house_rent_analysis

August 14, 2022

[1]:
```python
import pandas as pd
import numpy as np
import seaborn as sys
from matplotlib import pyplot as plt
```

[2]:
```python
data = pd.read_csv("train.csv")
```

[3]:
```python
data.head()
```

[3]:
```
      area_type   availability                  location       size  society  \
0  Built-up Area         19-Dec  Electronic City Phase II      2 BHK   Coomee
1      Plot Area  Ready To Move          Chikka Tirupathi  4 Bedroom  Theanmp
2    Carpet Area  Ready To Move               Uttarahalli      3 BHK      NaN
3  Built-up Area  Ready To Move        Lingadheeranahalli      3 BHK  Soiewre
4  Built-up Area  Ready To Move                  Kothanur      2 BHK      NaN

  total_sqft  bath  balcony   price data_category
0       1056   2.0      1.0   39.07         train
1       2600   5.0      3.0  120.00         train
2       1440   2.0      3.0   62.00         train
3       1521   3.0      1.0   95.00         train
4       1200   2.0      1.0   51.00         train
```

[4]:
```python
data.tail()
```

[4]:
```
          area_type   availability                  location       size  \
13269    Carpet Area  Ready To Move                Whitefield  5 Bedroom
13270  Built-up Area  Ready To Move             Richards Town      4 BHK
13271    Carpet Area  Ready To Move  Raja Rajeshwari Nagar      2 BHK
13272  Built-up Area         18-Jun           Padmanabhanagar      4 BHK
13273  Built-up Area  Ready To Move              Doddathoguru      1 BHK

       society total_sqft  bath  balcony  price data_category
13269  ArsiaEx       3453   4.0      0.0  231.0         train
13270      NaN       3600   5.0      NaN  400.0         train
13271  Mahla T       1141   2.0      1.0   60.0         train
13272  SollyCl       4689   4.0      1.0  488.0         train
13273      NaN        550   1.0      1.0   17.0         train
```

```
[5]: print(data.keys())
```

```
Index(['area_type', 'availability', 'location', 'size', 'society',
       'total_sqft', 'bath', 'balcony', 'price', 'data_category'],
      dtype='object')
```

```
[6]: data['area_type'].value_counts().keys()
```

```
[6]: Index(['Built-up Area', 'Carpet Area', 'Plot Area'], dtype='object')
```

```
[7]: data['location'].value_counts().keys()
```

```
[7]: Index(['Whitefield', 'Sarjapur  Road', 'Electronic City', 'Kanakpura Road',
            'Thanisandra', 'Yelahanka', 'Uttarahalli', 'Hebbal', 'Marathahalli',
            'Raja Rajeshwari Nagar',
            …
            'Maruthi Extension', 'Okalipura', 'Old Town', 'Vasantapura main road',
            'Bapuji Layout', '1st Stage Radha Krishna Layout',
            'BEML Layout 5th stage', 'Kannur', 'singapura paradise',
            'Abshot Layout'],
           dtype='object', length=1288)
```

```
[8]: data['size'].value_counts().keys()
```

```
[8]: Index(['2 BHK', '3 BHK', '4 Bedroom', '4 BHK', '3 Bedroom', '1 BHK',
            '2 Bedroom', '5 Bedroom', '6 Bedroom', '1 Bedroom', '7 Bedroom',
            '8 Bedroom', '5 BHK', '9 Bedroom', '6 BHK', '7 BHK', '1 RK',
            '10 Bedroom', '9 BHK', '8 BHK', '11 BHK', '11 Bedroom', '10 BHK',
            '14 BHK', '13 BHK', '12 Bedroom', '27 BHK', '43 Bedroom', '16 BHK',
            '19 BHK', '18 Bedroom'],
           dtype='object')
```

