A Project Report entitled

Prediction of the ideal price of Residential properties based on their characteristics

By

Amol Ghodke

(Date-15/02/2022)

Contents

Introduction

House price forecasting is an important topic of real estate. The literature attempts to derive useful knowledge from historical data of property markets. Machine learning techniques are applied to analyze historical property transactions in the region to discover useful models for house buyers and sellers.
Real estate sector often complies to demand based pricing and Hence there are rise and drop in the prices time and again So in order to tackle the issue in such a way that the customers are at a benefit we need to formulate a model which is successful in determining the price of the property based on its amenities rather than the demand. Also such a model would prove beneficial to the investors in order to understand the trend of the property valuation in the particular location.

Objectives to be achieved

- To identify the variables affecting house prices, e.g. area, number of rooms, bathrooms, etc.
- To create various models using different ML algorithms that quantitatively relates house prices with variables such as number of rooms, area, number of bathrooms, etc.
- To know and compare the accuracy of the model, i.e. how well these models can predict house prices.

Machine Learning Algorithms applied

- Multiple Linear Regression
- Decision Tree
- Random Forest
- XGB Boost Regressor
- Gradient Boosting Regressor
- KNN
- AdaBoostRegressor
- LightGBM
- Neural Network

The Data

- The data has been sourced from **https://www.kaggle.com/harlfoxem/housesalesprediction**

- This dataset contains house sale prices for King County, which includes Seattle. It includes homes sold between May 2014 and May 2015

- The data set contains 21 columns and 21613 rows along with the following features:-
  Date: *Date house was sold*
  Price: *Price of the house*
  Bedrooms: *Number of Bedrooms/House*
  Bathrooms: *Number of bathrooms/House*
  Sqft_Living: *square footage of the home*
  Sqft_Lot: *square footage of the lot*
  Floors: *Total floors (levels) in house*
  Waterfront: *whether the house has a view to a waterfront*
  View: *Has been viewed or not*
  Condition: *How good the condition is (Overall)*
  Grade: *grade given to the housing unit, based on King County grading system*
  Sqft_Above: *square footage of house apart from basement*
  Sqft_Basement: *square footage of the basement*
  Yr_Built: *Built Year*
  Yr_Renovated: *Year when house was renovated*
  Zipcode: *Zip*
  Lat: *Latitude coordinate*
  Long: *Longitude coordinate*
  Sqft_Living15: *Living room area in 2015(implies — some renovations)*
  Sqft_Lot15: *lot Size area in 2015(implies — some renovations)*

Data Analysis

- First and foremost the data has been checked for presence of missing values and the information about, what kind of data types are our variables.
- As for where the missing values have been encountered they were replaced by the median values.
- The describe function is used to see the percentile's and the statistical summary of the dataset.

.

	count	mean	std	min	25%	50%	75%	max
ID	21613.0	4.580302e+09	2.876566e+09	1.000102e+06	2.123049e+09	3.904930e+09	7.308900e+09	9.900000e+09
Price	21609.0	5.401984e+05	3.673890e+05	7.500000e+04	3.219500e+05	4.500000e+05	6.450000e+05	7.700000e+06
No of Bedrooms	21613.0	3.370842e+00	9.300618e-01	0.000000e+00	3.000000e+00	3.000000e+00	4.000000e+00	3.300000e+01
No of Bathrooms	21613.0	2.114757e+00	7.700693e-01	0.000000e+00	1.750000e+00	2.250000e+00	2.500000e+00	8.000000e+00
Flat Area (in Sqft)	21613.0	2.079861e+03	9.183029e+02	2.900000e+02	1.430000e+03	1.910000e+03	2.550000e+03	1.354000e+04
Lot Area (in Sqft)	21613.0	1.510464e+04	4.141992e+04	5.200000e+02	5.040000e+03	7.617500e+03	1.068500e+04	1.651359e+06
No of Floors	21613.0	1.494309e+00	5.399889e-01	1.000000e+00	1.000000e+00	1.500000e+00	2.000000e+00	3.500000e+00
Overall Grade	21613.0	7.623467e+00	1.105439e+00	1.000000e+00	7.000000e+00	7.000000e+00	8.000000e+00	1.000000e+01
Area of the House from Basement (in Sqft)	21613.0	1.788312e+03	8.279295e+02	2.900000e+02	1.190000e+03	1.560000e+03	2.210000e+03	9.410000e+03
Basement Area (in Sqft)	21613.0	2.915090e+02	4.425750e+02	0.000000e+00	0.000000e+00	0.000000e+00	5.600000e+02	4.820000e+03
Age of House (in Years)	21613.0	4.699486e+01	2.937341e+01	3.000000e+00	2.100000e+01	4.300000e+01	6.700000e+01	1.180000e+02
Renovated Year	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	5.350446e+01	9.800100e+04	9.803300e+04	9.806500e+04	9.811800e+04	9.819900e+04
Latitude	21613.0	4.756005e+01	1.385618e-01	4.715590e+01	4.747100e+01	4.757180e+01	4.767800e+01	4.777760e+01
Longitude	21613.0	-1.222139e+02	1.408266e-01	-1.225190e+02	-1.223280e+02	-1.222300e+02	-1.221250e+02	-1.213150e+02
Living Area after Renovation (in Sqft)	21613.0	1.986532e+03	6.853891e+02	3.990000e+02	1.490000e+03	1.840000e+03	2.360000e+03	6.210000e+03
Lot Area after Renovation (in Sqft)	21613.0	1.276846e+04	2.730418e+04	6.510000e+02	5.100000e+03	7.620000e+03	1.008300e+04	8.712000e+05

- From the describe function the following kind of inferences can be drawn:
  - When we observe the number of bedroom columns , the dataset has a house where the house has 33 bedrooms , seems to be a massive house whereas there is also a house with no bedrooms
  - Maximum square feet is 13,450 where as the minimum is 290.

- As for the next step the categorical variables that are No of times visited, Condition of the House and the Waterfront view are converted to the numerical values as for like the Condition of the house is rated as such Bad: 1, Okay: 2,Fair : 3,Good : 4,Excellent : 5 while the Waterfront view is mapped such that the yes value equals to 1 and the no value equals to 0.
- Also in order to take into consideration whether the renovation of the house would affect its price firstly its checked whether the house was ever renovated and then if yes how many hears have passed since its renovation is calculated and stored in a column named Years since renovation.

