z

House Price Prediction

FINAL REPORT - SUBMISSION

SANTHOSH KUMAR | DSBA SEPT-B BATCH | 04-09-2022

Table of Context Page

1. List of Figures………………………………………………………………………………………. 3
2. List of Tables………………………………………………………………………………… 4
3. Abstract………………………………………………………………………………………… 5
4. Introduction - What did you wish to achieve while doing the project...6-7
5. EDA - Uni-variate / Bi-variate / multi-variate analysis to understand relationship b/w variables. - Both visual and non-visual understanding of the data..7-8
6. Data Cleaning and Pre-processing - Approach used for identifying and treating missing values and outlier treatment (and why) - Need for variable transformation (if any) - Variables removed or added and why (if any) 9-15
7. Model building - Clear on why was a particular model(s) chosen. - Effort to improve model performance...16-26
8. Model validation - How was the model validated? Just accuracy, or anything else too..26-27
9. Final interpretation / recommendation - Very clear and crisp on what recommendations do you want to give to the management / client...28-29
10. Appendix A (Univariate & Bi-Variate Plots) ……………………………………. 30-37

A1 - Bedroom Vs Count
A2 - Quality of houses
A3- Ceilings in the house
A4- Bathroom in the houses
A5- Ceil vs Ceil_measure
A6- Furnished Houses Count
A7- Price vs Month & year
A8- Living measure vs Price
A9- Sight vs living measure
A10- Living measure vs year built
A11- Living measure vs Price vs renovated houses
A12- Living measure vs furnished
A13- Missing Data (In terms of %)

List of Figures Page

1. Geo Locations of the House based on Zip Code…………………………………
1.1 Zip Codes frequency …………………………………………………………………. 8
2. Bi-Variate Plots……………………………………………………………………………….
2.1 Pair Plot……………………………………………………………………………………. 11
2.2 Heat Map………………………………………………………………………………… 12

3. Box plot ……………………………………………………………………………………………..

3.1 Box plot before outlier treatment ……………………………………………14

3.2 Box plot after outlier treatment……………………………………………… 15

4. Hyper Tuning the model using Gride search cv……………………………………

4.1 Gridsearch CV iteration 1………………………………………………………… 23

4.2 Gridsearch CV iteration 2……………………………………………………… 23

4.3 Gridsearch CV iteration 3………………………………………………………24

4.4 Gridsearch CV iteration 4………………………………………………………24

4.5 Gridsearch CV iteration 5……………………………………………………… 25

5. Metrics graphs

5.1 R2 metrics of train vs test……………………………………………………… 26

5.2 Significant variables used in modelling…………………………………… 27

5.3 confidence interval………………………………………………………………….. 27

6. Metrics evaluation…………………………………………………………………………………28

6.1 R-square value ……………………………………………………………………… 28

6.2 MAPE metrics………………………………………………………………………………29

List of Tables Page

1. Classification Modelling

1.1 Linear Regression………………………………………………………………… 17
1.2 Linear Ridge Regression…………………………………………………… 18
1.3 KNN regressor………………………………………………………………… 18
1.4 Support Vector Regressor…………………………………………………. 19

2.Tree Based Modelling

2.1Decision Tree Regressor………………………………………………………. 19
2.2 Random Forest Regressor…………………………………………………… 20

3. Ensemble Techniques

3.1 Gradient Boosting…………………………………………………………………. 21

3.2 Bagging Regressor…………………………………………………………………. 22

4. Summary of the modelling

4.1 Consolidated metrics of all the models………………………………… 24

5. Hyper Tuning the model ………………………………………………………………….

5.1 Gradient Boosting Using grid search cv………………………………. 26

5.2 Performance of the model before hyper tuning……………………… 28

5.3 Performance of the model after hyper tuning…………………………. 28

Abstract:

Purpose:

This report contains the House price prediction modal that helps both the builder to determine the better price for the house that they are building

Modelling Approach

For Predicting the House prices classification techniques are used because The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data. In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, **Yes or No, 0 or 1, Spam or Not Spam, cat or dog,** etc. Classes can be called as targets/labels or categories.

Models Used

- **Regression approach**: Linear regression
- **Frequency Based:** Knn Regressor
- **Decision Tree:** Random Forest & Decision Tree regressor
- **Ensemble methods:** Gradient Boosting & Bagging regressor

Jupiter notebook used as tool for EDA, Modelling, visualization

1. Introduction (Business Understanding) - What did you wish to achieve while doing the project

House Price prediction, is important to drive Real Estate efficiency. As earlier, House prices were determined by calculating the acquiring and selling price in a locality. Therefore, the House Price prediction model is very essential in filling the information gap and improve Real Estate efficiency. With this model, we would be able to better predict the prices. Accurately estimating the value of real estate is an important problem for many stakeholders including house owners, house buyers, agents, creditors, and investors

House prices increase every year, so there is a need for a system to predict house prices in the future. House price prediction can help the developer determine the selling price of a house and can help the customer to arrange the right time to purchase a house. There are many factors that influence the price of a house which include physical conditions few are listed below

- square footage of the home
- Locality (Urban or Rural)
- Number of Bedrooms per house
- Number of restrooms per house
- How good the condition is (Overall)
- Total floors (levels) in house
- Built Year

Location is an important factor in shaping the price of a house. This is because the location determines the prevailing land price. In addition, the location also determines the ease of access to public facilities, such as schools, campus, hospitals and health centres, as well as family recreation facilities such as malls, culinary tours, or even offer a beautiful scenery

Companies like Casagrande etc. use House price prediction to determine the better price for the house that they are building. For example, a developer would like to build in an area where prices are going up. However even a large developer is likely only building 100 homes. The incremental revenue from better price predictions is worth maybe $100k.

As people don't know the features/aspects which cumulate property price, we can provide them House-Buying-Selling guiding services in the area so they can buy or sell their property with most suitable price tag and they didn't lose their hard-earned money by offering low price or keep waiting for buyers by putting high prices.

A person, an educated party would want to know all aspects that give a house its value. For example, if we want to sell a house and we don't know the price which we can take, as it can't be too low or too high. To find house price we usually try to find similar properties in our neighborhood and based on collected data we trying to assess our house price.

we can provide them House-Buying-Selling guiding services in the area so they can buy or sell their property with most suitable price tag

Buyers won't lose their hard-earned money by offering low price or keep waiting for buyers by putting high prices

2. EDA - Uni-variate / Bi-variate / multi-variate analysis to understand relationship b/w variables

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

After loading data into our panda's library data frame, we can now try to understand the kind of data we have with us.

• Shape of the data is (21613, 23) i.e., 23 columns & 21613 rows

• We have more than 21k records having 23 features

• Data has been collected from June 2014 to April 2015

• Exactly we have 11 months data for houses sold between June 2014to April 2015

Data info

- In the dataset, we have more than 21k records and 23 columns, out of which
- 12 features are of float type
- 4 features are of integer type
- 7 feature is of object type (we may need to convert this object type to specific datatype
- We don't have any duplicate record in our dataset. So, we can say we have more than 21k Unique records

These columns provide below information

1. **cid:** Notation for a house. Will not of our use. So, we will drop this column
2. **dayhours:** Represents Date, when house was sold.
3. **price:** It's our TARGET feature, that we have to predict based on other feature's
4. **room_bed:** Represents number of bedrooms in a house
5. **room_bath:** Represents number of bathrooms
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 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 zip code as name implies
18. **lat:** Represents Latitude 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 i.e., area of both living and lot

The "**dayhours**" column has timestamp which will not be useful in prediction since it was not recorded properly hence removed the time stamp values in the data & converted to month-Year date-time data type & renamed the column to "month-year"

Variable transformation

- The "**dayhours**" column has timestamp which will not be useful in prediction since it was not recorded properly hence removed the time stamp values in the data & converted to month-Year date-time data type & renamed the column to "month-year"
- Longitude variable was in object data type that needs to be converted in to float

Exploratory Data Analysis (Univariate & bi-Variate)
Data Describe & skewness

- **CID**: House ID/Property ID. Not used for analysis
- **price**: Our target column value is in 75k - 7700k range. As Mean > Median, it's Right-Skewed.
- **room_bed**: Number of bedrooms range from 0 - 33. As Mean slightly > Median, it's slightly Right-Skewed.
- **room_bath**: Number of bathrooms range from 0 - 8. As Mean slightly < Median, it's slightly Left-Skewed.
- **living_measure:** square footage of house ranges from 290 - 13,540. As Mean > Median, it's Right-Skewed.
- **lot_measure**: square footage of lot ranges from 520 - 16,51,359. As Mean almost double of Median, it's Highly Right-Skewed.
- ceil: Number of floors range from 1 - 3.5 As Mean ~ Median, it's almost Normal Distributed.
- **coast**: As this value represent whether house has waterfront view or not. It's categorical column. From above analysis we got know, very few houses have waterfront view.
- **sight**: Value ranges from 0 - 4. As Mean > Median, it's Right-Skewed
- **condition**: Represents rating of house which ranges from 1 - 5. As Mean > Median, it's Right-Skewed
- **quality**: Representing grade given to house which range from 1 - 13. As Mean > Median, it's Right-Skewed.
- **ceil_measure:** square footage of house apart from basement ranges in 290 - 9,410. As Mean > Median, it's Right-Skewed.
- **basement:** Square footage house basement ranges in 0 - 4,820. As Mean highlty > Median, it's Highly Right-Skewed.
- **yr_built:** House built year ranges from 1900 - 2015. As Mean < Median, it's Left-Skewed.
- **yr_renovated:** House renovation year only 2015. So, this column can be used as Categorical Variable for knowing whether house is renovated or not.
- **zipcode:** House Zip Code ranges from 98001 - 98199. As Mean > Median, it's Right-Skewed.
- **lat**: Latitude ranges from 47.1559 - 47.7776 As Mean < Median, it's Left-Skewed.
- **long**: Longitude ranges from -122.5190 to -121.315 As Mean > Median, it's Right-Skewed.
- **living_measure15:** Value ranges from 399 to 6,210. As Mean > Median, it's Right-Skewed.
- **lot_measure15:** Value ranges from 651 to 8,71,200. As Mean highly > Median, it's Highly Right-Skewed.
- furnished: Representing whether house is furnished or not. It's a Categorical Variable
- **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