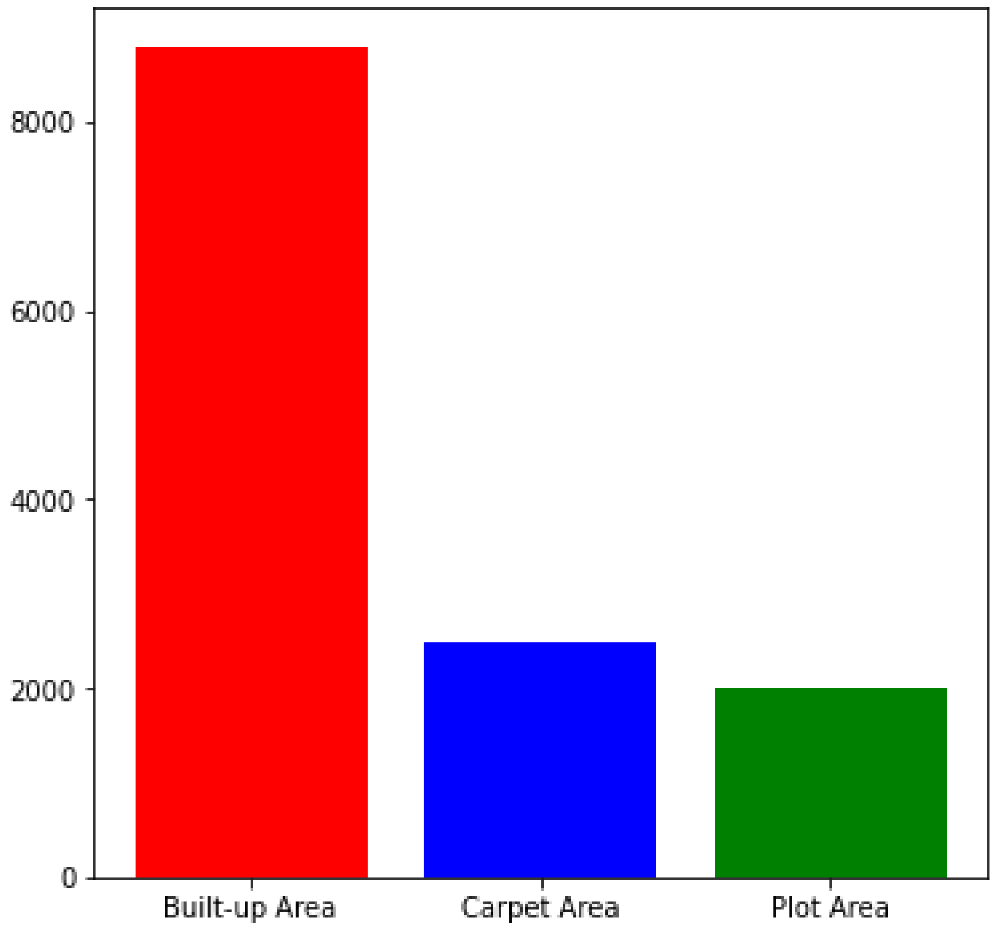
```
[9]: data['bath'].value_counts().keys()
```

```
[9]: Float64Index([ 2.0,  3.0,  4.0,  1.0,  5.0,  6.0,  7.0,  8.0,  9.0, 10.0, 12.0,
                   13.0, 11.0, 16.0, 27.0, 40.0, 15.0, 14.0, 18.0],
                  dtype='float64')
```

```
[10]: data['balcony'].value_counts().keys()
```

```
[10]: Float64Index([2.0, 1.0, 3.0, 0.0], dtype='float64')
```