Statistical Analysis of Price Feature:-

```
##Statistical Analysis of Price Feature
print("Price Min")
print(data['Price'].min())
print("Price Mean")
print(data['Price'].mean())
print("Price Median")
print(data['Price'].median())
print("Price Max")
print(data['Price'].max())
print("Price Std")
print(data['Price'].std())
print("Skewness: %f" % data['Price'].skew())
print("Kurtosis: %f" % data['Price'].kurt())
```

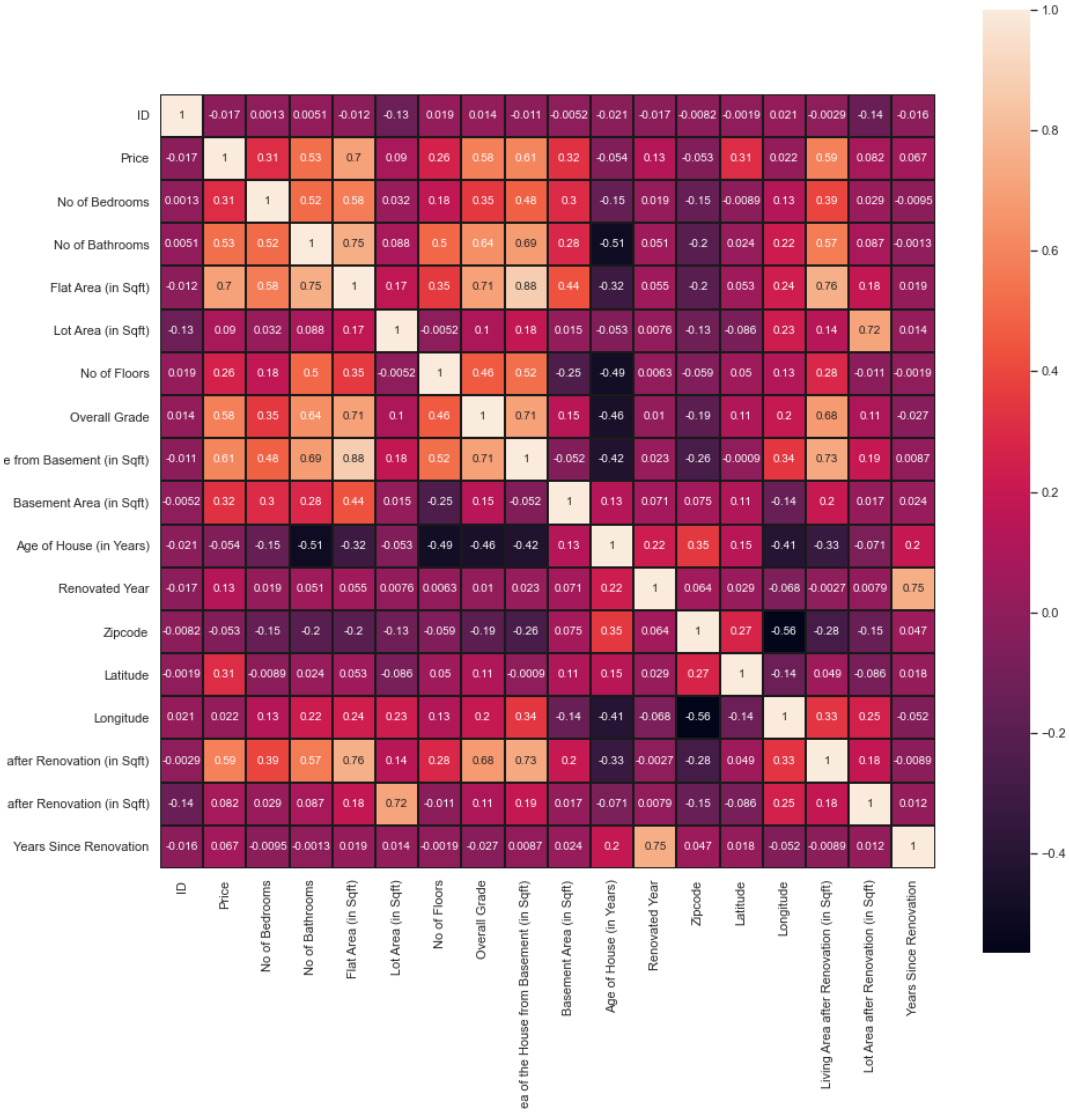
```
Price Min
75000.0
Price Mean
540198.4357443658
Price Median
450000.0
Price Max
7700000.0
Price Std
367388.99446065293
Skewness: 4.021472
Kurtosis: 34.517725
```

Following inference is drawn:-

- Minimum price of the house is 75000.
- Mean of the price of the house is 540198.4357
- Median of the price is 450000.
- Maximum price is 7700000.
- Standard deviation of the price data is 367388.9944.
- Skewness value is 4.021472. It refers to a distortion or asymmetry that deviates from the symmetrical bell curve, or normal distribution, in a set of data.
- Kurtosis is a statistical measure that defines how heavily the tails of a distribution differ from the tails of a normal distribution. For our data kurtosis is 34.5177.

Data Visualization

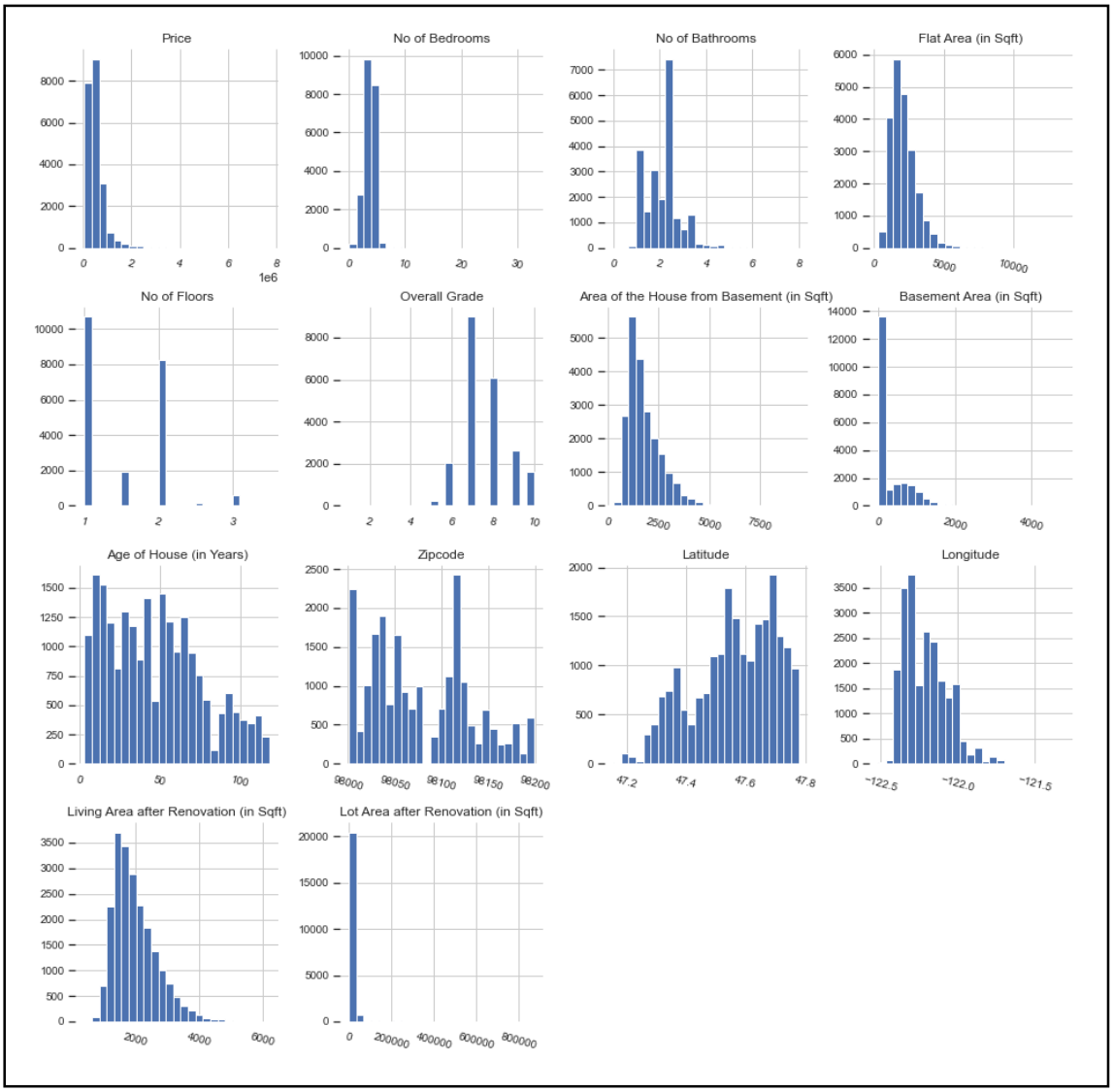
- The first visualization that is performed is the correlation heatmap that depicts the correlation values and the nature and strength of the correlation using colours.



