CAPSTONE PROJECT

Business Report

HOUSE PRICE PREDICTION

A house value is simply more than location and square footage. Like the features that make up a person, an educated party would want to know all aspects that give a house its value.

For example, you want to sell a house and you don't know the price which you may expect it can'tbe too low or too high.

To find house price you usually try to find similar properties in your neighborhood and based on gathered data you will try to assess your house price.

MAY 29 2022

PARAKANDLA MANISHA
PGPDSBA Great Learning Online

Table of Contents:

List of Figures:

List of Tables:

Introduction- Problem Understanding

Defining problem statement:

Whenever any individual/business wants to sell or buy a house, they generally face this kind of issue as they don’t have clear understanding on the price which they should offer. Due to this there is a chance that they might offer too low or high price for the property. Hence, we can analyse the available data of the properties/houses in the area and can predict the price. We need to find out how these attributes/features influence the house prices. Right pricing is very important aspect for selling the house. It is very important to understand that what are the factors and how they are influencing the house price.

Need of the study/project:

we need to predict the right price of the house based on the study results from attributes/features and result of the models performed from the dataset. To find out the important aspects that reflect on the sale price of the houses.

Understanding business/social opportunity:

Many people don't know the features/aspects which accumulate property price, we can provide them House Buying & Selling guidance services in the area so they can buy or sell their property with most suitable price tag and they won't lose their money by offering low price or keep waiting for the buyers by putting high prices.

Objective

Our objective is to Build a model which will predict the house price and its relevant features that manipulate the sale price when selected features based on analysis are passed into to the model.

- we need to analyse the data and find out the significant/Important features from the given features dataset which affects the house price the most.
- Build best feasible model to predict the house price with 95% confidence level.

Data Dictionary:

1. cid: a notation for a house
2. dayhours: Date house was sold
3. price: Price is prediction target
4. room_bed: Number of Bedrooms/House
5. room_bath: Number of bathrooms/bedrooms
6. living_measure: square footage of the home
7. lot_measure: square footage of the lot
8. ceil: Total floors (levels) in house
9. coast: House which has a view to a waterfront
10. sight: Has been viewed
11. condition: How good the condition is (Overall)
12. quality: grade given to the housing unit, based on grading system
13. ceil_measure: square footage of house apart from basement
14. basement: square footage of the basement
15. yr_built: Built Year
16. yr_renovated: Year when house was renovated
17. zipcode: zip
18. lat: Latitude coordinate
19. long: Longitude coordinate
20. living_measure15: Living room area in 2015(implies-- some renovations) This might or might not have affected the lotsize area
21. lot_measure15: lotSize area in 2015(implies-- some renovations)
22. furnished: Based on the quality of room
23. total_area: Measure of both living and lot

Data Report

Understanding how data was collected in terms of time, frequency, and methodology:

First, we import the necessary libraries for loading and analyzing the data.

Reading the data using pandas and importing 'innercity.xlsx' file

	cid	dayhours	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	...	basement	yr_built	yr_renovated	zi
0	3876100940	20150427T000000	600000	4.0	1.75	3050.0	9440.0	1	0	0.0	...	1250.0	1966	0	
1	3145600250	20150317T000000	190000	2.0	1.00	670.0	3101.0	1	0	0.0	...	0.0	1948	0	
2	7129303070	20140820T000000	735000	4.0	2.75	3040.0	2415.0	2	1	4.0	...	0.0	1966	0	
3	7338220280	20141010T000000	257000	3.0	2.50	1740.0	3721.0	2	0	0.0	...	0.0	2009	0	
4	7950300670	20150218T000000	450000	2.0	1.00	1120.0	4590.0	1	0	0.0	...	0.0	1924	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
21608	203600600	20150310T000000	685530	4.0	2.50	3130.0	60467.0	2	0	0.0	...	0.0	1996	0	
21609	625049281	20140521T000000	535000	2.0	1.00	1030.0	4841.0	1	0	0.0	...	110.0	1939	0	
21610	424069018	20140905T000000	998000	3.0	3.75	3710.0	34412.0	2	0	0.0	...	800.0	1978	0	
21611	7258200055	20150206T000000	262000	4.0	2.50	1560.0	7800.0	2	0	0.0	...	0.0	1997	0	
21612	8805900430	20141229T000000	1150000	4.0	2.50	1940.0	4875.0	2	0	0.0	...	0.0	1925	0	

21613 rows × 23 columns

Table 1. Sample of the data

The dataset is having 21613 rows and 23 columns(features) and the data collected is based on houses sold over 2014 & 2015 years respectively. This, we can observe from the date & timestamp in 'dayhours' column. The feature 'dayhours' is having date with the timestamp included, and the datatype is of object. We can further modify the column for better analysis.

let's check out the columns/features we have in the dataset:

```
Index(['cid', 'dayhours', 'price', 'room_bed', 'room_bath', 'living_measure',
       'lot_measure', 'ceil', 'coast', 'sight', 'condition', 'quality',
       'ceil_measure', 'basement', 'yr_built', 'yr_renovated', 'zipcode',
       'lat', 'long', 'living_measure15', 'lot_measure15', 'furnished',
       'total_area'],
      dtype='object')
```

The above columns/ features explained:

1. **cid:** Notation for a house. May not be useful for analysis. So, we will drop this column
2. **dayhours:** Represents Date when the house was sold. May not be useful for analysis, can be dropped.
3. **price:** It's our TARGET feature, that we must predict based on other featues
4. **room_bed:** Represents number of bedrooms in a house
5. **room_bath:** Represents number of bathrooms in a house
6. **living_measure:** Represents square footage of house
7. **lot_measure:** Represents square footage of lot
8. **ceil:** Represents number of floors in house
9. **coast:** Represents whether house has waterfront view. It seems to be a categorical variable. We will see in our further data analysis
10. **sight:** Represents how many times sight has been viewed.
11. **condition:** Represents the overall condition of the house. It's the kind of rating given to the house.
12. **quality:** Represents grade given to the house based on grading system
13. **ceil_measure:** Represents square footage of house apart from basement
14. **basement:** Represents square footage of basement
15. **yr_built:** Represents the year when house was built
16. **yr_renovated:** Represents the year when house was last renovated
17. **zipcode:** Represents zipcode as name implies
18. **lat:** Represents Lattitude co-ordinates
19. **long:** Represents Longitude co-ordinates
20. **living_measure15:** Represents square footage of house, when measured in 2015 year as house area may or may not change after renovation if any happened
21. **lot_measure15:** Represents square footage of lot, when measured in 2015 year as lot area may or may not change after renovation if any done
22. **furnished:** Tells whether house is furnished or not. It seems to be categorical variable as description implies
23. **total_area:** Represents total area that is area of both living and lot

Descriptive summary of the data:

	count	mean	std	min	25%	50%	75%	max
cid	21613.0	4.580302e+09	2.876566e+09	1.000102e+06	2.123049e+09	3.904930e+09	7.308900e+09	9.900000e+09
price	21613.0	5.401822e+05	3.673622e+05	7.500000e+04	3.219500e+05	4.500000e+05	6.450000e+05	7.700000e+06
room_bed	21505.0	3.371355e+00	9.302886e-01	0.000000e+00	3.000000e+00	3.000000e+00	4.000000e+00	3.300000e+01
room_bath	21505.0	2.115171e+00	7.702481e-01	0.000000e+00	1.750000e+00	2.250000e+00	2.500000e+00	8.000000e+00
living_measure	21596.0	2.079861e+03	9.184961e+02	2.900000e+02	1.429250e+03	1.910000e+03	2.550000e+03	1.354000e+04
lot_measure	21571.0	1.510458e+04	4.142362e+04	5.200000e+02	5.040000e+03	7.618000e+03	1.068450e+04	1.651359e+06
sight	21556.0	2.343663e-01	7.664376e-01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	4.000000e+00
quality	21612.0	7.656857e+00	1.175484e+00	1.000000e+00	7.000000e+00	7.000000e+00	8.000000e+00	1.300000e+01
ceil_measure	21612.0	1.788367e+03	8.281025e+02	2.900000e+02	1.190000e+03	1.560000e+03	2.210000e+03	9.410000e+03
basement	21612.0	2.915225e+02	4.425808e+02	0.000000e+00	0.000000e+00	0.000000e+00	5.600000e+02	4.820000e+03
yr_renovated	21613.0	8.440226e+01	4.016792e+02	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	2.015000e+03
zipcode	21613.0	9.807794e+04	5.350503e+01	9.800100e+04	9.803300e+04	9.806500e+04	9.811800e+04	9.819900e+04
lat	21613.0	4.756005e+01	1.385637e-01	4.715590e+01	4.747100e+01	4.757180e+01	4.767800e+01	4.777760e+01
living_measure15	21447.0	1.987066e+03	6.855196e+02	3.990000e+02	1.490000e+03	1.840000e+03	2.360000e+03	6.210000e+03
lot_measure15	21584.0	1.276654e+04	2.728699e+04	6.510000e+02	5.100000e+03	7.620000e+03	1.008700e+04	8.712000e+05
furnished	21584.0	1.967198e-01	3.975279e-01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	1.000000e+00

Table 2 Descriptive summary of the data

From the Data description we can observe that,

- The five number/statistical summary distribution of the categorical columns.
- From count, we can see that there are missing/null values present in the data which need to be further analysed and imputed accordingly.
- We can also observe that few of the numeric variables are categorised as 'object' datatype, those features need to be changed into numeric datatype.
- we will check the detailed data description again, after the datatypes of few other features are modified.

From the Info () function below we observe that,

- The total number of non-null values present in the data. Here, we can observe null/missing values in each feature clearly.
- The dataset is having 21613 rows and 23 columns.
- variables having float64 datatype: 12
- variables having int64 datatype: 4
- variables having object datatype: 7
- Datatypes for few features need to be modified.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 23 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   cid               21613 non-null  int64
 1   dayhours          21613 non-null  object
 2   price             21613 non-null  int64
 3   room_bed          21505 non-null  float64
 4   room_bath         21505 non-null  float64
 5   living_measure    21596 non-null  float64
 6   lot_measure       21571 non-null  float64
 7   ceil              21571 non-null  object
 8   coast             21612 non-null  object
 9   sight             21556 non-null  float64
 10  condition         21556 non-null  object
 11  quality           21612 non-null  float64
 12  ceil_measure      21612 non-null  float64
 13  basement          21612 non-null  float64
 14  yr_built          21612 non-null  object
 15  yr_renovated      21613 non-null  int64
 16  zipcode           21613 non-null  int64
 17  lat               21613 non-null  float64
 18  long              21613 non-null  object
 19  living_measure15  21447 non-null  float64
 20  lot_measure15     21584 non-null  float64
 21  furnished         21584 non-null  float64
 22  total_area        21584 non-null  object
dtypes: float64(12), int64(4), object(7)
memory usage: 3.8+ MB
```

cleaning the $'s in the data

The features ceil, coast, condition, yr_built, long, total_area is having $'s. First, I have changed the $ to NaN values, later imputed them while treating the missing values.

We are done with the cleaning for columns/features. Now let's go check the data for missing values and impute them if necessary.

```
cid                 0
dayhours            0
price               0
room_bed          121
room_bath         108
living_measure     17
lot_measure        42
ceil               72
coast              31
sight              57
condition          85
quality             1
ceil_measure        1
basement            1
yr_built           15
yr_renovated        0
```

```
zipcode               0
lat                   0
long                 34
living_measure15    166
lot_measure15        29
furnished            29
total_area           68
dtype: int64
```

Percentage of null values present in the data is: 0.04057743025031231

	Null Values	Data Type	No. of Unique Values
cid	0	int64	21436
dayhours	0	object	372
price	0	int64	3625
room_bed	121	float64	12
room_bath	108	float64	30
living_measure	17	float64	1038
lot_measure	42	float64	9765
ceil	72	float64	6
coast	31	float64	2
sight	57	float64	5
condition	85	float64	5
quality	1	float64	12
ceil_measure	1	float64	946
basement	1	float64	306
yr_built	15	float64	116
yr_renovated	0	int64	70
zipcode	0	int64	70
lat	0	float64	5034
long	34	float64	752
living_measure15	166	float64	774
lot_measure15	29	float64	8682
furnished	29	float64	2
total_area	68	float64	11144

Table 3 Null/ Missing value detail

Let us start imputing the missing values one by one:

room_bed: we have 121 null/missing values present in the 'room_bed' feature and these need to be treated.

array([4., 2., 3., 1., 5., 6., **nan**, 7., 10., 8., 9., 33., 11.])

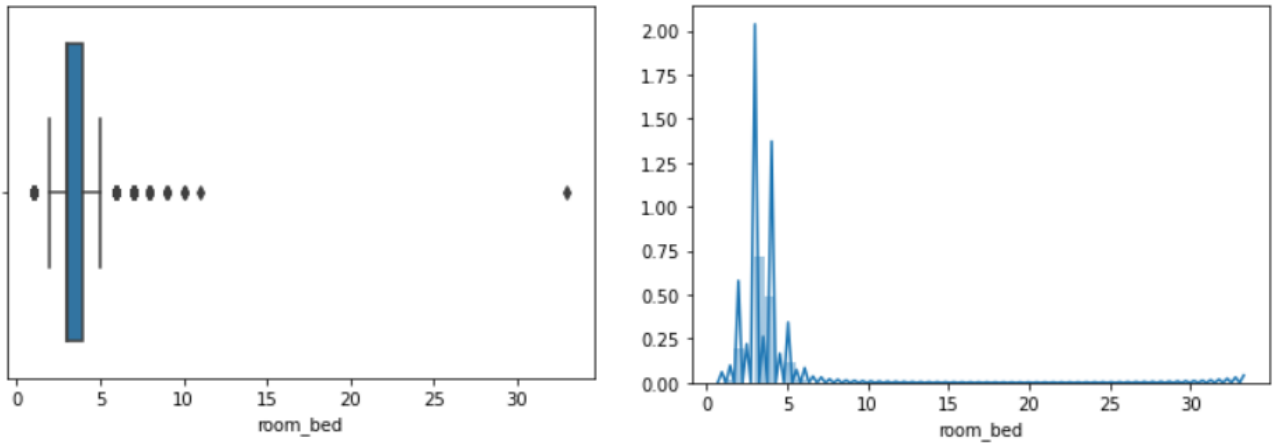


Figure 1 Boxplot & dist plot

```
count    21492.000000
mean         3.373395
std          0.926866
min          1.000000
25%          3.000000
50%          3.000000
75%          4.000000
max         33.000000
Name: room_bed, dtype: float64
```

There are significant number of outliers present in the room_bed feature. Hence, I choose to impute the missing values with the Median. Because median is resistant to outliers.

The data looks to be right-skewed (long tail in the right).

room_bath: we have 108 null/missing values present in the 'room_bath' feature and these need to be treated.

array ([1.75, 1. , 2.75, 2.5 , 1.5 , 3.5 , 2. , 2.25, 3. , 4. , 3.25, 3.75, **nan**, 5. , 0.75, 5.5 , 4.25, 4.5 , 4.75, 8. , 6.75, 5.25, 6. , 0. , 1.25, 5.75, 7.5 , 6.5 , 0.5 , 7.75, 6.25])

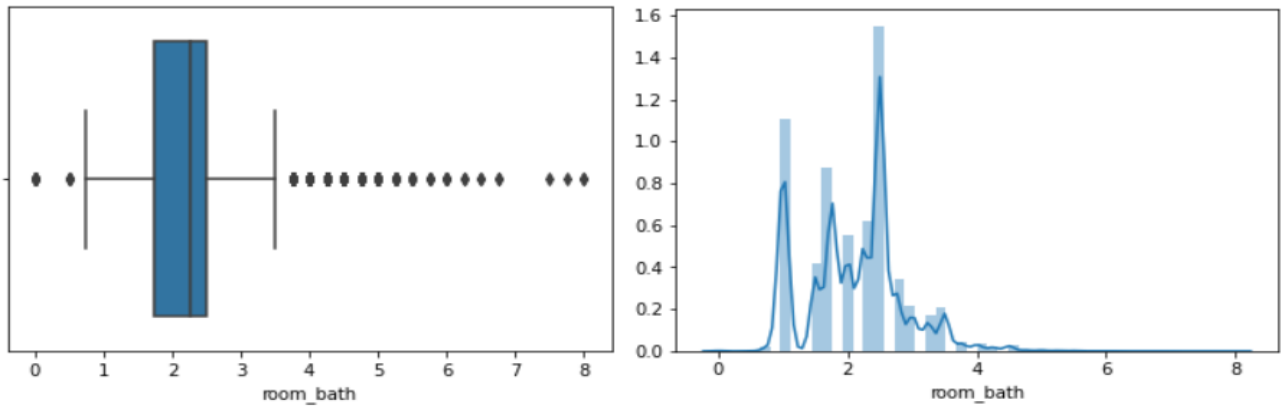


Figure 2 Boxplot & dist plot - room_bath