Univariate Analysis (Refer the raw output of visualization in Appendix section)

- Most of the houses/properties have 3 or 4 bedrooms
- Majority of the properties have bathroom in the range of 1.0 to 2.5
- Above graph confirming the same, that most properties have 1 and 2 floor
- There are 1625 properties which have the highest quality rating & 273 properties have lowest rating
- The vertical lines at each point represent the inter quartile range of values at that point
- Most properties are not furnished. Furnish column need to be converted into categorical column
- The mean price of the houses tends to be high during March, April, May as compared to that of September, October, November, December period.
- There is clear increment in price of the property with increment in the living measure but there seems to be one outlier to this trend. Need to evaluate the same
- The above graph also justifies that: Properties with higher price have more no. of sights compared to that of houses with lower price.
- We will create new variable: House land ratio - This is proportion of living area in the total area of the house. We will explore the trend of price against this house land ratio.
- Renovated properties have higher price than others with same living measure space.
- Furnished houses have higher price than that of the non-furnished houses

Latitude & longitude

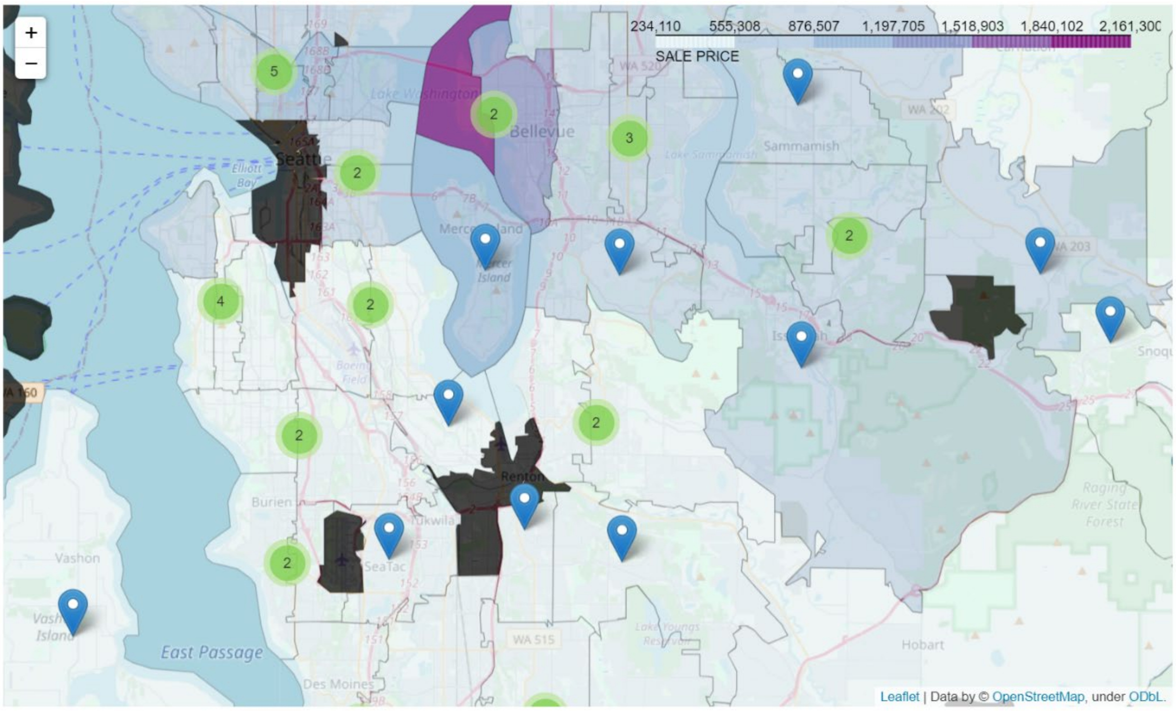


Fig 1.1 Zip Code in the City

- All the Houses are sold on Washington is based on the zip code, the prices vary drastically from the above map
- As you can see the price scale from the map, we see that highest price is concentrated in the centre of the map
- Lowest price is happening in the coastal region in the map

Bi-Variate Analysis:

Pair-plot

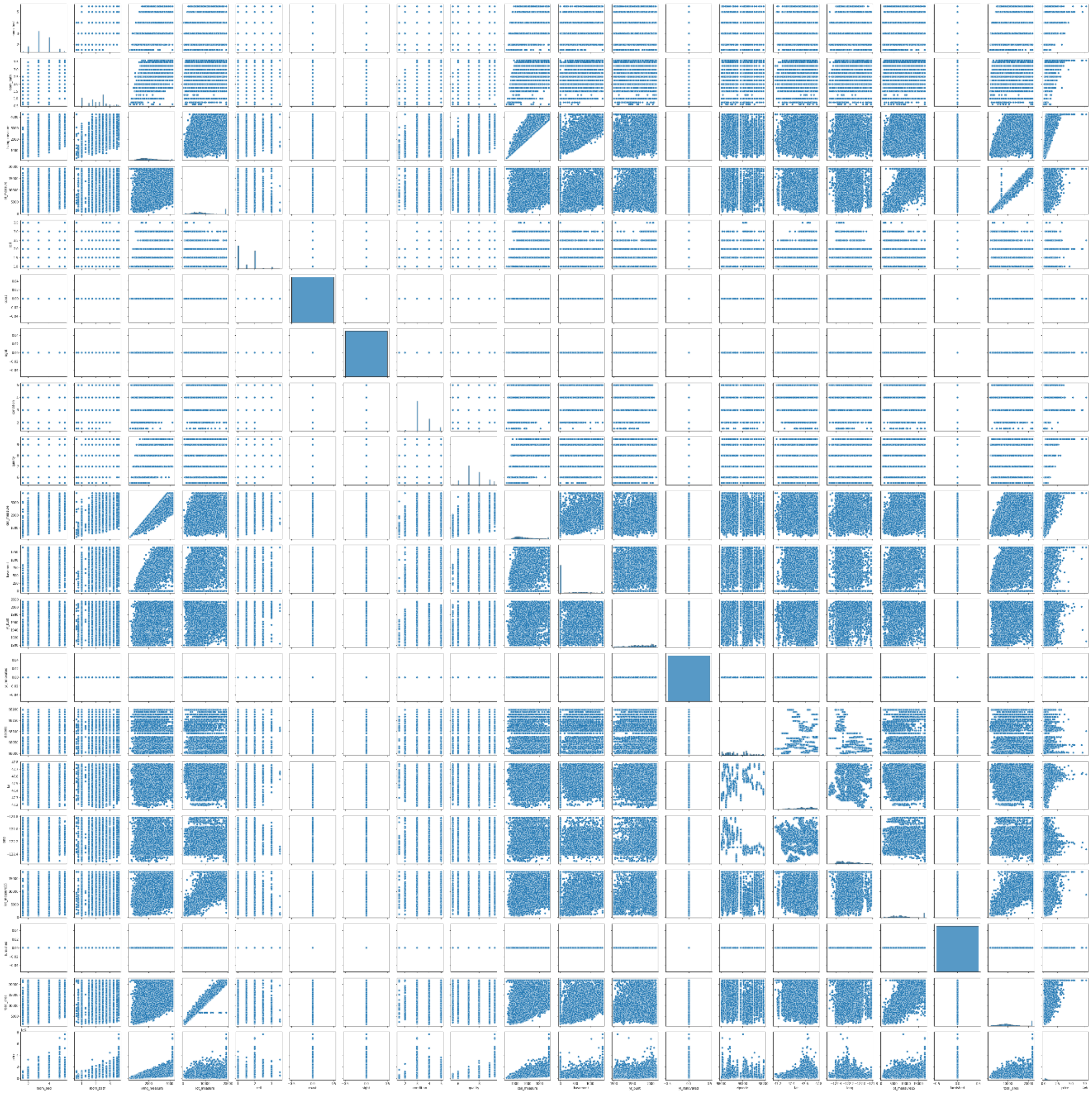


Fig 2.1 Pair plot for Bi-Variate Analysis

From above pair plot, we observed/deduced below

1. **price:** price distribution is Right-Skewed as we deduced earlier from our 5-factor analysis

2. **room_bed:** our target variable (price) and room_bed plot is not linear. Its distribution has lot of gaussians
3. **room_bath:** It's plot with price has **somewhat linear relationship**. Distribution has number of gaussians.
4. **living_measure:** Plot against price has **strong linear relationship**. It also has linear relationship with room bath variable. So **might remove one of these 2**. Distribution is Right-Skewed.
5. **lot_measure: No clear relationship** with price.
6. **ceil: No clear relationship** with price. We can see, it's **had 6 unique values** only. Therefore, we can **convert this column into categorical column** for values.
7. **coast: No clear relationship** with price. Clearly, it's **categorical variable with 2 unique values**.
8. **sight: No clear relationship** with price. This has **5 unique values**. Can be **converted to Categorical variable**.
9. **condition: No clear relationship** with price. This has **5 unique values**. Can be **converted to Categorical variable**.
10. **quality: Somewhat linear relationship with price**. Has **discrete values from 1 - 13. Can be converted to Categorical variable**.
11. **ceil_measure: Strong linear relationship with price**. Also, with room_bath and living_measure features. Distribution is **Right-Skewed**.
12. **basement: No clear relationship** with price.
13. **yr_built: No clear relationship** with price.
14. **yr_renovated: No clear relationship** with price. Have **2 unique values. Can be converted to Categorical Variable** which tells whether house is renovated or not.
15. **zipcode, lat, long: No clear relationship** with price or any other feature.
16. **living_measure15: Somewhat linear relationship with target feature**. It's same as living measure. Therefore, we can drop this variable.
17. **lot_measure15: No clear relationship** with price or any other feature.
18. **furnished: No clear relationship** with price or any other feature. **2 unique values so can be converted to Categorical Variable**

total_area: No clear relationship with price. But it has **Very Strong linear relationship with lot_measure**. So, one of it can be dropped.