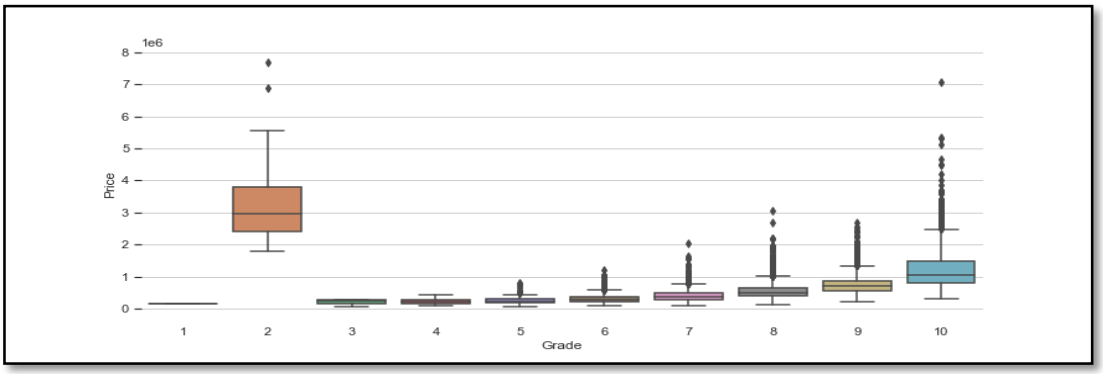
Following Figure represents the heatmap for the data the inference that can be drawn from the following fig are:

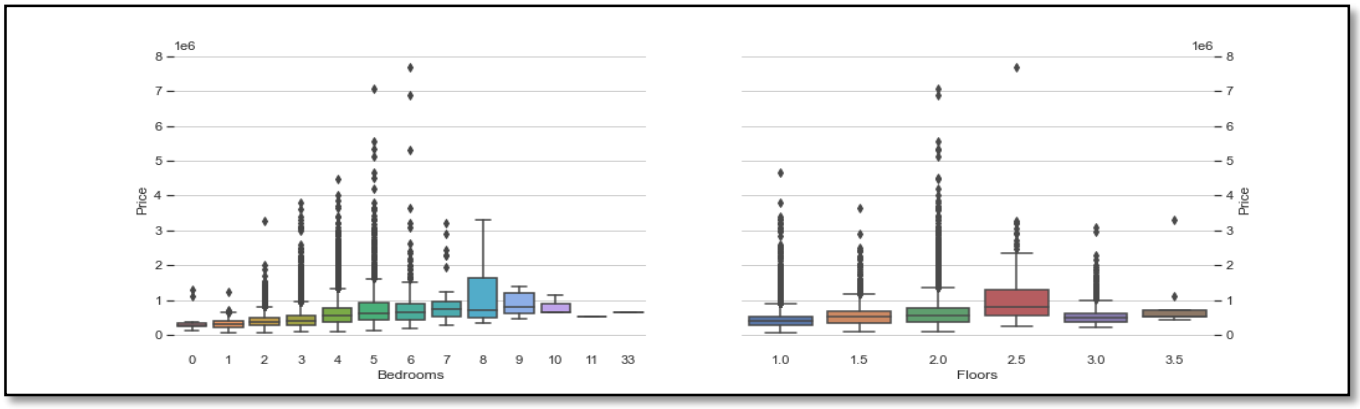
- The Price of the house has a positive correlation with all the features except the Age of the house.
- Also there is strong positive correlation between the price of the house and the flat area, and a good correlation between the price and no of floors, the living area after renovation, and the area of house from the basement.
- Even though there is a positive correlation between years since renovation, lot area, lot area after renovation and the price the value is very small.

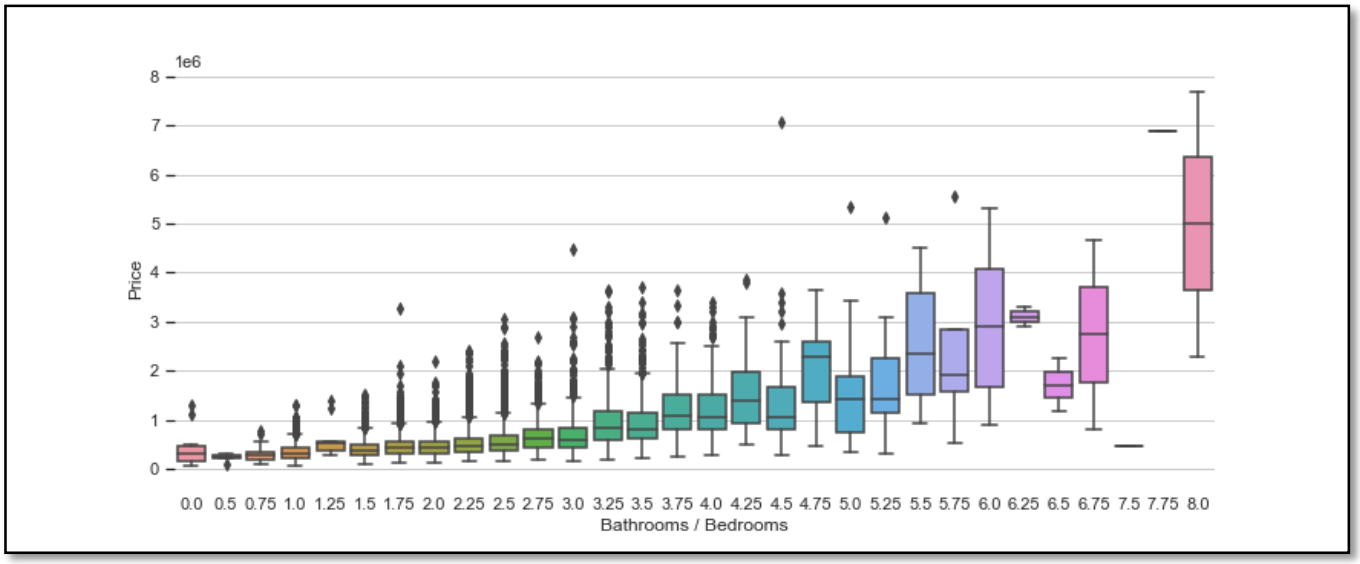
- The next visualization performed is Histograms which organizes a group of data points for different number of characteristics like Price, no. of bedroom, Flat area, etc.

