  1.74601825e+04
```

```
 9.70414143e+03 -3.06395769e+03  1.82116427e+04  2.01669464e+04
 4.59723315e+03  5.59581746e+04  2.70601530e+04  1.62149018e+04
 1.86345703e+04  3.60214350e+03  5.83252345e+04  7.21100964e+02
 5.32930683e+04  5.66160624e+04  4.37083117e+03  2.57927317e+04
 1.68064330e+04  3.48355187e+04  4.48541187e+04  6.09238683e+03
 8.22190555e+03  2.05193932e+04 -5.56344685e+03  2.05537819e+04
 1.22519306e+04  2.52445791e+04  1.70163164e+04  3.25276770e+04
 2.62680724e+04  4.65644628e+04  4.69048579e+03  3.41775617e+04
 1.76002417e+04  2.03776379e+04  9.17459028e+03 -3.62504962e+03
 2.67674544e+04  4.05581526e+04  1.92254251e+04  2.52552081e+04
 6.22403996e+04  3.64548681e+03  5.45619345e+04  1.42600897e+04
 1.58603742e+04  1.69951615e+04  5.27215850e+04  1.10351908e+03
 5.41422013e+04  5.44288621e+04  5.69846717e+04 -1.39244318e+03
 1.94342103e+04  1.69489718e+04  1.52394353e+04  1.20075171e+04
 1.26701903e+04  4.48781548e+04  4.53394420e+04  5.41611931e+04
 1.88045700e+04  1.60032704e+04  2.37899079e+04  1.69839767e+04
 2.49037887e+04  5.54308902e+04  5.49825282e+04  2.66198590e+04
 2.86280897e+04  1.18443690e+02  5.53435501e+04  3.67555133e+04
 4.90085132e+04  7.11371706e+03  1.55181453e+04  2.36711752e+04
 2.87660628e+04  1.10843114e+04  3.00075593e+03  1.28358786e+04
 8.10834591e+03  7.82897191e+03  1.52703314e+04  1.68261879e+04
-2.79252915e+03  3.41378307e+04  1.81780278e+04  3.52409385e+04
 1.88591523e+04  5.45377682e+04  4.54908368e+04  2.98287092e+04
 9.12395720e+03  6.39874530e+04  4.60423351e+04  7.39036735e+04
 3.14168008e+04  1.79528028e+04  5.35480605e+04  9.24452278e+03
 4.87584666e+04  5.17583994e+04  3.67827721e+04  3.15158115e+02
 3.55281887e+04  9.69113689e+03  4.78755030e+04 -2.04349530e+03
 2.82736444e+04  1.74374056e+04  7.01459834e+04  5.31775231e+04
 2.11617430e+03  2.97375714e+04  5.05320467e+04  2.79038298e+04
 1.51488017e+04  9.68484805e+03  1.02595584e+04  2.83024419e+04
 1.61635893e+04  2.92275236e+04  3.29003026e+04  4.01797473e+04
 1.01689327e+04  1.21582307e+04  8.55502005e+03  1.51698105e+04
 1.62615387e+04  3.71763399e+04  4.37955682e+03  2.04059789e+04
 1.52021370e+04  3.46932537e+04  4.53394420e+04  4.20592602e+04
 4.61258106e+04  3.66807272e+04  7.06930273e+03  7.47460866e+03
 4.73076006e+04  5.20272004e+04  1.34032419e+04  2.00392698e+04
-1.29633723e+02  1.81545671e+04  4.99863359e+04  5.84068633e+02
