PROJECT SAUDDY
DATA MINING ND INSIGHT DISCOVERY

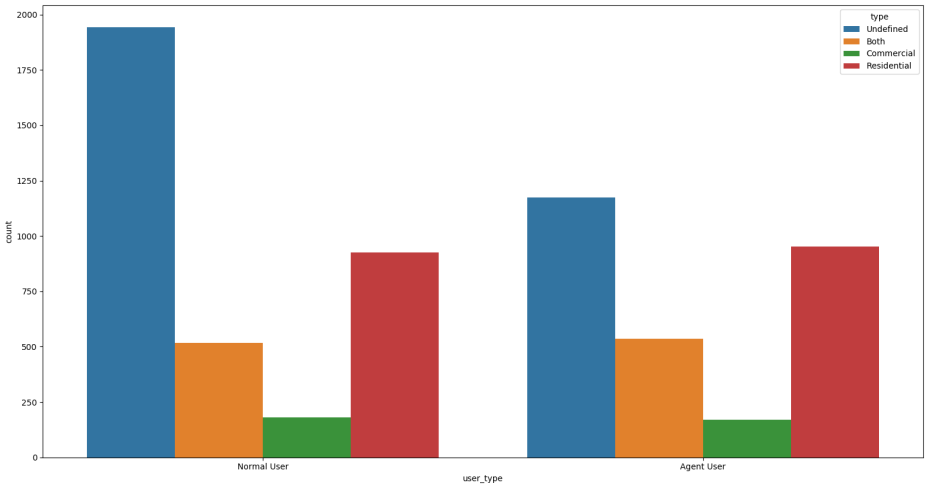
INDONESIAN HOUSING PRICES

By :
E. KENNETH

For :
SAUDDY

TITLE
MACHINE
LEARNER/DATA
SCIENTIST

March 18, 2019



Contents

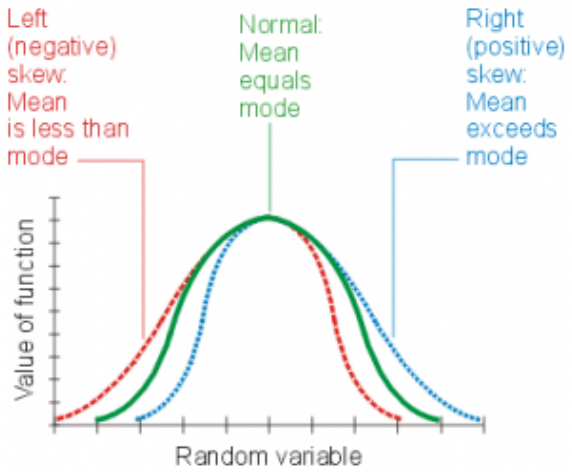
List of Figures

List of Tables

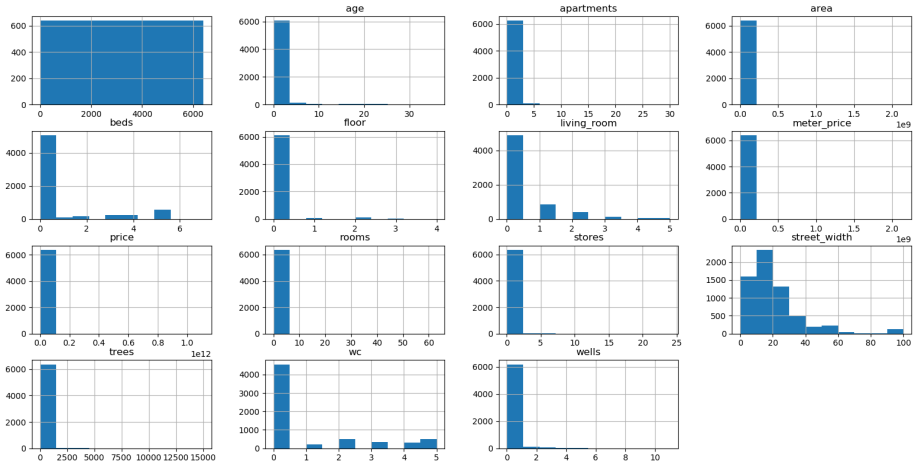
1 Introduction

For this project i have taken the standard machine learning approach to solving every analysis problem. I began by taking a statistical measure of the data i am working, then followed by preprocessing of the data in an understandable form. I will begin by expplaining the two basic terms to introduce us to the type of data we are working on.

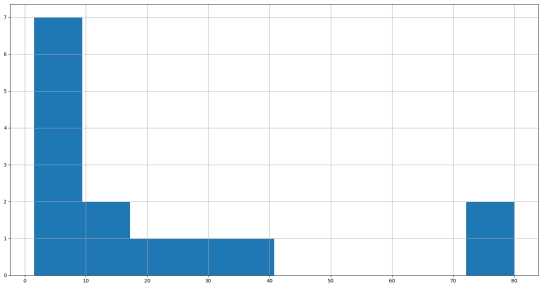
- Skew: Skewness is asymmetry in a statistical distribution, in which the curve appears distorted or skewed either to the left or to the right. As explained in the graph below.



In our case, the data is positive skewed as the mean and median exceeds the mode price of the houses. In essence, the average price of houses exceeds the highest price of any particular house. This can be seen in the histogram below.



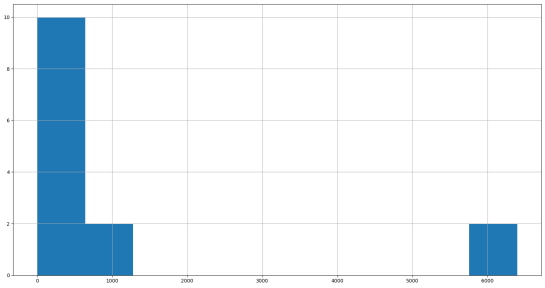
And more clearly here.



Skewness plot(positively skew)

This is merely to show that prices of the houses are not evenly distributed. and that the highest prices are concentrated within a certain range. This will be revealed much later during exploration.
Meter price, area and tree have very make the dataset positively skewed and this could be for a reason. In our case, it means these variables are responsible for high prices of houses(although not in every case.) We will find out more about this in later stage.

- kurtosis: the sharpness of the peak of a frequency-distribution curve. Just as explained above for prices we would also observe that there are area where prices are very high and areas where it is very low.



Kurtosis plot

High kurtosis in a data set is an indicator that data has heavy tails or outliers. If there is a high kurtosis, then, we need to investigate why do we have so many outliers. It indicates a lot of things, maybe wrong data entry or other things. Investigate in next section!