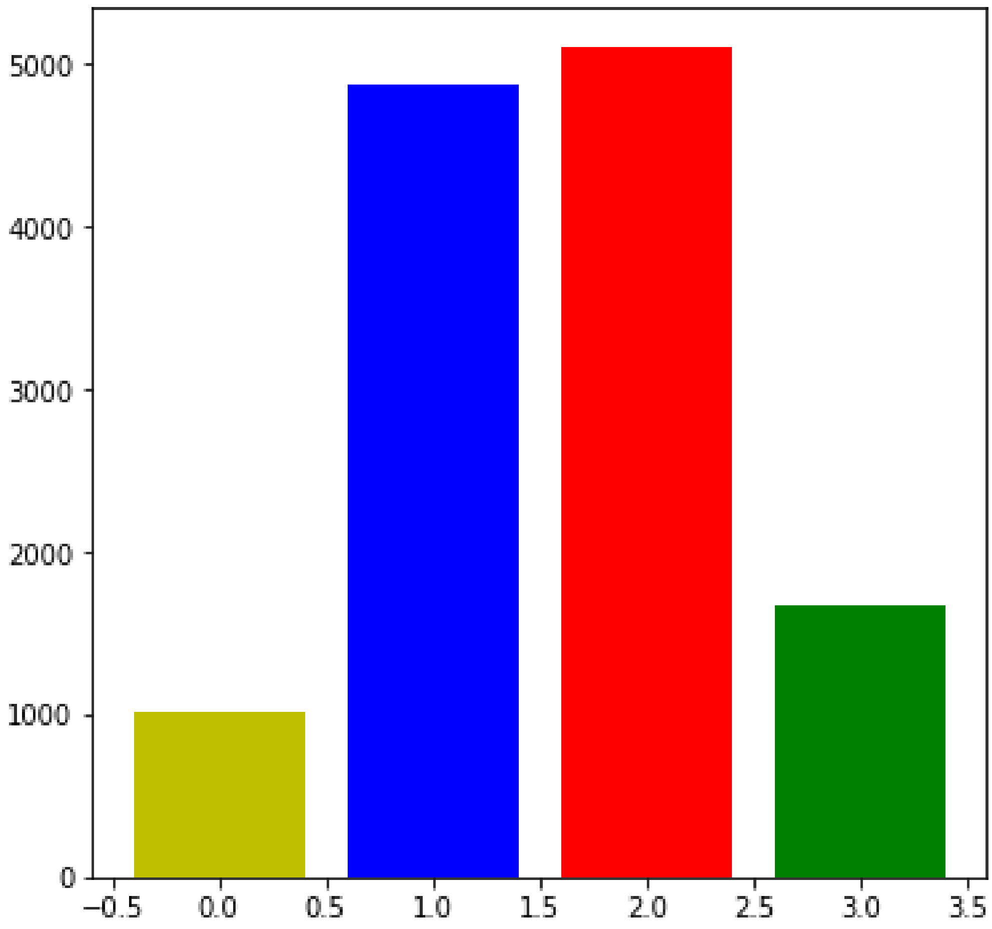
```
[11]: plt.figure(figsize=(6,6))
      plt.bar(list(data['area_type'].value_counts().keys()),list(data['area_type'].
        ↪value_counts()),color=["r","b","g"])
      plt.show()
```



```
[12]: data['area_type'].value_counts()
```

```
[12]: Built-up Area    8779
      Carpet Area     2488
      Plot Area       2007
      Name: area_type, dtype: int64
```

```
[13]: plt.figure(figsize=(6,6))
      plt.bar(list(data['balcony'].value_counts().keys()),list(data['balcony'].
       ↪value_counts()),color=["r","b","g","y"])
      plt.show()
```



[14]:
```
data['balcony'].value_counts()
```

[14]:
```
2.0    5101
1.0    4880
3.0    1669
0.0    1019
Name: balcony, dtype: int64
```