 1.18644785e+04  7.04995998e+04  4.44958097e+03  2.71059529e+04
 1.60719159e+04  5.20895816e+03  9.66067482e+02  6.88161434e+04
 1.12426005e+04  2.99640830e+04  1.80609139e+04  2.69229721e+04
 7.65181310e+04  3.82830356e+04  4.68547976e+03  5.29720476e+03
 6.74274661e+03  1.39126246e+04  9.72612070e+02  6.57851516e+03
 1.50661421e+04  1.80478265e+04 -1.48110905e+03  2.94807970e+04
 2.35771988e+04  2.83482067e+04  4.31320507e+04  7.55980322e+03
 1.65152903e+04  2.65055313e+04  4.55336227e+04  1.26871723e+04
 4.90906659e+04  7.14169959e+04  3.25218310e+04  4.09147988e+04
 2.03204321e+04  1.81892663e+04  1.70951431e+03  3.30972001e+04
 4.18483589e+04  1.59405840e+04  1.16214903e+04  1.12264018e+04
 5.85028394e+04 -1.74722949e+02  5.78074859e+04  1.13033866e+04
 9.75148702e+03  1.20843037e+04  5.65430292e+04  4.03045817e+04
 2.12684388e+04  1.10351908e+03  5.54445323e+04  9.44197095e+03
 3.82728265e+04 -3.01332461e+03  3.01543775e+04  2.84793930e+04
 1.30615768e+04  1.72893609e+04  7.15109012e+04  1.66563567e+04
 2.77325729e+04  3.41378307e+04  1.74601825e+04  9.59614284e+03
 1.00303960e+04  8.64066496e+03  1.24572117e+04  1.76863354e+04
 3.01343195e+04  1.49737366e+04  1.67809891e+04  1.37502866e+04
```

```
 1.68877355e+04  4.36651388e+03  2.16881299e+04  5.35740401e+04
 8.06862213e+03  2.53935700e+04  6.21332426e+03  1.92353017e+04
-1.79939090e+03 -4.23677462e+03  2.57788725e+04  2.39331729e+04
 1.36548072e+04  3.73488968e+04  1.73808595e+04  5.41229996e+04
 6.65978916e+03  4.70685559e+04  4.33987139e+03  1.69560475e+04
 5.05161980e+04  5.10893268e+04  5.29735114e+04  2.91856372e+04
-1.43177030e+03  1.62765143e+04  1.75831131e+04  4.03996223e+04
 1.10843114e+04 -5.17362349e+02  4.71303307e+04  1.62950557e+04
 4.32244225e+04  7.55868274e+03  1.25026655e+04  3.70106869e+03
 9.15584824e+04  6.85189002e+03  5.84855724e+03  1.95315648e+04
 1.04872706e+03  1.16124286e+04  1.16656926e+04  2.33964551e+03
 2.40361047e+04  1.19055953e+04  3.86682476e+04  3.35877324e+04
 1.38112145e+04  4.95681513e+04  2.01418854e+04  3.30481379e+04
 1.15215627e+04  1.76863354e+04  4.03881759e+04  2.97124217e+03
 1.92029520e+04  4.44958097e+03 -2.94018134e+03  1.10071079e+03
 1.27019017e+04  3.60684850e+04  3.34911313e+04  2.01687777e+03
 1.80228453e+04  2.73442656e+04  5.37986434e+03  3.53612807e+04
 2.41216892e+04  2.00061183e+04  5.64900595e+04  4.91117796e+03
 4.83560777e+03  3.41753171e+04  1.94383962e+04  5.34439734e+04