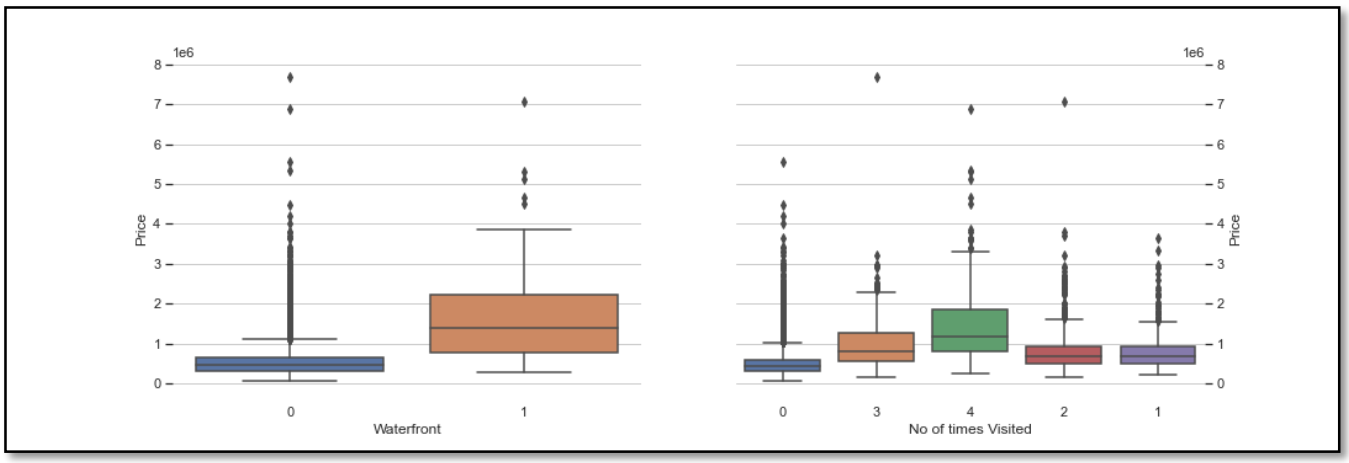


- The various box plots have been plotted to study the various features in relation to the sale price.



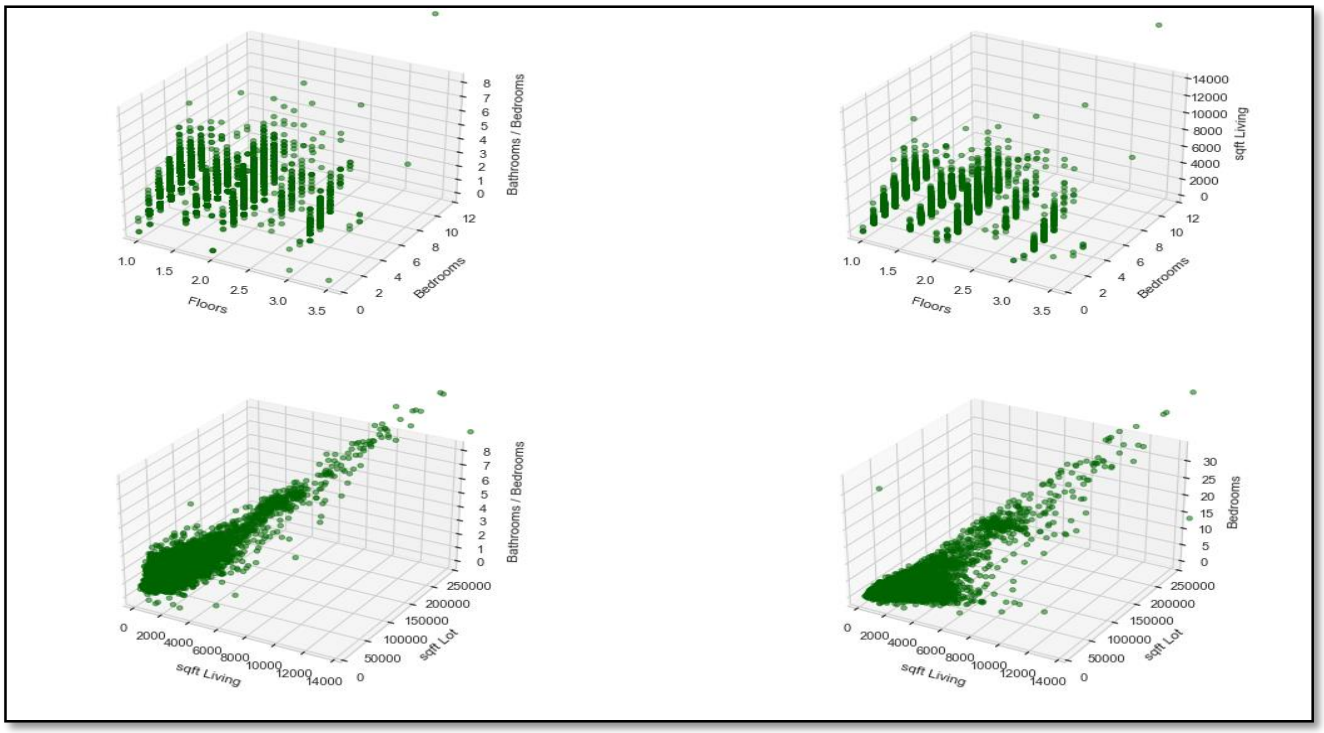






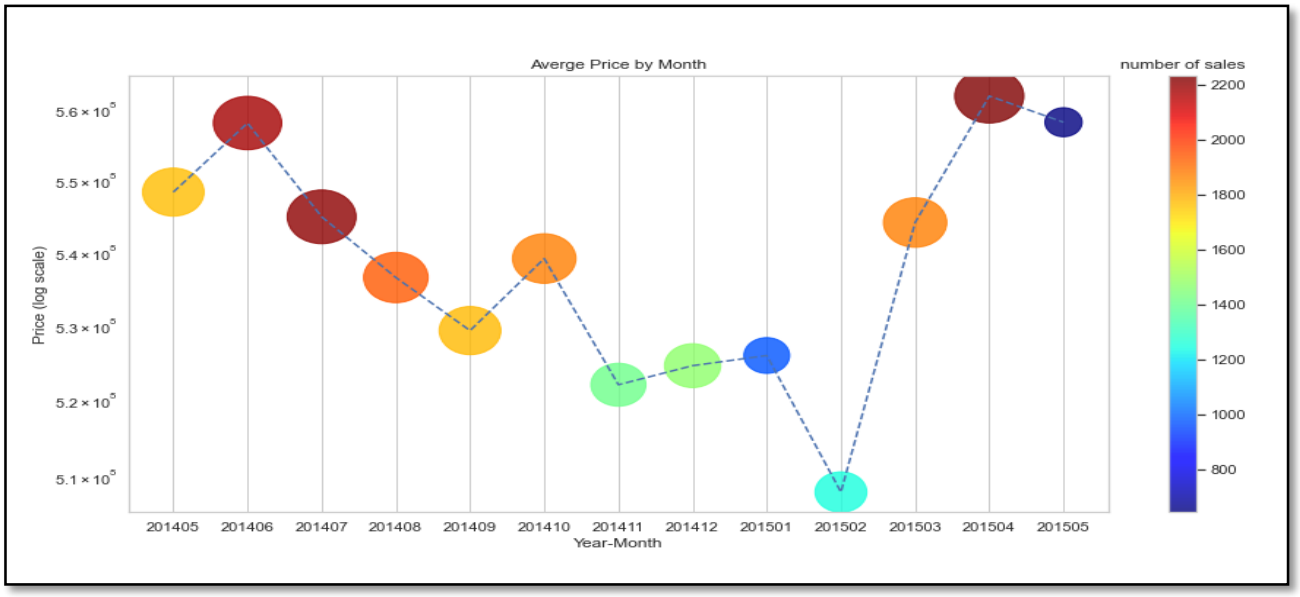
- Clearly, as the features "bathrooms", "grade", "bedrooms" increase, so does the Sale Price. This is most evident in case of the features bathrooms and grade.
- Waterfront is a binary variable, the distribution of housing price is quite different for the two groups. So it is likely to be a good predictor.
- The following plots depict the relationship between the three variables