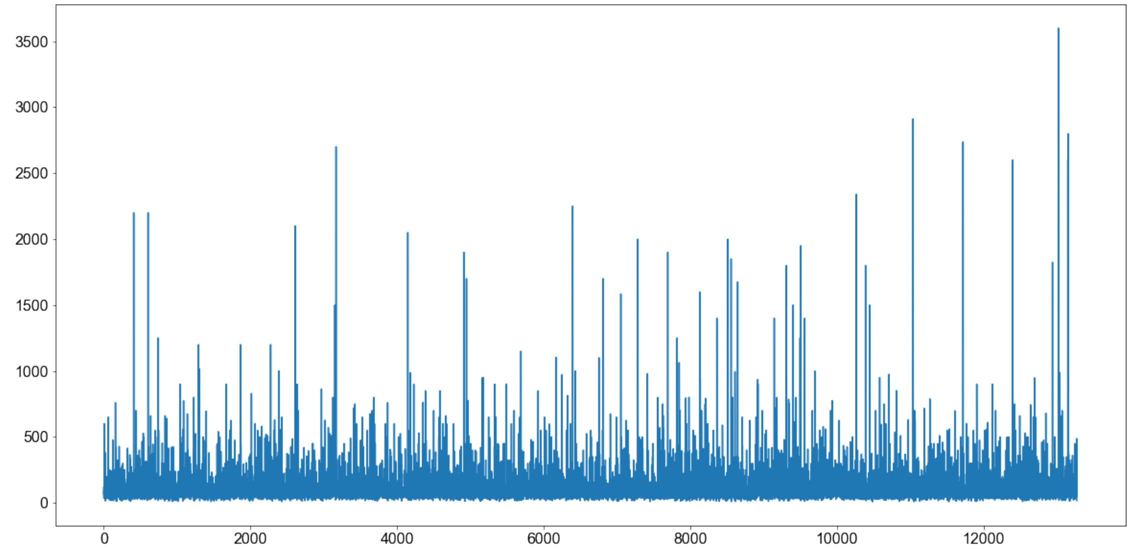
[15]:
```
data.isnull().sum()
```

[15]:
```
area_type          0
availability       0
location           1
size              16
society         5472
total_sqft         0
bath              73
balcony          605
```

```
price                 0
data_category         0
dtype: int64
```

[16]:

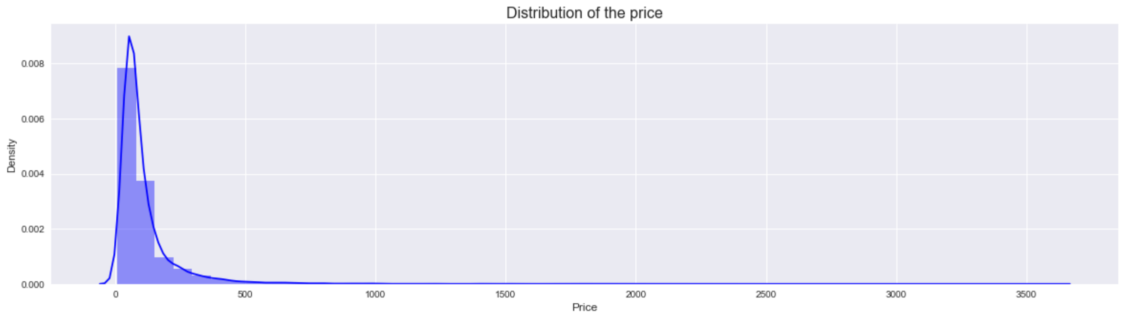
```
data['price'].plot(figsize=(20,10), fontsize = 16)
plt.style.use("seaborn")
plt.show()
```



[17]:

```
plt.figure(figsize=(20,5))
sys.distplot(data['price'],color='blue')
plt.title('Distribution of the price', fontsize=16)
plt.xlabel('Price', fontsize=12)
plt.show()
```

```
C:\Users\Sushan Shivagiri\AppData\Local\Programs\Python\Python310\lib\site-
packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



[18]:
```
data.shape
```

[18]:
```
(13274, 10)
```

[19]:
```
print(data.keys())
```

```
Index(['area_type', 'availability', 'location', 'size', 'society',
       'total_sqft', 'bath', 'balcony', 'price', 'data_category'],
      dtype='object')
```

[20]:
```
data = data.
 ↪drop(['area_type','society','balcony','availability','data_category'], axis␣
 ↪= 'columns')
```

[21]:
```
data.shape
```

[21]:
```
(13274, 5)
```

[22]:
```
data = data.dropna()
```

[23]:
```
data.isnull().sum()
```

[23]:
```
location      0
size          0
total_sqft    0
bath          0
price         0
dtype: int64
```

[24]:
```
data.shape
```

[24]:
```
(13200, 5)
```

0.0.1 Feature Engineering