Heatmap for Co-relation

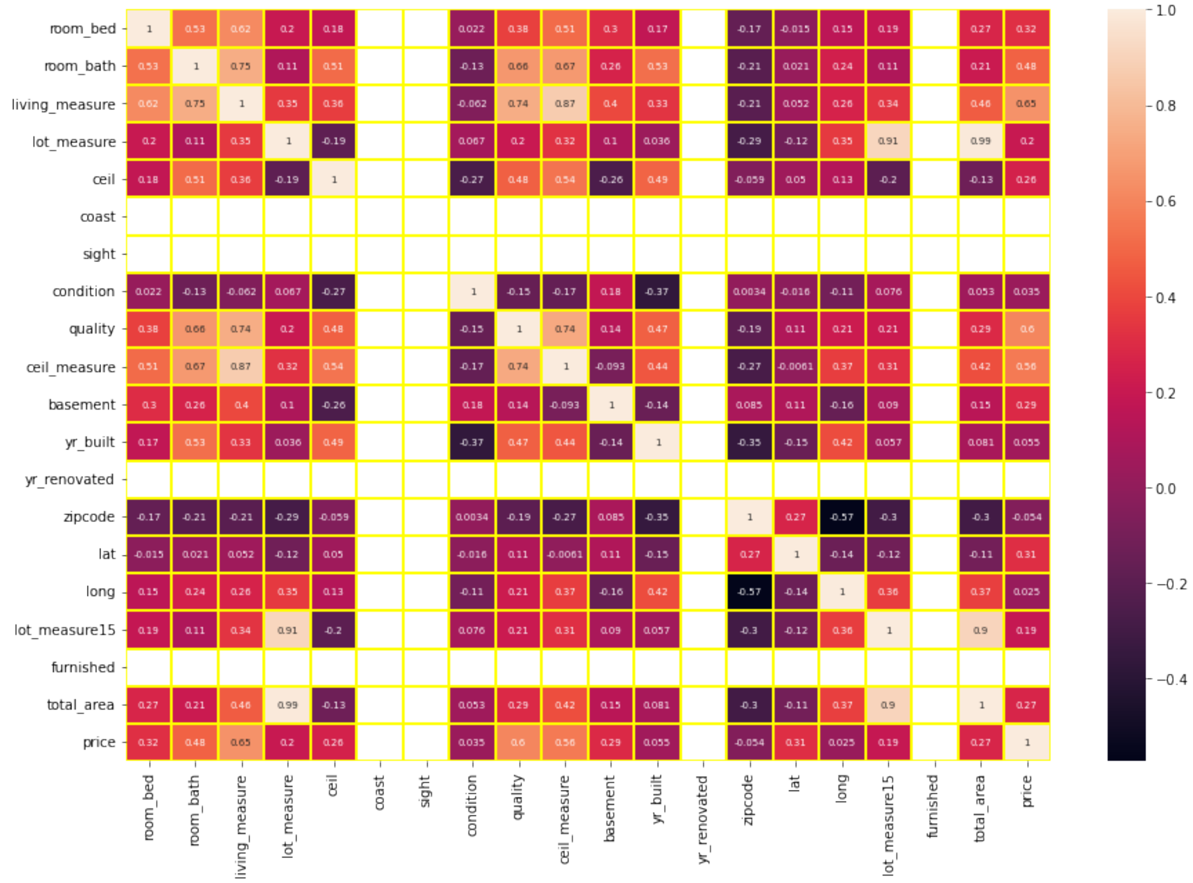


Fig 2.2 heat map for co-relation

We have linear relationships in below featues as we got to know from above matrix

1. **price**: room_bath, living_measure, quality, living_measure15, furnished
2. **living_measure**: price, room_bath. So, we can consider dropping 'room_bath' variable.
3. **quality**: price, room_bath, living_measure
4. **ceil_measure**: price, room_bath, living_measure, quality
5. **living_measure15**: price, living_measure, quality. So, we can consider dropping living_measure15 as well. As it's giving same info as living_measure.
6. **lot_measure15**: lot_measure. Therefore, we can consider dropping lot_measure15, as it's giving same info.
7. **furnished**: quality
8. **total_area**: lot_measure, lot_measure15. Therefore, we can consider dropping total_area feature as well. As it's giving same info as lot_measure.

3. Data Cleaning and Pre-processing - Approach used for identifying and treating missing values and outlier treatment (and why) - Need for variable transformation (if any) - Variables removed or added and why (if any)

Checking for the missing Values in the data: (For Raw output refer Appendix)

The overall percentage of data that is missing is important. Generally, if less than 20 -30% of values are missing then it is acceptable to ignore them by the industry standards.

It also important that if the column is considered as the important for building the model at that we cannot drop the column instead we can drop null values from the column and use it for prediction

- As we can see from the above image, all the variables have missing values in it
- “living_measure15” column has 70% of the data is missing, with this this variable will not help the model for a better prediction of the house price hence removed from the data set
- “room_bed”, “room_bath” also has data missing close to 50% and these two columns are important aspects for the price prediction, hence not removed from the data set.
- For the rest of columns, the data missing is less than 30% hence we can go-ahead & remove them

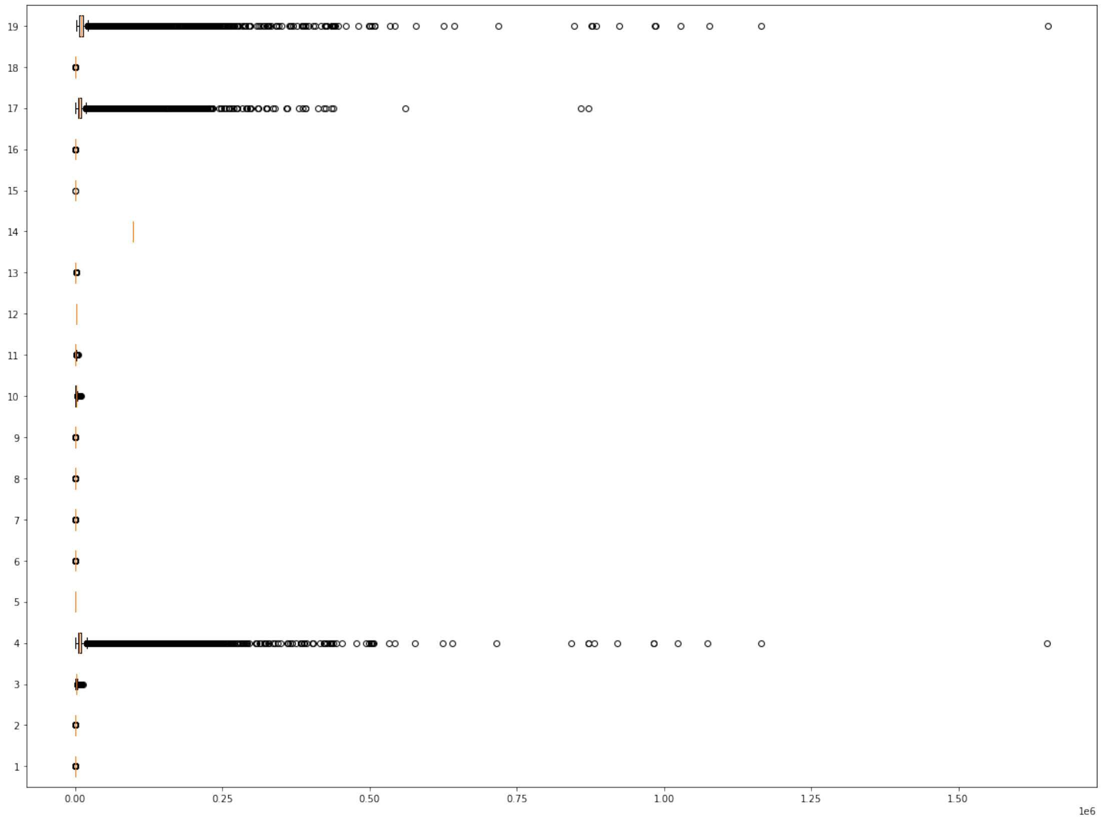


Fig 3.1 Box plot for checking outliers

Outlier Treatment is not performed for the target variable “**price**” in the data set

Outliers are treated using the IQR method

IQR is used to measure variability by dividing a data set into quartiles. The data is sorted in ascending order and split into 4 equal parts. Q1, Q2, Q3 called first, second and third quartiles are the values which separate the 4 equal parts.

- Q1 represents the 25th percentile of the data.
- Q2 represents the 50th percentile of the data.
- Q3 represents the 75th percentile of the data.

IQR is the range between the first and the third quartiles namely Q1 and Q3: IQR = Q3 – Q1. The data points which fall below Q1 – 1.5 IQR or above Q3 + 1.5 IQR are outliers.