2 Exploratory analysis

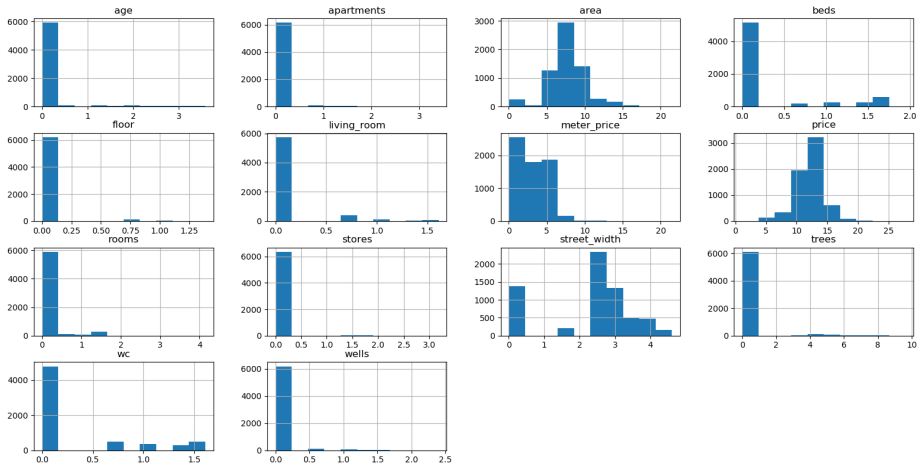
This section deals with both preprocessing of the data and exploration to understand the trends and possible existence of outliere. We will also talk a little about the outliers and how to deal with them.
We begin by wrting a function to return either a standardized data, scaled log data or normalized data as the case may be.
you can see the notebook here: Notebook

```
def standardize_houseprize(df, standardize = None,
                           logg = None, normalize = None):
  df = df.copy(deep = True)
  #drop all objects
  #and leaving all float64 and int64 datatypes
  for ii in hosue_df.columns:
```

```
  if hosue_df[ii].dtype == object:
    df = df.drop(ii, axis = 1)

'''
#standardize values
      x - mean of x
z = --------------------
        sd of x

#log values

z = log(x)

#normalize values

        x - min(x)
z = --------------------
        max(x) - min(x)
'''

#standard deviation
def stdev(df):
  return np.std(df, axis = 0)
#mean deviation
def mean_dev(df):
  return df - np.mean(df, axis = 0)
#log of data
def logg_dat(df):
  return np.log(df)

#standardized values for columns
if standardize:
  for ii, ij in enumerate(df.columns):
    print(ii, ij)
    df['{}'.format(ij)] = mean_dev(df.loc[:, '{}'.format(ij)])/\
                          stdev(df.loc[:, '{}'.format(ij)])
elif logg:
  df = logg_dat(df)
```

```
    df = df.replace([np.inf, -np.inf, np.nan], 0)
  elif normalize:
    for ii, ij in enumerate(df.columns):
      df['{}'.format(ij)] = (df.loc[:, '{}'.format(ij)] - \
                             min(df.loc[:, '{}'.format(ij)]))/\
      (max(df.loc[:, '{}'.format(ij)]) - min(df.loc[:, '{}'.format(ij)]))
  else:
    pass

  return df

df = standardize_houseprize(hosue_df)
df_standard = standardize_houseprize(hosue_df, standardize = True)
log_data = standardize_houseprize(hosue_df, logg=True)
df_normal = standardize_houseprize(hosue_df, normalize = True)
```

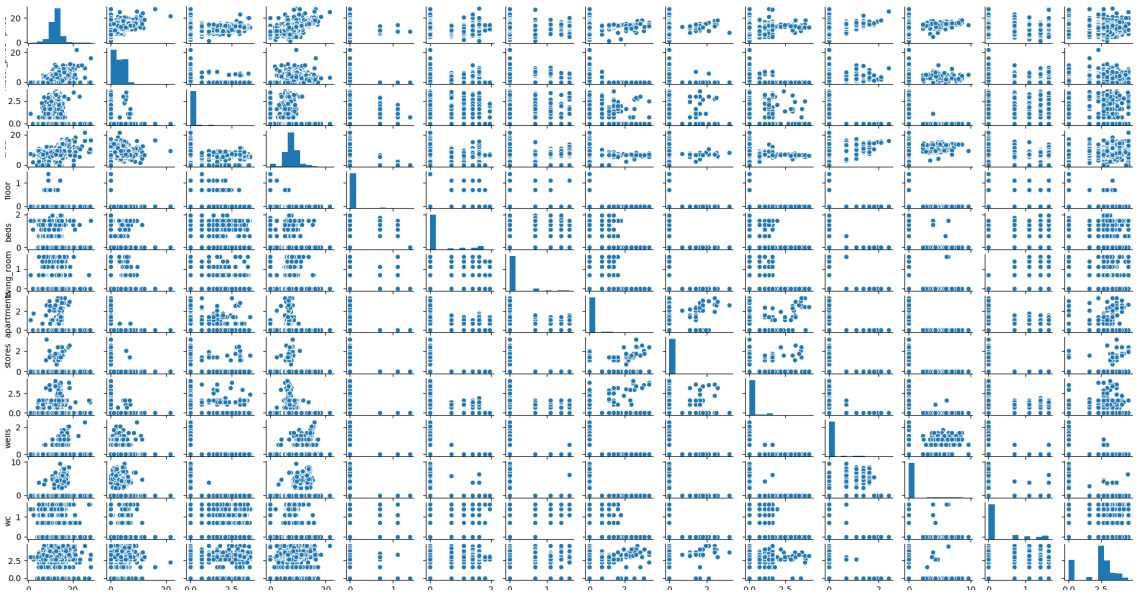
The Essence of standardizing the data is to remove the unbalance effect in the dataset. And to ensure it is scalled within a certain range. the formula for this is found inside the function above. In most cases i prefer the log_data since it is easy to visualize.



Plot of log data

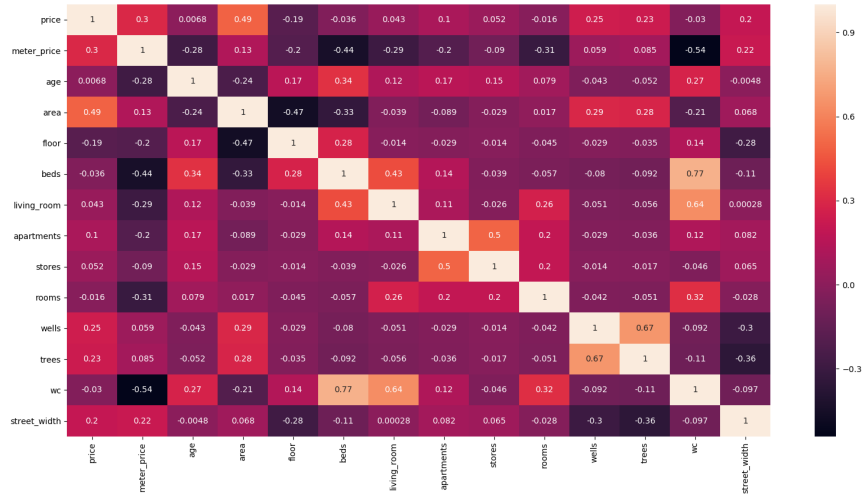
The following code shows the distribution of the pairplot as they are correlated to eachother.

```
sns.pairplot(log_data)
```



Pairplot of log data

However, this is not enough to see the corrlation between numerical features. Hence, we need a heatmap for this reason. The essence of the heat is to plot using a range of color intensity to show the weak or stron correlation between feature.



Heatmap of log data

Here are the conclusions from the heatmap.