- If markers are close to forming a straight line in any direction, then the correlation between those variables is high. If the markers are equally distributed in the graph, the correlation is low, or zero.



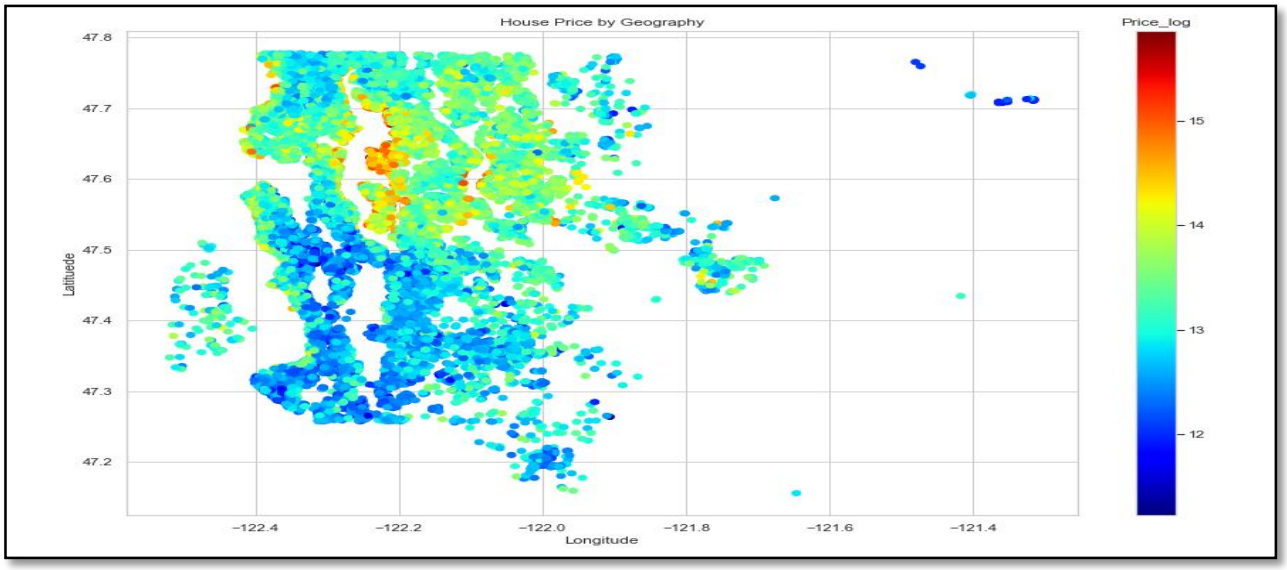
The seasonality of housing price

- In the following plot the y axis is representing the housing price on log scale, the color of the bubbles shows the number of sales. When the number of sales is high, the housing price is also high, which is quite reasonable. But sometimes, the market might response a bit slowly, that's why we can see in May of 2015, even though there are not many of sales, the price is still high. We can see there is clear seasonality, so to create some month indicators might be a good idea.
- From the figure we can say that the lowest price was in the 2015 Feb (02) and number of sales is in between1200 to 1400.
- Highest price is in the 2015 April (04) and the number of sales is in nearly 2200.



Housing Price and Location

- As we can see from this graph, the northern area is generally more expensive than the southern.
- Most of the million-dollar houses (right stars in the graph) are around the hollow area in the north. That should be Lake Washington. Latitude and Longitude can be very good features if we want to build a model to predict the housing price. And they are on a more granular level than zip code. So it should provide more information than zip code.
- Another point to note from the graph is that the relationship between latitude/longitude and housing price is not linear, so linear model might not work very well. But tree based models should work better in this case.



Dataset Preparation (Splitting and Scaling)

Data is divided into the Train set and Test set. We use the Train set to make the algorithm learn the data's behavior and then check the accuracy of our model on the Test set.

- Features (X): The columns that are inserted into the model will be used to make predictions i.e. all the features except the price
- Prediction (y): Target variable that will be predicted by the features i.e. Price

```
X = data.drop('Price',axis =1).values
y = data['Price'].values
print(X.shape, y.shape)
y = np.reshape(y, (-1,1))
y.shape
```

- Train data is set to be as 90% and 10% of the data to be the test data .
- Feature scaling will helps to view all the variables from the same lens (same scale), it will also help the models learn faster.

```
In [124]: X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.10)

In [125]: from sklearn.preprocessing import StandardScaler
          sc_x=StandardScaler()
          X_train=sc_x.fit_transform(X_train)
          X_test=sc_x.transform(X_test)

In [126]: print(X_train.shape,y_train.shape)
          print(X_test.shape,y_test.shape)

          (19448, 18) (19448, 1)
          (2161, 18) (2161, 1)
```

Evaluation Data frame for All the Models

Comparative analysis is performed based on various performance metrics such as k-Fold Cross Validation, Adjusted R-Square, R-Square and Root Mean Square Error.

- K fold Cross validation:- By using k-fold cross-validation, we are able to "test" the model on k different data sets, which helps to ensure that the model is generalizable. The cross_val_score from sklearn is used to calculate the cross validation score.
- Adjusted R^2 :-It indicates how well data points fit a curve or line, but adjusts for the number of terms in a model. The Adjusted R-squared takes into account the number of independent variables used for predicting the target variable. In doing so, we can determine whether adding new variables to the model actually increases the model fit.

$$Adjusted\ R^2 = \{1 - [\frac{(1 - R^2)(n-1)}{(n-k-1)}]\}$$

- R-squared : -It is a statistical measure of how close the data are to the fitted regression line. It is the percentage of the response variable variation that is explained by a linear model. In general, the higher the R-squared, the better the model fits the data.

