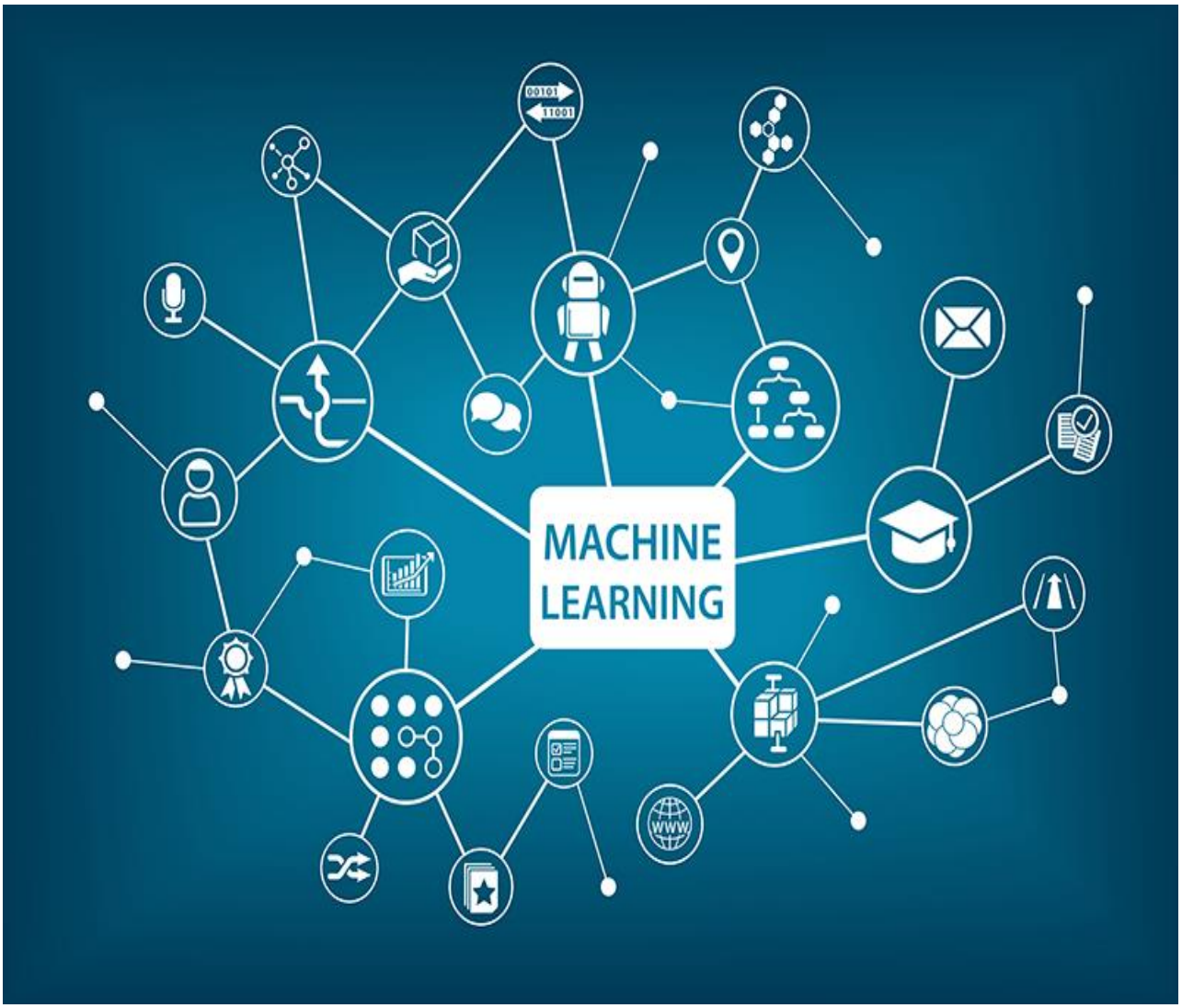
greatlearning
Learning for Life

Capstone Project
Greatlearning
FINAL REPORT



By :
Amit, Arun, Sakthi
ML Capstone Decemper'19 Group 2

Contents

PROBLEM STATEMENT

The dataset "innercity" consists of housing prices in Seattle, Washington, USA with other features. Each column in the given data set is a particular building measurement/information w.r.t target variable price. The main objective here is to analyse the data with visualization and then predict house prices by building a generalized model.