 3.10554591e+04  7.63693735e+04  3.53612807e+04  2.82100364e+04
 3.13038370e+04  8.64066496e+03 -4.06504141e+03  1.84731255e+04
 1.94162950e+04  1.71017550e+04 -4.06170950e+03  3.02457603e+04
 2.55826073e+04  1.02918754e+04  4.25721181e+03  9.64880333e+03
 2.28429155e+04  5.66312788e+04 -2.83825949e+03  4.92351221e+04
 1.94397627e+04  1.65723719e+04  5.14518989e+04  1.22294179e+04
 2.79715184e+04  9.68029970e+03 -6.43409331e+02  1.01888116e+03
 2.90555572e+04  2.49854210e+04  3.88292030e+04  7.03951961e+03
 1.10007036e+04  1.86291752e+04  1.40593286e+04  4.16184844e+04
 3.05106583e+03  1.33593570e+04  5.09463502e+03  2.65263480e+04
 4.38707164e+04  2.74104264e+04  2.46515694e+04  9.79067351e+03
 2.20092522e+03  7.36489908e+03  5.08820460e+04  5.65334028e+04
 2.63750135e+04 -3.91867087e+03  1.24966206e+04 -2.08349586e+03
 5.23569531e+03  2.45208475e+04  1.10843114e+04  2.07185162e+03
 9.42745509e+03  1.20901767e+04  6.83009451e+04  3.18880116e+04
 2.83851015e+04  2.37202374e+04  2.88949046e+04  1.56250075e+04
 1.26683044e+04  3.11642138e+03  2.34270913e+03  2.16816102e+04
 2.47191097e+04  1.87598866e+04  4.81196426e+04  5.76940560e+04
 2.18249172e+04  1.95137208e+03  3.26440881e+03  3.14206381e+03
-9.57354138e+02 -3.69828292e+03  1.53318409e+04 -3.64894416e+03
 1.48950500e+04  3.09440914e+03  1.66836409e+04  2.39027858e+03
 2.12021073e+04  4.10031680e+04  3.64548681e+03  2.69850155e+04
 1.91396425e+04  5.30119198e+04  1.37757446e+04  2.03534993e+04
 2.87173082e+04  1.20965133e+04  2.57824358e+04  5.62928687e+04
 6.15552710e+03  2.48866058e+04  2.08373659e+04  1.35685411e+04
 2.68052387e+04  2.56487923e+04  1.20664560e+04  4.93530309e+03
 3.67292723e+04 -1.86685934e+03  3.20625992e+04  7.36074014e+03
 3.96317741e+04  5.80787252e+04  4.86893682e+03 -3.30892421e+03
 3.22762898e+04  3.03376180e+03  6.20334294e+04  2.26270412e+04
 5.33313845e+04  1.15953441e+04  2.25614019e+04  1.38296142e+04
 5.08090363e+04  4.50978473e+04  1.96256398e+04  1.40571594e+04
 5.53286680e+04  4.56073281e+04  5.60625185e+04  2.99104785e+04
 1.43113785e+04 -4.75609408e+03  5.16215900e+04  1.58005273e+04
 3.12261889e+04  1.89588476e+04  2.07445058e+04  4.67252943e+04
 1.93077769e+04  4.27040933e+04  2.90456334e+04  4.55461307e+04
 5.64834151e+04  1.56250075e+04  2.59193626e+04  1.10369319e+04
 1.72703624e+04  7.92756185e+03  3.53461780e+04  1.00799365e+04
```

```