```
count    21505.000000
mean         2.115171
std          0.770248
min          0.000000
25%          1.750000
50%          2.250000
75%          2.500000
max          8.000000
Name: room_bath, dtype: float64
```

There are significant number of outliers present in the room_bath feature. Hence, I choose to impute the missing values with the **Median**. Because median is resistant to outliers.

living_measure: we can see that 17 missing values in the 'living_measure' feature

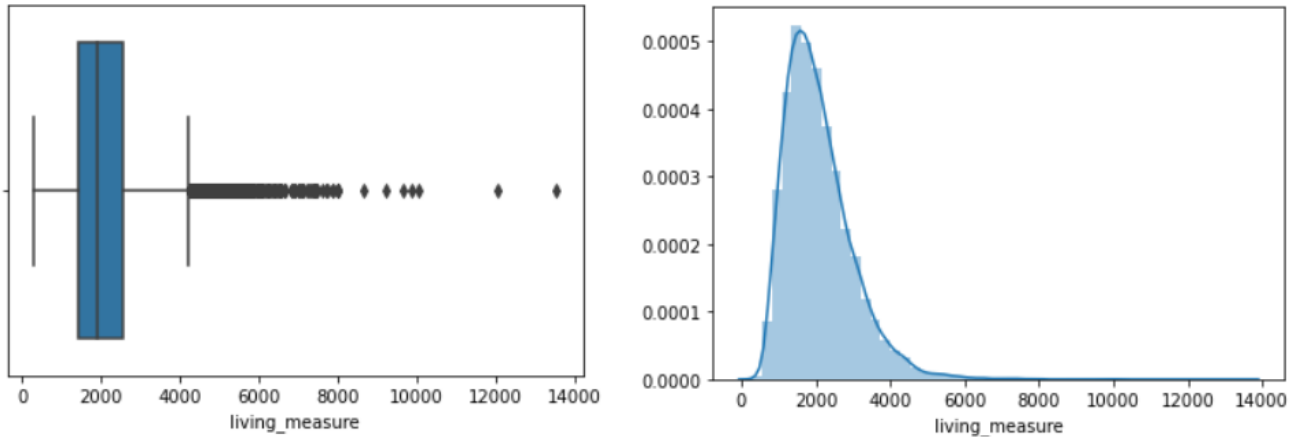


Figure 3 Boxplot & dist plot - living_measure

We can observe that, there are significant number of outliers present in the living_measure feature and the data is right-skewed. Hence, I choose to impute the missing values with the **Median**.

lot_measure: There are 42 missing values present in the 'lot_measure' feature.

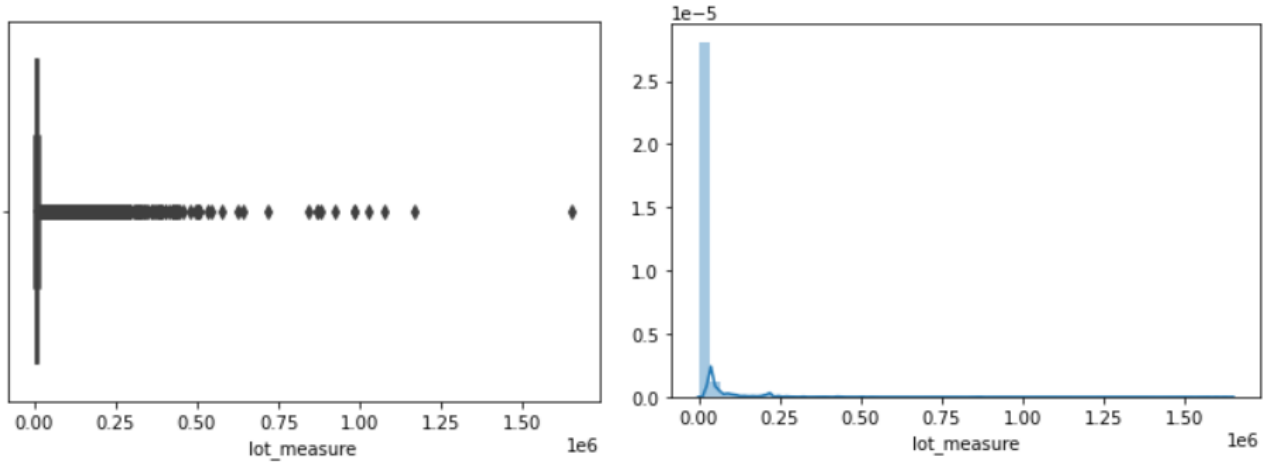


Figure 4 Boxplot & dist plot - lot_measure

We can observe that, there are significant number of outliers present in the lot_measure feature and the data is right-skewed. Hence, I choose to impute the missing values with the **Median**.

Similarly, for '**ceil'** as well I have imputed the null values with median. As it is a numeric continuous variable.

coast: The feature 'coast' seems to be categorical which represents whether house has waterfront view or not. We can Impute the missing values in 'coast' with **Mode** in this case.

Sight: I have considered sight as numeric variable. 0 for sight is not viewed, rest for number of times viewed.

array ([0., 4., 2., 3., 1., **nan**])

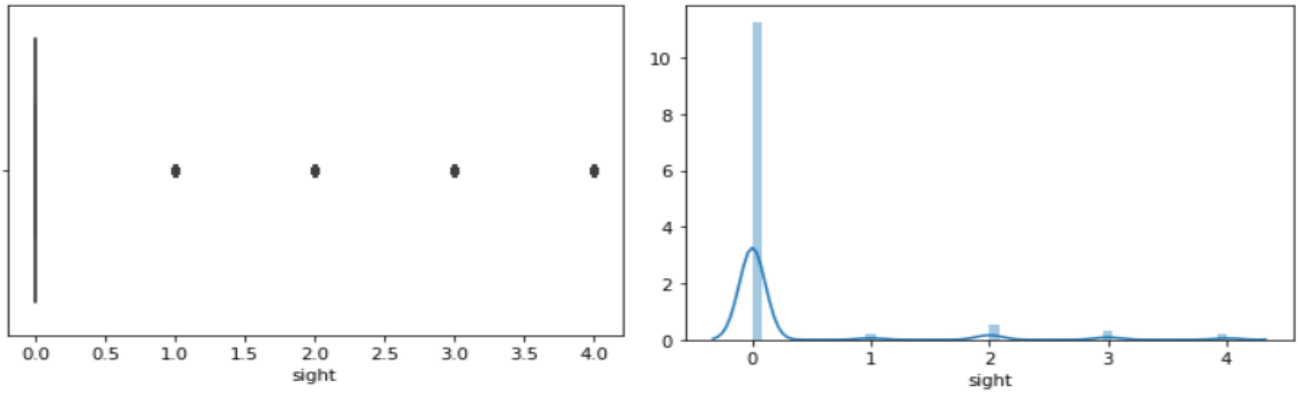


Figure 5 Boxplot & dist plot – sight

We can observe that, there are significant number of outliers present in the lot_measure feature and the data is right-skewed. Hence, I choose to impute the missing values with the **Median**.

The features **'Condition'** and **'quality'** are categorical variables. Hence, I have imputed the missing values with **Mode.**

ceil_measure: The feature **'ceil_measure'** having only one missing value.

```
2497., 3056., 2594., 1767., 2542., 2216., 1061., 2341., 5450.,
2963., 3202., 1329., 1216., 1491., 1397.,  992., 1741., 2099.,
2665., 2208., 2557., 2313.,  902., 1606.,  995., 3691., 1489.,
1078., 3078., 3362., 3597., 2415., 2798., 1782., 5140., 5220.,
4420., 5584., 1435., 2577., 2015., 5670., 1212., 1087., 1984.,
3154., 4800., 1921., 2793., 5010., 1611., 2181., 2518., 2311.,
2656., 2452., 5610., 6640.,  944., 2238., 1536., 1296., 1714.,
4600., 2092., 2014., 2932., 1805., 2413., 2115., 6380., 1876.,
5980., 2329., 2601., 3555., 2165., 1976., 1094., 1556., 4330.,
3002., 2174., 2782., 2717., 2568.,  844.,  962., 1764., 1248.,
3236., 1451., 3745., 2068., 3274., 3052., 2547., 2233., 3915.,
2198., 4450., 5090., 2844., 5530., 1084., 1904., 2531., 3118.,
6530., 2628., 1425., 1628., 1571., 3136., 2331., 5310.,  901.,
3674., 1834., 1396., 6120., 6085., 2356., 6090., 1422., 3001.,
6290., 2835., 5490.,  833., 8020., 3265.,  410., 1726.,  988.,
3064., 1528., 2598., 2448., 2395.,  828., 1811.,  866.,   nan,
 894., 2382., 4285., 1595., 1463., 4133., 2064., 5770., 4870.,
1105., 2253.])
```

As it is a continuous numeric feature, I have imputed the NaN value with the median.

basement: The feature basement also having only one missing value. I have imputed it with Median.

```
2110., 2050., 4130., 1008., 2330., 2030.,  516.,  704., 2580.,
 915.,  172., 1510.,  602., 2550., 1610., 1284., 1281., 2170.,
1798., 2240., 2070., 1930., 1880., 2020.,  508.,  295., 2360.,
2720., 2160.,  435.,  225., 2220., 1860., 1840., 2590., 2130.,
2490.,  862., 3000., 2310., 2150.,  556., 1852.,  475., 1548.,
1960.,  235., 2610.,  875., 1024., 2190.,  415.,  792.,  768.,
1248., 1275.,   20., 2850., 1525., 2120., 1913., 2250.,   65.,
1770., 1750., 2570., 2500.,  588.,  266., 2350., 1481.,  274.,
 248.,  935., 1245., 2196.,  243., 2810.,   nan,  906., 1920.,
2180.])
```

yr_built: Represents the year when house was built. It can be considered as categorical variable. There are 15 missing values present in this feature. As it is a categorical variable, I have imputed the missing values with **Mode**.

Long: The 'feature' long is having 34 missing values. I have considered long as a numeric categorical variable. Hence, imputed with Median.

Living_measure15: Represents square footage of house, when measured in 2015 year as house area may or may not change after renovation if any happened.

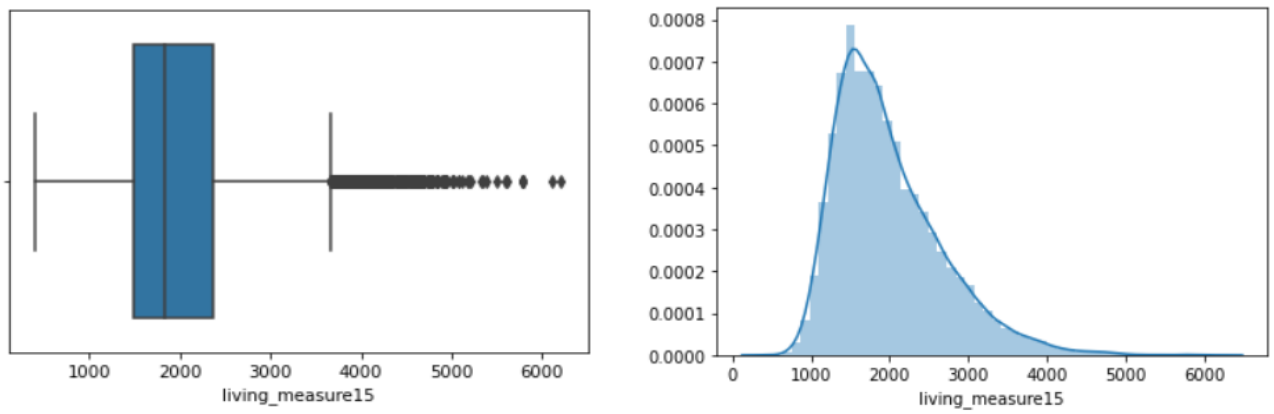


Figure 6 Boxplot & dist plot - living_measure15

We can observe that the feature 'living_measure15' is having outliers, The distribution is normal and slightly right skewed.

The feature 'leaving_measure' have 116 missing values. These NaN values need to be imputed with median as it is a continuous numeric variable.

lot_measure15: The feature 'lot_measure15' has 29 missing values. I have imputed the NaN values with median as it is a continuous numeric variable.

Let's see the boxplot and distribution for 'lot_measure15':

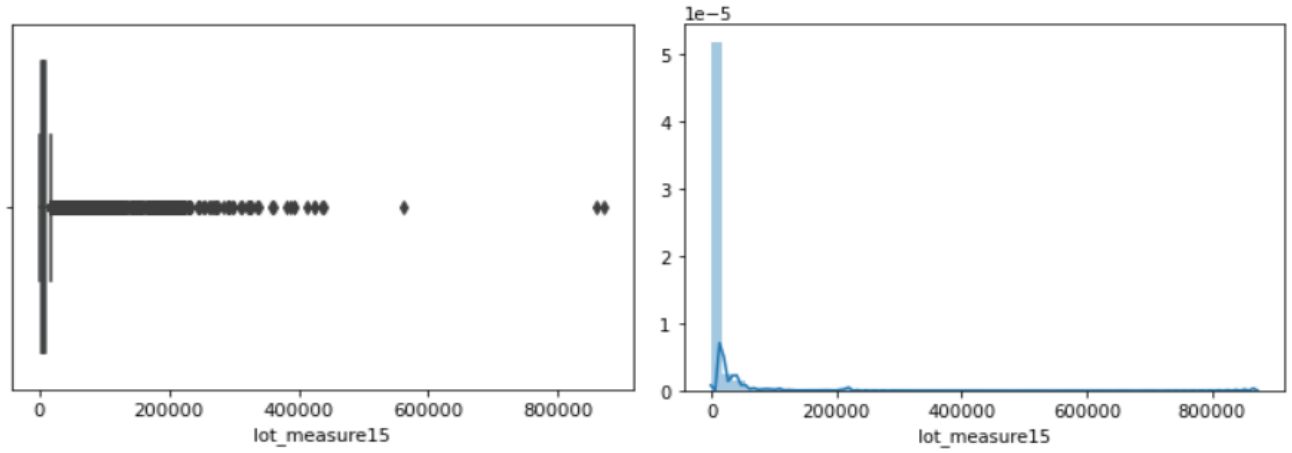


Figure 7 Boxplot & dist plot - lot_measure15

We can observe that, there are outliers in 'lot_measure15' feature and the distribution of the data is highly right skewed. These outliers need to be treated further for future analysis.

furnished: This is a categorical variable represents house is furnished or not. Hence, I have imputed the 29 missing values with mode.

total_area: total area is a measure of living and lot. There are 68 missing values present in this column. As total_area is a continuous numeric variable I choose to impute with median.

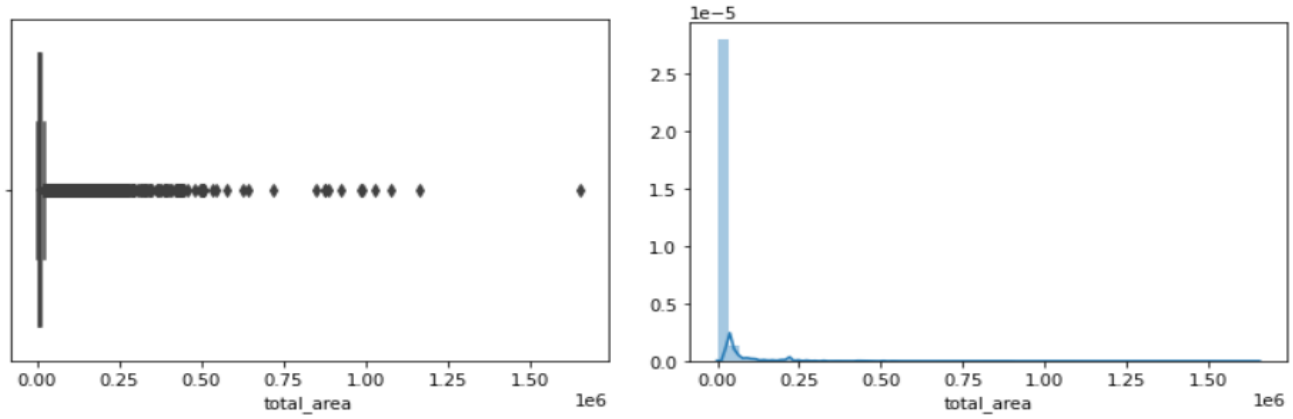


Figure 8 Boxplot & dist plot - total_area

We can observe that, there are outliers in 'total_area' feature and the distribution of the data is highly right skewed. These outliers need to be treated further for future analysis.

All the missing values are imputed. Now, the data is clean and do not have any missing data/information.

Let's check the info and description of data after imputing the outliers.

I will perform Exploratory Data Analysis i.e., Univariate & Bivariate analysis after imputing the missing values. So that we get better visualization and proper insights from the data

Detailed Description of data for each individual variable:

	count	mean	std	min	25%	50%	75%	max
cid	21613.0	4.580302e+09	2.876566e+09	1.000102e+06	2.123049e+09	3.904930e+09	7.308900e+09	9.900000e+09
price	21613.0	5.401822e+05	3.673622e+05	7.500000e+04	3.219500e+05	4.500000e+05	6.450000e+05	7.700000e+06
room_bed	21613.0	3.371304e+00	9.246877e-01	1.000000e+00	3.000000e+00	3.000000e+00	4.000000e+00	3.300000e+01
room_bath	21613.0	2.115845e+00	7.683799e-01	0.000000e+00	1.750000e+00	2.250000e+00	2.500000e+00	8.000000e+00
living_measure	21613.0	2.079727e+03	9.181472e+02	2.900000e+02	1.430000e+03	1.910000e+03	2.550000e+03	1.354000e+04
lot_measure	21613.0	1.509003e+04	4.138466e+04	5.200000e+02	5.043000e+03	7.618000e+03	1.066000e+04	1.651359e+06
ceil	21613.0	1.494147e+00	5.390116e-01	1.000000e+00	1.000000e+00	1.500000e+00	2.000000e+00	3.500000e+00
coast	21613.0	7.449220e-03	8.598879e-02	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	1.000000e+00
sight	21613.0	2.337482e-01	7.655206e-01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	4.000000e+00
condition	21613.0	3.407718e+00	6.499332e-01	1.000000e+00	3.000000e+00	3.000000e+00	4.000000e+00	5.000000e+00
quality	21613.0	7.656827e+00	1.175465e+00	1.000000e+00	7.000000e+00	7.000000e+00	8.000000e+00	1.300000e+01
ceil_measure	21613.0	1.788356e+03	8.280848e+02	2.900000e+02	1.190000e+03	1.560000e+03	2.210000e+03	9.410000e+03
basement	21613.0	2.915090e+02	4.425750e+02	0.000000e+00	0.000000e+00	0.000000e+00	5.600000e+02	4.820000e+03
yr_built	21613.0	1.971039e+03	2.938506e+01	1.900000e+03	1.951000e+03	1.975000e+03	1.997000e+03	2.015000e+03
yr_renovated	21613.0	8.440226e+01	4.016792e+02	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	2.015000e+03
zipcode	21613.0	9.807794e+04	5.350503e+01	9.800100e+04	9.803300e+04	9.806500e+04	9.811800e+04	9.819900e+04
lat	21613.0	4.756005e+01	1.385637e-01	4.715590e+01	4.747100e+01	4.757180e+01	4.767800e+01	4.777760e+01
long	21613.0	-1.222139e+02	1.407590e-01	-1.225190e+02	-1.223280e+02	-1.222300e+02	-1.221250e+02	-1.213150e+02
living_measure15	21613.0	1.985936e+03	6.830025e+02	3.990000e+02	1.490000e+03	1.840000e+03	2.360000e+03	6.210000e+03
lot_measure15	21613.0	1.275964e+04	2.726932e+04	6.510000e+02	5.100000e+03	7.620000e+03	1.008000e+04	8.712000e+05
furnished	21613.0	1.964558e-01	3.973264e-01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	1.000000e+00
total_area	21613.0	1.716808e+04	4.156534e+04	1.423000e+03	7.040000e+03	9.575000e+03	1.297000e+04	1.652659e+06

Table 4 Detailed Description of data for each individual variable

1. **CID:** Notation/ID for the house. Not useful for our analysis
2. **Dayhours:** 5 factor analysis is reflecting for this column
3. **price:** Our target column value is in 75k - 7700k range. As Mean > Median, it is **Right Skewed**.
4. **room_bed:** Number of bedrooms range from 0 - 33. The distribution is **slightly Right Skewed.**
5. **room_bath:** Number of bathrooms range from 0 - 8. As Mean is slightly < Median, it is **slightly Left Skewed**.
6. **living_measure:** square footage of house ranges from 290 - 13,540. As Mean > Median, it is **Right Skewed**.
7. **lot_measure:** square footage of lot ranges from 520 - 16,51,359. As Mean almost double the Median, it's **Highly Right Skewed**.
8. **ceil:** Number of floors range from 1 - 3.5 As Mean > Median, it's **almost Normal Distributed**.
9. **coast:** As this value represent whether house has waterfront view or not. It's **categorical column**. From above analysis we got to know, very few houses have the waterfront view.
10. **sight:** Value ranges from 0 - 4. As Mean > Median, it's **Right Skewed**
11. **condition:** Represents rating of house which ranges from 1 - 5. As Mean > Median, it's **Right Skewed**
12. **quality:** Representing grade given to house which range from 1 - 13. As Mean > Median, it's **Right Skewed**.

13. **ceil_measure:** square footage of house apart from basement ranges in 290 - 9,410. As Mean > Median, it's **Right Skewed**.
14. **basement:** Square footage house basement ranges in 0 - 4,820. As Mean highly > Median, it's **Highly Right Skewed**.
15. **yr_built:** House built year ranges from 1900 - 2015. As Mean < Median, it's **Left Skewed**.
16. **yr_renovated:** House renovation year only 2015. So, this column can be used as **Categorical Variable** which refers whether house is renovated or not.
17. **zipcode:** House zipcode ranges from 98001 - 98199. As Mean > Median, it's **Right Skewed**.
18. **lat:** Lattitude ranges from 47.1559 - 47.7776 As Mean < Median, it's **Left Skewed**.
19. **long:** Longitude ranges from -122.5190 to -121.315 As Mean > Median, it's **Right Skewed**.
20. **living_measure15:** Value ranges from 399 to 6,210. As Mean > Median, it's **Right Skewed**.
21. **lot_measure15:** Value ranges from 651 to 8,71,200. As Mean highly > Median, it's **Highly Right Skewed**.
22. **furnished:** Representing whether house is furnished or not. It's a **Categorical Variable**
23. **total_area** Total area of house ranges from 1,423 to 16,52,659. As Mean is almost double of Median, it's **Highly Right Skewed.**

From above analysis we can observe that,

- Most columns distribution is Right-Skewed and only few features are Left-Skewed (like room_bath, yr_built, lat).
- We have columns which are Categorical in nature are:
  coast, yr_built, yr_renovated , quality, condition, furnished.

There are no duplicate values present in the data.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 23 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   cid               21613 non-null  int64
 1   dayhours          21613 non-null  object
 2   price             21613 non-null  int64
 3   room_bed          21613 non-null  float64
 4   room_bath         21613 non-null  float64
 5   living_measure    21613 non-null  float64
 6   lot_measure       21613 non-null  float64
 7   ceil              21613 non-null  float64
 8   coast             21613 non-null  float64
 9   sight             21613 non-null  float64
 10  condition         21613 non-null  float64
 11  quality           21613 non-null  float64
 12  ceil_measure      21613 non-null  float64
 13  basement          21613 non-null  float64
 14  yr_built          21613 non-null  float64
 15  yr_renovated      21613 non-null  int64
 16  zipcode           21613 non-null  int64
 17  lat               21613 non-null  float64
 18  long              21613 non-null  float64
 19  living_measure15  21613 non-null  float64
 20  lot_measure15     21613 non-null  float64
 21  furnished         21613 non-null  float64
 22  total_area        21613 non-null  float64
dtypes: float64(18), int64(4), object(1)
memory usage: 3.8+ MB
```