```
[25]: data['size'].unique()
```

```
[25]: array(['2 BHK', '4 Bedroom', '3 BHK', '4 BHK', '6 Bedroom', '3 Bedroom',
             '1 BHK', '1 RK', '1 Bedroom', '8 Bedroom', '2 Bedroom',
             '7 Bedroom', '5 BHK', '7 BHK', '6 BHK', '5 Bedroom', '11 BHK',
             '9 BHK', '9 Bedroom', '27 BHK', '10 Bedroom', '11 Bedroom',
             '10 BHK', '19 BHK', '16 BHK', '43 Bedroom', '14 BHK', '8 BHK',
             '12 Bedroom', '13 BHK', '18 Bedroom'], dtype=object)
```

```
[26]: data['BHK'] = data['size'].apply(lambda x: int(x.split(" ")[0]))
```

```
[27]: data.head(2)
```

```
[27]:                    location       size total_sqft  bath   price  BHK
      0  Electronic City Phase II      2 BHK       1056   2.0   39.07    2
      1          Chikka Tirupathi  4 Bedroom       2600   5.0  120.00    4
```

0.0.2 Exploring total_sqft feature

```
[28]: def is_float(x):
          try:
              float(x)
          except:
              return False
          return True
```

```
[29]: data[~data['total_sqft'].apply(is_float)].head(10)
```

```
[29]:                location       size   total_sqft  bath    price  BHK
      30            Yelahanka      4 BHK  2100 - 2850   4.0  186.000    4
      122              Hebbal      4 BHK  3067 - 8156   4.0  477.000    4
      137    8th Phase JP Nagar    2 BHK  1042 - 1105   2.0   54.005    2
      165            Sarjapur      2 BHK  1145 - 1340   2.0   43.490    2
      188            KR Puram      2 BHK  1015 - 1540   2.0   56.800    2
      548         Hennur Road      2 BHK  1195 - 1440   2.0   63.770    2
      659           Yelahanka      2 BHK  1120 - 1145   2.0   48.130    2
      670        Bettahalsoor  4 Bedroom  3090 - 5002   4.0  445.000    4
      770  Banashankari Stage VI   2 BHK  1160 - 1195   2.0   59.935    2
      847     Bannerghatta Road    2 BHK  1115 - 1130   2.0   58.935    2
```

```
[30]: def convert_sqft_to_number(x):
          tokens = x.split("-")
          if len(tokens) == 2:
              return (float(tokens[0])+float(tokens[1]))/2
          try:
              return float(x)
```

```
    except:
        return None
```

[31]:
```
data = data.copy()
data["total_sqft"] = data["total_sqft"].apply(convert_sqft_to_number)
data.head(10)
```

[31]:
```
                   location       size  total_sqft  bath   price  BHK
0  Electronic City Phase II      2 BHK      1056.0   2.0   39.07    2
1          Chikka Tirupathi  4 Bedroom      2600.0   5.0  120.00    4
2               Uttarahalli      3 BHK      1440.0   2.0   62.00    3
3        Lingadheeranahalli      3 BHK      1521.0   3.0   95.00    3
4                  Kothanur      2 BHK      1200.0   2.0   51.00    2
5                Whitefield      2 BHK      1170.0   2.0   38.00    2
6          Old Airport Road      4 BHK      2732.0   4.0  204.00    4
7              Rajaji Nagar      4 BHK      3300.0   4.0  600.00    4
8              Marathahalli      3 BHK      1310.0   3.0   63.25    3
9              Gandhi Bazar  6 Bedroom      1020.0   6.0  370.00    6
```

[32]:
```
from sklearn.preprocessing import LabelEncoder
```

[33]:
```
lb = LabelEncoder()
```

[34]:
```
data['location'] = lb.fit_transform(data['location'])
```

[35]:
```
data
```

[35]:
```
       location       size  total_sqft  bath   price  BHK
0           402      2 BHK      1056.0   2.0   39.07    2
1           300  4 Bedroom      2600.0   5.0  120.00    4
2          1160      3 BHK      1440.0   2.0   62.00    3
3           739      3 BHK      1521.0   3.0   95.00    3
4           698      2 BHK      1200.0   2.0   51.00    2
…          …          …           …     …        …    …
13269      1233  5 Bedroom      3453.0   4.0  231.00    5
13270       985      4 BHK      3600.0   5.0  400.00    4
13271       953      2 BHK      1141.0   2.0   60.00    2
13272       888      4 BHK      4689.0   4.0  488.00    4
13273       379      1 BHK       550.0   1.0   17.00    1