After treating the outlier, the box plot looks clean

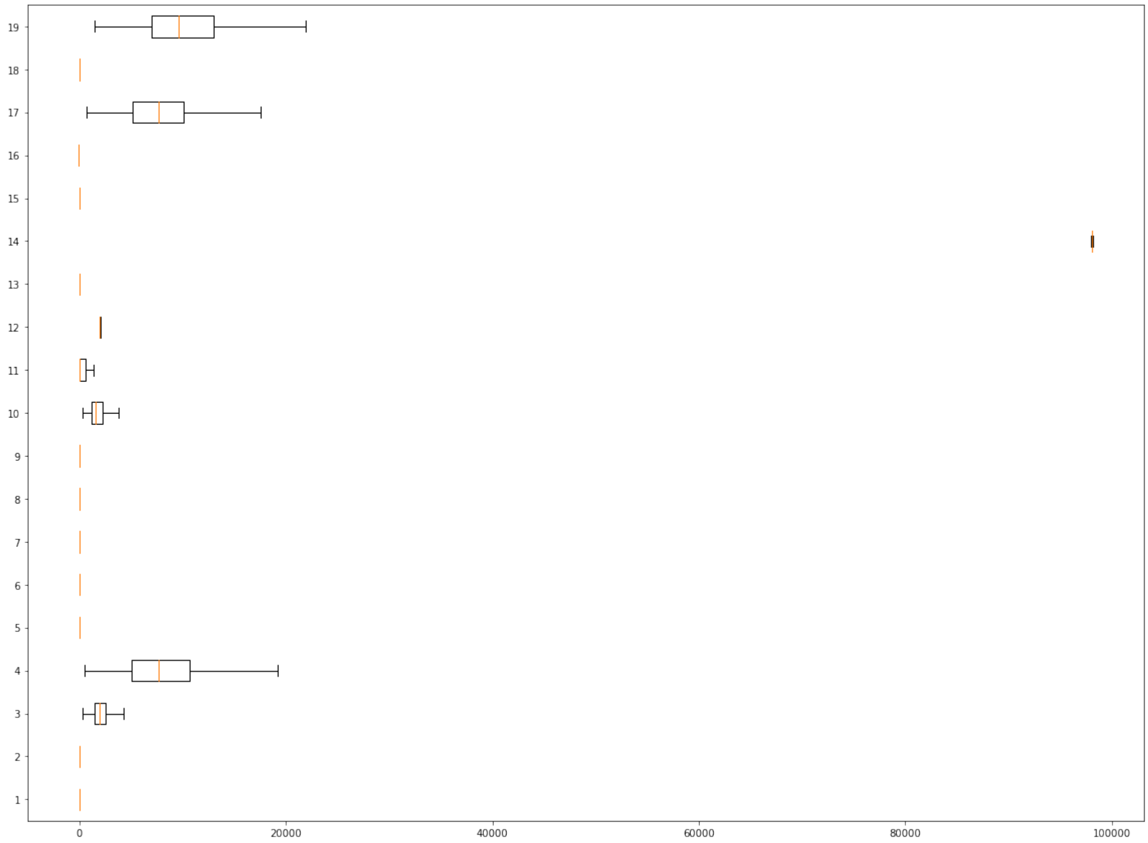


Fig 3.1 Box plot after treating outliers

Scaling the Data
Machine Learning algorithms which are mostly dependent on the distance of the features which vary heavily in terms of scaling may not weigh all these in the same way. The parameter which has smaller values does not mean to be seen as less importance Hence, to make the predictions do not vary just because of the 'X' vars are of different scales, we can apply few techniques like Feature Scaling, Standardization or Normalization

Some of the methods for scaling the data are "Normalization", "Standardization"

Min-Max Normalization: This technique re-scales a feature or observation value with distribution value between 0 and 1.

Standardization function is to make data points cantered about the mean of all the data points presented in a feature with a unit standard deviation. This means the mean of the data point will be zero and the standard deviation will be 1.

This technique also tries to scale the data point between zero to one but in it, we don't use max or minimum. Here we are working with the mean and the standard deviation.

For the House-Price-Prediction regression problem, I've used the standardization scaling method for model building

Encoding the Data using Dummies
Dummy coding scheme is similar to one-hot encoding. This categorical data encoding method transforms the categorical variable into a set of binary variables (also known as dummy variables). In the case of one-hot encoding, for N categories in a variable, it uses N binary variables. The dummy encoding is a small improvement over one-hot-encoding. Dummy encoding uses N-1 features to represent N labels/categories

For the House-Price-Prediction, used Dummy encoding technique

Variance Inflation Factor (VIF)
Variance Inflation Factor (VIF) is used to detect the presence of multicollinearity. Variance inflation factors (VIF) measure how much the variance of the estimated regression coefficients is inflated as compared to when the predictor variables are not linearly related

Multicollinearity (Refer VIF output in Appendix)

multicollinearity does not reduce a model's overall predictive power.it can produce estimates of the regression coefficients that are not statistically significant. In a sense, it can be thought of as a kind of double-counting in the model.

When two or more independent variables are closely related or measure almost the same thing, then the underlying effect that they measure is being accounted for twice (or more) across the variables

The table mentioned to the left is the Variation inflation factor for the house-price-prediction features

From the VIF results we can see that columns “total_area”, “lot_measure”, “living_measure”, “ceil_measure”, “basement” has VIF score greater than 5

Even though VIF score is high for the above-mentioned columns, I’ve not removed the above-mentioned columns for modelling because from the real-estate domain perspective apart from all the columns people would give more focus to the “total_area”, “Living_measure”, “lot_measure”, “ceil_measure” – purely from the domain perspective these columns will add more value while the prediction, hence these columns are not removed for model building

4. Model building - Clear on why was a particular model(s) chosen. - Effort to improve model performance

Train & Test Split:

The train-test split is a technique for evaluating the performance of a machine learning algorithm. It can be used for classification or regression problems and can be used for any supervised learning algorithm.

The procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model and is referred to as the training dataset. The second subset is not used to train the model; instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values. This second dataset is referred to as the test dataset.

- **Train Dataset**: Used to fit the machine learning model.
- **Test Dataset**: Used to evaluate the fit machine learning model.

The objective is to estimate the performance of the machine learning model on new

For this House-Price-Prediction regression problem split the data as mentioned below

Train Data: 75%
Test Data: 25%

Model 1: Linear Regression

In this technique, the dependent variable is continuous, independent variable(s) can be continuous or discrete, and nature of regression line is linear.

Linear Regression establishes a relationship between dependent variable (Y) and one or more independent variables (X) using a best fit straight line (also known as regression line).

It is represented by an equation Y=a+b*X + e, where a is intercept, b is slope of the line and e is error term. This equation can be used to predict the value of target variable based on given predictor variable(s).

Linear Regression is very sensitive to Outliers. It can terribly affect the regression line and eventually the forecasted values.

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
Linear Regression	0.6255	0.625688	0.391486	0.350973	0.646468	0.590058	0.348168	0.350604	2.473046

Table 1.1 Linear Regression

Model 2: Ridge Regression

Ridge Regression is a technique used when the data suffers from multicollinearity (independent variables are highly correlated). In multicollinearity, even though the least squares estimate (OLS) are unbiased, their variances are large which deviates the observed value far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors.

Ridge Regression can be represented as y=a+ b*x

The assumptions of this regression are same as least squared regression except normality is not to be assumed

Ridge regression shrinks the value of coefficients but doesn’t reach zero, which suggests no feature selection feature

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
Ridge Regression	0.625511	0.625691	0.391489	0.350960	0.64648	0.590058	0.348168	0.350596	2.473488

Table 1.2 Ridge Regression

Model 3: KNN Regressor

KNN algorithm can be used for both classification and regression problems. The KNN algorithm uses '**feature similarity**' to predict the values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set.

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
KNN Regressor	0.635605	0.617201	0.380936	0.316817	0.999346	0.025373	0.000644	0.002263	2.218185

Table 1.3 KNN Regression

Model 4: Support vector regressor

Support Vector Regression is a supervised learning algorithm that is used to predict discrete values. Support Vector Regression uses the same principle as the SVMs. The basic idea behind SVR is to find the best fit line. In SVR, the best fit line is the hyperplane that has the maximum number of points.

Unlike other Regression models that try to minimize the error between the real and predicted value, the SVR tries to fit the best line within a threshold value. The threshold value is the distance between the hyperplane and boundary line. The fit time complexity of SVR is more than quadratic with the number of samples which makes it hard to scale to datasets with more than a couple of 10000 samples.

For large datasets, Linear SVR or SGD Regressor is used. Linear SVR provides a faster implementation than SVR but only considers the linear kernel. The model produced by Support Vector Regression depends only on a subset of the training data, because the cost function ignores samples whose prediction is close to their target.

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
Support Vector Regressor	0.721221	0.539857	0.291446	0.272436	0.858765	0.372956	0.139096	0.196000	1.695654

Table 1.4 Support Vector Regression

Model 5: Decision Tree Regressor

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called **root node**

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
Decision Tree	0.748356	0.512598	0.262757	0.254180	0.858779	0.349772	0.122340	0.205993	1.665546

Table 2.1 Decision Tree Regressor

Ensemble techniques: Boosting & bagging

Model 6: Gradient Boosting

Gradient boosting is one of the most popular machine learning algorithms for tabular datasets. It is powerful enough to find any nonlinear relationship between your model target and features and has great usability that can deal with missing values, outliers, and high cardinality categorical values on your features without any special treatment. While you can build barebone gradient boosting trees using some popular libraries such as XGBoost or LightGBM without knowing any details of the algorithm, you still want to know how it works when you start tuning hyper-parameters, customizing the loss functions

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
Gradient Boosting	0.785167	0.493704	0.224585	0.225631	0.868491	0.359886	0.129518	0.202751	1.616828

Table 3.1 Gradient Boosting regressor

Model 7: Random Forest

Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

Random Forest has multiple decision trees as base learning models

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
Random Forest	0.804996	0.451404	0.203856	0.208023	0.975683	0.154755	0.023949	0.077728	1.361259

Table 2.2 Random Forest Regressor

Summary of all the models

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
Linear Regression	0.6255	0.625688	0.391486	0.350973	0.646468	0.590058	0.348168	0.350604	2.473046
Ridge Regression	0.625511	0.625691	0.391489	0.350960	0.64648	0.590058	0.348168	0.350596	2.473488
KNN Regressor	0.635605	0.617201	0.380936	0.316817	0.999346	0.025373	0.000644	0.002263	2.218185
Support Vector Regressor	0.721221	0.539857	0.291446	0.272436	0.858765	0.372956	0.139096	0.196000	1.695654
Decision Tree	0.748356	0.512598	0.262757	0.254180	0.858779	0.349772	0.122340	0.205993	1.665546
Gradient Boosting	0.785167	0.493704	0.224585	0.225631	0.868491	0.359886	0.129518	0.202751	1.616828
Random Forest	0.804996	0.451404	0.203856	0.208023	0.975683	0.154755	0.023949	0.077728	1.361259