  R-squared = (TSS-RSS)/TSS

  Here, TSS= Total sum of squares and RSS= Residual sum of squares

- Root Mean Square Error:- Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). It tells you how concentrated the data is around the line of best fit.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

The following dataframe will be used to store the values of the various performance metrics for all the models as it will help us in comparing them with ease.

```
evaluation = pd.DataFrame({'Model': [],
                           'Details':[],
                           'Root Mean Squared Error (RMSE)':[],
                           'R-squared (training)':[],
                           'Adjusted R-squared (training)':[],
                           'R-squared (test)':[],
                           'Adjusted R-squared (test)':[],
                           '5-Fold Cross Validation':[]})
```

Machine Learning Models

In this project a number of machine learning algorithms are used to predict the price of the house on the basis of all the features that are available. In past various linear based and non-parametric algorithms are used for forecasting real estate price but are ineffective in observing anomalies in the data. Therefore, we used ensemble-based algorithm like Random Forest and deep learning-based algorithm like Neural Networks to perform a comparative analysis against the typical used algorithms like Multiple Linear Regression, Polynomial Regression and K Nearest Neighbors. As explained before each of these models is subjected to the performance metrics decided earlier.

Model 1: **Multiple Linear Regression**

The most important and common question is whether there is a statistical relationship between an explanatory variable (usually denoted by X) and a response variable (usually denoted by y). To solve this problem, a typical way is to apply regression analysis to model and quantify this statistical relationship.

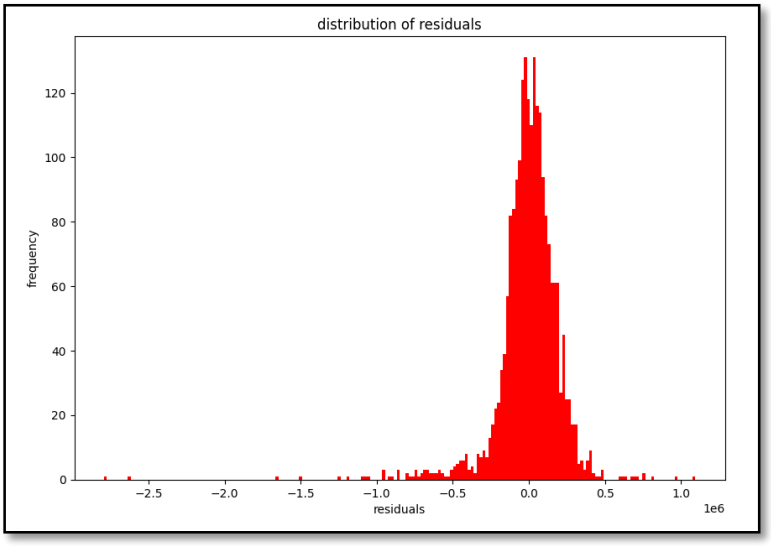
The goal here is to model a relationship between y , the dependent variable and x , .. , x_p , the independent variable where p is the number of these variable . The general form of this is given by the following equation : y = f(x) + ε

Since we had several independent variables (features) to predict the dependent variable (Price), This was in the case of multiple linear regression (the ^ means estimation), given by :

$$\hat{y} = \hat{f}(x) + \varepsilon$$
$$\hat{f}(x) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

Where ε is the error term, f(x) is the estimation of the function of x , $\hat{y}$ the predicted result i.e. the predicted price, $\beta_{0.....p}$ are the unknown parameters that we want to find, $x_{0....p}$ the independent variable i.e. the various features like no of bedrooms , flat area etc.

By visualizing the residual it can be seen that it is normally distributed (proof of having linear relationship with the dependent variable). Below given figure is the distribution of residuals for the model deployed.



The model yields an accuracy of 0.66277 i.e. 66.2277 %

Model 2: Gradient Boosting Regressor

Gradient Boosting for regression builds an additive model in a forward stage wise fashion. It allows for the optimization of arbitrary differentiable loss functions. In each stage, a regression tree is fit on the negative gradient of the given loss function. The idea of boosting came out of the idea of whether a weak learner can be modified to become better. A weak hypothesis is defined as one whose performance is at least slightly better than random chance. The Objective is to minimize the loss of the model by adding weak hypothesis using a gradient descent like procedure. This class of algorithms was described as a stage-wise additive model. This is because one new weak learner is added at a time and existing weak learners in the model are frozen and left unchanged.

GradientBoostingRegressor(n_estimators=1000, max_depth=5, loss='ls ', min_samples_split=2 ,learning_rate=0.1)

The model yields an accuracy of 85.98%

Model 3: k-Nearest neighbor's regression

k-Nearest neighbors (k-NN) is a non-parametric algorithm that can be used for both classification and regression problems. Since we are dealing with regression, we are concerned with determining a numerical value for the unknown variable. Taking the mean of the nearest neighbors is a simple approach. However some of those k points could be much more distant than others. To combat that, weights can be added to each neighbor according to some function dependent on the distance. One approach is to weigh them inversely proportional to the distance. In many cases the latter approach, inverse distance weighted average, performs better than uniform weights, i.e. no weights. The number of neighbors to include in the calculation (i.e. the size of k) is in practice determined by trial, comparing prediction errors for different values of k.

Here in this model the value of k is set as 5 which yields an accuracy of 0.7724 i.e. 77.24%

Model 4: Random forest regression

Random forest is an algorithm which can be used both for classification and regression. Random forest models are constructed by using a collection of decision trees based on the training data. Instead of taking the target value from a single tree, the Random forest algorithm makes a prediction on the average prediction of a collection of trees.

The number of trees in the Random forest is an important hyper parameter of the algorithm called 'n_estimators' and the more trees used the more will overfitting be prevented. In this model we have estimated the value of n_estimators to be 400.The tradeoff however, is an increase in the computation time needed. Another hyper parameter of the Random forest algorithm is the 'criterion' hyper parameter which determines what error metric to use for measuring the quality of splits in the trees in the Random forest. The value can be either mean squared error or mean absolute error. A third hyper parameter that is particularly interesting for this study since there are

a lot of features in the data set is 'max_features' which controls the number of features to consider when building the trees.

The model yields an accuracy of 87.472 %

Model 5: XGBoost Regressor

The trees in XGBoost take into consideration the past prediction value for a certain datum and create a new tree that splits the existing data as best as possible to maximize the 'gain' in prediction (if tree-splits are created that don't lead to too much gain, based on a hyper-parameter set threshold, the tree is then pruned to prevent overfitting). Gradient Boosted trees create several models, taking past trees and factoring in their predictions to create a new tree, with the purpose of minimizing their prediction error. Once trained, the algorithm will add all built trees' predictions to make its final regression.

```
xgb = xgboost.XGBRegressor(n_estimators=800, learning_rate=0.05,reg_alpha= 0.1, subsample=0.95,colsample_bytree=1, max_depth=6)
```