We can observe that datatypes for few features are changed after imputation. we will be modifying them for further analysis. Let's review the info () again after changing the datatypes for features.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 23 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   cid               21613 non-null  int64
 1   dayhours          21613 non-null  object
 2   price             21613 non-null  int64
 3   room_bed          21613 non-null  float64
 4   room_bath         21613 non-null  float64
 5   living_measure    21613 non-null  float64
 6   lot_measure       21613 non-null  float64
 7   ceil              21613 non-null  float64
 8   coast             21613 non-null  object
 9   sight             21613 non-null  float64
 10  condition         21613 non-null  object
 11  quality           21613 non-null  object
 12  ceil_measure      21613 non-null  float64
 13  basement          21613 non-null  float64
 14  yr_built          21613 non-null  object
 15  yr_renovated      21613 non-null  object
 16  zipcode           21613 non-null  int64
 17  lat               21613 non-null  float64
 18  long              21613 non-null  float64
 19  living_measure15  21613 non-null  float64
 20  lot_measure15     21613 non-null  float64
 21  furnished         21613 non-null  object
 22  total_area        21613 non-null  float64
dtypes: float64(13), int64(3), object(7)
memory usage: 3.8+ MB
```

Now, in the dataset, we have 21613 records for all the variables(no null/missing values) and 23 columns, out of which

- 13 features are of float64 type
- 3 features are of integer type
- 7 feature is of object type

	count	mean	std	min	25%	50%	75%	max
cid	21613.0	4.580302e+09	2.876566e+09	1.000102e+06	2.123049e+09	3.904930e+09	7.308900e+09	9.900000e+09
price	21613.0	5.401822e+05	3.673622e+05	7.500000e+04	3.219500e+05	4.500000e+05	6.450000e+05	7.700000e+06
room_bed	21613.0	3.371304e+00	9.246877e-01	1.000000e+00	3.000000e+00	3.000000e+00	4.000000e+00	3.300000e+01
room_bath	21613.0	2.115845e+00	7.683799e-01	0.000000e+00	1.750000e+00	2.250000e+00	2.500000e+00	8.000000e+00
living_measure	21613.0	2.079727e+03	9.181472e+02	2.900000e+02	1.430000e+03	1.910000e+03	2.550000e+03	1.354000e+04
lot_measure	21613.0	1.509003e+04	4.138466e+04	5.200000e+02	5.043000e+03	7.618000e+03	1.066000e+04	1.651359e+06
ceil	21613.0	1.494147e+00	5.390116e-01	1.000000e+00	1.000000e+00	1.500000e+00	2.000000e+00	3.500000e+00
sight	21613.0	2.337482e-01	7.655206e-01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	4.000000e+00
ceil_measure	21613.0	1.788356e+03	8.280848e+02	2.900000e+02	1.190000e+03	1.560000e+03	2.210000e+03	9.410000e+03
basement	21613.0	2.915090e+02	4.425750e+02	0.000000e+00	0.000000e+00	0.000000e+00	5.600000e+02	4.820000e+03
zipcode	21613.0	9.807794e+04	5.350503e+01	9.800100e+04	9.803300e+04	9.806500e+04	9.811800e+04	9.819900e+04
lat	21613.0	4.756005e+01	1.385637e-01	4.715590e+01	4.747100e+01	4.757180e+01	4.767800e+01	4.777760e+01
long	21613.0	-1.222139e+02	1.407590e-01	-1.225190e+02	-1.223280e+02	-1.222300e+02	-1.221250e+02	-1.213150e+02
living_measure15	21613.0	1.985936e+03	6.830025e+02	3.990000e+02	1.490000e+03	1.840000e+03	2.360000e+03	6.210000e+03
lot_measure15	21613.0	1.275964e+04	2.726932e+04	6.510000e+02	5.100000e+03	7.620000e+03	1.008000e+04	8.712000e+05
total_area	21613.0	1.716808e+04	4.156534e+04	1.423000e+03	7.040000e+03	9.575000e+03	1.297000e+04	1.652659e+06

Table 5 Descriptive summary of Continuous features

Exploratory Data Analysis

Performing Univariate, Bivariate, Multivariate analysis and understanding each feature one by one.

Note: I have performed EDA after cleaning the data and missing value treatment for better analysis and visualization.

Univariate Analysis:

cid: Notation/ID for a house.

The feature 'cid' is appearing multiple times, it seems data contains house which is sold multiple times. We can observe that, we have 176 properties that were sold more than once in the given data.

dayhours: Represents the date when the house was sold. Before analyzing this feature, I will take the copy of the data for model building.

Converting the dayhours to 'month_year' as sale month-year is relevant for analysis

```
0       April-2015
1       March-2015
2      August-2014
3     October-2014
4    February-2015
Name: month_year, dtype: object
```

Number of houses sold for every month_year:

```
April-2015        2231
July-2014         2211
June-2014         2180
August-2014       1940
October-2014      1878
March-2015        1875
September-2014    1774
May-2014          1768
December-2014     1471
November-2014     1411
February-2015     1250
January-2015       978
May-2015           646
Name: month_year, dtype: int64
```

We can observe that, most of the houses were sold in April - 2015 & July - 2014.

I have used groupby for month_year & price with aggregator as mean to find the mean of the houses sold for each month_year

```
month_year
April-2015        561933.463021
August-2014       536527.039691
December-2014     524602.893270
February-2015     507919.603200
January-2015      525963.251534
July-2014         544892.161013
June-2014         558123.736239
March-2015        544057.683200
May-2014          548166.600113
May-2015          558193.095975
November-2014     522058.861800
October-2014      539127.477636
September-2014    529315.868095
Name: price, dtype: float64
```

- This represents the timeline for sale data of houses is from May-2014 to May-2015.
- From the mean sale data, we can observe that, April month have the highest mean price.
- It represents that in April month highest sale of houses happened.

price: It's our TARGET feature, that we have to predict based on other features.

Descriptive summary of target variable:

```
count    2.161300e+04
mean     5.401822e+05
std      3.673622e+05
min      7.500000e+04
25%      3.219500e+05
50%      4.500000e+05
75%      6.450000e+05
max      7.700000e+06
Name: price, dtype: float64
```

Let's view the distribution of data using plot:

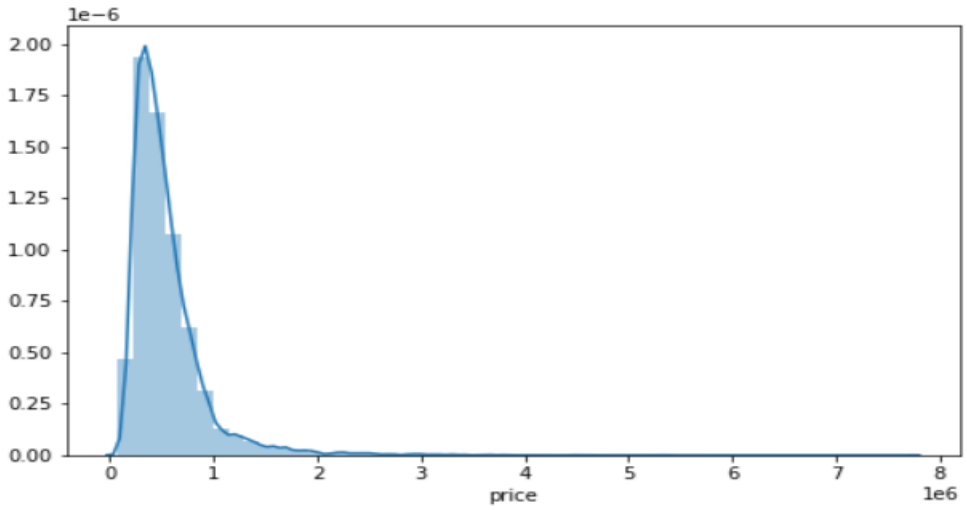


Figure 9 distribution plot for price(Target Variable)

Observations from the plot:

- The Price is ranging from 75,000 to 77,00,000.
- The distribution is right - skewed.

room_bed: Represents the number of bedrooms in a house

```
3.0     9888
4.0     6854
2.0     2747
5.0     1595
6.0      270
1.0      197
7.0       38
8.0       13
9.0        6
10.0       3
33.0       1
11.0       1
Name: room_bed, dtype: int64
```

Here, the value 33 seems to be an outlier. This need to be checked before treating it.

	cid	dayhours	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	condition	quality	ceil_measure	basement	yr_
16913	2402100895	2014-06-25	640000	33.0	1.75	1620.0	6000.0	1.0	0.0	0.0	5.0	7.0	1040.0	580.0	19

This is sure to be an outlier, because for a 33-bedroom property the price given is 640000 which is very low. This outlier needs to be treated.

Let's view the data using count plot:

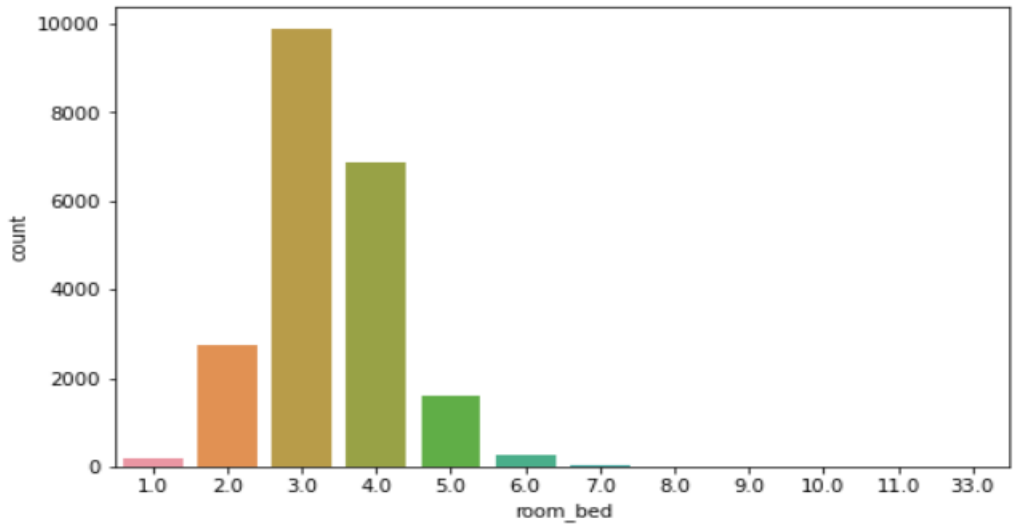


Figure 10 count plot - room_bed

Observations from the plot: Most of the properties/houses are of 3-bedroom & 4-bedroom type.

room_bath: Represents the number of bathrooms in a house

```
0.00      10
0.50       4
0.75      72
1.00    3829
1.25       9
1.50    1439
1.75    3031
2.00    1917
2.25    2147
2.50    5358
2.75    1178
3.00     750
3.25     588
3.50     726
3.75     155
4.00     135
4.25      78
4.50     100
4.75      23
5.00      21
5.25      13
5.50      10
5.75       4
6.00       6
6.25       2
6.50       2
6.75       2
7.50       1
7.75       1
8.00       2
Name: room_bath, dtype: int64
```

Let's view the data using count plot:

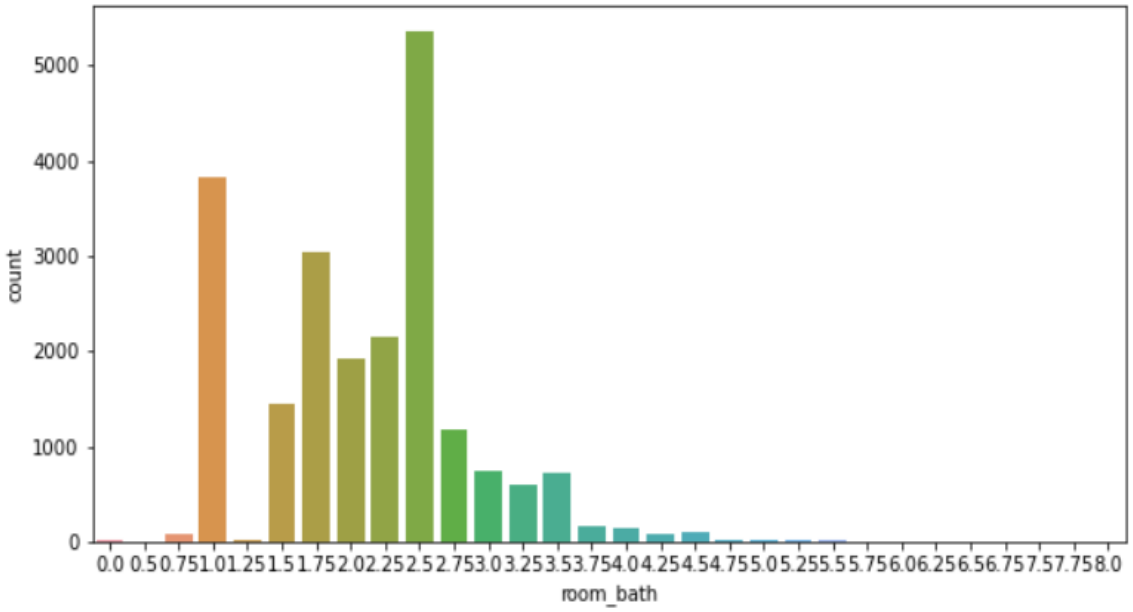


Figure 11 count plot - room_bath

Observations from the plot:

Majority of the properties/houses having bathroom in the range of 1.0 to 2.5

Skewness: 0.5102509663719975

Let’s view the distribution of data using plot:

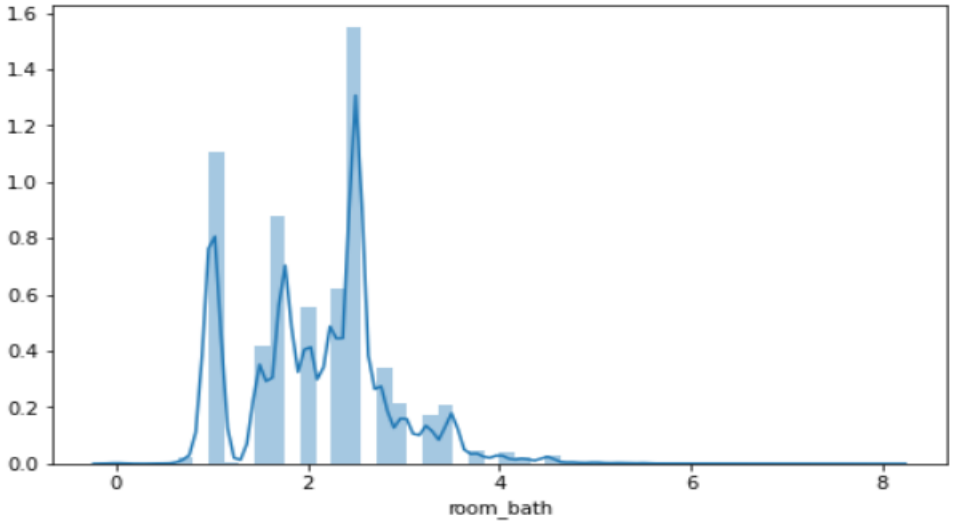


Figure 12 dist plot -room_bath

The distribution is slightly left skewed.

living_measure: square footage of house