* Note that this is ofcourse without the CATEGORICAL VARIABLE whic will be introduced later during analysis phase

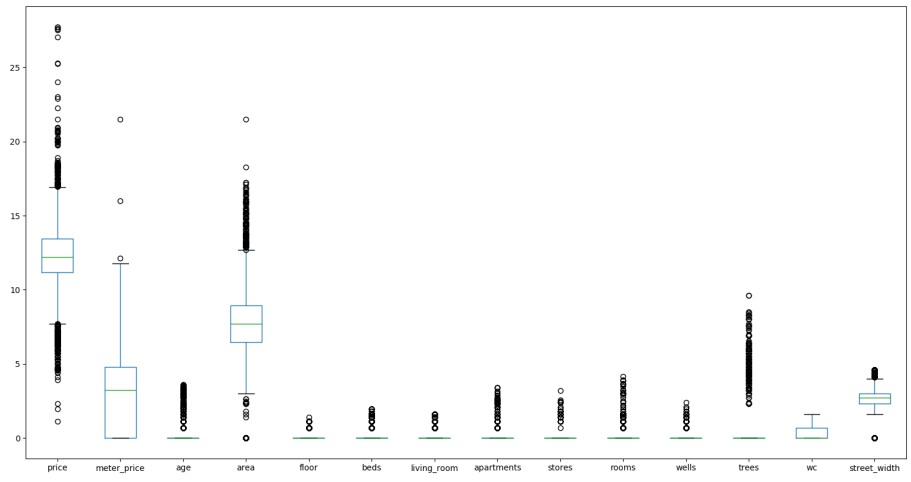
1. PRICE has a close but not so high correlation with AREA (as price increases, area slightly increases)
2. PRICE also has some correlation with METER PRICE (as price increases, meter price slightly increases)
3. PRICE also has some correlation with WELLS (same here: increase in number of wells is a slight increase in well)
4. PRICE also has some correlation with TREES (same)
5. PRICE has a much lower correlation with APPARTMENT (same)
6. PRICE has a close but not so high correlation with STREET WIDTH (same)

7. FLOOR has some correlation with BEDS (number of floors also determines the numbers of beds)

8. APPARTMENT has some correlation with ROOMS (the higher the numbers of appartments, the higher the room number)

9. Floor has some correlation with STORES (same case here. increase in one causes an increases in the other)

Since we know the features with high correlation, we know what is responsible for high prices and co. What we would like to do next is check for the presence of outliers.
We introduce a **boxplot** to see how much of these outliers are present in the data.



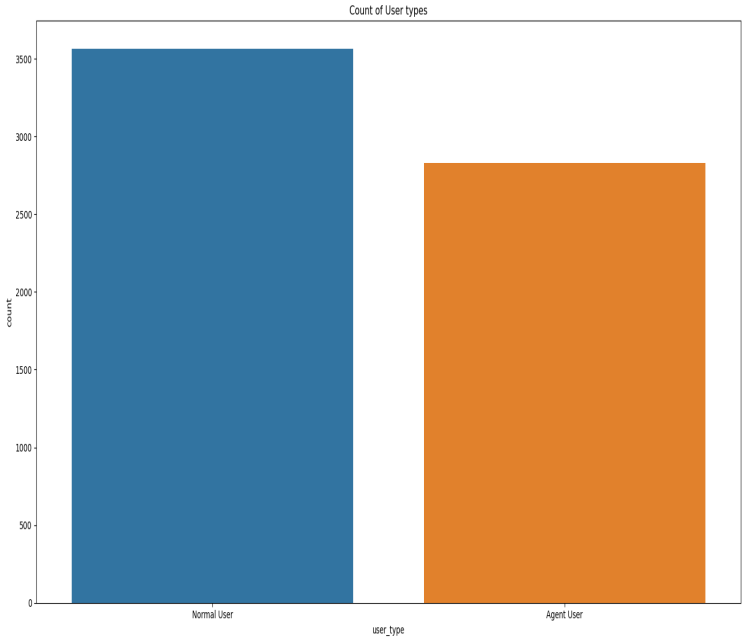
Boxplot of log data

Above you will find at the tail of the box above the extent of the outliers present in the data. To remove this data, we would be sacrificing alot of useful information also and this would affect our model in further section as you would find out.

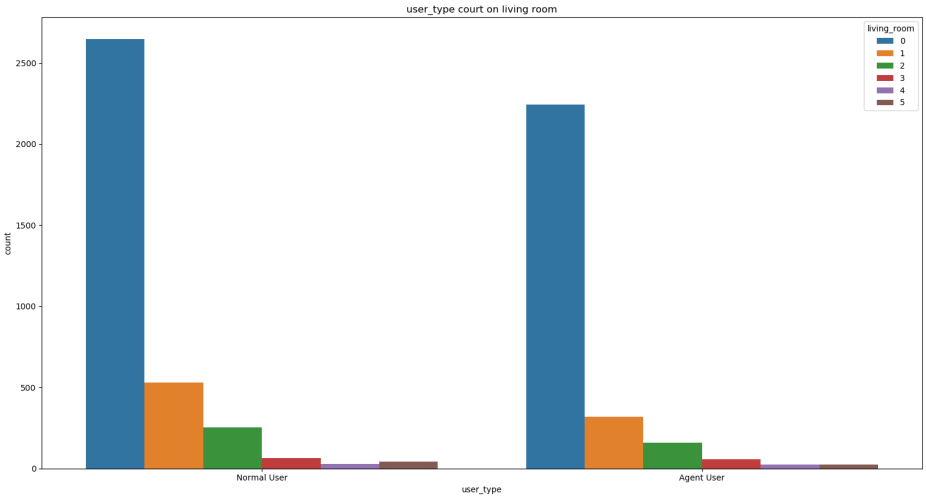
Further exploration before removing outliers

This section is meant to extract what we think may be underlying trends in the dataset

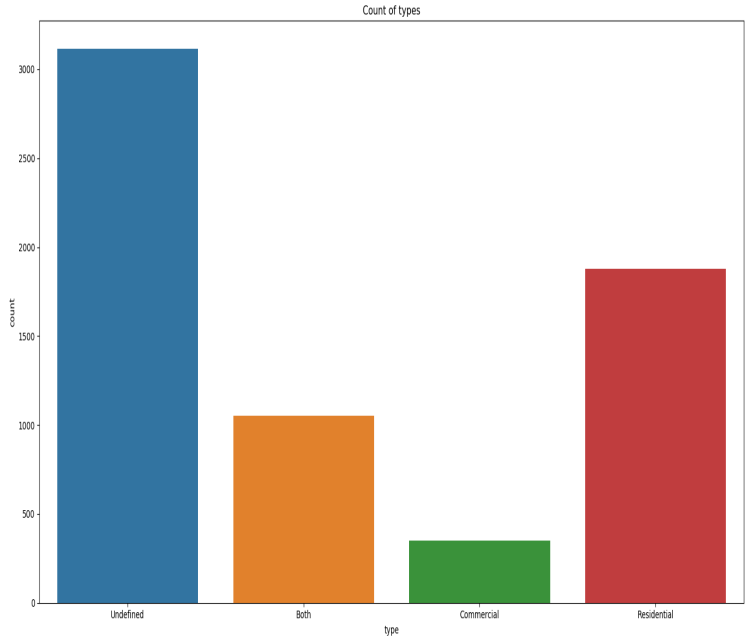
1. **Grouping**: grouping by **Type, user_type and rent_period**