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

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
             gamma=0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.05, max_delta_step=0,
             max_depth=6, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=800, n_jobs=8,
             num_parallel_tree=1, predictor='auto', random_state=0,
             reg_alpha=0.1, reg_lambda=1, scale_pos_weight=1, subsample=0.95,
             tree_method='exact', validate_parameters=1, verbosity=None)
```

The XGBoost model yields an accuracy of 86.7905%

Model 6: Light Gradient Boosting Machine (LightGBM)

LightGBM is a gradient boosting framework that uses a tree-based learning algorithm. LightGBM has faster training speed with lower memory usage compare to XGBoost . Moreover, it can also support parallel and GPU learning or handle large scale of data. Even though both of the aforementioned boosting models follow Gradient Boosting Algorithm, XGBoost grows tree level-wise and LightGBM grows tree leaf-wise. There are also detailed differences in modelling making them outstanding among different gradient boosting models.

The model with following parameters is applied:-

LGBMRegressor(learning_rate=0.05, metric='mae', n_estimators=1000, objective='regression')

The Light GBM Model yields an accuracy of 85.021%

Model 7: AdaBoost Regressor

AdaBoost algorithm, short for Adaptive Boosting, is a Boosting technique used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights assigned to incorrectly classified instances. Boosting is used to reduce bias as well as variance for supervised learning.

It works on the principle of learners growing sequentially. Except for the first, each subsequent learner is grown from previously grown learners. In simple words, weak learners are converted into strong ones. The AdaBoost algorithm works on the same principle as boosting with a slight difference.

The model with following parameters is applied:-

AdaBoostRegressor (n_estimators=45, learning_rate=0.2, loss='exponential')

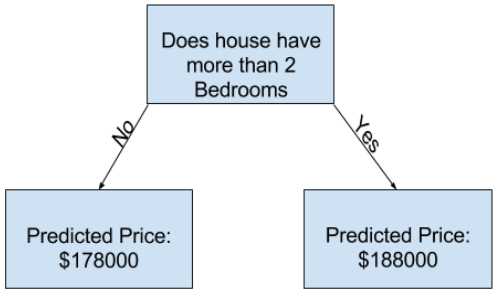
The accuracy is found to be as low as 65.2099%.

Model 8: Decision Tree Regressor

Decision Tree Regression, as the name suggests it uses tree like structure to build regression and classification models. 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 has two or more branches, each representing values for the attribute tested. Leaf node represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

The decision tree model yields an accuracy of 69.3525%

The following fig represents a sample decision tree.



Comparative analysis of all the above applied Algorithms

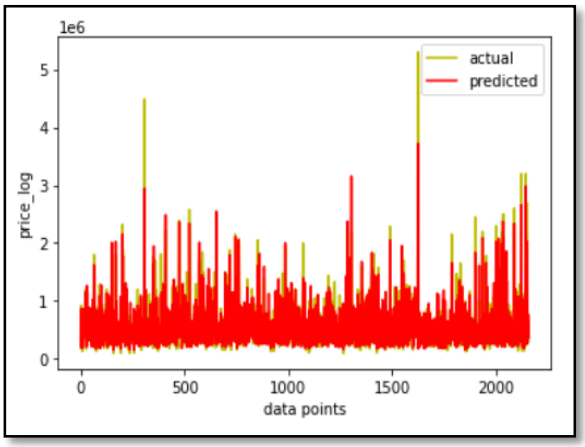
The following table depicts the 5 fold cross validation score, R squared , Adjusted R squared and RMSE values of all the above discussed models:

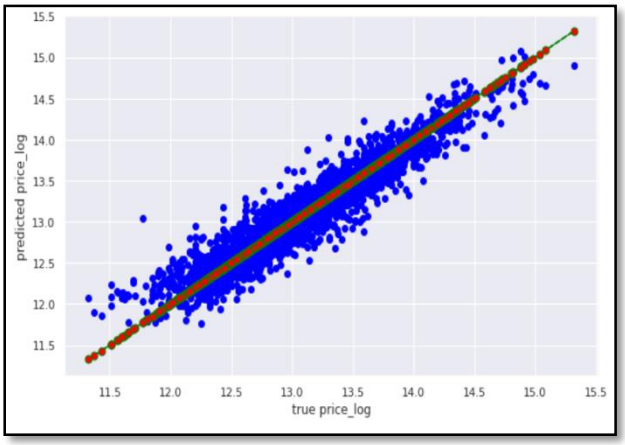
	Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	Adjusted R-squared (training)	R-squared (test)	Adjusted R-squared (test)	5-Fold Cross Validation
7	KNN(neighbhors=5)	All Features	175088.725	0.848	0.848	0.772	0.771	0.502
0	Multiple Linear Regression(feauture scaling)	All features	213145.468	0.677	0.676	0.663	0.660	0.668
3	AdaBoostRegressor	All features	216493.303	0.705	0.705	0.652	0.649	0.673
5	Decision Tree regression	All Features	203195.640	0.999	0.999	0.694	0.691	0.738
6	Random Forest Regression	All Features	129913.940	0.982	0.982	0.875	0.874	0.874
4	LightGBM	All features	124950.054	0.976	0.976	0.884	0.883	0.879
2	Gradient Boosting Regressor	All features	127232.171	0.985	0.985	0.880	0.879	0.887
1	XGBoost	All features	123895.973	0.980	0.980	0.886	0.885	0.892

From the table it is evident that the XGBoost model and the Gradient Boosting Regressor model are the best two models out of all the implemented models as they have the Highest R squared (test) values , Adjusted R squared (test) values and even the cross validation score is higher than all the models.

The worst model proved to be the Adaboost Regressor followed by the multiple linear Regression as they have very low scores for R squared and Adjusted R squared even though their cross validation score are higher than the KNN.

Let's focus on the XGBoost model and look at the plots in regards to the predictions the model makes:-



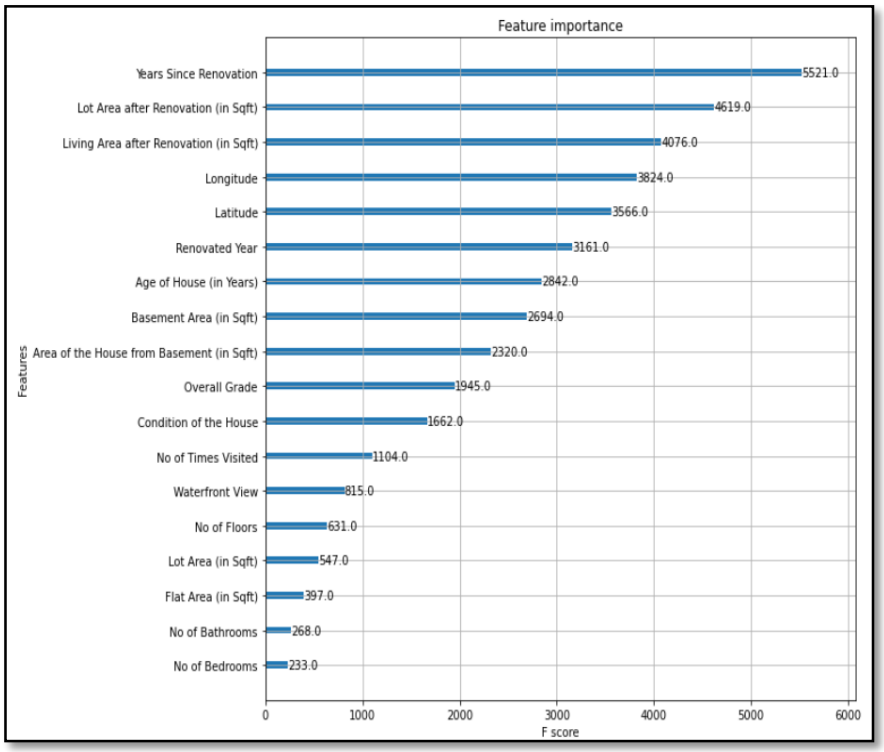


id	Test Data	Predicted Price	Difference
8562770430	567500	552626	14873
5694500105	595000	598267	3267
1446400715	279999	238642	41357
8677300720	616000	615878	121
7137970130	339999	316651	23347
7853220210	563500	609668	46168
2617370090	338500	435745	97245
8648100130	306500	307281	781
7702010030	551000	544181	6818
2008000130	360499	409669	49169

The following table depicts the difference between the actual price values from the dataset and the prices predicted by our model and it is evident that the difference is low.

Also from the graphs above it can be clearly inferred that the model effectively predicts the house prices.

Feature Importance For XGBoost Model



From adjacent bar graph it is evident that the following features have the most importance:-

- Years Since Renovation
- Lot area after Renovation
- Living Area after Renovation
- Longitude
- Latitude

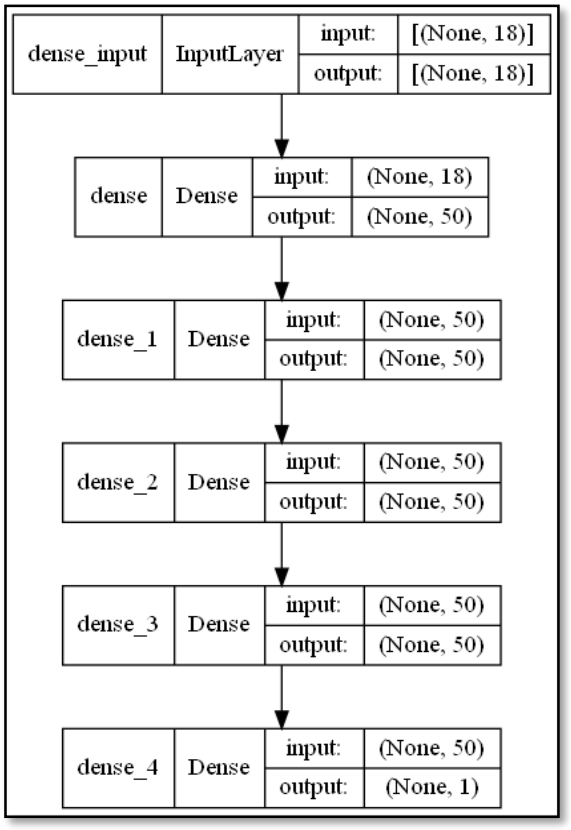
Neural Network Model

MLP is the multilayer perceptron it is a part of artificial neural networks it has the same structure of a single layer perceptron with one or more hidden layers, in this the hidden layer will be directly connected to the input layer here the input values are presented in the perceptrons, the perceptrons will classify any linear separable set of inputs if the input values presented to the perceptron, and if the predicted output is as same as the desired output, then the performance is considered satisfactory and we know that no changes to the weights are made, if it does not match then the weights need to be changed to reduce the error.

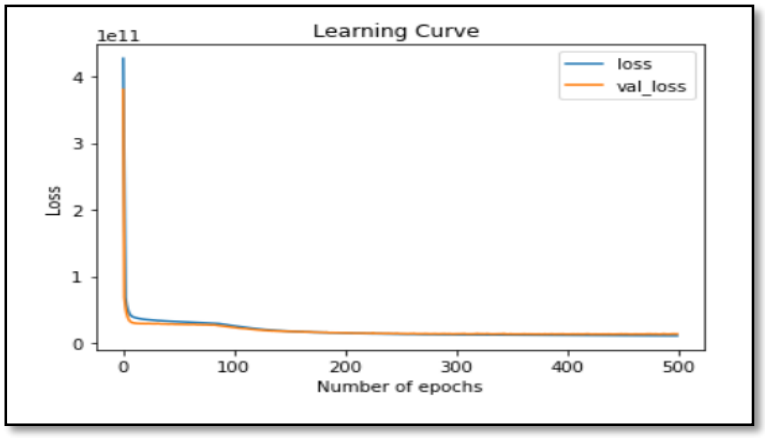
Here we have developed a sequential model with 1 input layer , 3 hidden layers and 1 output layer with 50 neurons each with relu activation function the optimizer is adam and the loss calculated is the mean squared error.

This model is then fit for 500 epochs with 128 batch size.

Following is the model summary and the representation of the network:-