 3.46202975e+03  7.15615343e+04  3.22749850e+04  1.73855746e+04
 2.18631950e+04  3.85473416e+04  7.70770816e+04  1.42230665e+03
 3.69614190e+04  1.56214562e+04  5.09019844e+04  1.64628854e+04
 4.39827210e+03  1.37993488e+04  1.52391202e+04  4.58182047e+04
 8.41032625e+03  5.96678355e+04  7.79674959e+04  3.53589245e+04
 1.40593286e+04  6.40017951e+04 -1.28007145e+03  1.81031179e+03
 2.06207294e+04  9.68859949e+03 -4.24493145e+03  2.75789919e+03
 1.25252657e+04  7.05426421e+04  3.57150957e+04  3.17570773e+04
 7.70296902e+03  7.05671461e+04  4.35061322e+04  2.91964216e+04
 2.39027858e+03  5.44951477e+04  1.90663559e+04  1.34663189e+04
 8.82654708e+03  4.59244571e+04  5.54673870e+04  1.60336380e+04
 1.44067625e+04  6.21369278e+04  1.44192884e+04  2.14149615e+04
 7.95762554e+03  5.42432606e+04  5.19115700e+04  5.12227716e+04
 8.73661935e+02  4.11407339e+04 -1.52417584e+03  2.25063980e+04
 4.66427157e+04  5.51270421e+04  1.06146768e+04  4.86219795e+04
 2.87166565e+04  2.50865089e+04  3.73944295e+04  1.47615527e+03
 2.77671021e+04  1.09221255e+04  7.39942980e+04  6.75025418e+04
-2.92091906e+03  9.12316230e+04  1.36676270e+04  1.93904192e+04
 2.70881879e+04  2.63169722e+04  2.47789809e+04  3.14208251e+04
 6.07683591e+04  6.55507284e+02  4.30742377e+04 -3.53264407e+03
 2.20377645e+04  5.86011575e+04  1.76955492e+04  1.97713060e+04
 1.74601825e+04  6.27297423e+04  1.51658374e+04  1.57253980e+04
 7.36489908e+03  4.30915886e+03  2.87873832e+04  4.59675063e+04
 1.49671003e+04  5.41953519e+04  1.62680832e+04 -2.67202834e+03
 1.45709873e+04  1.08539996e+04  1.00219436e+04  7.57225043e+03
 1.63103244e+04  2.23548692e+04 -2.36403201e+03  4.96188407e+04
 1.16124286e+04  3.87974934e+03  1.81196446e+04  2.95997744e+04
 2.52695690e+04  1.24498518e+04  5.59770801e+04  4.71787655e+04
 1.91339127e+04  4.81834088e+04  2.17698569e+04  2.47613367e+04
 1.98152626e+04  1.85422958e+04  1.25026655e+04  4.54677363e+03
 3.57841108e+01  2.86358021e+04  6.39786266e+03  5.78545698e+04
 1.93379996e+04  5.15409216e+04  1.75696035e+04  1.28449403e+04
 4.63331776e+04  1.42091109e+04  7.26574895e+04  2.70058858e+04
 4.25021897e+04  1.91766400e+04  1.56721544e+02  1.34042603e+04
 1.56236824e+04  6.26467252e+04  1.80719075e+04  4.11426575e+04
 4.64213645e+03  5.91120246e+04  3.63544756e+04  1.94479125e+04
 6.90149389e+04  2.13486822e+03  5.13721066e+04  2.39027858e+03
 3.33176355e+04  1.80995068e+04  6.13768232e+04  3.13179003e+04