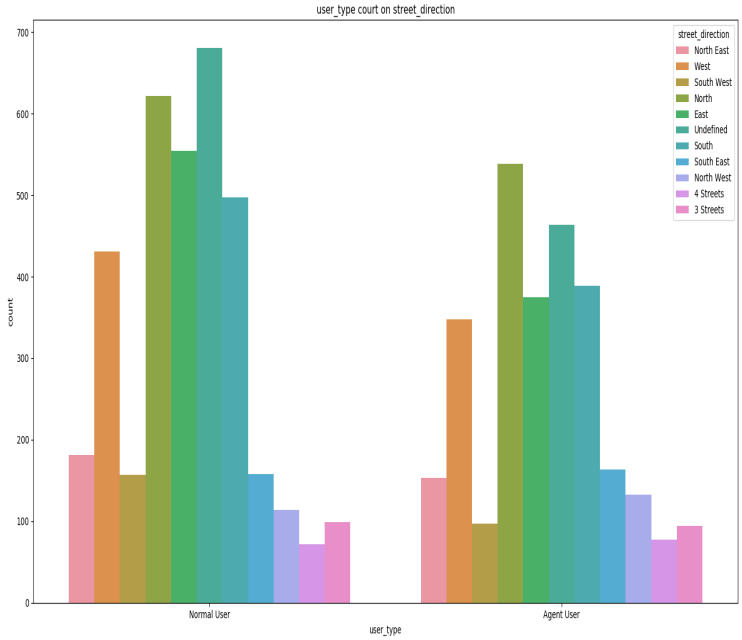
- Here you will find out Normal Users are more than Agent users.i.e people tend to buy houses themselves than use agents(this is probably because agents fee are high.)



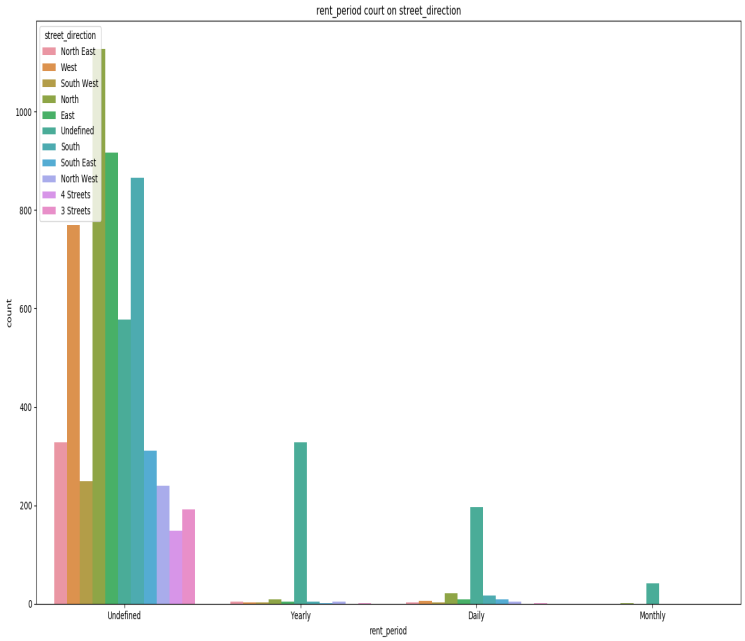
- Here we have more sale of houses with zero living room than any other



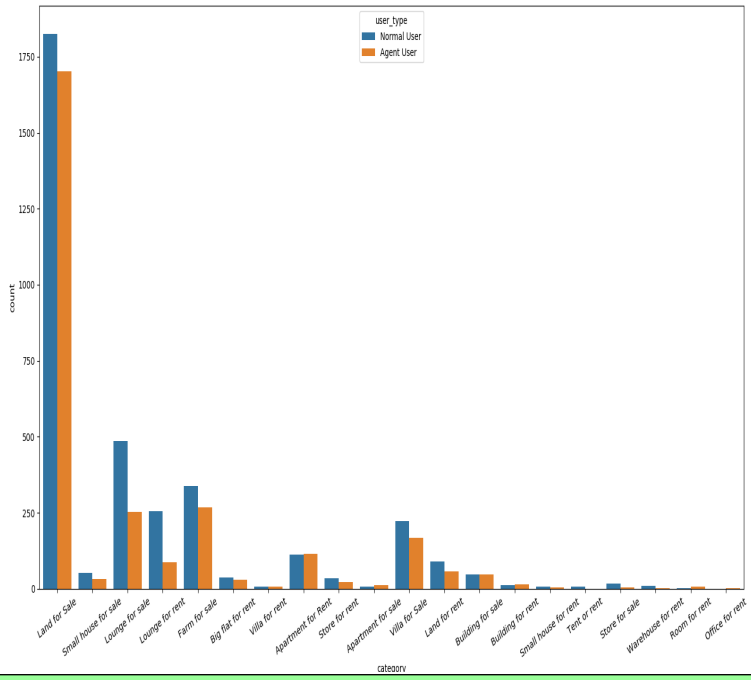
- Here there is alot of undefined types, followed by Resident houses in the dataset. Clients buy more residential houses than commercial places.



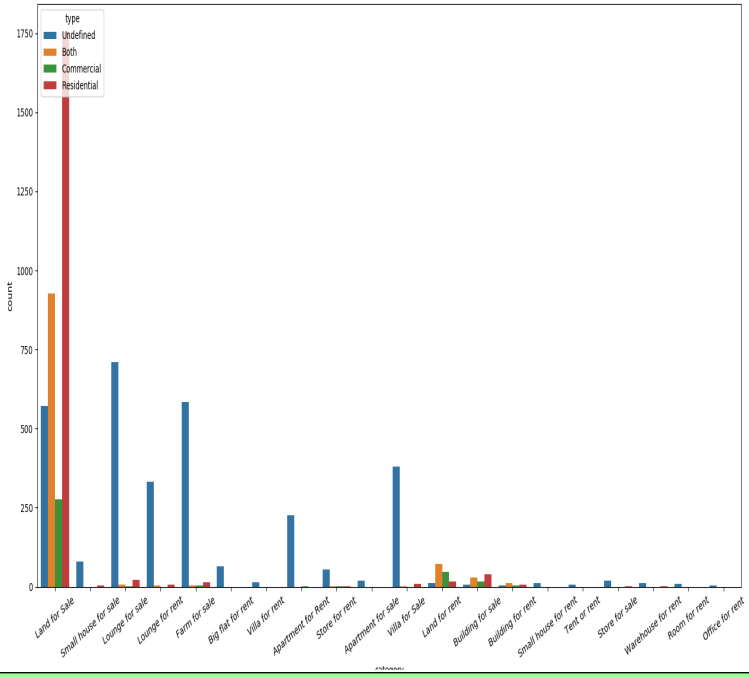
- Clients or buyers tend to buy houses on the North side and West side more than any other street side.