```
model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 50)                950

 dense_1 (Dense)             (None, 50)                2550

 dense_2 (Dense)             (None, 50)                2550

 dense_3 (Dense)             (None, 50)                2550

 dense_4 (Dense)             (None, 1)                 51

=================================================================
Total params: 8,651
Trainable params: 8,651
Non-trainable params: 0
_________________________________________________________________
```



The above Learning curve ensures us that the model is a good fit as the loss and the validation loss converge steadily.

The Mean Absolute Error ,Mean Squared Error (MSE) ,Root Mean Squared Error (RMSE), Explained Variance Score ,Rsquared matrices are used to evaluate the performance of the model.

	Mean Squared Error (MSE)	Root Mean Squared Error (RMSE)	Mean Absolute Error (MAE)	Explained Variance Score	R squared
0	1.373929e+10	117214.698263	68932.821271	0.881546	0.88028

Here we can observe that the R squared value is significantly higher than any of the algorithms except the XGBoost that we had applied hence we can say that the Neural network model will be best suited for the prediction of the house prices as it can be further enhanced to achieve more accuracy.

Conclusion

- The models are able to predict the house prices successfully with an acceptable accuracy.
- The XGBoost Regressor model had the highest R squared value making it the best model among the other model such as Light GBM, Gradient Boosting Regressor, Random Forest, Decision Tree etc.
- Years Since Renovation , Lot area after Renovation , Living Area after Renovation , Longitude, Latitude are observed to be the most dominant features.
- No of Bathrooms, No of Bedrooms and Flat area proved to the least useful features to predict the price.
- The AdaBoost Regressor model proved to be the worst performing model with the Lowest R squared value.
- The Neural Network model proved to be very effective with the R squared value as High as 0.88.There is a further scope of improvement in the Neural Network model which can lead to a better performing model.

References

1) Ayush Varma, Abhijit Sarma, Sagar Doshi, Rohini Nair - "Housing Price Prediction Using Machine Learning and Neural Networks" 2018, IEEE.
2) G.Naga Satish, Ch.V.Raghavendran, M.D.Sugnana Rao, Ch.Srinivasulu "House Price Prediction Using Machine Learning". IJITEE, 2019.
3) CH. Raga Madhuri, G. Anuradha, M. Vani Pujitha -" House Price Prediction Using Regression Techniques: A Comparative Study" 2019 in (ICSSS), IEEE.
4) Sifei Lu, Zengxiang Li, Zheng Qin , Xulei Yang , Rick Siow Mong Goh - "A hybrid regression technique for house prices prediction" 2017,IEEE
5) Vincy Joseph, Anuradha Srinivasaraghavan- "Machine Learning".
6) Trevor Hastie, Robert Tibshirani, Jerome Friedman- "The Elements of Statistical Learning".
7) Tom M Mitchell- "Machine Learning"
8) Saleh Hyatt- "Machine Learning Fundamentals"
9) An Introduction to Statistical Learning: With Applications in R
10) **Github Link-https://github.com/amolghodke25/House-price-Prediction**