Table 4.1 Consolidated metrics of all the models

1. **Linear regression** – model has accuracy 64% in training set & 62% in validation set, hence Linear regression model cannot be considered for the tuning will look for the other better model performance
2. **KNN** – model has overfitted in training data results close to 100% & even though it has performed well in training data still the KNN regressor has performed poorly in the validation set resulting in only 63% accuracy – KNN model will not be considered for the Tuning
3. **Decision Tree Regressor** - Similar to KNN model, DTR model is overfitted in the training data with accuracy 100% & performed poorly in the validation set with accuracy 68% hence DTR model will not be considered for further tuning
4. **Gradient Boosting regressor** - It's an ensemble technique which performed very good on both training data & validation set train data accuracy – 86%, validation set accuracy – 78% GBR model has good RMSE_val – 47%, MAPE – 12% is good and MSE value is 0.2022 which is good hence GBR is considered for tuning which might result the prediction in the validation set
5. **Bagging Regressor** - model has given excellent score in train data which is 97% seems close to overfitting & BGR model performed well in validation set and has accuracy 80%
6. **Random Forest** – Model has performed similarly to the BGR model has accuracy close to 97% & performed well in validation set as well with accuracy 80%

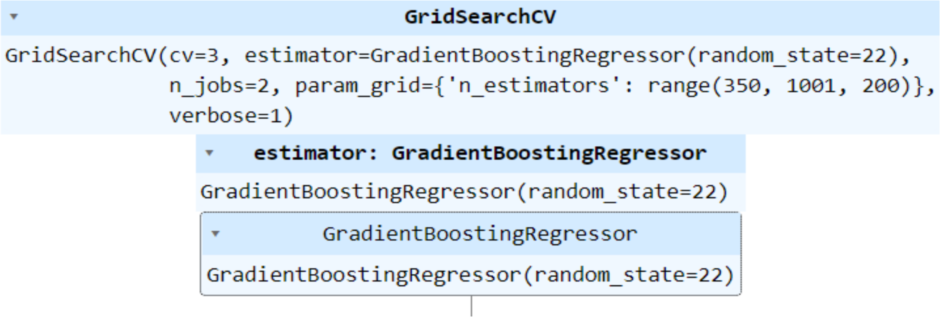
Gradient Boost Regressor Tuning using gride search CV method:

GridSearchCV is the process of performing hyperparameter tuning in order to determine the optimal values for a given model. As mentioned above, the performance of a model significantly depends on the value of hyperparameters. Note that there is no way to know in advance the best values for hyperparameters so ideally, we need to try all possible values to know the optimal values. Doing this manually could take a considerable amount of time and resources and thus we use GridSearchCV to automate the tuning of hyperparameters

1.**estimator**: Pass the model (Gradient Boost Regressor).

2.**params_grid**: the dictionary object that holds the hyperparameters you want to try
3.**scoring**: evaluation metric that you want to use, you can simply pass a valid string/ object of evaluation metric
4.**cv:** number of cross-validation you have to try for each selected set of hyperparameters
5.**verbose:** you can set it to 1 to get the detailed print out while you fit the data to GridSearchCV
6.**n_jobs:** number of processes you wish to run in parallel for this task if it -1 it will use all available processors.
Tuned the GBR model with 6 iterations of grid Search CV params

Iteration 1:

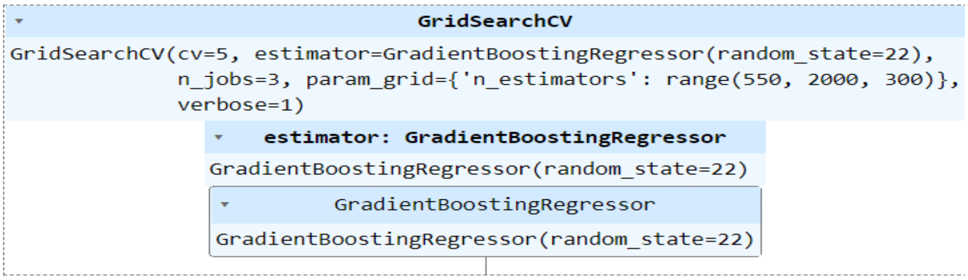


```
GridSearchCV
GridSearchCV(cv=3, estimator=GradientBoostingRegressor(random_state=22),
             n_jobs=2, param_grid={'n_estimators': range(350, 1001, 200)},
             verbose=1)
estimator: GradientBoostingRegressor
GradientBoostingRegressor(random_state=22)
GradientBoostingRegressor
GradientBoostingRegressor(random_state=22)
```

Fig 4.1 Grid search CV iteration 1

For Iteration 1 with n_estimator : 550 the score was 81%

Iteration 2:

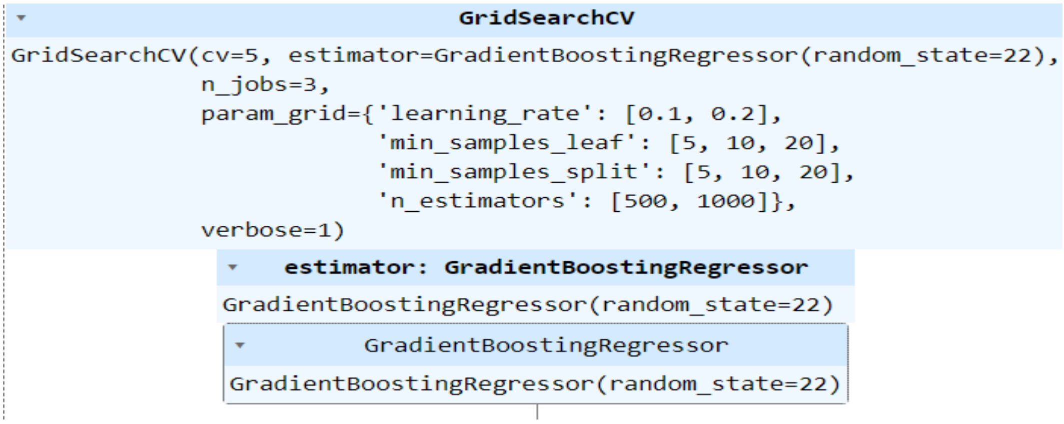


```
GridSearchCV
GridSearchCV(cv=5, estimator=GradientBoostingRegressor(random_state=22),
             n_jobs=3, param_grid={'n_estimators': range(550, 2000, 300)},
             verbose=1)
estimator: GradientBoostingRegressor
GradientBoostingRegressor(random_state=22)
GradientBoostingRegressor
GradientBoostingRegressor(random_state=22)
```

Fig 4.2 Grid search CV iteration 2

For Iteration 2 with n_estimator : 550 the score was increased to 82%

Iteration 3:


GridSearchCV
estimator: GradientBoostingRegressor
GradientBoostingRegressor

```
GridSearchCV(cv=5, estimator=GradientBoostingRegressor(random_state=22),
             n_jobs=3,
             param_grid={'learning_rate': [0.1, 0.2],
                         'min_samples_leaf': [5, 10, 20],
                         'min_samples_split': [5, 10, 20],
                         'n_estimators': [500, 1000]},
             verbose=1)
```

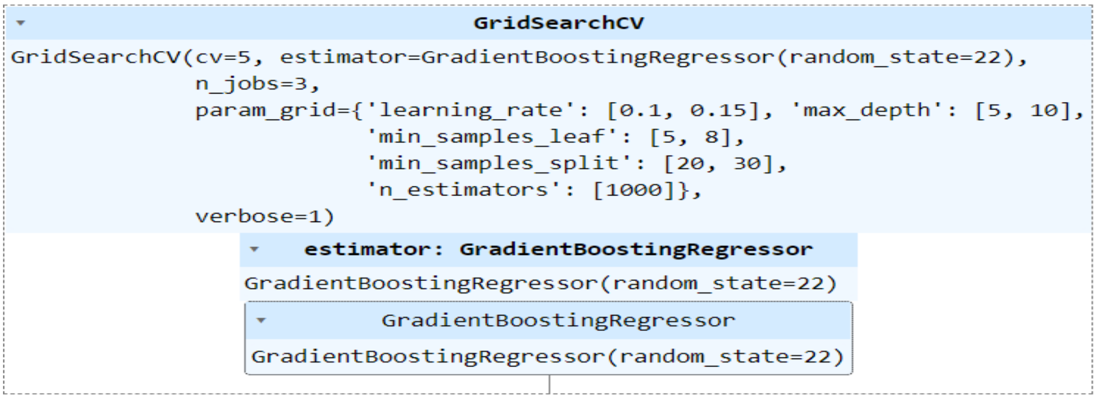
```
GradientBoostingRegressor(random_state=22)
```

```
GradientBoostingRegressor(random_state=22)
```

Fig 4.3 Grid search CV iteration 3

For Iteration 3 with n_estimator : 100 the score was increased to 83.3%

Iteration 4:


GridSearchCV
estimator: GradientBoostingRegressor
GradientBoostingRegressor

```
GridSearchCV(cv=5, estimator=GradientBoostingRegressor(random_state=22),
             n_jobs=3,
             param_grid={'learning_rate': [0.1, 0.15], 'max_depth': [5, 10],
                         'min_samples_leaf': [5, 8],
                         'min_samples_split': [20, 30],
                         'n_estimators': [1000]},
             verbose=1)
```

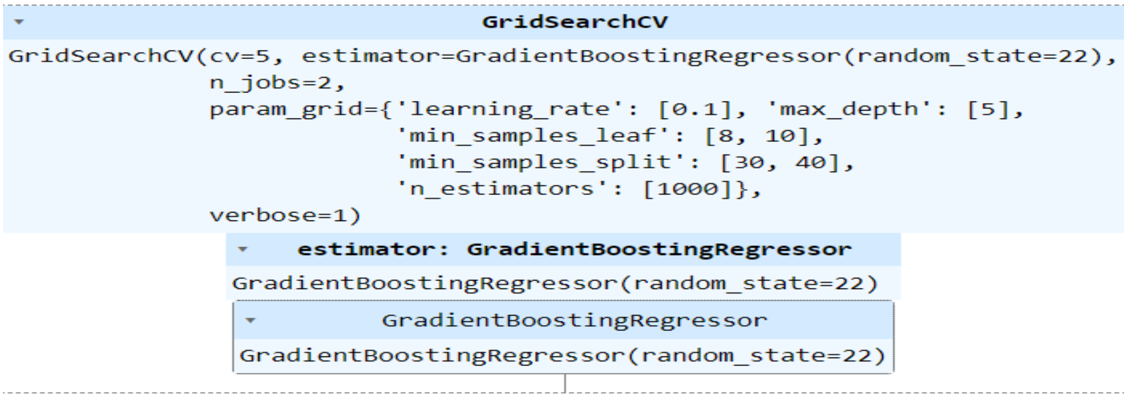
```
GradientBoostingRegressor(random_state=22)
```

```
GradientBoostingRegressor(random_state=22)
```