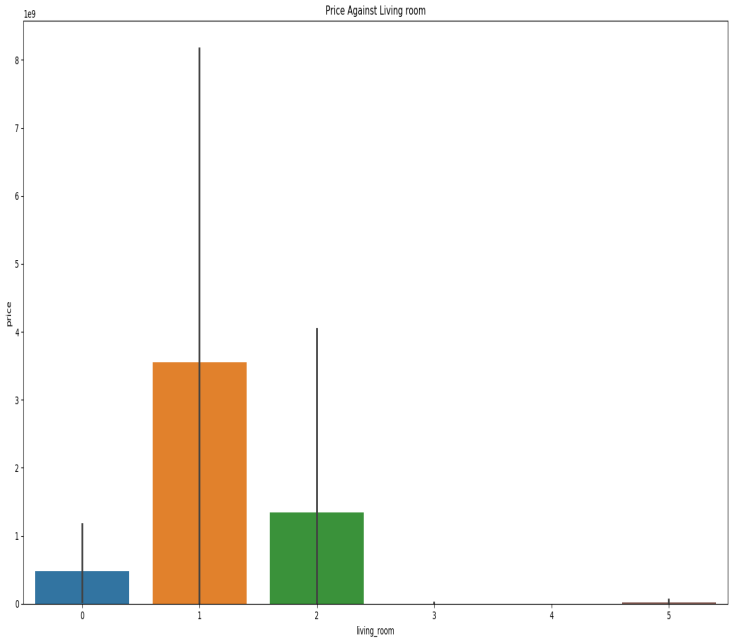
- How rent differ from those who live on a particular street. You would observe majority of the data fall under undefined(meaning we dont know if they prefer to pay yearly, daily, week, etc.)



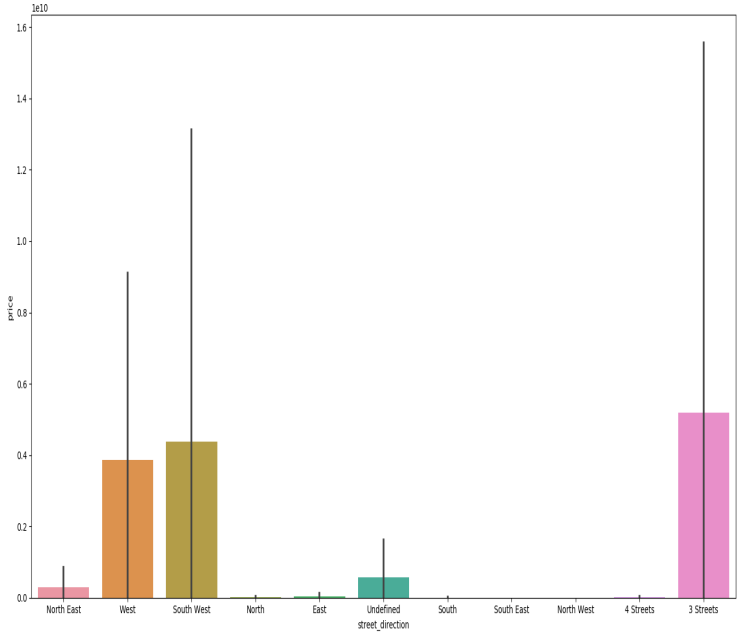
- Here we try to visualize the count of user_type using categories. people tend to go for **Land for sale**, followed next to those who buy **Lounge for sale** and then **Villa for sale**



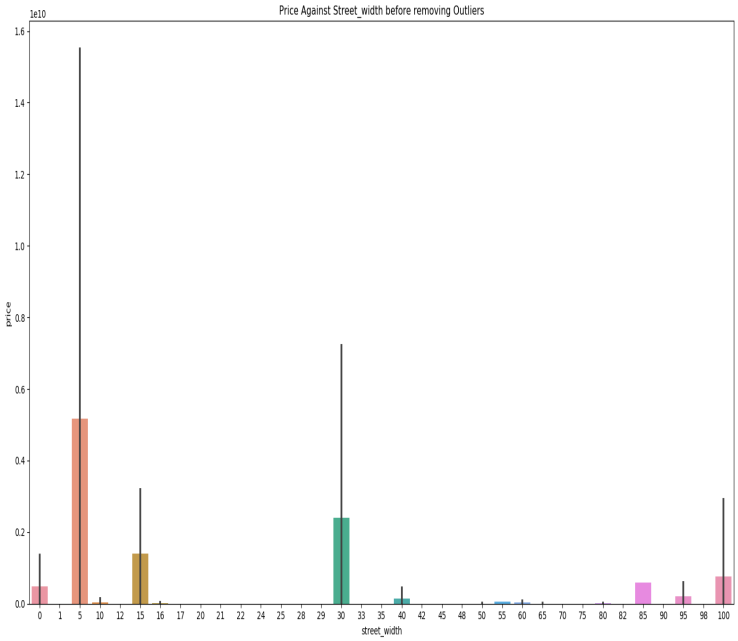
- Here we try to visualize the count of type using categories. Earlier, we noticed more people buying residential houses, an obvious phenomenon here also.



- Price against living room. **Prices** are much higher for houses with 1 living room

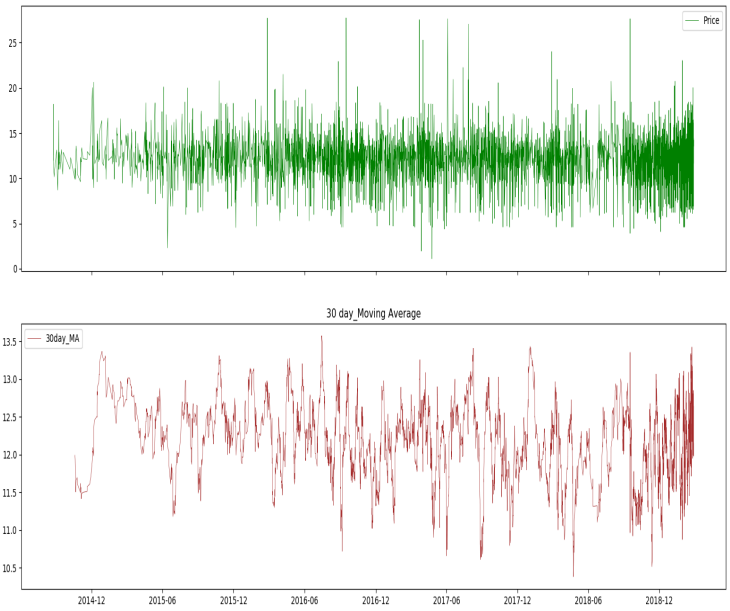


- Prices are highest for houses on **3street**, then **South-west and West**. This could be because of the presence of either Villa or Lounge.

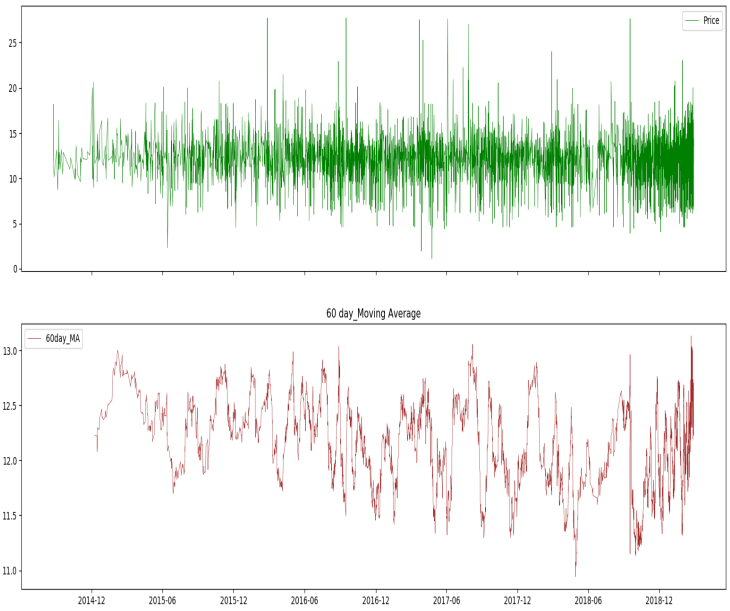


- Prices are highest places with street width of 3, must be residential houses. Places where street width is 30 also have high price, could be commercial areas.