```
Skewness : 1.4735169838222357

count    21613.000000
mean      2079.727155
std        918.147155
min        290.000000
25%       1430.000000
50%       1910.000000
75%       2550.000000
max      13540.000000
Name: living_measure, dtype: float64
```

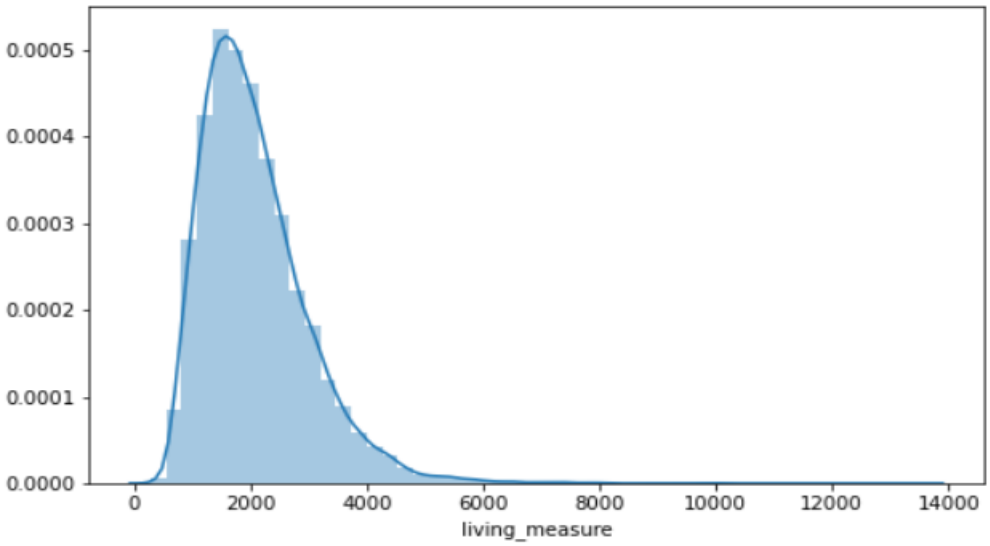


Figure 13 dist plot - living_measure

Observations from the plot:

Square footage of house ranges from 290 - 13,540 and Mean > Median. Data is skewed as visible from plot, and its distribution is normal & right skewed.

Let's check for the outliers in living_measure using boxplot:

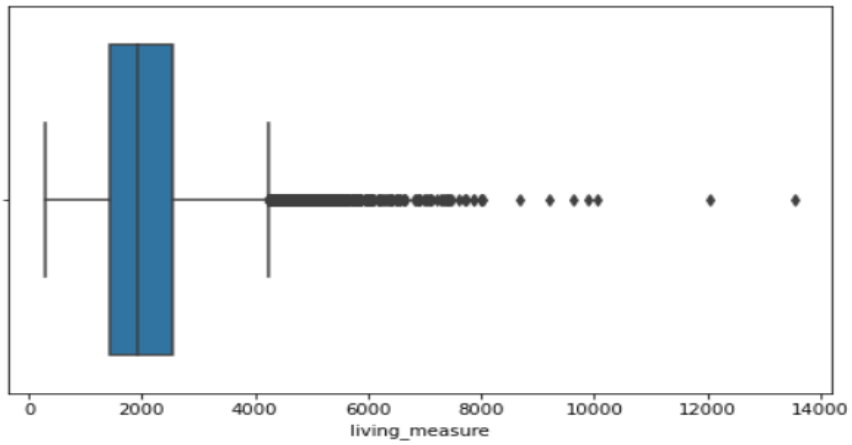


Figure 14 boxplot - living_measure

The feature 'living_measure' is having outliers. These outliers need to be further analyzed and treated.

checking the no. of data points with Living measure greater than 8000

	cid	dayhours	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	condition	quality	ceil_measure	basement
1068	6762700020	2014-10-13	7700000	6.0	8.00	12050.0	27600.0	2.5	0.0	3.0	4.0	13.0	8570.0	3480.0
1245	1924059029	2014-06-17	4670000	5.0	6.75	9640.0	13068.0	1.0	1.0	4.0	3.0	12.0	4820.0	4820.0
7928	1225069038	2014-05-05	2280000	7.0	8.00	13540.0	307752.0	3.0	0.0	4.0	3.0	12.0	9410.0	4130.0
10639	9208900037	2014-09-19	6890000	6.0	7.75	9890.0	31374.0	2.0	0.0	4.0	3.0	13.0	8860.0	1030.0
10718	9808700762	2014-06-11	7060000	5.0	4.50	10040.0	37325.0	2.0	1.0	2.0	3.0	11.0	7680.0	2360.0
12794	2470100110	2014-08-04	5570000	5.0	5.75	9200.0	35069.0	2.0	0.0	0.0	3.0	13.0	6200.0	3000.0
20038	1247600105	2014-10-20	5110000	5.0	5.25	8010.0	45517.0	2.0	1.0	4.0	3.0	12.0	5990.0	2020.0
20193	2303900035	2014-06-11	2890000	5.0	6.25	8670.0	64033.0	2.0	0.0	4.0	3.0	13.0	6120.0	2550.0
20746	6072800246	2014-07-02	3300000	5.0	6.25	8020.0	21738.0	2.0	0.0	0.0	3.0	11.0	8020.0	0.0

Table 6 Table showing no. of data points with Living measure greater than 8000

Observations:

- we can see that there are 9 houses which have more than 8000 living_measure.
- These can be considered as outliers and need to be treated.

lot_measure: square footage of lot

```
Skewness : 13.084880210575367

count    2.161300e+04
mean     1.509003e+04
std      4.138466e+04
min      5.200000e+02
25%      5.043000e+03
50%      7.618000e+03
75%      1.066000e+04
max      1.651359e+06
Name: lot_measure, dtype: float64
```

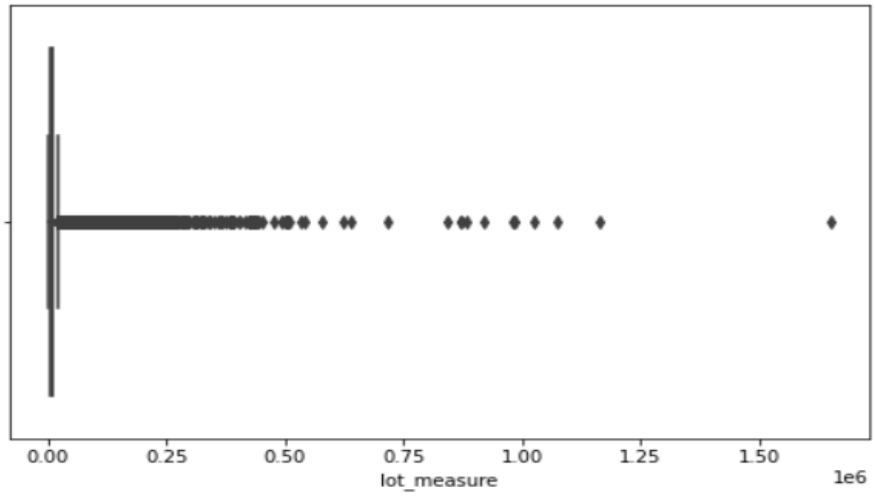


Figure 15 Boxplot - lot_measure

Observations:

- Square footage of lot ranges from 520 - 16,51,359.
- As Mean almost double the Median, it's Highly Right-Skewed.
- We can see that the data is skewed and has outliers as visible from plot above.

checking the no. of data points with Lot measure greater than 1250000

	cid	dayhours	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	condition	quality	ceil_measure	basement
11674	1020069017	2015-03-27	700000	4.0	1.0	1300.0	1651359.0	1.0	0.0	3.0	4.0	6.0	1300.0	0.0

we can observe that, we have only 1 record with more than 1250000. This is an outlier need to be treated further.

ceil: Total floors (levels) in house

```
1.0    10647
2.0     8210
1.5     1977
3.0      610
2.5      161
3.5        8
Name: ceil, dtype: int64
```

Most of the houses have 1 floor.

Let's view the data using count plot:

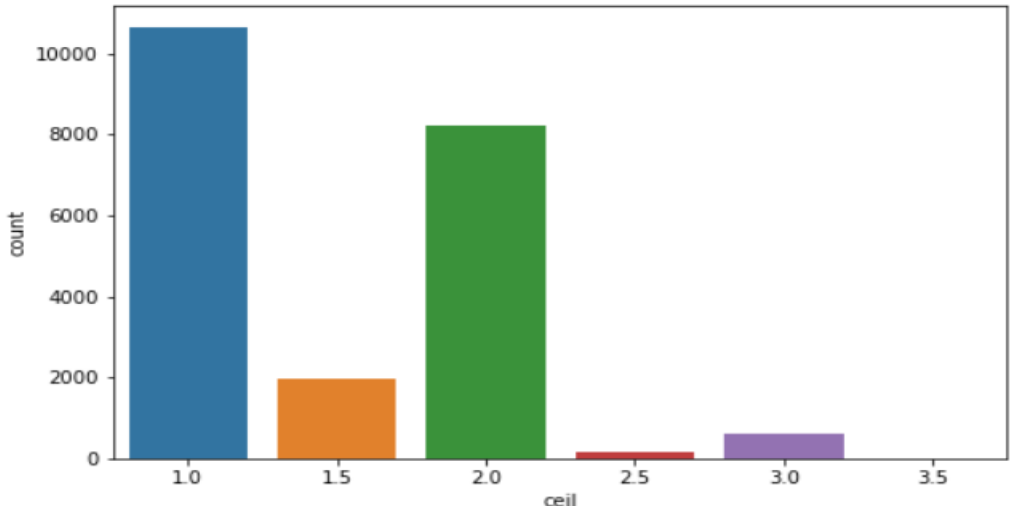


Figure 16 count plot – ceil

From the above plot, we can observe that 1 & 2 floored houses are more in the data.

coast: This is a categorical variable represents whether house has waterfront view or not.

```
0.0    21452
1.0      161
Name: coast, dtype: int64
```

We can observe that most of the houses do not have waterfront view.

sight: represents the number of times sight has been viewed.

```
0.0    19494
2.0      959
3.0      510
1.0      332
4.0      318
Name: sight, dtype: int64
```

We can observe that most of the houses has not been viewed. Here, 0 represents that the sight has not been viewed and the others are number of times the sight has been viewed.

condition: overall condition of the house (how good the house is). This is a categorical variable. It represents the ranking of the house based on condition

```
3.0    14063
4.0     5655
5.0     1694
2.0      171
1.0       30
Name: condition, dtype: int64
```

condition represents rating of house which ranges from 1 – 5.

Most of the houses are rated as 3 and above for its overall condition.

quality: Grade given to the housing unit based on grading system. quality represents the grade given to house which range from 1 - 13.

```
7.0     8982
8.0     6067
9.0     2615
6.0     2038
10.0    1134
11.0     399
5.0      242
12.0      90
4.0       29
13.0      13
3.0        3
1.0        1
Name: quality, dtype: int64
```

Let's view the data using count plot:

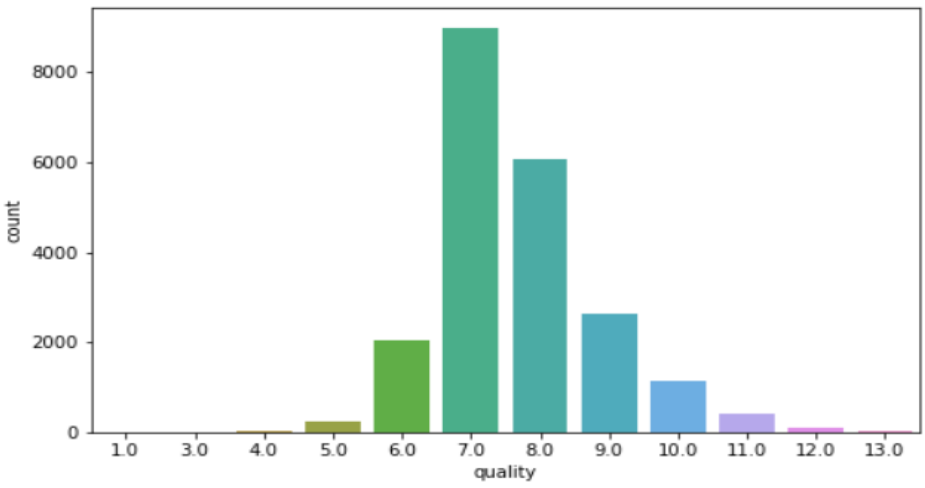


Figure 17 count plot – quality

Most of the properties/houses have quality rating between 6 to 10.

let's check the no. of data points with quality rating as 13

	cid	dayhours	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	condition	quality	ceil_measure	basement
388	853200010	2014-07-01	3800000	5.0	5.50	7050.0	42840.0	1.0	0.0	2.0	4.0	13.0	4320.0	2730.0
1068	6762700020	2014-10-13	7700000	6.0	8.00	12050.0	27600.0	2.5	0.0	3.0	4.0	13.0	8570.0	3480.0
3271	7237501190	2014-10-10	1780000	4.0	3.25	4890.0	13402.0	2.0	0.0	0.0	3.0	13.0	4890.0	0.0
3649	2426039123	2015-01-30	2420000	5.0	4.75	7880.0	24250.0	2.0	0.0	2.0	3.0	13.0	7880.0	0.0
4371	1725059316	2014-11-20	2390000	4.0	4.00	6330.0	13296.0	2.0	0.0	2.0	3.0	13.0	4900.0	1430.0
8420	9831200500	2015-03-04	2480000	5.0	3.75	6810.0	7500.0	2.5	0.0	0.0	3.0	13.0	6110.0	700.0
10639	9208900037	2014-09-19	6890000	6.0	7.75	9890.0	31374.0	2.0	0.0	4.0	3.0	13.0	8860.0	1030.0
10832	4139900180	2015-04-20	2340000	4.0	2.50	4500.0	35200.0	1.0	0.0	0.0	3.0	13.0	4500.0	0.0
11459	1068000375	2014-09-23	3200000	6.0	5.00	7100.0	18200.0	2.5	0.0	0.0	3.0	13.0	5240.0	1860.0
12794	2470100110	2014-08-04	5570000	5.0	5.75	9200.0	35069.0	2.0	0.0	0.0	3.0	13.0	6200.0	3000.0
16985	2303900100	2014-09-11	3800000	3.0	4.25	5510.0	35000.0	2.0	0.0	4.0	3.0	13.0	4910.0	600.0
20193	2303900035	2014-06-11	2890000	5.0	6.25	8670.0	64033.0	2.0	0.0	4.0	3.0	13.0	6120.0	2550.0
20547	3303850390	2014-12-12	2980000	5.0	5.50	7400.0	18898.0	2.0	0.0	3.0	3.0	13.0	6290.0	1110.0

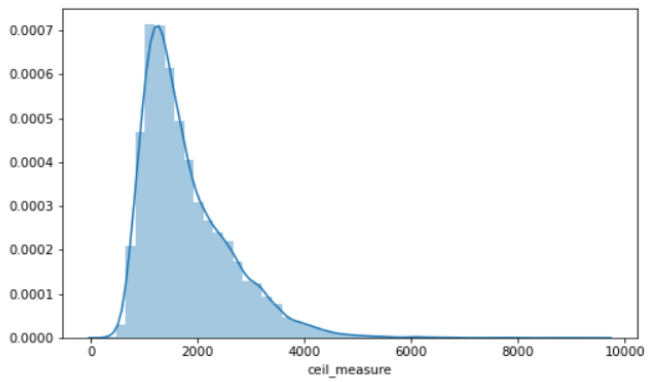
Figure 18 showing no. of data points with quality rating as 13

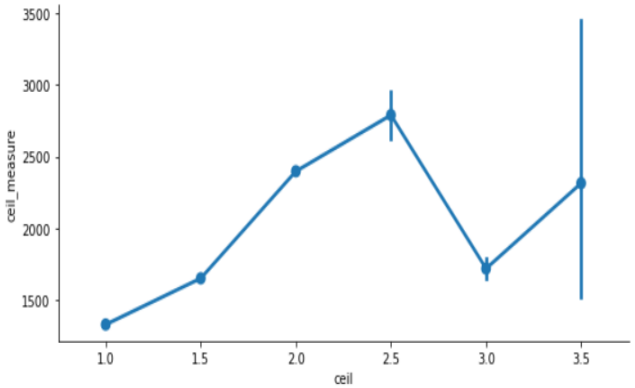
We can observe that, there are only 13 properties which have the highest quality rating.

ceil_measure: square footage of house apart from basement ranges in 290 - 9,410

```
Skewness : 1.4468098702392473

count    21613.000000
mean      1788.355989
std        828.084833
min        290.000000
25%       1190.000000
50%       1560.000000
75%       2210.000000
max       9410.000000
Name: ceil_measure, dtype: float64
```





As Mean > Median, ceil_measure is right skewed.

The vertical lines at each point represent the IQR (inter quartile range) of values at that point.

basement: square footage house basement ranges in 0 - 4,820.

Let's view the distribution of data using plot:

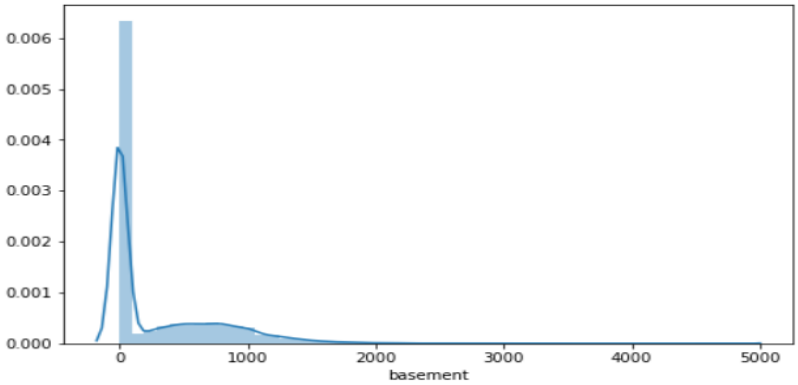


Figure 19 dist plot – basement

Here, we can see 2 gaussians, which tells us there are houses which do not have basements and some have the basements.

(13126, 24)

Houses have zero measure of basement means that they do not have basements. Around 60% of the houses do not have the basement.

plotting the boxplot for properties which have basements only:

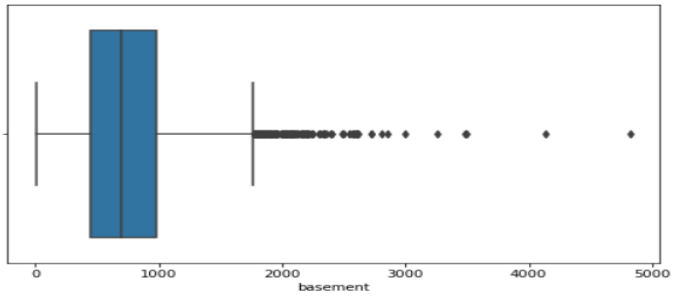


Figure 20 boxplot – basement

We can observe that there are outliers, these need to be treated.

Checking the no. of data points with 'basement' greater than 4000

	cid	dayhours	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	condition	quality	ceil_measure	basement	y
1245	1924059029	2014-06-17	4670000	5.0	6.75	9640.0	13068.0	1.0	1.0	4.0	3.0	12.0	4820.0	4820.0	
7928	1225069038	2014-05-05	2280000	7.0	8.00	13540.0	307752.0	3.0	0.0	4.0	3.0	12.0	9410.0	4130.0	

We have only 2 properties with more than 4,000 measure basement

Distribution of houses having basement:

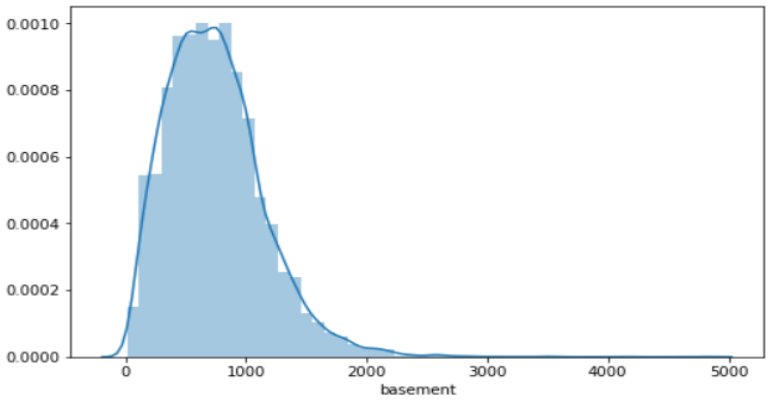


Figure 21 Distribution of houses having basement

The distribution is right skewed.

yr_built: This is a categorical variable which refers to the year house was built. House built year ranges from 1900 - 2015.

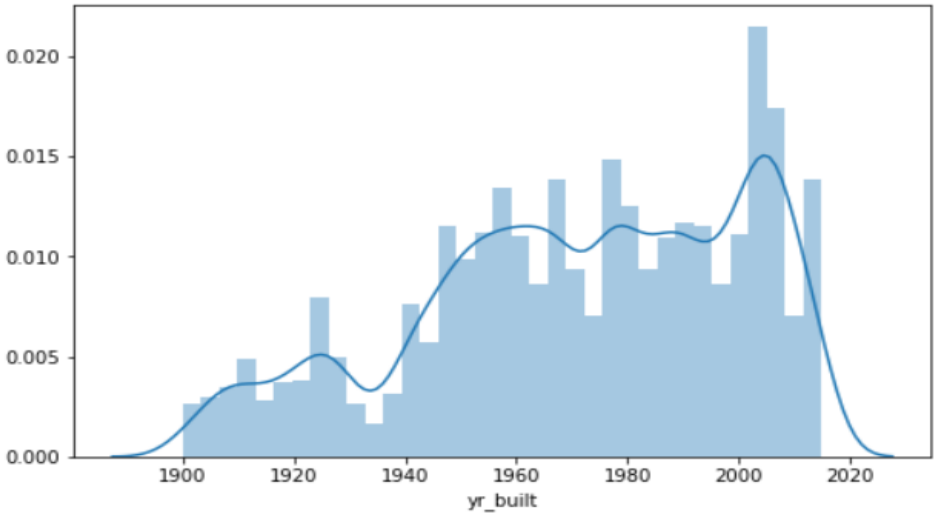


Figure 22 dist plot - yr_built

- The distribution is left skewed.
- we can observe that, the houses range from new to very old.
- The built year of the houses range from 1900 to 2014 and we can see the upward trend with time

yr_renovated: year when the house was renovated. House renovation year only 2015. Hence, this column can be used as the categorical column/feature.