Fig 4.4 Grid search CV iteration 4

For Iteration 4 with n_estimator: 100 the score was increased to 83.4%

Iteration 5:



```
GridSearchCV
GridSearchCV(cv=5, estimator=GradientBoostingRegressor(random_state=22),
             n_jobs=2,
             param_grid={'learning_rate': [0.1], 'max_depth': [5],
                         'min_samples_leaf': [8, 10],
                         'min_samples_split': [30, 40],
                         'n_estimators': [1000]},
             verbose=1)
    estimator: GradientBoostingRegressor
    GradientBoostingRegressor(random_state=22)
          GradientBoostingRegressor
    GradientBoostingRegressor(random_state=22)
```

Fig 4.5 Grid search CV iteration 5

For Iteration 5 with n_estimator: 1000 the score was decreased to 83.1%

GridSearch CV method helped to identify the best params for the GBR model with 5 iterations only iteration 4 as given the best score compared to other iterations, hence finalizing the iteration 4 as the best params and build the GBR model

Performance of the model after Hyper-Tuning

Model	Test score	Test RMSE	MSE Test	MAE Test	Train Score	Train RMSE	MSE Train	MAE Train	MAPE Test
Gradient Boosting Using Grid search CV	0.804671	0.45188	0.204196	0.927447	0.267311	0.071455	0.804671	0.45188	0.204196

Table 5.1 Gradient Boosting using Grid Search CV

- For Iteration 4 with n_estimator: 100 the score was increased to 83.4%
- After improving the model - Training data score is improved from 87 % to 92.7%
- Test data score has improved from 78% to 81%
- Mean absolute percentage error has been drastically reduced from 16% to 12% which is very good for the GBR model
- RMSE value of both train & test has been reduced and stays close to 0 which indicates the model has predicted well in the test data with less error
- MSE value of both train & test has been reduced well model which indicates the model has performed well in both train & test

Training data score is improved from 87 % to 92.7%

Test data score has improved from 78% to 81%

Both Training & Test validation has been improved drastically with GBR model

Metrics of the model before tuning:

GBR – Model	Training Data	Test Data
Score	86.1%	78%
RMSE	0.359886	0.473904
MSE	0.129518	0.224585
MAPE	16%	16%

Table 5.2 Perfomance of Gradient Boosting Regressor before Hyper Tuning

Metrics of the model after Tuning:

GBR – Model	Training Data	Test Data
Score	92.7%	80.5%
RMSE	0.267311	0.45188
MSE	0.071455	0.204196
MAPE	12%	12%

Table 5.3 Perfomance of Gradient Boosting Regressor after Hyper Tuning

From the above tables, we can clearly say that after hyper-Tuning the GBR model using grid search CV has been significantly improved the model

Mean absolute percentage error has been drastically reduced from 16% to 12% which is very good for the GBR model

RMSE value of both train & test has been reduced and stays close to 0 which indicates the model has predicted well in the test data with less error

MSE value of both train & test has been reduced well model which indicates the model has performed well in both train & test.

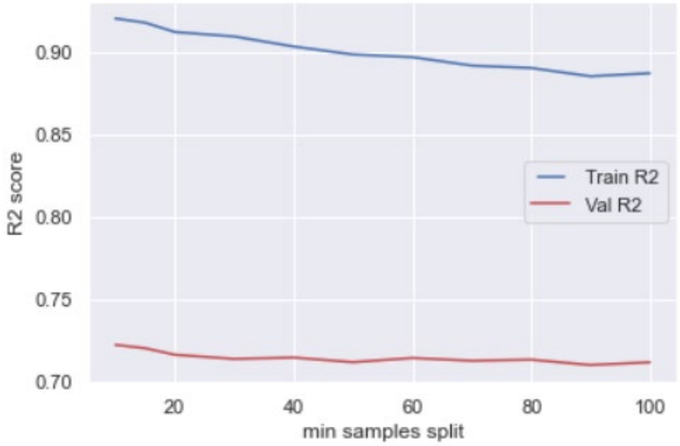


Fig 5.1 R2 Score vs sample split

Significant Variables from the model

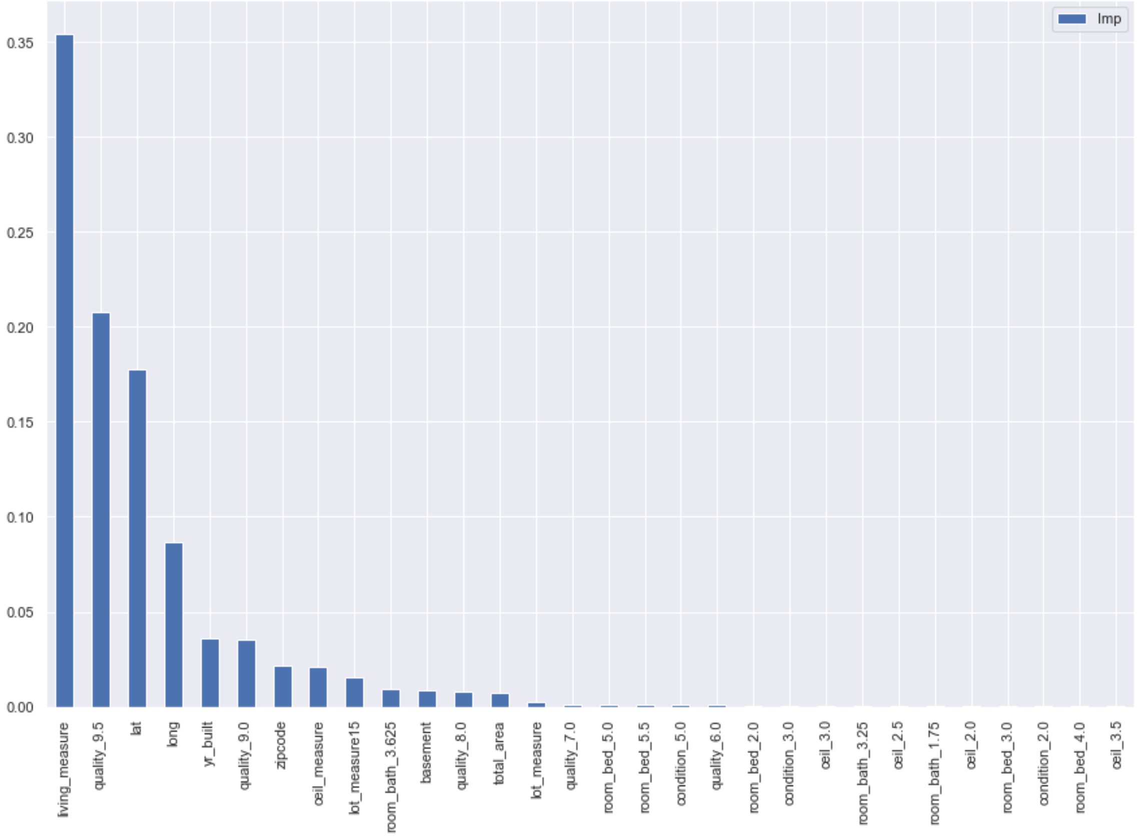


Fig 5.2 Significant variables used in modelling

Co-efficient of the model:

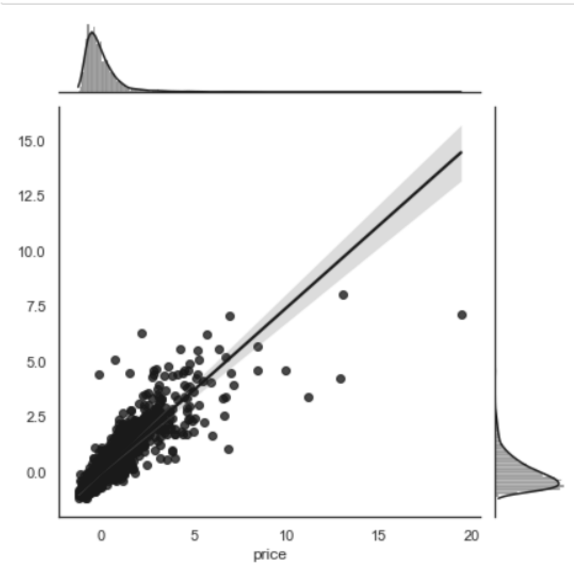
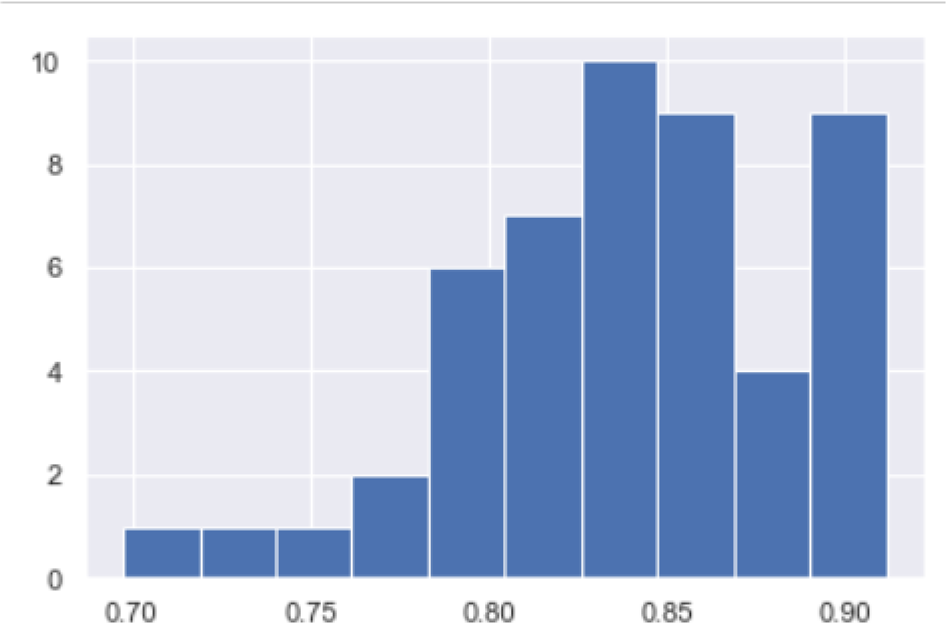