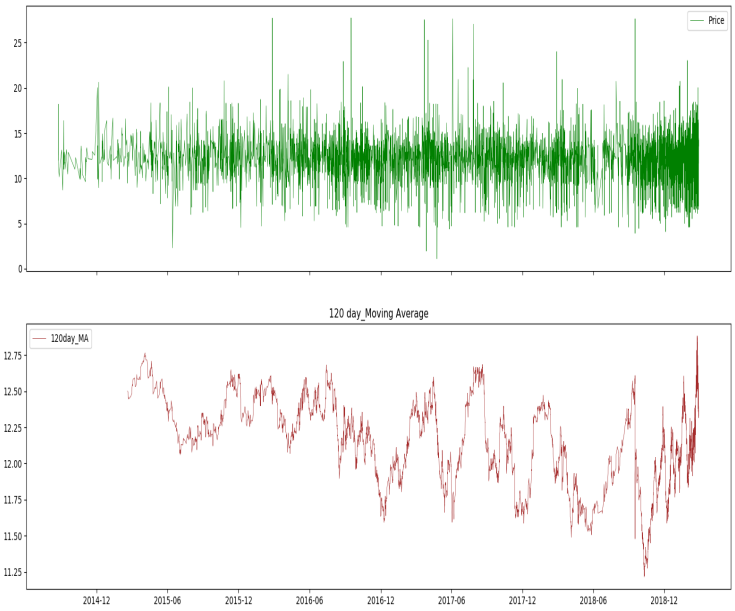
2. **Moving Average**: The trend of the price overtime is explored using moving averages. Here we have conducted both monthly, annual and biannual analysis of the price trend over the period of 5years.