```
(914, 24)
```

We can see that, only 914 houses were renovated out of 21613 houses

plot for the houses which are renovated:

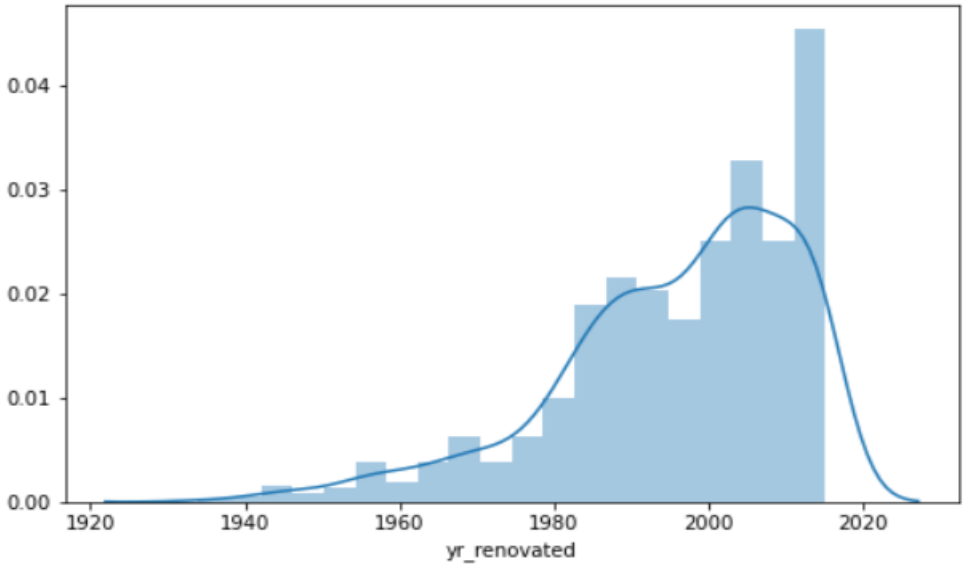


Figure 23 dist plot - ye_renovated

The distribution for the houses which are renovated is left skewed.

zip, lat, long: These three features represent the address/place where the houses are located.

zipcode: House zipcode ranges from 98001 - 98199.

lat: Latitude ranges from 47.1559 - 47.7776

long: Longitude ranges from -122.5190 to -121.315

for **zip code** I have used value_counts () function, then I observed that zipcode **-** 98103 having 603 hou ses. Which refers to greater number of houses sold in the same area.

From the Zip codes, I come to know that all this data is from USA (Seattle/Washington and other diffe rent states around USA)

Using **lat, long** ranges I have made a map in the Jupyter file. Which gives the complete understanding of the cities/areas, where the houses/properties are located, and which cities/states are having the hi ghest number of houses for sale.

This map is interactive, we can zoom in and zoom out to view the datapoints/ houses.

Here's a quick glance of the map, which was created using lat, long. I have taken the maximum range of lat & long using value_counts () to print this map.

We need to import the necessary libraries as well to print the map to the jupyter notebook.

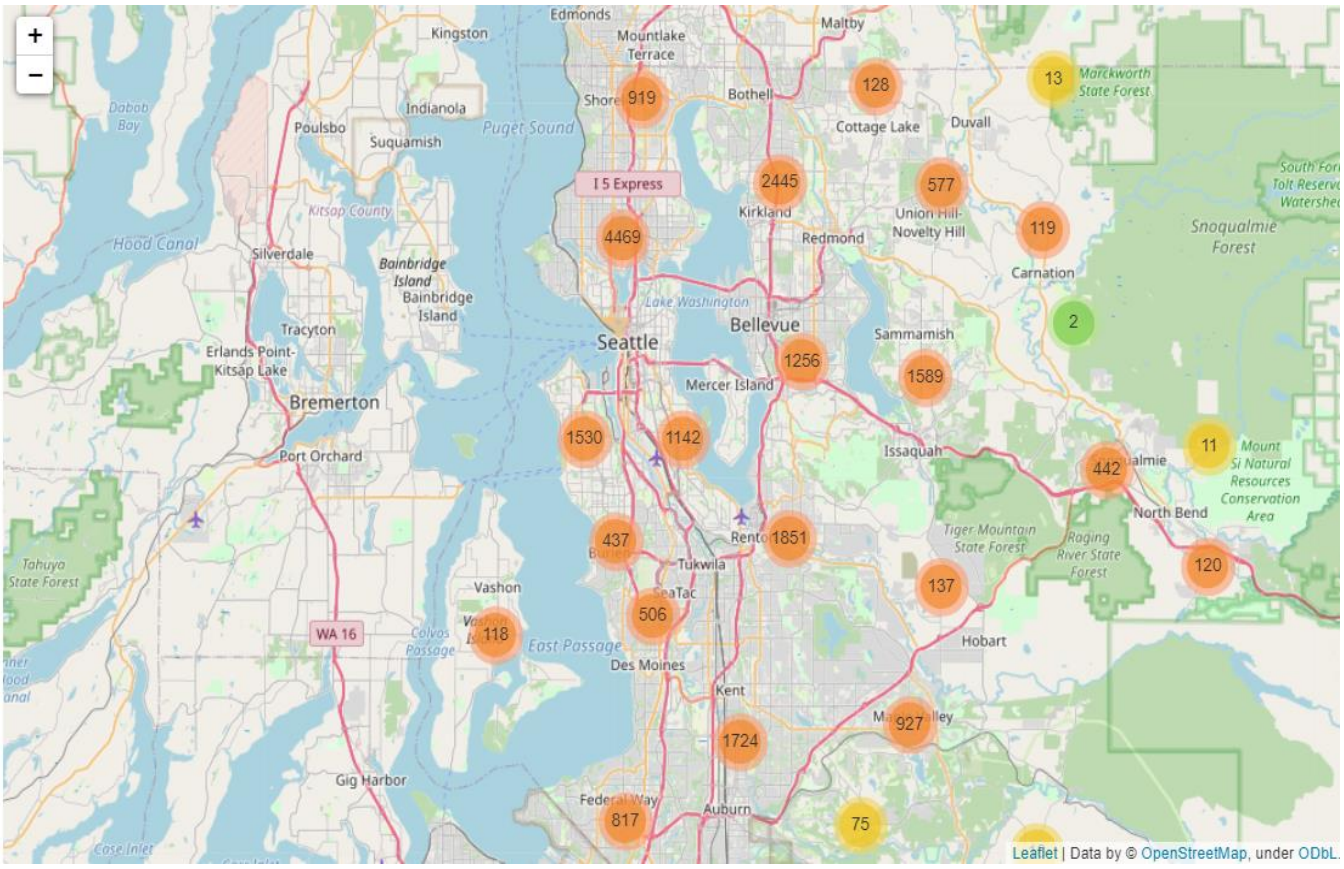


Figure 24 lat- long map

furnished: This is a categorical variable which refers to whether the house is furnished or not.

```
0.0    17367
1.0     4246
Name: furnished, dtype: int64
```

Most of the houses are not furnished.

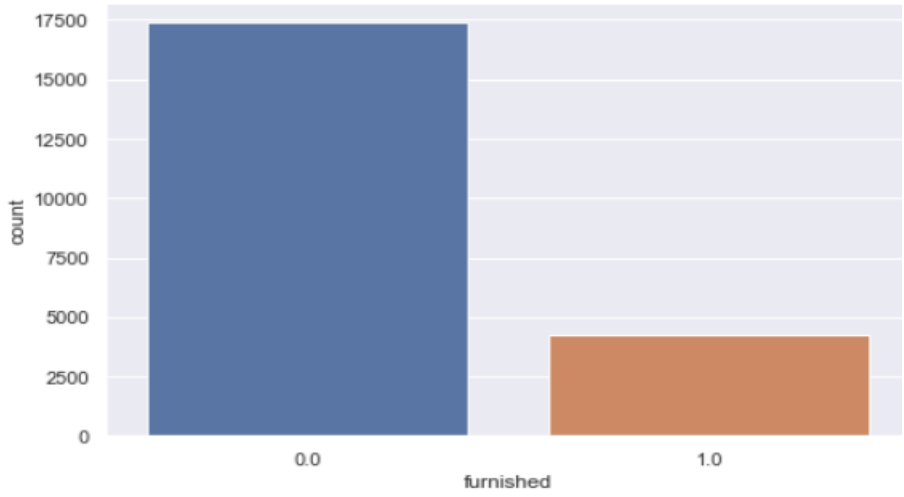


Figure 25 count plot - furnished

Bivariate Analysis:

Month_year vs price:

month_year	mean	median	size
April-2015	561933.463021	476500	2231
August-2014	536527.039691	442100	1940
December-2014	524602.893270	432500	1471
February-2015	507919.603200	425545	1250
January-2015	525963.251534	438500	978
July-2014	544892.161013	465000	2211
June-2014	558123.736239	465000	2180
March-2015	544057.683200	450000	1875
May-2014	548166.600113	465000	1768
May-2015	558193.095975	455000	646
November-2014	522058.861800	435000	1411
October-2014	539127.477636	446900	1878
September-2014	529315.868095	450000	1774

Table 7 group by month_year – price

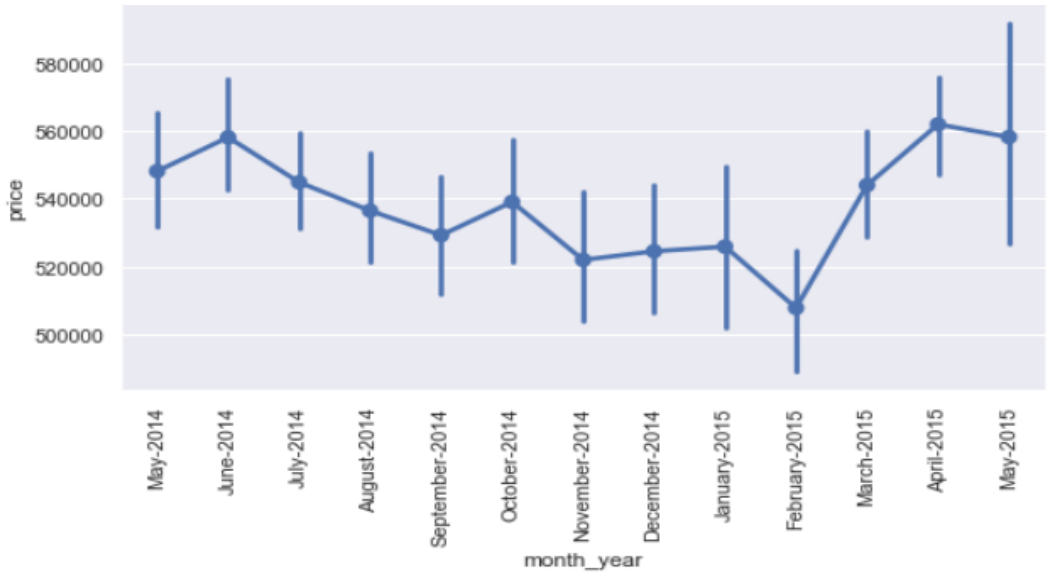


Table 8 factor plot month_year – price

- Month_year in which house is sold and price is not influenced by it.
- There are outliers and can be easily seen.
- The mean of the sale price of houses tends to be high during March, April & May compared to that of September, October, November, December period.

room_bed vs price:

room_bed	mean	median	size
1.0	3.189286e+05	299000.0	197
2.0	4.013572e+05	373500.0	2747
3.0	4.666820e+05	413682.5	9888
4.0	6.357284e+05	549950.0	6854
5.0	7.867329e+05	619000.0	1595
6.0	8.274895e+05	652500.0	270
7.0	9.514478e+05	728580.0	38
8.0	1.105077e+06	700000.0	13
9.0	8.939998e+05	817000.0	6
10.0	8.200000e+05	660000.0	3
11.0	5.200000e+05	520000.0	1
33.0	6.400000e+05	640000.0	1

Table 9 group by room_bed – price

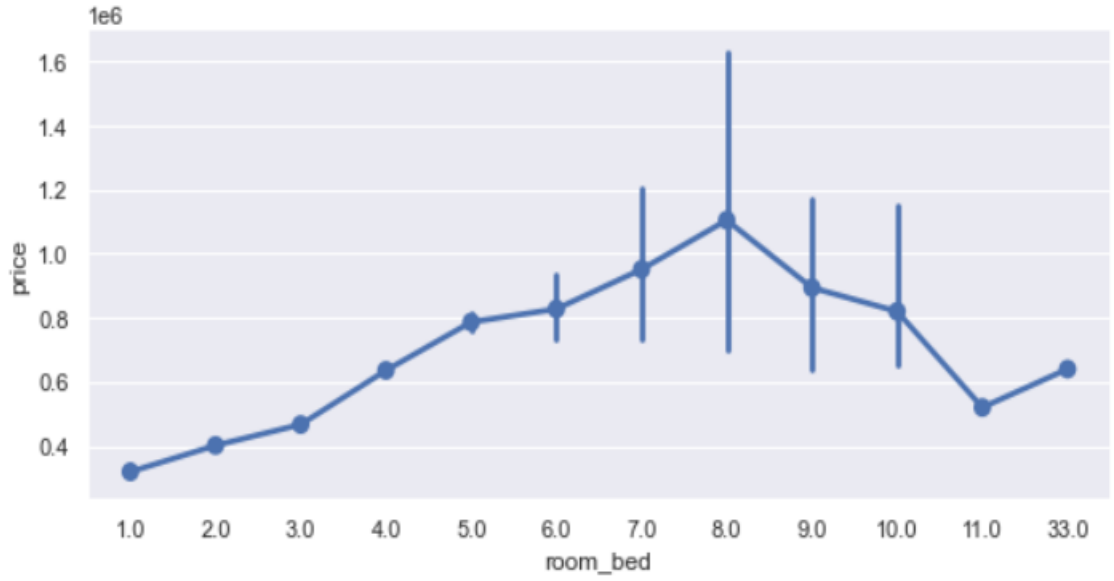


Figure 26 factor plot room_bed - price

- From the above analysis, outliers can be seen easily.
- Mean and median of price increases with number bedrooms per house up to a point and then drops
- we can observe a clear increasing trend in price with room_bed.

room_bath vs price:

room_bath	mean	median	size
0.00	4.490950e+05	317500	10
0.50	2.373750e+05	264000	4
0.75	2.945209e+05	273500	72
1.00	3.470889e+05	320000	3829
1.25	6.217722e+05	516500	9
1.50	4.091834e+05	370000	1439
1.75	4.549409e+05	422800	3031
2.00	4.575184e+05	422500	1917
2.25	5.325099e+05	470000	2147
2.50	5.536595e+05	499950	5358
2.75	6.605538e+05	605000	1178
3.00	7.087325e+05	600000	750
3.25	9.710299e+05	837352	588
3.50	9.329520e+05	823250	726
3.75	1.198179e+06	1070000	155
4.00	1.271616e+06	1060000	135
4.25	1.535072e+06	1390000	78
4.50	1.334211e+06	1060000	100
4.75	2.022300e+06	2300000	23
5.00	1.674167e+06	1430000	21
5.25	1.817962e+06	1420000	13
5.50	2.522500e+06	2340000	10
5.75	2.492500e+06	1930000	4
6.00	2.948333e+06	2895000	6
6.25	3.095000e+06	3095000	2
6.50	1.710000e+06	1710000	2
6.75	2.735000e+06	2735000	2
7.50	4.500000e+05	450000	1
7.75	6.890000e+06	6890000	1
8.00	4.990000e+06	4990000	2

Table 10 group by room_bath – price

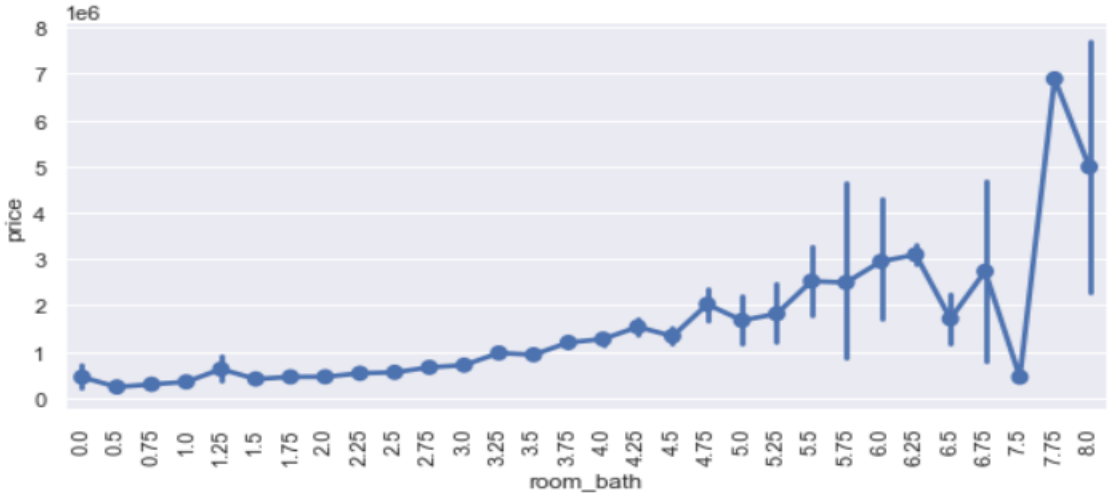


Figure 27 factor plot room_bath - price

- Outliers can be seen easily.
- Overall mean and median price increases with increasing room_bath.
- There is an upward trend in price with increase in room_bath.

living_measure vs price:

```
count    21613.000000
mean      2079.727155
std        918.147155
min        290.000000
25%       1430.000000
50%       1910.000000
75%       2550.000000
max      13540.000000
Name: living_measure, dtype: float64
```

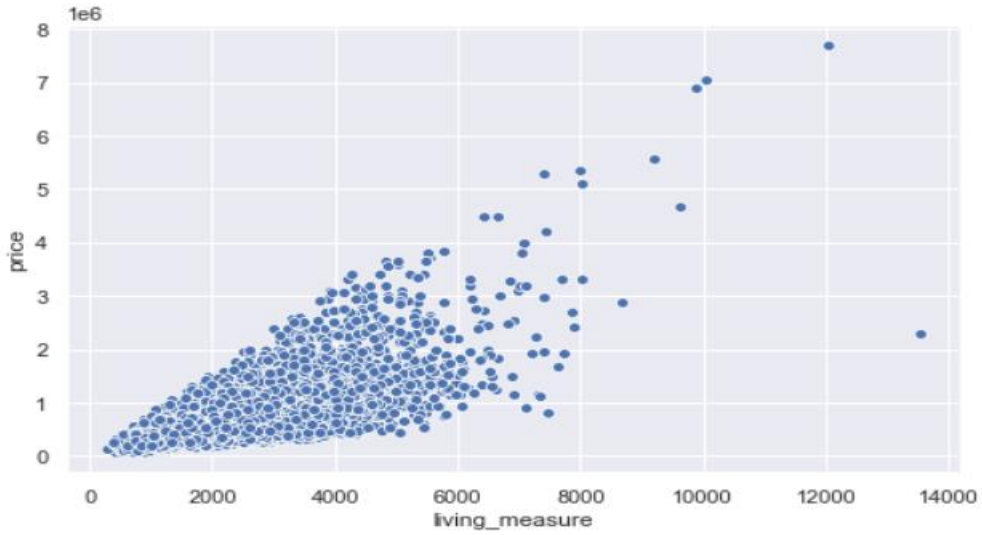


Figure 28 Scatter plot living_measere – price

- Price increases with increase in living measure.
- There is clear increment in the price of the house with increment in the living measure
- We can observe that there is one outlier to this trend. This need to be taken care of.

lot_measure vs price:

```
count    2.161300e+04
mean     1.509003e+04
std      4.138466e+04
min      5.200000e+02
25%      5.043000e+03
50%      7.618000e+03
75%      1.066000e+04
max      1.651359e+06
Name: lot_measure, dtype: float64
```

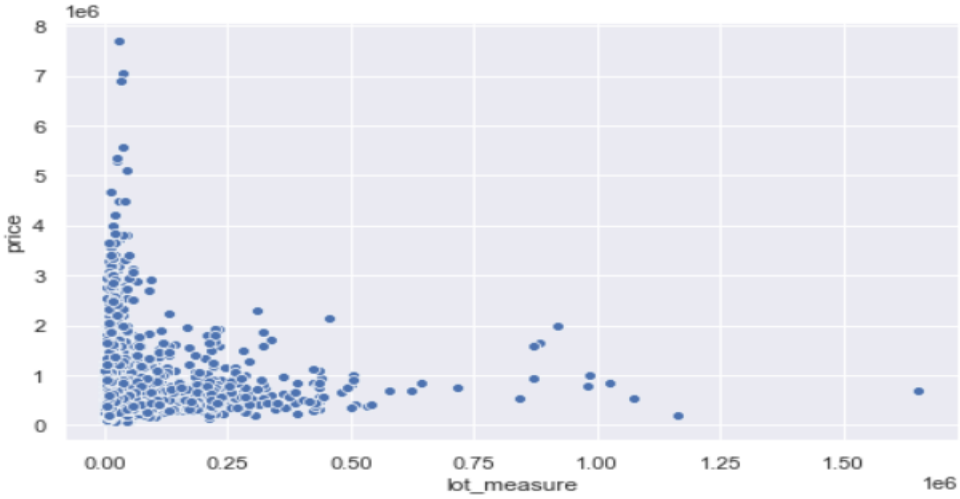


Figure 29 scatter plot lot_measure – price

data value range is very large. So, breaking it get better view, lot_measure <25000

```
count    19717.000000
mean      7760.594817
std       4245.511393
min        520.000000
25%       5000.000000
50%       7260.000000
75%       9612.000000
max      24969.000000
Name: lot_measure, dtype: float64
```

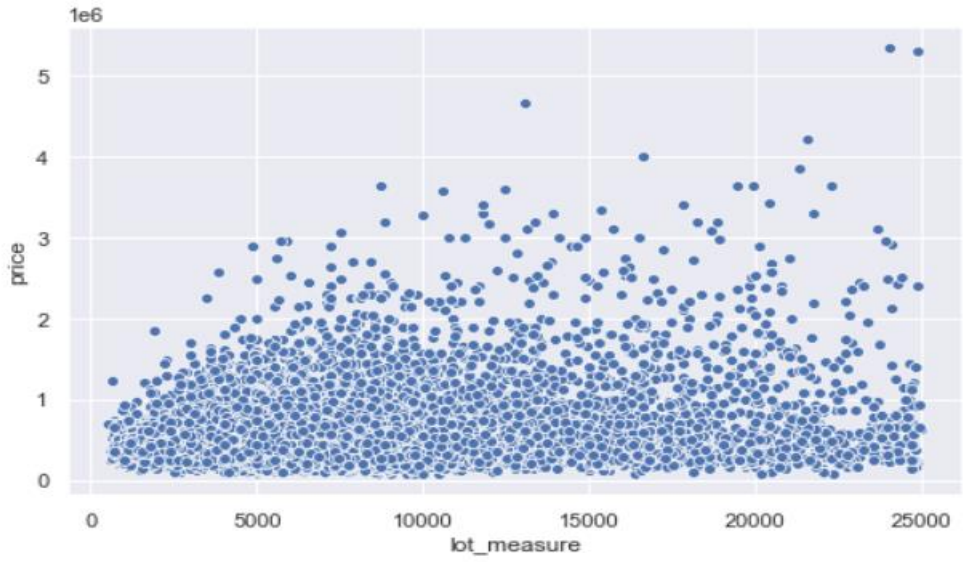


Figure 30 scatter plot lot_measure - price 2

- We can see that almost 95% of the houses have <25000 lot_measure.
- But there is no clear trend between lot_measure and price.

lot_measure >100000

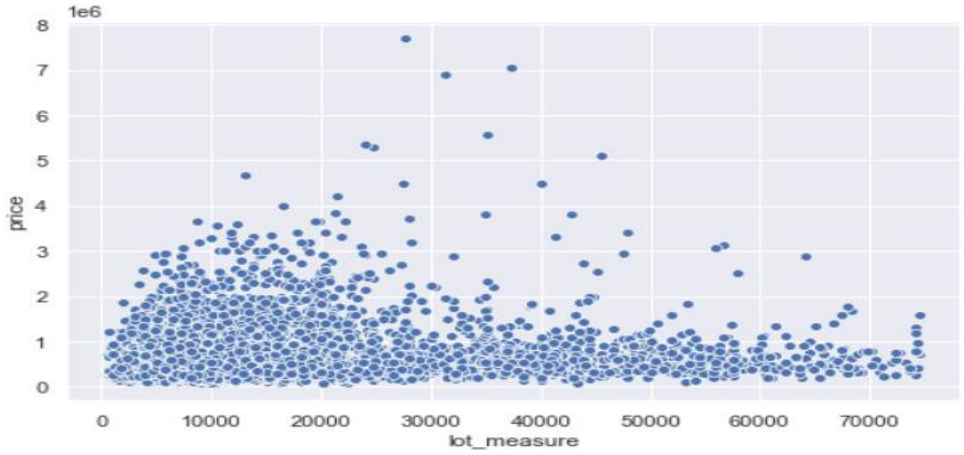


Figure 31 scatter plot lot_measure - price 3

Price increases with increase in lot measure.

ceil vs price:

ceil	mean	median	size
1.0	4.420451e+05	390000	10647
1.5	5.590374e+05	521000	1977
2.0	6.491210e+05	543250	8210
2.5	1.061021e+06	799200	161
3.0	5.831248e+05	490500	610
3.5	9.339375e+05	534500	8

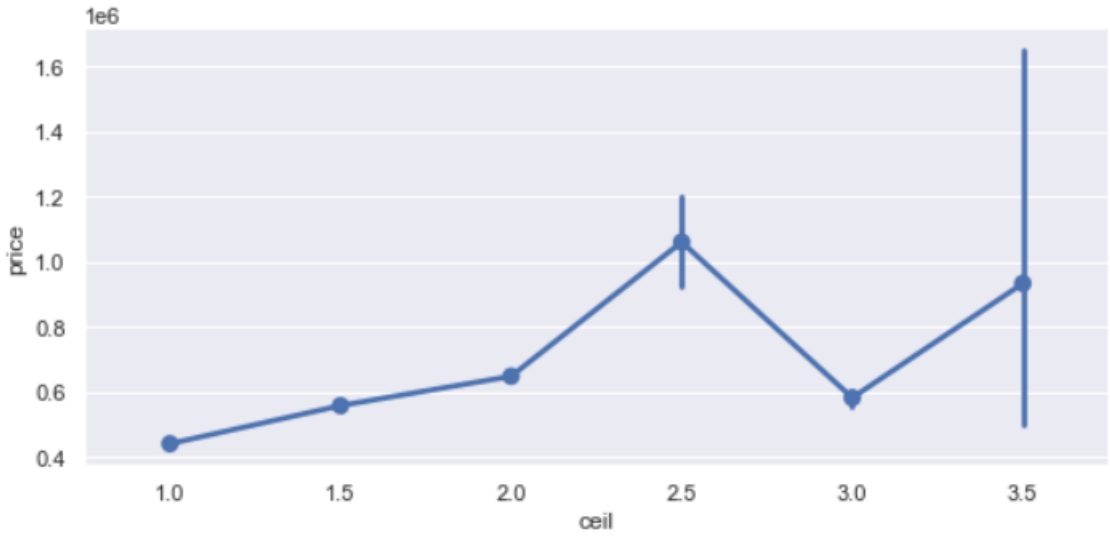


Figure 32 factor plot ceil - price

- Median price increases initially and then falls.
- There is a slight upward trend in price with ceil.

Coast vs price, living_measure:

	living_measure		price	
	median	mean	median	mean
coast				
0.0	1910.0	2071.571042	450000	5.317155e+05
1.0	2830.0	3166.465839	1410000	1.668301e+06

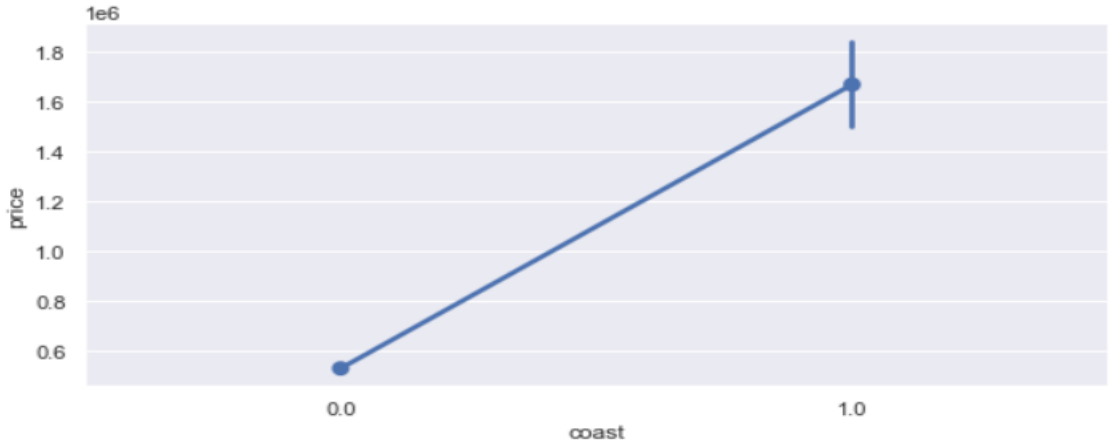


Figure 33 factor plot coast – price

- The mean and median of waterfront view is high. However, such houses would be very small compared to non-waterfront houses.
- living_measure mean, and median is greater for the waterfront houses.
- The houses with waterfront view are expected to have higher price compared to that of non-waterfront view houses.

Sight vs price, living_measure:

sight - have outliers. The house sighted more have high price (mean and median) and have large living area as well.

	price			living_measure		
	mean	median	size	mean	median	size
sight						
0.0	4.967417e+05	432500	19494	1997.843285	1850.0	19494
1.0	8.125186e+05	690944	332	2568.960843	2420.0	332
2.0	7.918609e+05	675000	959	2652.881126	2450.0	959
3.0	9.724684e+05	802500	510	3018.564706	2840.0	510
4.0	1.466554e+06	1190000	318	3354.433962	3070.0	318

Table 11 group by sight - price, living_measure

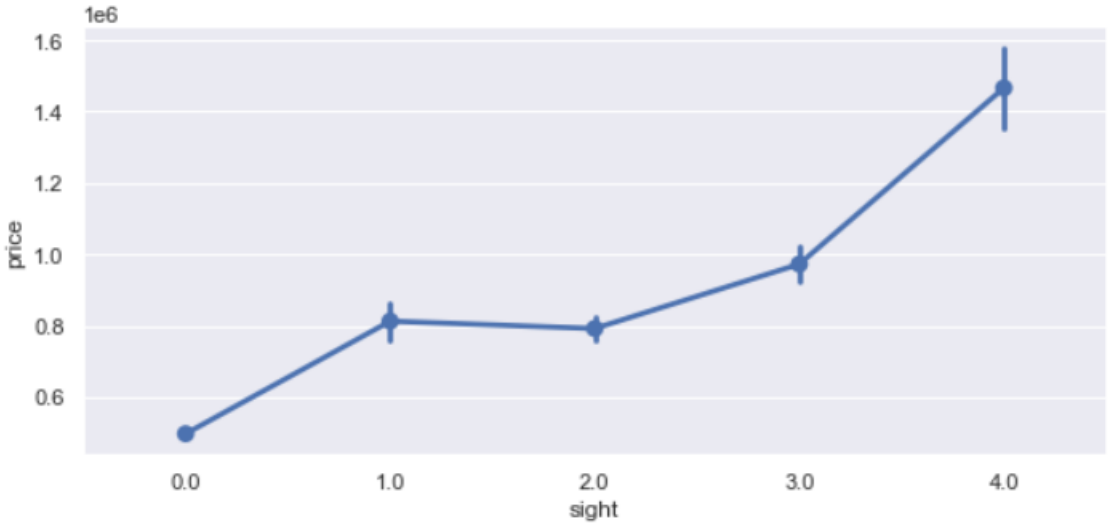


Figure 34 factor plot sight – price

- The feature sight is having outliers.
- The houses that are viewed more have high price (mean and median) and having large living area.
- Houses with high price have been viewed more compared to that of houses with low price.

In relation with price and living_measure:

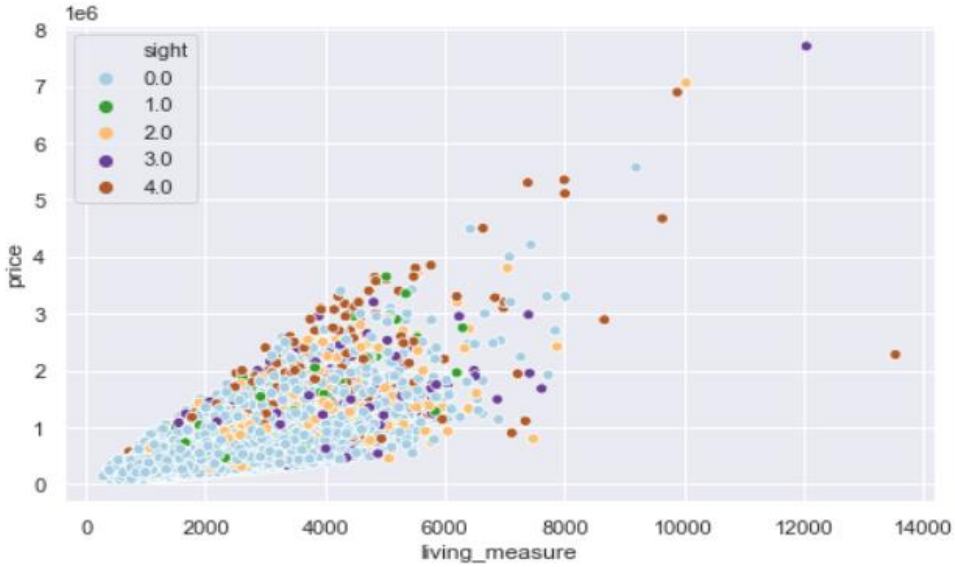


Figure 35 relation with price and living_measure

- Houses with high price with large living area are viewed more.
- Hence, Houses with higher price have a greater number of times viewed compared to that of houses with lower price.

Condition vs price, living_measure:

	price			living_measure		
	mean	median	size	mean	median	size
condition						
1.0	334431.666667	262500	30	1216.000000	1000.0	30
2.0	326423.327485	279000	171	1414.152047	1330.0	171
3.0	542364.148048	451000	14063	2148.572780	1970.0	14063
4.0	520643.219275	440000	5655	1950.352255	1820.0	5655
5.0	612515.489965	525444	1694	2022.563164	1880.0	1694

Table 12 group by condition, price, living_measure

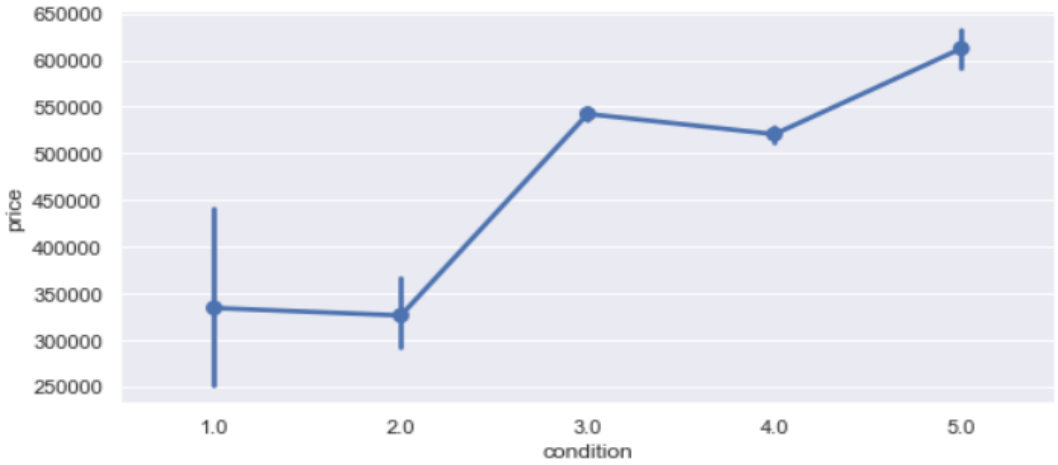


Figure 36 Factor plot condition – price

As the condition rating increases its price and living measure mean and median also increases.

quality vs price, living_measure:

	price			living_measure		
	mean	median	size	mean	median	size
quality						
1.0	1.420000e+05	142000.0	1	290.000000	290.0	1
3.0	2.056667e+05	262000.0	3	596.666667	600.0	3
4.0	2.143810e+05	205000.0	29	660.482759	660.0	29
5.0	2.485240e+05	228700.0	242	983.326446	905.0	242
6.0	3.019166e+05	275276.5	2038	1192.096173	1120.0	2038
7.0	4.025850e+05	375000.0	8982	1689.795703	1630.0	8982
8.0	5.429310e+05	510000.0	6067	2184.126257	2150.0	6067
9.0	7.737382e+05	720000.0	2615	2866.572084	2820.0	2615
10.0	1.072347e+06	914327.0	1134	3520.299824	3450.0	1134
11.0	1.497792e+06	1280000.0	399	4395.448622	4260.0	399
12.0	2.192500e+06	1820000.0	90	5471.588889	4965.0	90
13.0	3.710769e+06	2980000.0	13	7483.076923	7100.0	13

Table 13 group by quality - price, living_measure

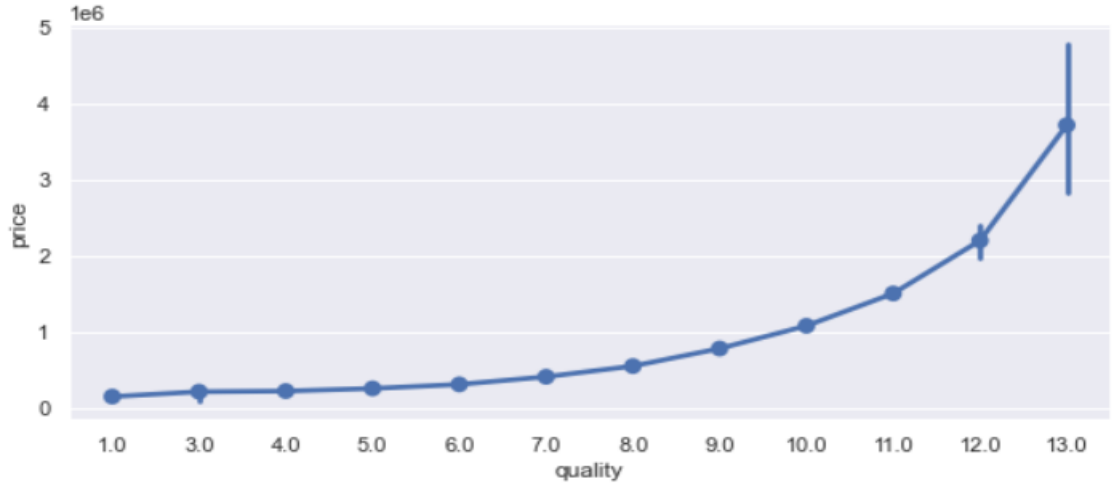


Figure 37 factor plot quality – price

- As grade increases price and living_measure also increased (mean and median).
- when viewed in relation with price and living_measure. Most houses are graded as 6 or more.

ceil_measure vs price:

```
count    21613.000000
mean      1788.355989
std        828.084833
min        290.000000
25%       1190.000000
50%       1560.000000
75%       2210.000000
max       9410.000000
Name: ceil_measure, dtype: float64
```

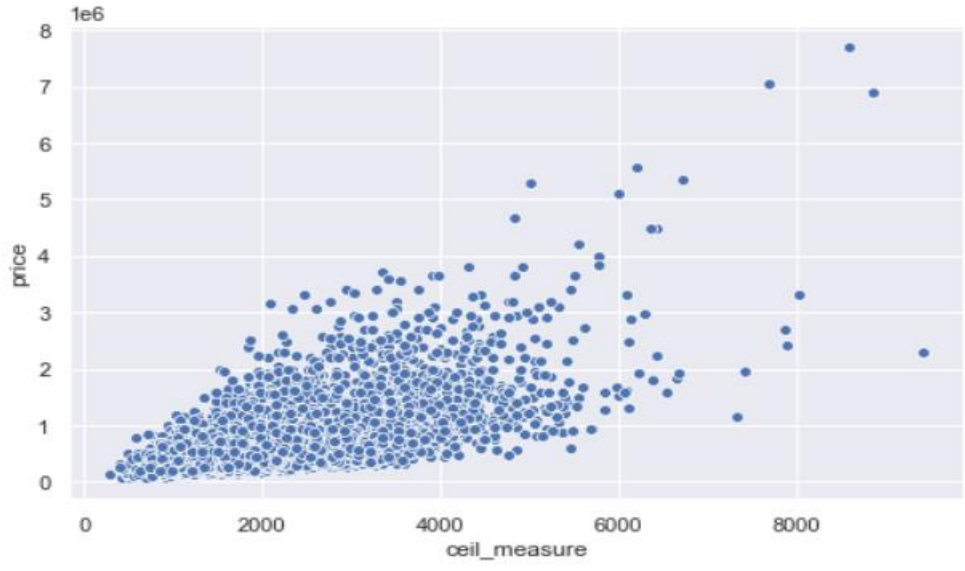


Figure 38 scatter plot ceil_measure - price

- price increases with increase in ceil measure.
- We can observe that, there is an upward trend in price with ceil_measure.

Basement vs price:

We will create the categorical variable for basement 'has_basement' for houses with basement and no basement. This categorical variable will be used for further analysis.

Binning Basement to analyze data. after binning, we can see data shows with basement houses are costlier and have higher living measure (mean & median).

	price			living_measure		
	mean	median	size	mean	median	size
has_basement						
No	486945.394789	411500	13126	1928.891818	1740.0	13126
Yes	622518.174384	515000	8487	2313.009191	2100.0	8487

Table 14 group by has_basement - price, living_measure

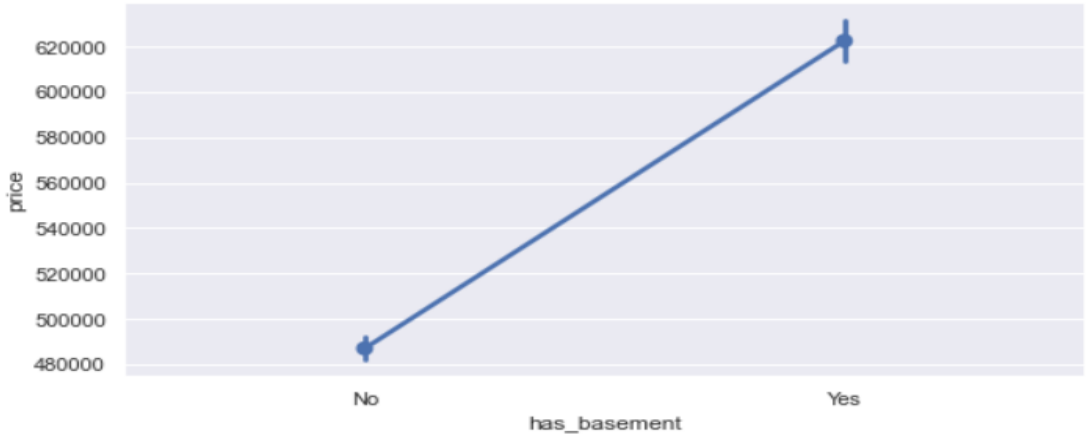


Figure 39 factor plot has_basement - price

basement - have higher price & living measure:

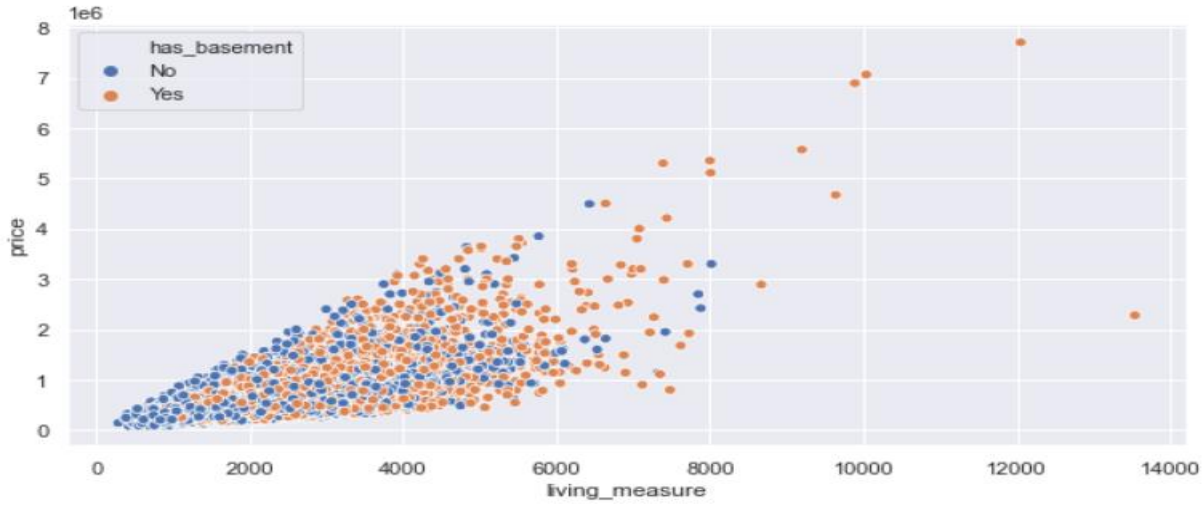


Figure 40 Scatterplot has_basement - price, living_measure

We can observe that, basement houses are costlier and have higher living measure (mean & median).

Yr_built: House built year ranges from 1900 - 2015

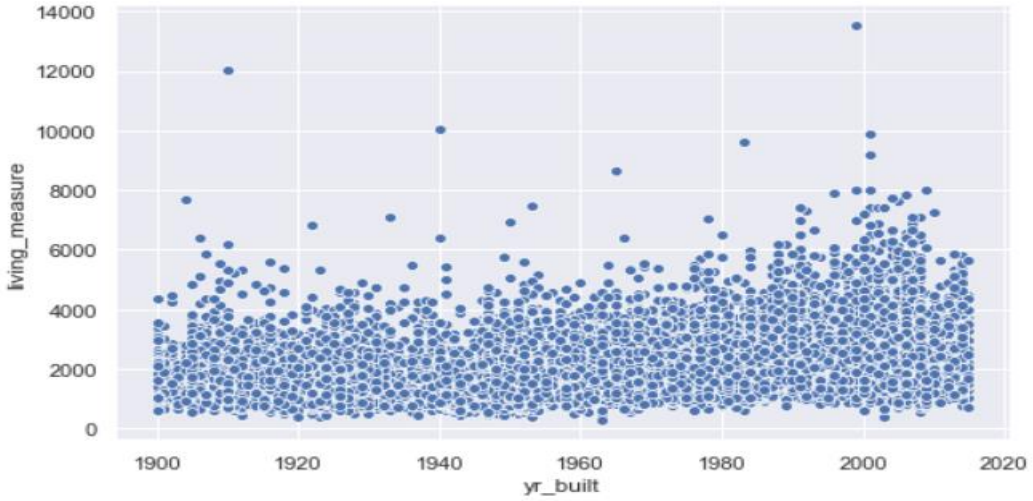


Figure 41 scatter plot yr_built - living_measure

We will create new variable: HouseLandRatio - This is proportion of living area in the total area of the house. We will explore the trend of price against this HouseLandRatio.

computing new variable as ratio of living_measure/total_area refers to - Land used for construction of house.