Fig 5.3 Co-efficient of the target variable

From the above Joint-Plot we can say that GBR model is positively co-related

Confidence Interval:



```
95.0 confidence interval 73.2% and 90.9%
```

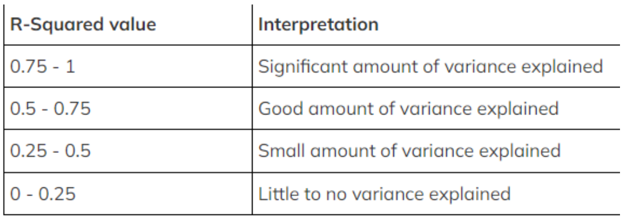
Fig 5.4 Confidence interval of the gradient boosting model

From the above graph we can conclude that GBR model confidence is 95% which means the model has will predict the outcomes correctly 95% of the time and only 5% of the times the model will fail to predict

5. Model validation - How was the model validated? Just accuracy, or anything else too?

For Evaluating the model, used RMSE, MAPE, MSE metrics

R-Squared is a measure of fit where the value ranges from 1, where all variance is explained, to 0 where none of the variance is explained. Of course, how good a score is will be dependent upon your use case, but in general R-Squared values would be interpreted as:



R-Squared value	Interpretation
0.75 - 1	Significant amount of variance explained
0.5 - 0.75	Good amount of variance explained
0.25 - 0.5	Small amount of variance explained
0 - 0.25	Little to no variance explained

Fig 6.1 R-Squared Metrics evaluation

Mean Squared Error (MSE) is the average squared error between actual and predicted values Squared error, also known as L2 loss, is a row-level error calculation where the difference between the prediction and the actual is squared. MSE is the aggregated mean of these errors, which helps us understand the model performance over the whole dataset.

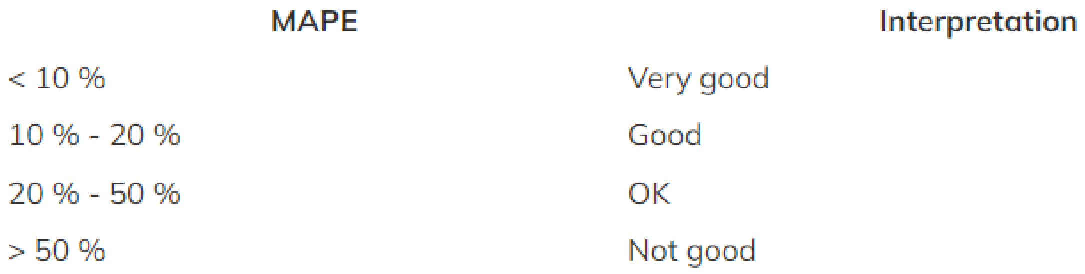
What is a good MSE value?

The closer your MSE value is to 0, the more accurate your model is. However, there is no 'good' value for MSE. It is an absolute value which is unique to each dataset and can only be used to say whether the model has become more or less accurate than a previous run.

MAPE (Mean Absolute Percentage Error) is an error metric used to measure the performance of regression machine learning models. It is a popular metric to use amongst data scientists as it returns the error as a percentage, making it both easy for end users to understand and simpler to compare model accuracy across use cases and datasets.

What is a good MAPE score?

MAPE returns error as a percentage, making it refreshingly easy to understand the 'goodness' of the error value. The lower the percentage, the more accurate the model, so 10% is better than 60%.



MAPE	Interpretation
< 10 %	Very good
10 % - 20 %	Good
20 % - 50 %	OK
> 50 %	Not good

Fig 6.2 MAPE Metrics

What is a good RMSE value?

The closer RMSE is to 0, the more accurate the model is. But RMSE is returned on the same scale as the target you are predicting for and therefore there isn’t a general rule for what is considered a ‘good’ value

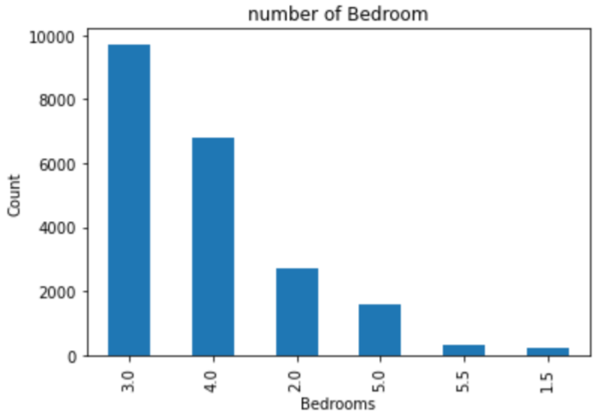
6. Final interpretation / recommendation

- From the above bar graph, we can see “Living_measure” is an important variable for buying a house
- “Quality_9.5” has been a second significant variable, so from this we can conclude that majority of the houses are sold were of quality – 9.5, people prefer the house with higher quality
- “yr_built” – has contributed well in the model, this indicates majority of the people who are buying houses preferred to know the year built
- “lat”, “long”, “ZipCode” – for a house the location is very important, these three columns combined together considered as significant columns. In the data with the zip code, we can able to find the city as Washington DC, with lat, long variables provide more information about the house, for ex: whether the house is located in the coastal region, center of the city etc.
- Based on the zip codes, lat, long – variables if we can able to bag the houses in to categories such as (coastal region, metropolitan, apartment, individual villa) which will help in predicting the price better
- Other significant variables are “room_bath3.65”, “room_bed5.0”, “lot_measure” also contributed well while predicting the model, from these we can conclude that houses sold were mostly with bathroom minimum 3 & bedroom 5
- Also, if the data can provide information such as (nearby railway Station, Airport, Hospital, schools & colleges) this information will also help to predict the price better
- This House-prediction was all about one city called Washington DC, where we are predicting only for this city, if we can add the house data similar to Washington DC cities like (Seattle, New York) the people who are buying/selling will get to know about the house prices across the metropolitan cities which helps them to conclude the price better & also gives them an additional choice of moving to another city or state
- April & may are the two months were most of the house were sold and the price of the house is higher when compared to the other months. I would not recommend a client to buy a house during the month of April & may

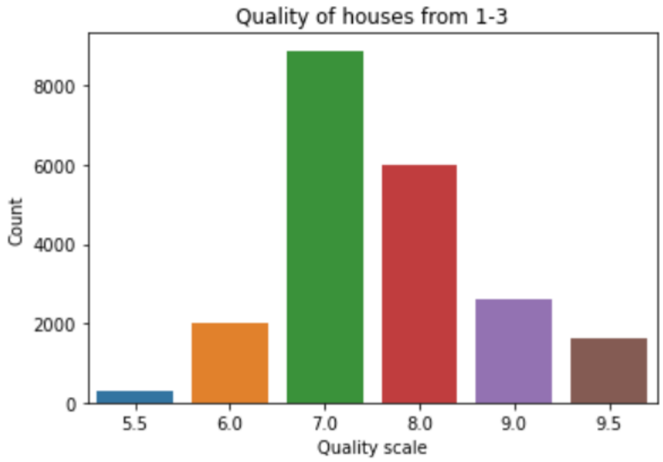
Appendix (Raw output from Jupiter note book)