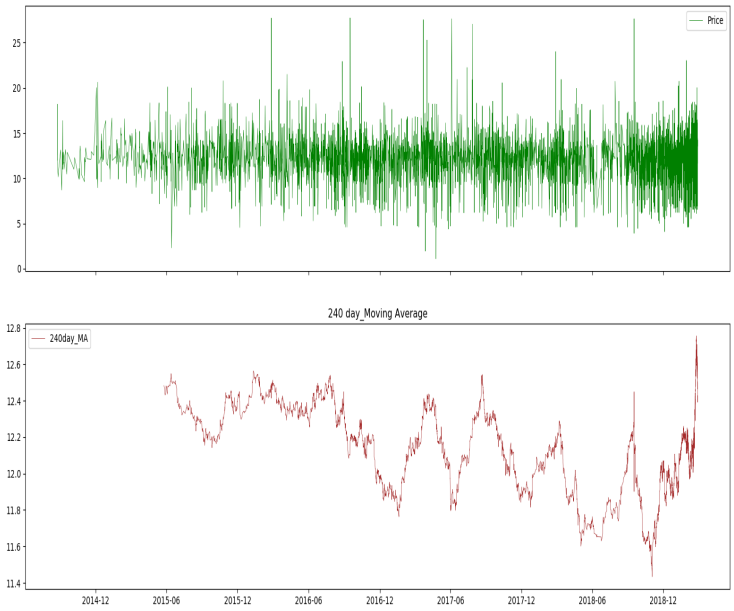
- 30 days moving average



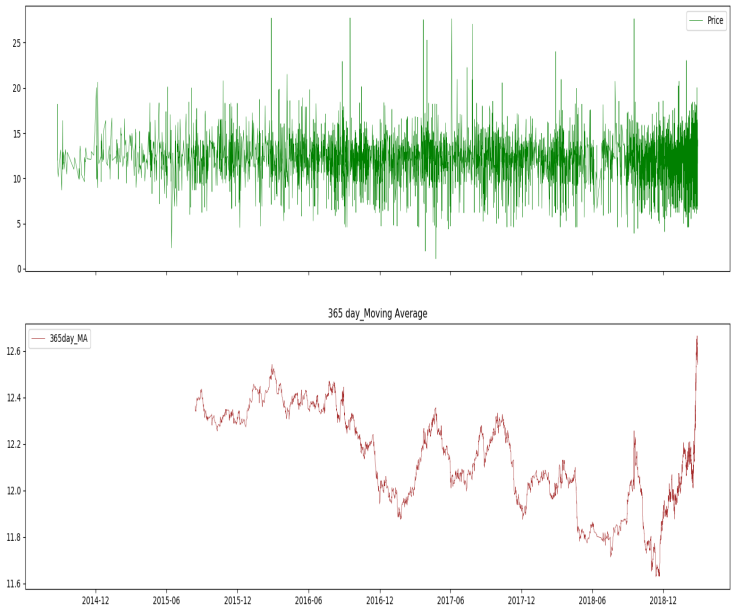
- 60 days moving average



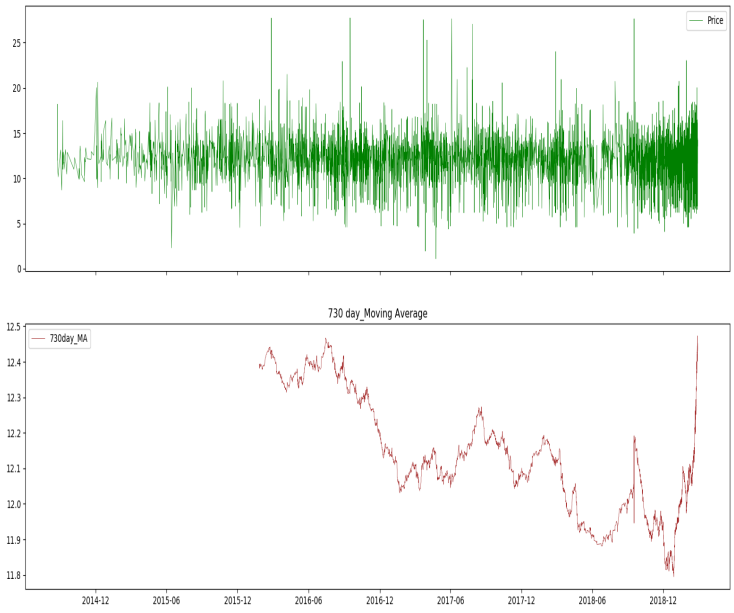
- 120 days moving average



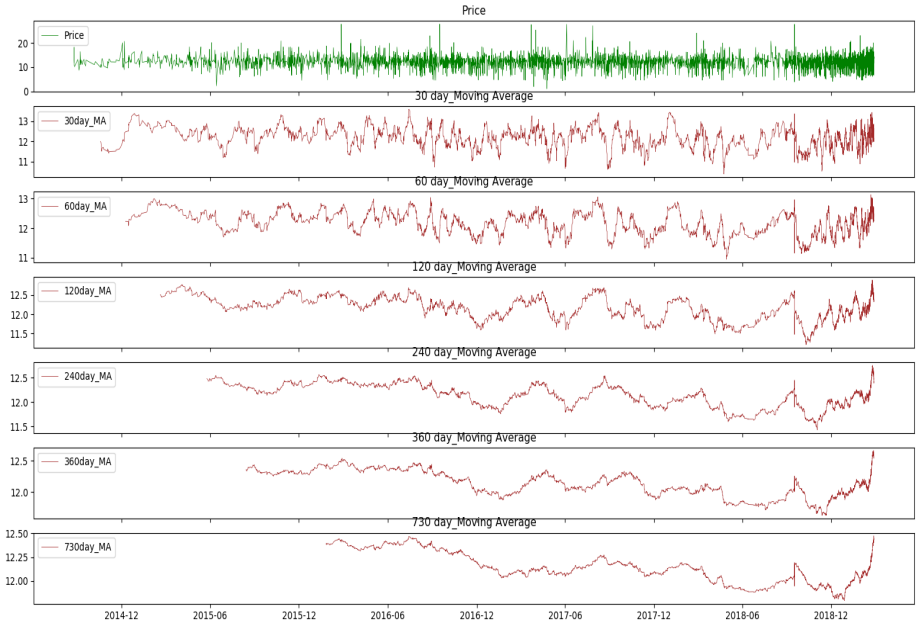
- 240 days moving average



- 356 days moving average



- 730 days moving average



- Putting it all together.

Trend Observation: For the past 3years until now, we can observe house prices declining towards the end of the year(in december) and then picking up somewhere around the January of the new year. More importantly, this trend of price declining and incline tends to happen at the middel(June) and towards end(december) of the year. You can check the image folder of the project to see the graphs more clearly.

3 Categorical data and Outliers

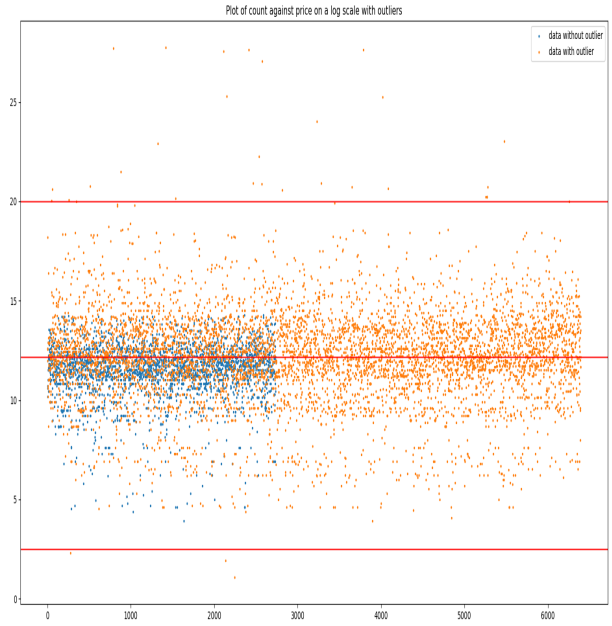
From here you can see how the outlier function effectively dealt with the data with having to set data.price limits to remove. Compare this two graph. The only problem is that more than half of the data was trimmed as outlier. As you can see below.

To do this we employ the use of the function **remove_outliers**. Where we particularly made use of percentile to remove the outliers. The following coed helps us achieve this result.

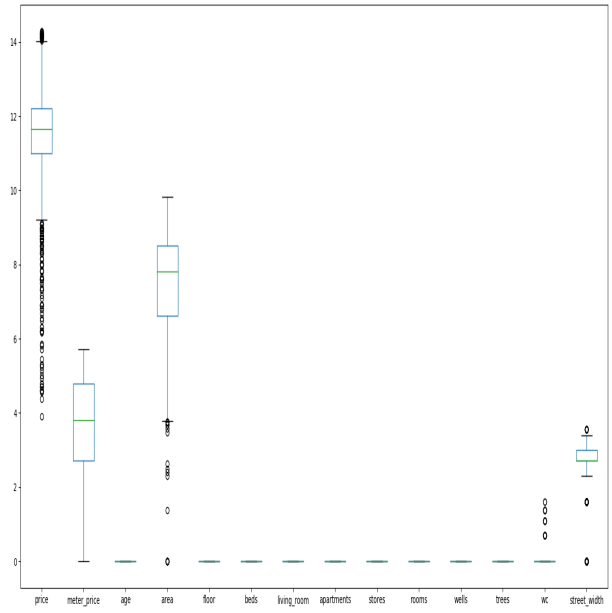
```
lower_quart = .25 #using the 25th percentile for the lower quartile
upper_quart = .75 #and the 75th percentile for the upper percentile
multiplier = 1.5
quart_1 = df.quantile(lower_quartile)
quart_2 = df.quantile(upper_quartile)
diff_quart = quart_2 - quart_1
df = df[~((df < (quart_1 - 1.5 * diff_quart)) |(df > (quart_2 + 1.5 *\
                diff_quart))).any(axis=1)]
```

* Note that df is the house price dataset

This would have a huge impact on our dataset by cutting out more than half the data.



Now lets see how the data look using the boxplot.



We have succeeded in removing most of the largely deviated data. Remember the removed data is responsible for the inbalance in our dataset by causing a positive skewness.
Removing this also affect our model as we would find out soon.

4 feature Importance and Extraction

In this section we require the use of a powerful boosting algorithm to understand which features affect housing price the most.
The finding are interesting as it shows us the features more likely to drive price fluctuations over the years.

Solidifying this result requires a machine learning approach called **Grid-search** to find the best estimation parameter for our model. This parameter would be used finally to predict the final outcome of the features.

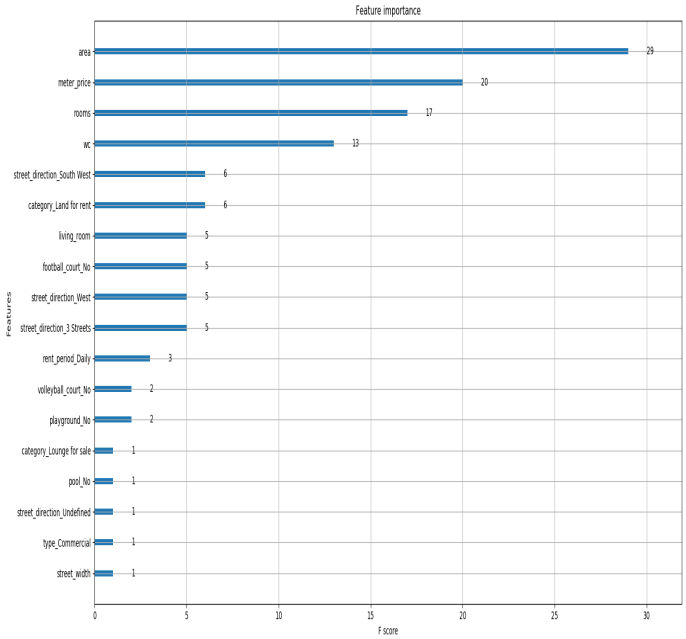
The code required for this stage is shown below.

```
#plot feature importance
def plot_features(model):
  figsize = [20, 16]
  fig, ax = plt.subplots(1, 1, figsize = figsize)
  return plot_importance(model)


def categorical_handler(df, standardize = None, remove_objects = True,\
                    logg = None, normalize = None, \
                    lower_quartile = None, upper_quartile = None, \
                    multiplier = None):

  df_dum = df.copy(deep = True)
  df_num = df.copy(deep = True)
  #seperate numerical variables
  for ii in df_num.columns:
    if df_num[ii].dtypes == object:
      df_num = df_num.drop(ii, axis = 1)
  #seperate categories
  for ii in df_dum.columns:
    if df_dum[ii].dtypes != object:
      df_dum = df_dum.drop(ii, axis = 1)
  #remove outliers
  quart_1 = df_num.quantile(lower_quart)
  quart_2 = df_num.quantile(upper_quart)
  diff_quart = abs(quart_1 - quart_2)
  df_num = df_num[~((df_num < (quart_1 - multiplier * diff_quart))\
            | (df_num > (quart_2 + multiplier * diff_quart))).any(axis=1)]
  #convert categorical variables to numerical var
  df_dum = pd.get_dummies(df_dum, dtype = float)
  #merge
  df = pd.merge(df_num.reset_index(drop = True),\
             df_dum.reset_index(drop = True), \
             left_index=True, right_index=True)
```