```
12235    16.0
14791    35.0
1742     17.0
17829    27.0
14810    17.0
Name: HouseLandRatio, dtype: float64
```

yr_renovated:

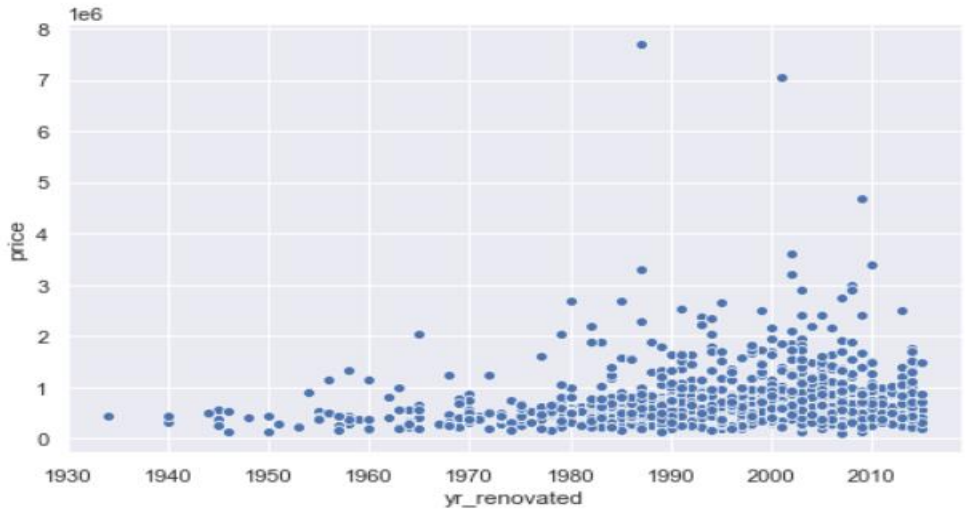


Figure 42 scatter plot yr_renovated - price

- Most houses are renovated after 1980's.
- We will create new categorical variable 'has_renovated' to categorize the property as renovated and non-renovated. we will use this categorical variable for further analysis.

Let's try to group yr_renovated - Binning Basement to analyse data

	price			HouseLandRatio		
	mean	median	size	mean	median	size
has_renovated						
No	530447.958597	448000	20699	22.069424	20.0	20699
Yes	760628.777899	600000	914	22.271335	21.0	914

Table 15 group by has_renovated - price, HouseLandRatio

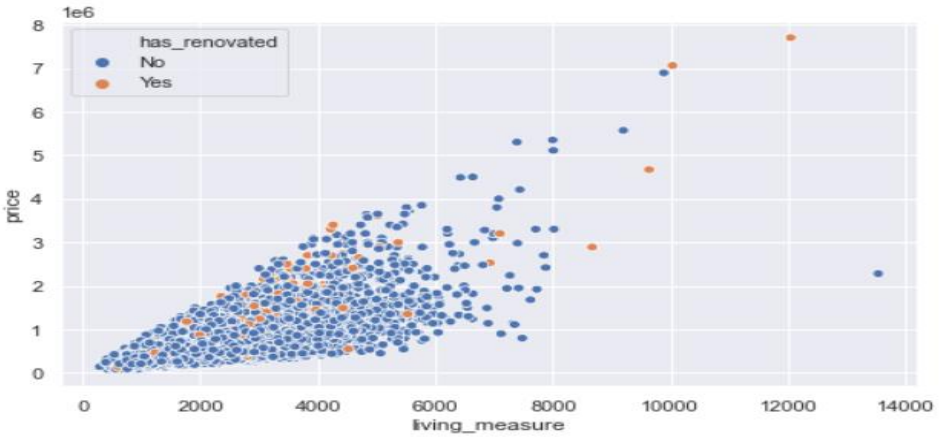


Figure 43 scatter plot living measure – price

- **has_renovated** - renovated houses have higher mean and median, however it does not confirm if the prices of house renovated actually increased or not.
- **HouseLandRatio** - Renovated house utilized more land area for construction of house.
- Renovated properties have higher price than others with same living measure space.

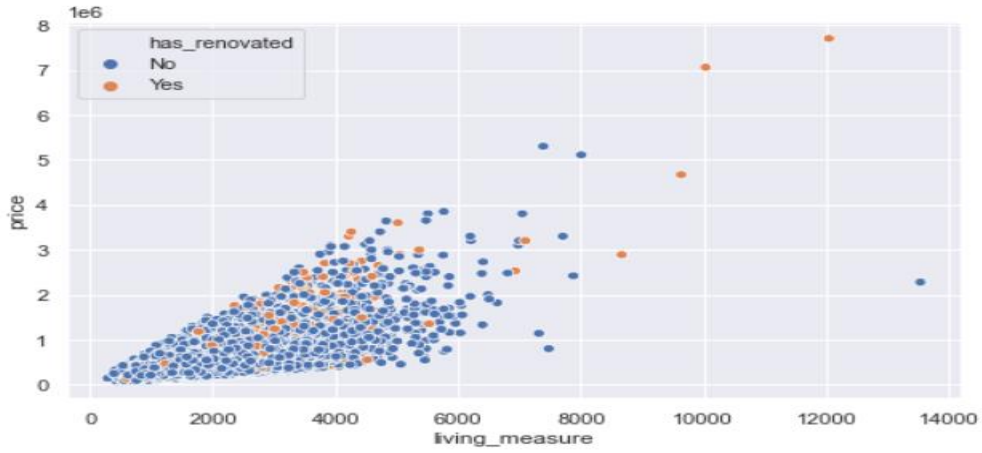


Figure 44 scatter plot has_renovated - price, living_measure

The variable 'has_renovated' have higher price and living measure.

Furnished vs price, living_measure, HouseLandRatio:

	price			living_measure			HouseLandRatio		
	mean	median	size	mean	median	size	mean	median	size
furnished									
0.0	437403.973628	401000	17367	1792.618184	1720.0	17367	21.506708	19.0	17367
1.0	960565.753179	810000	4246	3254.060999	3110.0	4246	24.414508	24.0	4246

Table 16 group by furnished - price, living_measure, HouseLandRatio

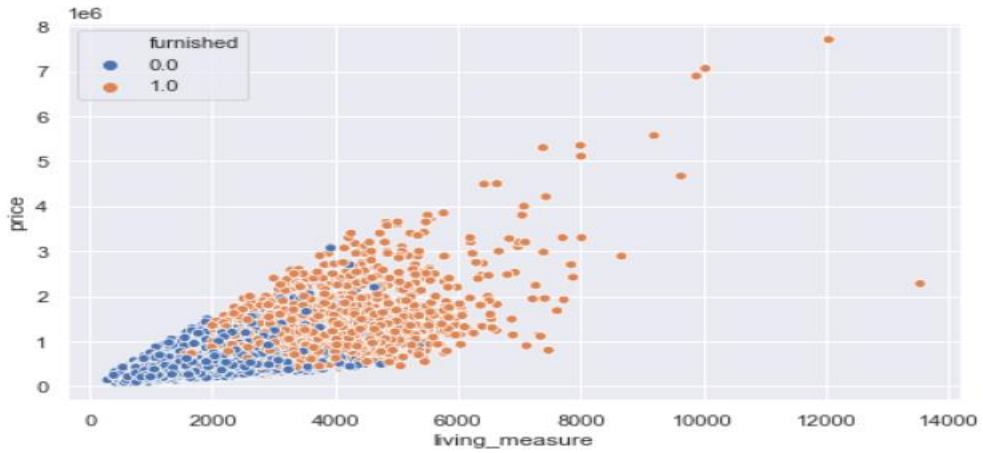


Figure 45 scatter plot furnished - price, living_measure

- Furnished houses have higher price value and has greater living_measure.
- Furnished houses have higher price than that of the non-furnished houses.

Correlation Matrix: correlation between the different features.

	cid	price	room_bed	room_bath	living_measure	lot_measure	ceil	sight	ceil_measure	basement	zipcode
cid	1.000000	-0.016797	0.001533	0.005768	-0.012151	-0.131789	0.018683	0.011776	-0.010812	-0.005151	-0.008224
price	-0.016797	1.000000	0.308288	0.524509	0.702066	0.089676	0.256826	0.396999	0.605594	0.323837	-0.053168
room_bed	0.001533	0.308288	1.000000	0.513940	0.577253	0.032560	0.176616	0.078712	0.478670	0.301800	-0.153567
room_bath	0.005768	0.524509	0.513940	1.000000	0.753265	0.087675	0.497953	0.186137	0.684131	0.282696	-0.203568
living_measure	-0.012151	0.702066	0.577253	0.753265	1.000000	0.172728	0.353502	0.284366	0.876342	0.434775	-0.199588
lot_measure	-0.131789	0.089676	0.032560	0.087675	0.172728	1.000000	-0.005368	0.074982	0.183413	0.015364	-0.129243
ceil	0.018683	0.256826	0.176616	0.497953	0.353502	-0.005368	1.000000	0.029612	0.523100	-0.244919	-0.058706
sight	0.011776	0.396999	0.078712	0.186137	0.284366	0.074982	0.029612	1.000000	0.167393	0.277243	0.084386
ceil_measure	-0.010812	0.605594	0.478670	0.684131	0.876342	0.183413	0.523100	0.167393	1.000000	-0.051916	-0.261155
basement	-0.005151	0.323837	0.301800	0.282696	0.434775	0.015364	-0.244919	0.277243	-0.051916	1.000000	0.074845
zipcode	-0.008224	-0.053168	-0.153567	-0.203568	-0.199588	-0.129243	-0.058706	0.084386	-0.261155	0.074845	1.000000
lat	-0.001891	0.306919	-0.010132	0.024578	0.052722	-0.085940	0.049876	0.006001	-0.000770	0.110538	0.267048
long	0.020885	0.021532	0.131071	0.222169	0.240062	0.228884	0.125163	-0.078051	0.343535	-0.144547	-0.563556
living_measure15	-0.003126	0.583825	0.391685	0.566262	0.753594	0.145003	0.277874	0.280910	0.729391	0.199378	-0.278340
lot_measure15	-0.138699	0.082597	0.030483	0.086769	0.183484	0.716048	-0.010868	0.072979	0.194175	0.017514	-0.147423
total_area	-0.131638	0.104849	0.045029	0.103866	0.194168	0.998169	0.002802	0.080998	0.202049	0.025003	-0.133481
HouseLandRatio	0.112790	0.155778	0.087234	0.322582	0.132455	-0.340407	0.546511	0.006500	0.094396	0.097298	0.183230

Table 17 Correlation Matrix

Pairplot:

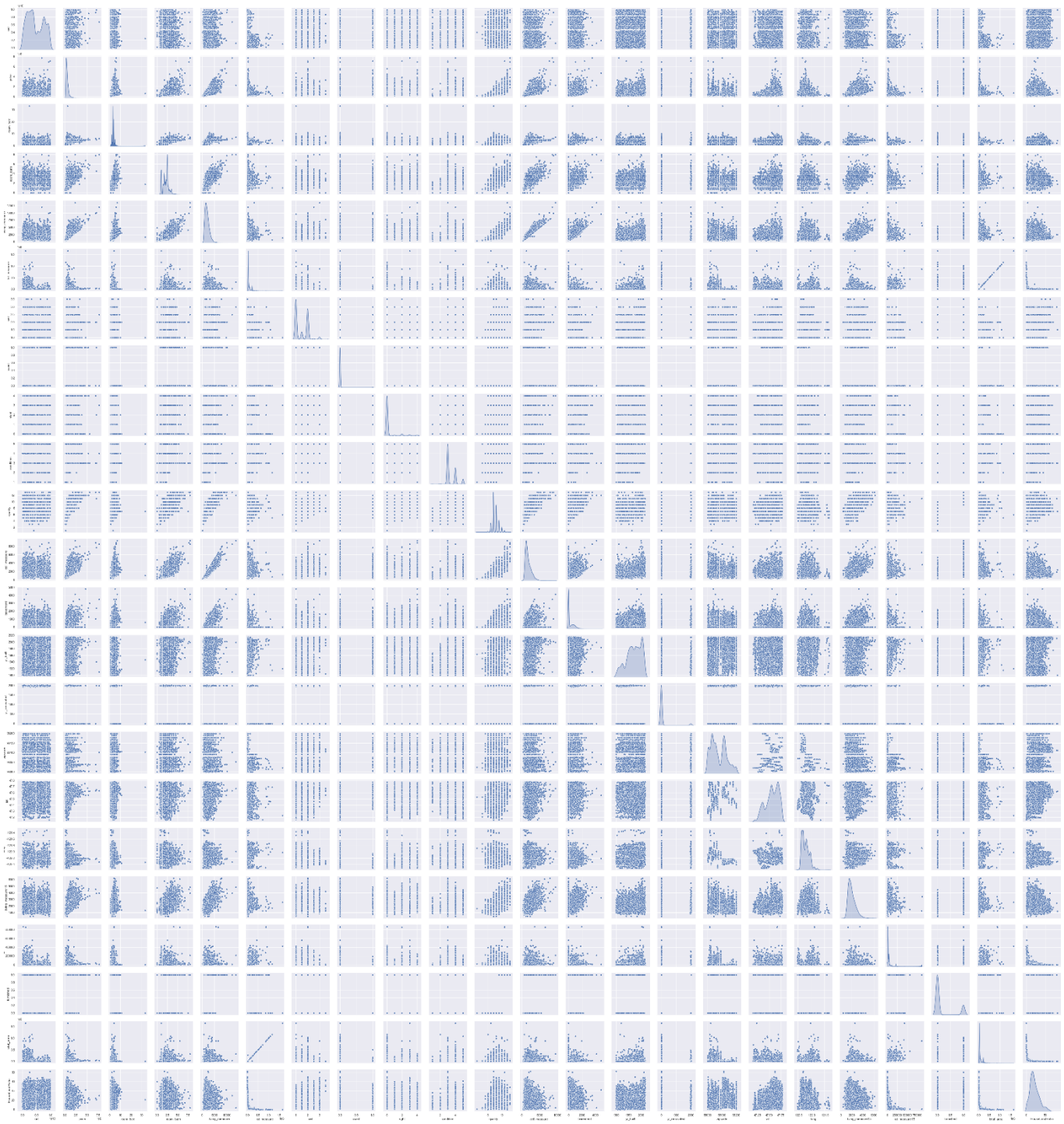


Figure 46 Pairplot

Observations from pairplot:

- **price:** distribution for price is Right-Skewed as we observed earlier from data description.
- **room_bed:** the target variable (price) and room_bed plot is not linear. Its distribution has lot of gaussians.

- **room_bath:** The plot for room_bath with price has somewhat linear relationship. Distribution has number of gaussians.
- **living_measure:** The Plot against price has strong linear relationship. It also has linear relationship with room_bath variable. Distribution is Right-Skewed.
- **lot_measure:** No clear relationship with price.
- **ceil:** No clear relationship with price. We can see that, it has 6 unique values**.** Therefore, we can convert this column into categorical column for values**.**
- **coast:** No clear relationship with price. Clearly, it's categorical variable with 2 unique values**.**
- **sight:** No clear relationship with price. This has 5 unique values**.** This can be converted to Categorical variable**.**
- **condition:** No clear relationship with price. This has 5 unique values**.** This can be converted to Categorical variable**.**
- **quality:** Somewhat linear relationship with price. It has discrete values from 1 - 13. This is a Categorical variable.
- **ceil_measure:** Strong linear relationship with price. Also, with room_bath and living_measure features. Distribution is Right Skewed.
- **basement:** No clear relationship with price.
- **yr_built**: No clear relationship with price.
- **yr_renovated:** No clear relationship with price. It has 2 unique values. which tells whether the house is renovated or not.
- **zipcode, lat, long**: No clear relationship with price or any other feature.
- **living_measure15:** Somewhat linear relationship with target feature. It's same as living_measure.
- **lot_measure15:** No clear relationship with price or any other feature.
- **furnished:** No clear relationship with price or any other feature.
- **total_area:** No clear relationship with price. But it has Very Strong linear relationship with lot_measure.

From the above correlation Matrix: (placed above the pairplot)

We have linear relationships in below featues from above correlation matrix

1. **price**: room_bath, living_measure, quality, living_measure15, furnished
2. **living_measure**: price, room_bath. Hence, we can consider dropping 'room_bath' variable. Or can be used in the analysis
3. **quality**: price, room_bath, living_measure
4. **ceil_measure**: price, room_bath, living_measure, quality
5. **living_measure15**: price, living_measure, quality.
6. **lot_measure15**: lot_measure.
7. **furnished**: quality
8. **total_area**: lot_measure, lot_measure15.

Heat Map:

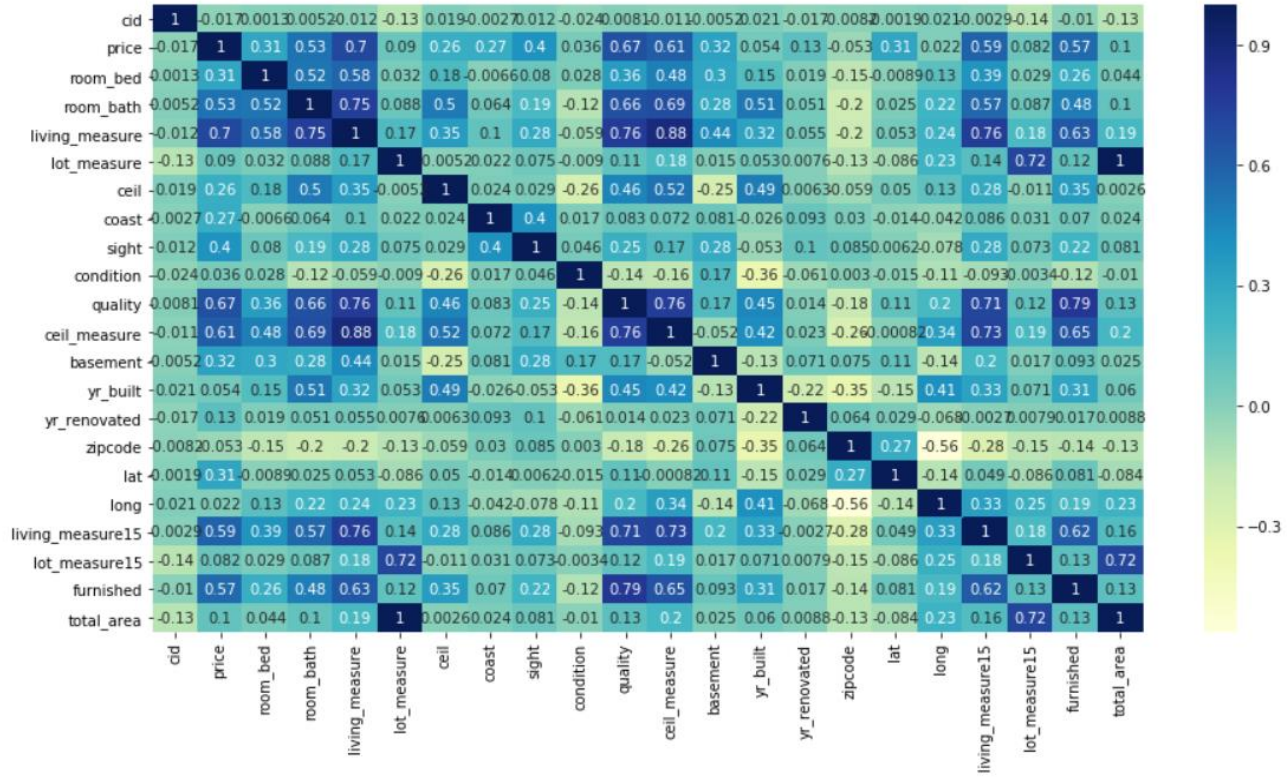


Figure 47 Heatmap

Data Preprocessing:

Outlier Treatment:

I have analyzed all the columns with outliers while performing the Exploratory data Analysis. As per my analysis, few columns have greater number of outliers. Almost 15% of the records in the dataset are the outliers. These outliers may contain some important information. I choose not to treat the outliers this time. I will perform the model building with the outliers.

I am neither imputing nor dropping the outliers at this point. I believe that building the model with outliers will give me better insights than treating them or dropping them. I won't consider dropping the outliers in any case anyway because, dropping the outliers may sometime leads to the loss of valuable information about the data. we can treat the outliers in the data either by imputing or capping the outliers, this might also give the better results.

This time, I am not going to treat the outliers. I will proceed to model building without treating outliers.

In my dataset, I feel each & every value is important and helps me to analyze more in detail about the data.

Hence, I choose not to treat the outliers this time.

Let's see the feature/columns and drop the unnecessary features:

```
Index(['cid', 'dayhours', 'price', 'room_bed', 'room_bath', 'living_measure',
       'lot_measure', 'ceil', 'coast', 'sight', 'condition', 'quality',
       'ceil_measure', 'basement', 'yr_built', 'yr_renovated', 'zipcode',
       'lat', 'long', 'living_measure15', 'lot_measure15', 'furnished',
       'total_area', 'month_year', 'has_basement', 'HouseLandRatio',
       'has_renovated'],
      dtype='object')
```

As we already have this information in other features. We will drop the unwanted columns from new copied data frame instance: cid, dayhours, yr_renovated, zipcode, lat, long.

	cid	dayhours	price	room_bed	room_bath	living_measure	lot_measure	ceil	coast	sight	condition	quality	ceil_measure	basement
12235	1425059178	2014-05-07	460000	3.0	2.00	1760.0	9055.0	2.0	0.0	0.0	4.0	7.0	1760.0	0.0
14791	7942601475	2014-05-20	345600	5.0	3.50	2800.0	5120.0	2.5	0.0	0.0	3.0	9.0	2800.0	0.0
1742	5652600185	2014-05-02	750000	3.0	1.75	2240.0	10578.0	2.0	0.0	0.0	5.0	8.0	1550.0	690.0
17829	3529200190	2014-05-14	325000	3.0	2.50	2220.0	6049.0	2.0	0.0	0.0	4.0	8.0	2220.0	0.0
14810	5631500992	2014-05-15	390000	3.0	2.50	2240.0	10800.0	2.0	0.0	0.0	3.0	8.0	2240.0	0.0

The instance of new dataframe for model created.

df_model columns:

```
Index(['cid', 'dayhours', 'price', 'room_bed', 'room_bath', 'living_measure',
       'lot_measure', 'ceil', 'coast', 'sight', 'condition', 'quality',
       'ceil_measure', 'basement', 'yr_built', 'yr_renovated', 'zipcode',
       'lat', 'long', 'living_measure15', 'lot_measure15', 'furnished',
       'total_area', 'month_year', 'has_basement', 'HouseLandRatio',
       'has_renovated'],
      dtype='object')
```

Dropping unnecessary columns in df_model:

'cid', 'dayhours', 'yr_renovated', 'zipcode', 'lat', 'long'

```
Index(['price', 'room_bed', 'room_bath', 'living_measure', 'lot_measure',
       'ceil', 'coast', 'sight', 'condition', 'quality', 'ceil_measure',
       'basement', 'yr_built', 'living_measure15', 'lot_measure15',
       'furnished', 'total_area', 'month_year', 'has_basement',
       'HouseLandRatio', 'has_renovated'],
      dtype='object')
```

Creating dummies for categorical variables:

'room_bed', 'room_bath', 'ceil', 'coast', 'sight', 'condition', 'quality', 'furnished', 'has_basement', 'has_renovated'

Getting dummies for columns ceil, coast, sight, condition, quality, yr_renovated, furnished.

The shape of df_model: (21613,79)

model_df columns:

```
Index(['price', 'living_measure', 'lot_measure', 'ceil_measure', 'basement',
       'yr_built', 'living_measure15', 'lot_measure15', 'total_area',
       'month_year', 'HouseLandRatio', 'room_bed_2.0', 'room_bed_3.0',
       'room_bed_4.0', 'room_bed_5.0', 'room_bed_6.0', 'room_bed_7.0',
       'room_bed_8.0', 'room_bed_9.0', 'room_bed_10.0', 'room_bed_11.0',
       'room_bed_33.0', 'room_bath_0.5', 'room_bath_0.75', 'room_bath_1.0',
       'room_bath_1.25', 'room_bath_1.5', 'room_bath_1.75', 'room_bath_2.0',
       'room_bath_2.25', 'room_bath_2.5', 'room_bath_2.75', 'room_bath_3.0',
       'room_bath_3.25', 'room_bath_3.5', 'room_bath_3.75', 'room_bath_4.0',
       'room_bath_4.25', 'room_bath_4.5', 'room_bath_4.75', 'room_bath_5.0',
       'room_bath_5.25', 'room_bath_5.5', 'room_bath_5.75', 'room_bath_6.0',
       'room_bath_6.25', 'room_bath_6.5', 'room_bath_6.75', 'room_bath_7.5',
       'room_bath_7.75', 'room_bath_8.0', 'ceil_1.5', 'ceil_2.0', 'ceil_2.5',
       'ceil_3.0', 'ceil_3.5', 'coast_1.0', 'sight_1.0', 'sight_2.0',
       'sight_3.0', 'sight_4.0', 'condition_2.0', 'condition_3.0',
       'condition_4.0', 'condition_5.0', 'quality_3.0', 'quality_4.0',
       'quality_5.0', 'quality_6.0', 'quality_7.0', 'quality_8.0',
       'quality_9.0', 'quality_10.0', 'quality_11.0', 'quality_12.0',
       'quality_13.0', 'furnished_1.0', 'has_basement_Yes',
       'has_renovated_Yes'],
      dtype='object')
```

	price	living_measure	lot_measure	ceil_measure	basement	yr_built	living_measure15	lot_measure15	total_area	month_year	HouseLandRatio
12235	460000	1760.0	9055.0	1760.0	0.0	1985.0	2010.0	9383.0	10815.0	May-2014	16.0
14791	345600	2800.0	5120.0	2800.0	0.0	1903.0	1780.0	5120.0	7920.0	May-2014	35.0
1742	750000	2240.0	10578.0	1550.0	690.0	1923.0	1570.0	10578.0	12818.0	May-2014	17.0
17829	325000	2220.0	6049.0	2220.0	0.0	1990.0	1980.0	7226.0	8269.0	May-2014	27.0
14810	390000	2240.0	10800.0	2240.0	0.0	1996.0	1900.0	9900.0	13040.0	May-2014	17.0

Now the data is ready for model building with the selected columns.

Business insights from EDA

→The data is not unbalanced in my case. I cannot see any unbalanced data in the dataset.

→The feature 'cid' is appearing multiple times, it means that data contains houses which were sold multiple times. We have 176 properties that were sold more than once in the given data.

→The timeline for sale data of houses is from May-2014 to May-2015. From the mean sale data, we can observe that, April month have the highest mean price

→The price of the house is ranging from 75,000 to 77,00,000.

→Most of the properties/houses are of 3-bedroom & 4-bedroom type.

→Majority of the properties/houses having bathroom in the range of 1.0 to 2.5

→Square footage of house ranges from 290 - 13,540.

→Square footage of lot ranges from 520 - 16,51,359.

→Most of the houses in the data are 1 & 2 floored houses.

→We can observe that most of the houses has not been viewed(sight). Many houses were viewed(sight) twice. Condition represents rating of house which ranges from 1 – 5. Most of the houses are rated as 3 and above for its overall condition.

→Majority of the properties/houses have quality rating between 6 to 10.

→Houses with zero measure of basement means that they do not have basements. Around 60% of the houses do not have the basement.

→The built year of the houses range from 1900 to 2014 and we can see the upward trend with time.

→Only 914 houses were renovated out of 21613 houses.

→Majority of the houses are non-furnished houses. Only 4246 houses are furnished out of 21613 houses.

→The average of the sale price of houses tends to be high during March, April & May compared to that of September, October, November, December period.

→We can observe that Mean and median of price increases with number bedrooms per house up to a point and then drops. There is a clear increasing trend in price with room_bed.

→The overall mean and median price increases with increasing room_bath. There is an upward trend in price with increase in room_bath.

→Price increases with increase in living measure. There is clear increment in the price of the house with increment in the living measure

→We can see that, almost 95% of the houses have <25000 lot_measure. But there is no clear trend between lot_measure and price.

→Price increases with increase in lot measure.

→The houses with waterfront view are expected to have higher price compared to that of non-waterfront view houses.

→The houses that are viewed more have high price (mean and median) and having large living area. Houses with high price have been viewed more compared to that of houses with low price.

→Houses with high price with large living area are viewed more.

→Houses with higher price have a greater number of times viewed compared to that of houses with lower price.

→As the condition rating increases its price and living measure mean and median also increases.

→When viewed in relation with price and living_measure. Most houses are graded as 6 or more.

→Price increases with increase in ceil measure. We can observe that, there is an upward trend in price with ceil_measure.

→Houses with basement are costlier and have higher living measure (mean & median).

→Most houses are renovated after 1980's. Renovated houses utilized more land area for construction of house. Renovated properties have higher price than others with same living measure space.

→Furnished houses have higher price value and has greater living_measure.