 1.71593551e+04  2.16498520e+04  5.54039291e+04  2.11400489e+04
-1.48281088e+03  3.21129008e+04 -5.36781909e+03  9.56061708e+03
 1.20075171e+04  1.20848664e+04  1.64908063e+04  2.76539005e+03
 3.87516367e+04  6.79420364e+04  5.17680976e+04  6.14363101e+04
-3.48201099e+03  1.86836326e+04  2.16852339e+04  2.16299892e+04
 4.88390067e+03  2.64357563e+04  3.49243627e+04  5.30144961e+04
 2.58650953e+04  1.89784274e+04  3.06553655e+04  5.67764454e+03
 2.97127170e+04  1.42293283e+04  2.38411748e+04  2.31434349e+04
 1.36250303e+04  4.08116731e+03  6.83912729e+03  1.90663559e+04
-7.86447954e+02  4.11562754e+04  2.42314164e+04  2.60258649e+04
 5.75542227e+03  1.20238282e+04  6.20209641e+03  2.41426195e+04
 2.19898802e+04  3.15498703e+04  1.57266589e+04  5.70020892e+04
 4.02532112e+04  4.45879545e+04  6.43806944e+04  1.12568013e+04
-4.14436907e+03  2.16852339e+04  8.96469689e+03  4.84569706e+04
 2.25796808e+04  6.63790050e+04  5.35391453e+04  2.17083166e+04
 1.70256791e+04  4.66810680e+04  8.68508670e+03  1.25923412e+03
 5.04872702e+04  9.06945766e+03  2.96975476e+04  5.52253148e+04
 1.44015575e+04  9.01882458e+03  1.09255743e+04  1.14751198e+04
```

```
 7.06407161e+03  5.30132808e+04  2.13577415e+04  5.15131668e+04
 1.51903731e+04  2.00268095e+03  3.22822395e+04  1.44842171e+04
 3.64341125e+04  2.69427783e+02  4.06893715e+04]
```

The coefficients

In [ ]:
```
print("Coefficients: \n", model.coef_)
```

```
Coefficients:
 [ 3179.93921755  4231.33143619 -1696.50924328  6839.41157078
   125.90743057  -603.67873485  1347.46666257 -1641.57863413
  -903.28931482 10644.93193236   -47.26380969 -5724.03875343
   210.06523484  -499.78493903]
```

The mean squared error

In [ ]:
```
print("Mean squared error: %.2f" % mean_squared_error(y_test, y_preds))
```

```
Mean squared error: 178731484.25
```

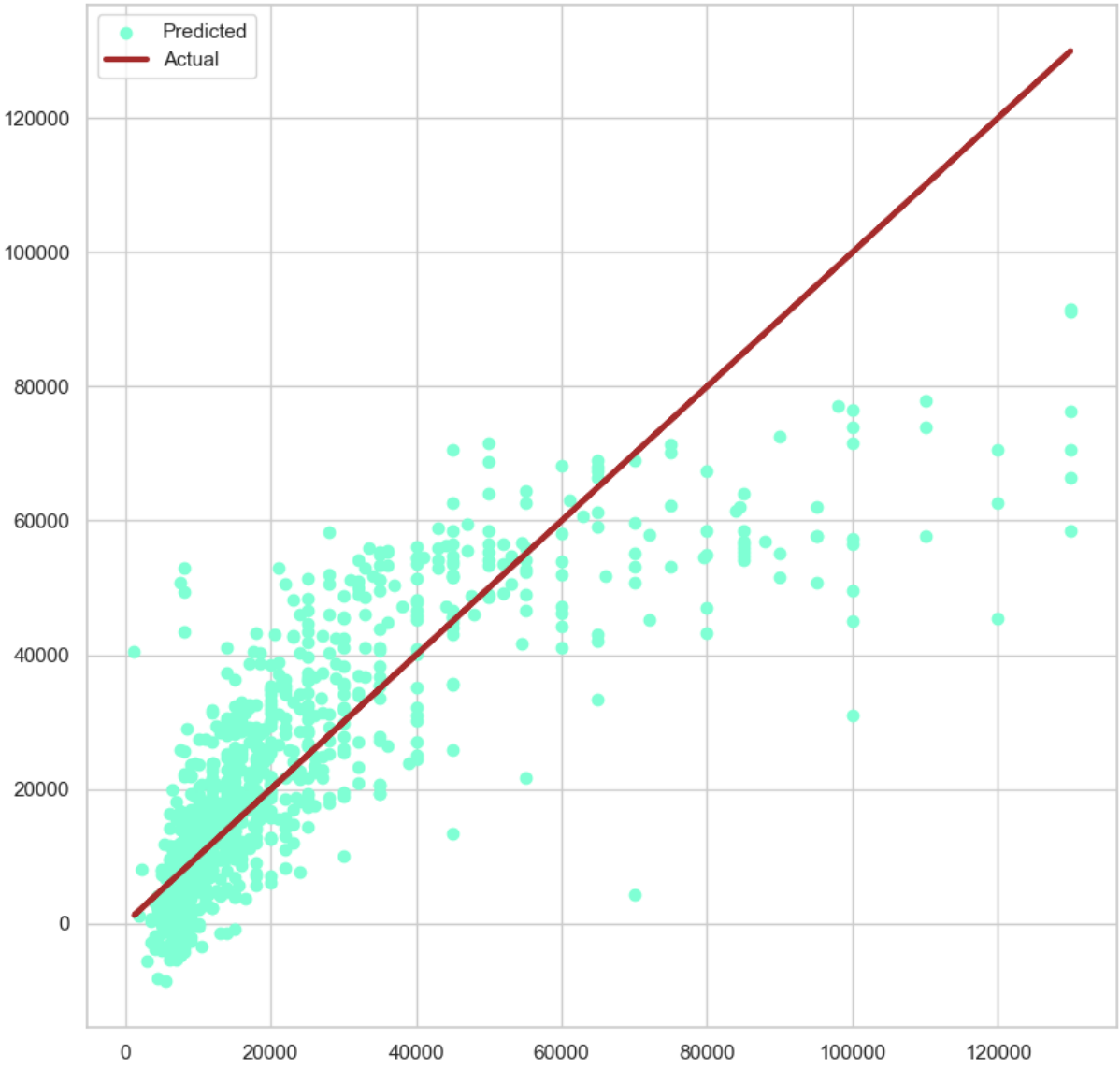
In [ ]:
```
print("Coefficient of determination: %.2f" % r2_score(y_test, y_preds))
```

```
Coefficient of determination: 0.65
```

Graph of the predicted and actual values

In [ ]:
```
plt.figure(figsize=(10, 10))
plt.scatter(y_test, y_preds, color='aquamarine', label='Predicted' )
plt.plot(y_test, y_test, color='brown', linewidth=3, label='Actual')
plt.legend()
```

Out[ ]:
```
<matplotlib.legend.Legend at 0x1e78d0168d0>
```


Predicted
Actual
120000
100000
80000
60000
40000
20000
0
0
20000
40000
60000
80000
100000
120000