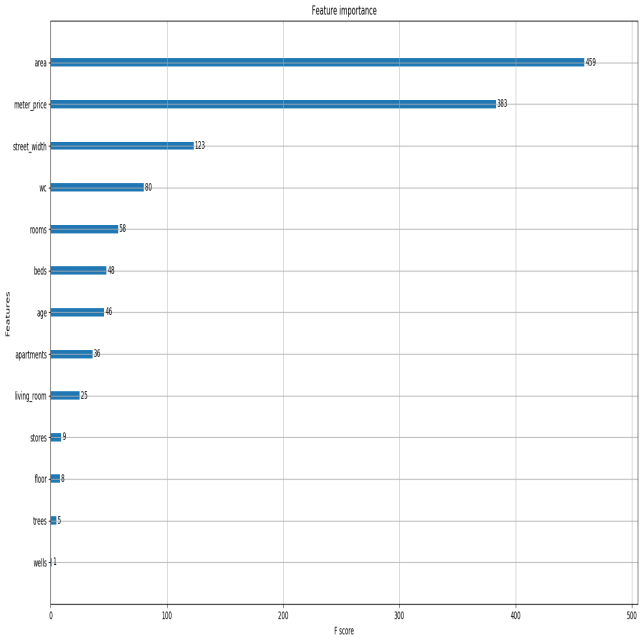
```
df.set_index(df_num.index, inplace = True)
#create additional time features

#standard deviation
def stdev(df):
  return np.std(df, axis = 0)
#mean deviation
def mean_dev(df):
  return df - np.mean(df, axis = 0)
#log of data
def logg_dat(df):
  return np.log(df)

#standardized values for columns
if standardize:
  for ii, ij in enumerate(df.columns):
    print(ii, ij)
    df['{}'.format(ij)] = mean_dev(df.loc[:, '{}'.format(ij)])\
                          /stdev(df.loc[:, '{}'.format(ij)])
    df = df.replace([np.inf, -np.inf, np.nan], 0)
elif logg:
  df = logg_dat(df)
  df = df.replace([np.inf, -np.inf, np.nan], 0)
elif normalize:
  for ii, ij in enumerate(df.columns):
    df['{}'.format(ij)] = (df.loc[:, '{}'.format(ij)] -
    min(df.loc[:, '{}'.format(ij)]))/\
    (max(df.loc[:, '{}'.format(ij)]) - min(df.loc[:, '{}'.format(ij)]))
    df = df.replace([np.inf, -np.inf, np.nan], 0)
else:
  pass


return df
```

The code begins by first converting the categorical data to numerical/float data, then removing outliers and finally standardizing the data.

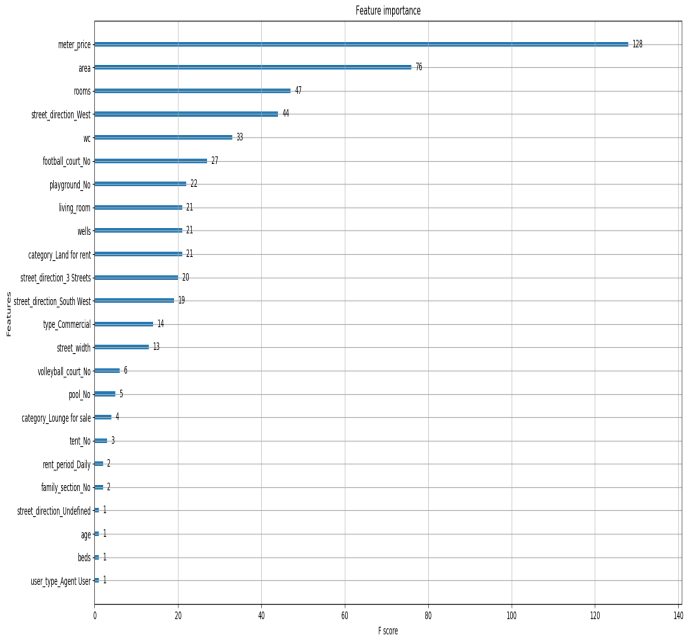
1. Feature importance on full datatset (dataset containing outliers.)



- Feature importance on understandadized data

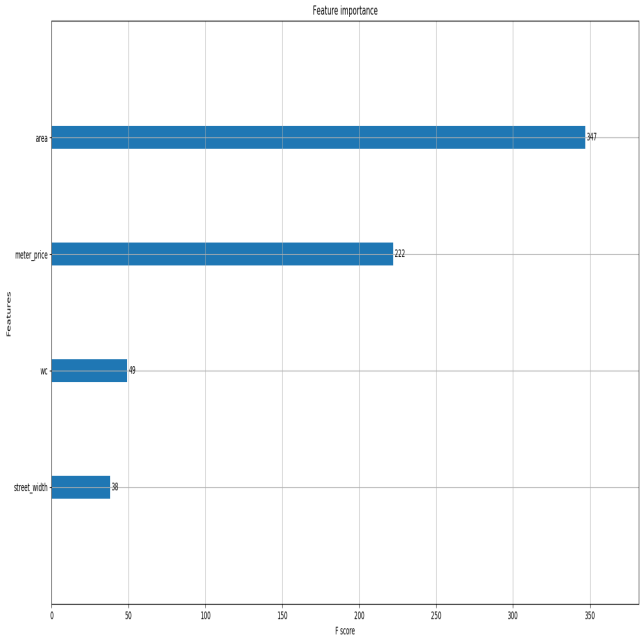


- Feature importance on log_data



- Feature importance on Standardized data. From all the 3 figures above, you will trend the trend of similar features **area, meter price, rooms, wc and street_direction_south_west, coming in the top 5 position.** This means that these features play a very final role in determining how much a house would cost at any given time of the year.

2. Feature importance in the absence of Outliers



- Feature importance after romoving outliers. You will find out that not a lot of features were present after the removal of outliers. leaving us with just **area, meter price, wc and street width as the major factors responsible for price fluctuation.**

5 Modeling

This is the section where i would be doing huge part of the work. It invloves the modeling an algorithm capable of predciting the price as well as forecasting future prices at any given time when provided the right variables.

This would be the next phase of the project and would require alot of computing time to produce the best model. Plus we would be creating far more relevant features capable of helping us significantly predict or fact house prices.

References

A Appendix

No provided at the moment. See doc folder if any.