Uni-Variate Analysis

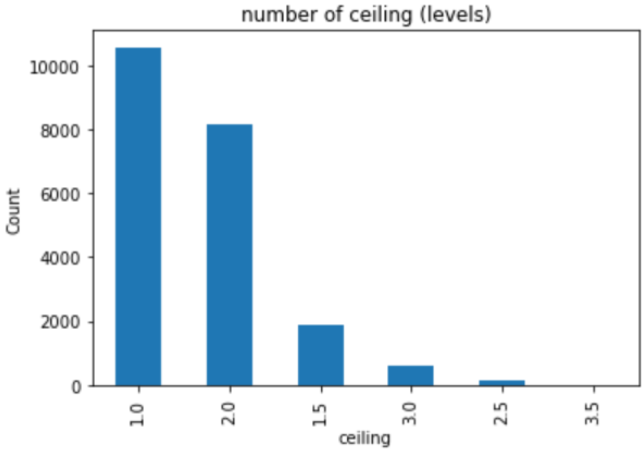
A1 – Bedroom Vs Count



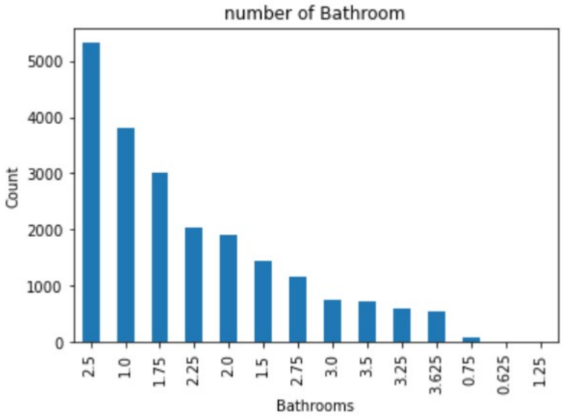
A2- Quality of houses



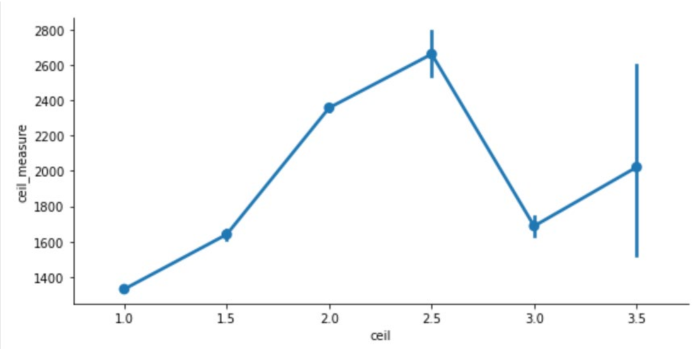
A3. Ceilings in the house



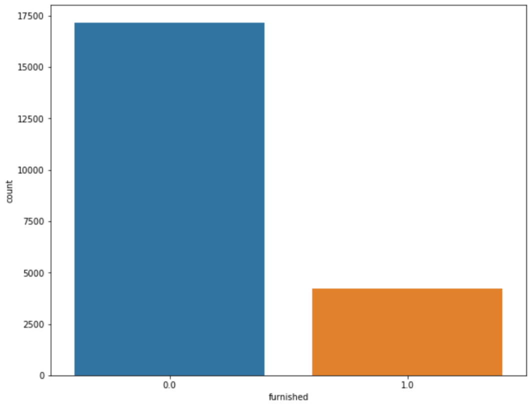
A4. Bathroom in the houses



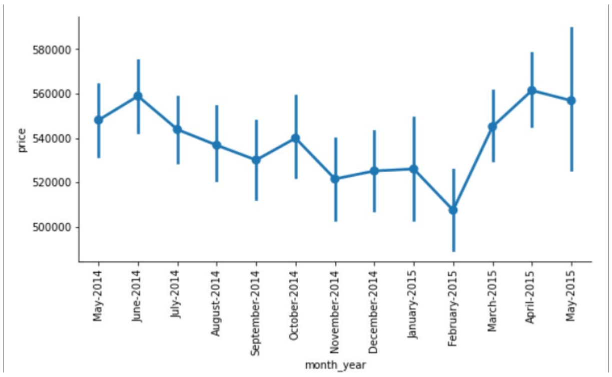
A5. Ceil vs Ceil_measure



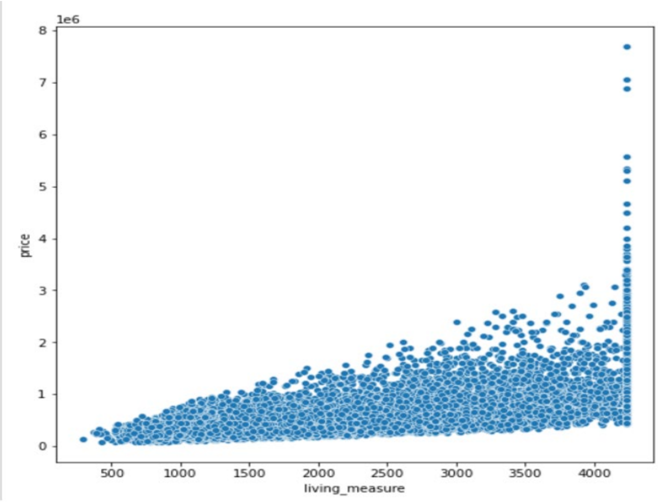
A6. Furnished Houses Count



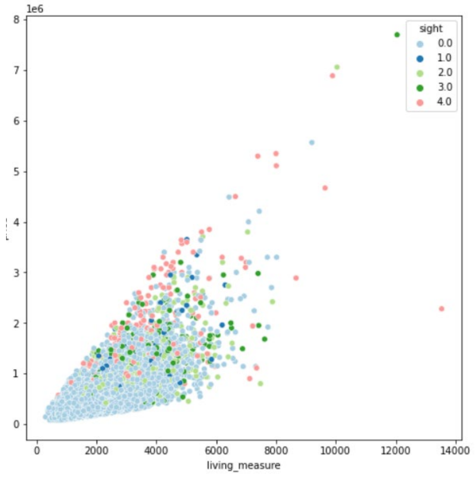
A7. Price vs Month & year



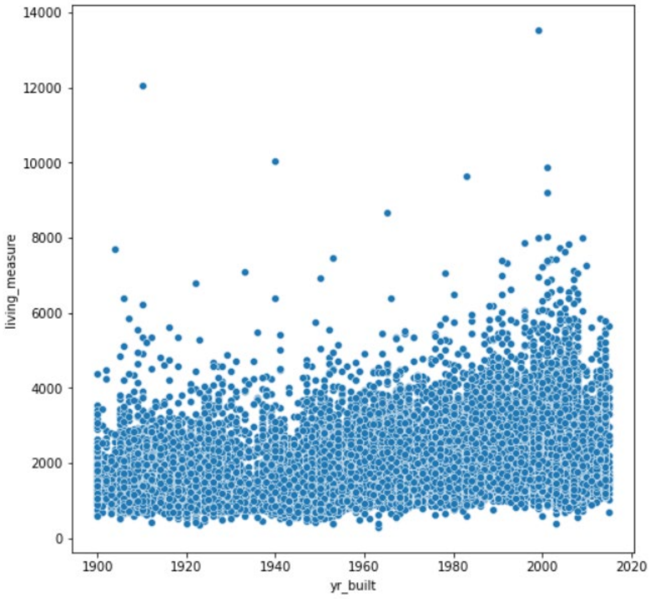
A8. Living measure vs Price



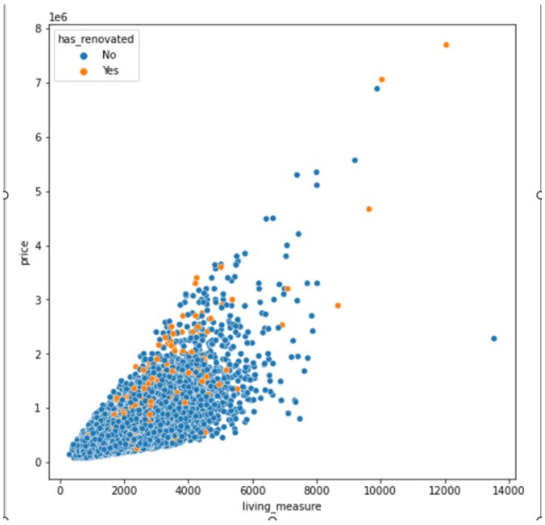
A9. Sight vs living measure



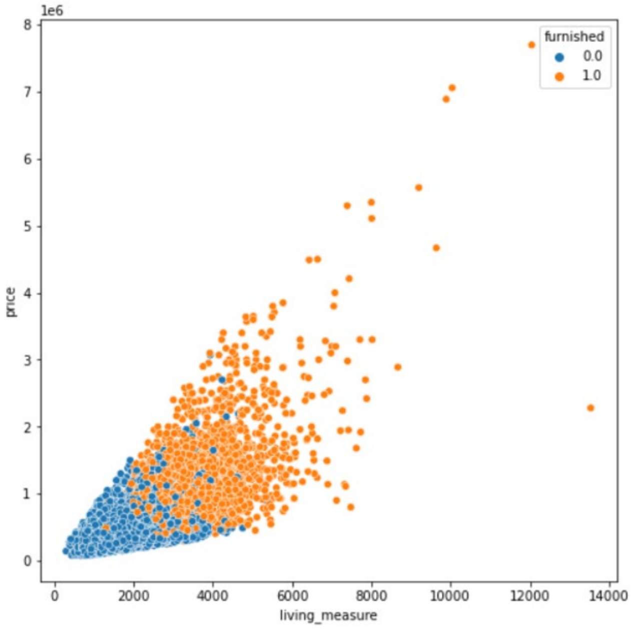
A10. Living measure vs year built



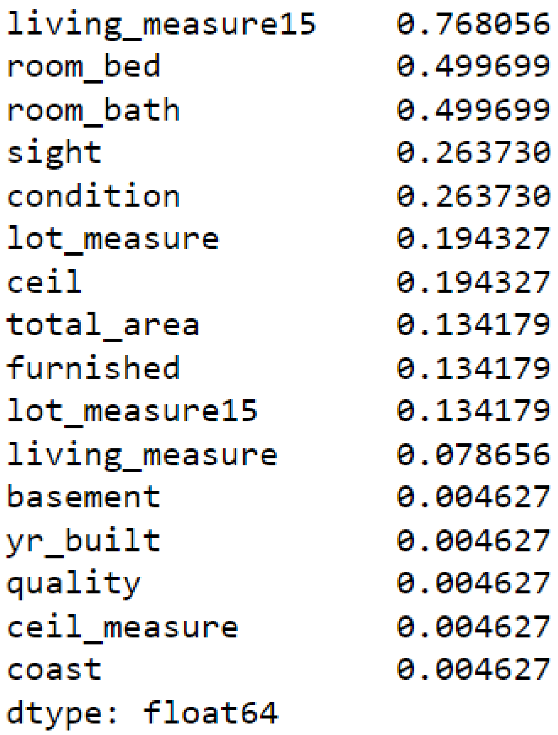
A11. Living measure vs Price vs renovated houses



A12. Living measure vs furnished

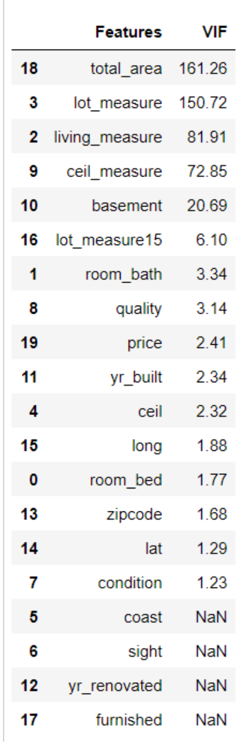


A13. Missing Data (In terms of %)



```
living_measure15    0.768056
room_bed            0.499699
room_bath           0.499699
sight               0.263730
condition           0.263730
lot_measure         0.194327
ceil                0.194327
total_area          0.134179
furnished           0.134179
lot_measure15       0.134179
living_measure      0.078656
basement            0.004627
yr_built            0.004627
quality             0.004627
ceil_measure        0.004627
coast               0.004627
dtype: float64
```

A14. VIF Table



	Features	VIF
18	total_area	161.26
3	lot_measure	150.72
2	living_measure	81.91
9	ceil_measure	72.85
10	basement	20.69
16	lot_measure15	6.10
1	room_bath	3.34
8	quality	3.14
19	price	2.41
11	yr_built	2.34
4	ceil	2.32
15	long	1.88
0	room_bed	1.77
13	zipcode	1.68
14	lat	1.29
7	condition	1.23
5	coast	NaN
6	sight	NaN
12	yr_renovated	NaN
17	furnished	NaN