[13200 rows x 6 columns]
```

[36]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 13200 entries, 0 to 13273
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
```

```
---  ------      --------------  -----
 0   location    13200 non-null  int32
 1   size        13200 non-null  object
 2   total_sqft  13200 non-null  float64
 3   bath        13200 non-null  float64
 4   price       13200 non-null  float64
 5   BHK         13200 non-null  int64
dtypes: float64(3), int32(1), int64(1), object(1)
memory usage: 670.3+ KB
```

[37]:
```
data.drop(['size'], axis=1, inplace = True)
```

[38]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 13200 entries, 0 to 13273
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   location    13200 non-null  int32
 1   total_sqft  13200 non-null  float64
 2   bath        13200 non-null  float64
 3   price       13200 non-null  float64
 4   BHK         13200 non-null  int64
dtypes: float64(3), int32(1), int64(1)
memory usage: 567.2 KB
```

[39]:
```
data
```

[39]:
```
       location  total_sqft  bath   price  BHK
0           402      1056.0   2.0   39.07    2
1           300      2600.0   5.0  120.00    4
2          1160      1440.0   2.0   62.00    3
3           739      1521.0   3.0   95.00    3
4           698      1200.0   2.0   51.00    2
…            …           …    …     …   …
13269      1233      3453.0   4.0  231.00    5
13270       985      3600.0   5.0  400.00    4
13271       953      1141.0   2.0   60.00    2
13272       888      4689.0   4.0  488.00    4
13273       379       550.0   1.0   17.00    1

[13200 rows x 5 columns]
```

[40]:
```
data['location'] = data['location'].astype('category')
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 13200 entries, 0 to 13273
```

```
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   location    13200 non-null  category
 1   total_sqft  13200 non-null  float64
 2   bath        13200 non-null  float64
 3   price       13200 non-null  float64
 4   BHK         13200 non-null  int64
dtypes: category(1), float64(3), int64(1)
memory usage: 583.8 KB
```

[41]:

```
data.shape
```

[41]:

```
(13200, 5)
```

[42]:

```
y = data['price']
x = data.drop(['price'], axis=1)
```

[43]:

```
x.shape, y.shape
```

[43]:

```
((13200, 4), (13200,))
```

0.1 Model Building

[44]:

```
from sklearn.model_selection import train_test_split
from sklearn import linear_model
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import RandomForestRegressor
```

[45]:

```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.30,␣
 ↪random_state=40)
```

[46]:

```
regressor = linear_model.LinearRegression()
```

[47]:

```
regressor.fit(X_train, y_train)
```

[47]:

```
LinearRegression()
```

[48]:

```
regressor.score(X_test, y_test)
```

[48]:

```
0.4376286651700696
```

[49]:

```
reg = GradientBoostingRegressor(random_state=0)
```

[50]:

```
reg.fit(X_train, y_train)
```

[50]:

```
GradientBoostingRegressor(random_state=0)
```

```
[51]: reg.score(X_test, y_test)
```

```
[51]: 0.554384572529143
```

```
[52]: regre = RandomForestRegressor(max_depth=10, random_state=0)
```

```
[53]: regre.fit(X_train, y_train)
```

```
[53]: RandomForestRegressor(max_depth=10, random_state=0)
```

```
[54]: regre.score(X_test, y_test)*100
```

```
[54]: 59.233232801733735
```

0.1.1 Random Forest Regressor is consider with the accuracy 59.94