EXPLORING DATA AND FINDINGS

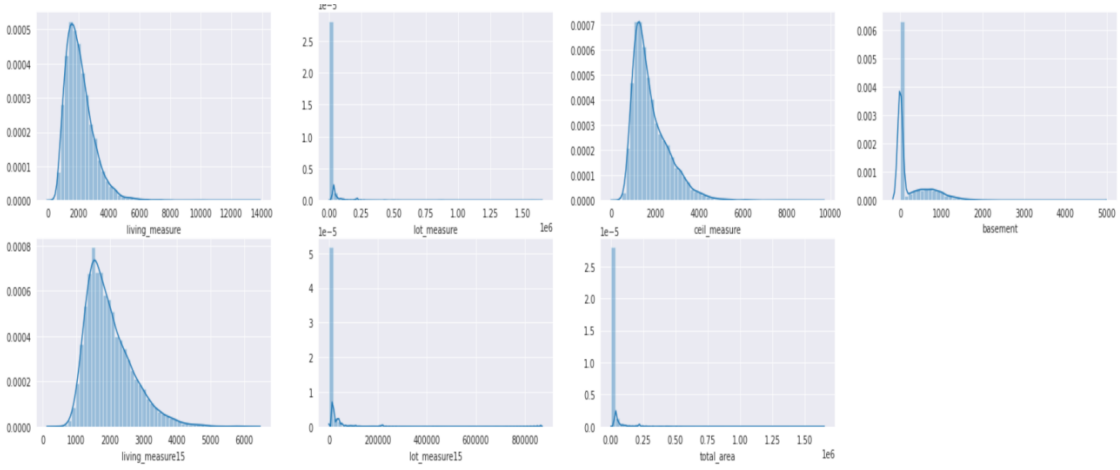
The data set in 'innercity.csv' is a single data source provided to us which consists of multiple variables where each column in the table represents particular building measurement information for that house property with the target variable as price.

1. Dataset info : The data set consists of 21613 records and 23 features. It contains three different data types of data which are int, float and object.

2. Missing Values : By using null function, we found that there's no missing or 'NaN' values present in the data set.

3. Duplicate Records: There are 177 house properties for which duplicate records are present but for such duplicate records values of prices and 'dayhours' attributes were found different while other feature values remain the same. These can be the cases where particular house property being sold more than once with different prices.

4. Distribution of features with their skewness : Features which are of continuous nature like living_measure, lot_measure, basement, total_area have high standard deviation which indicates that these features have a wide range of spread with presence of outliers. Same can be seen in the plots below.



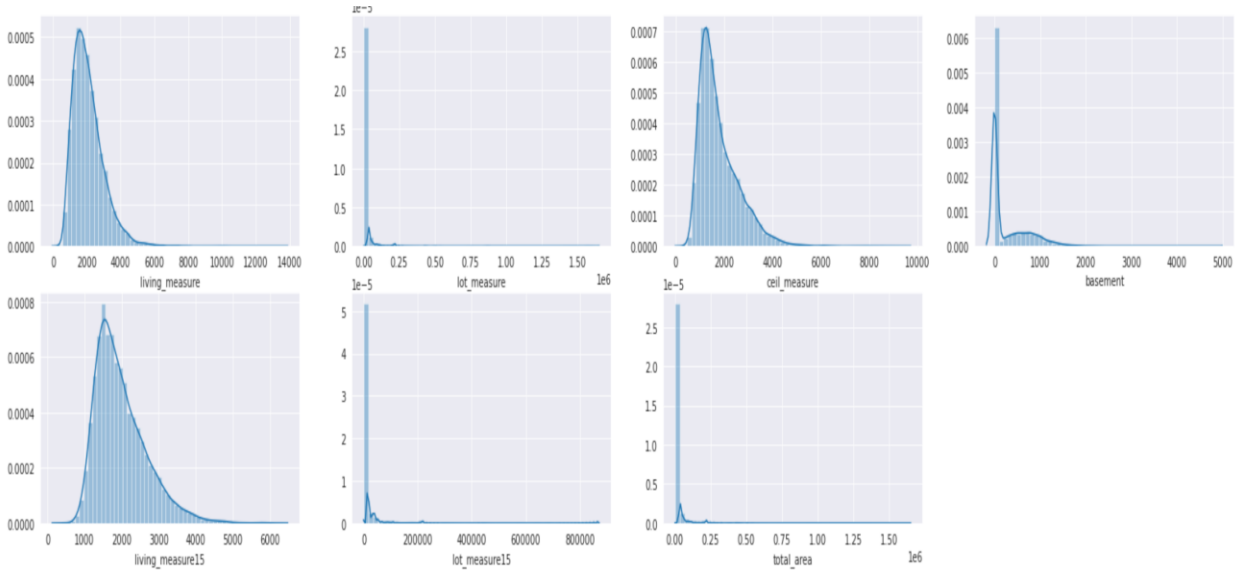
PROCESSING

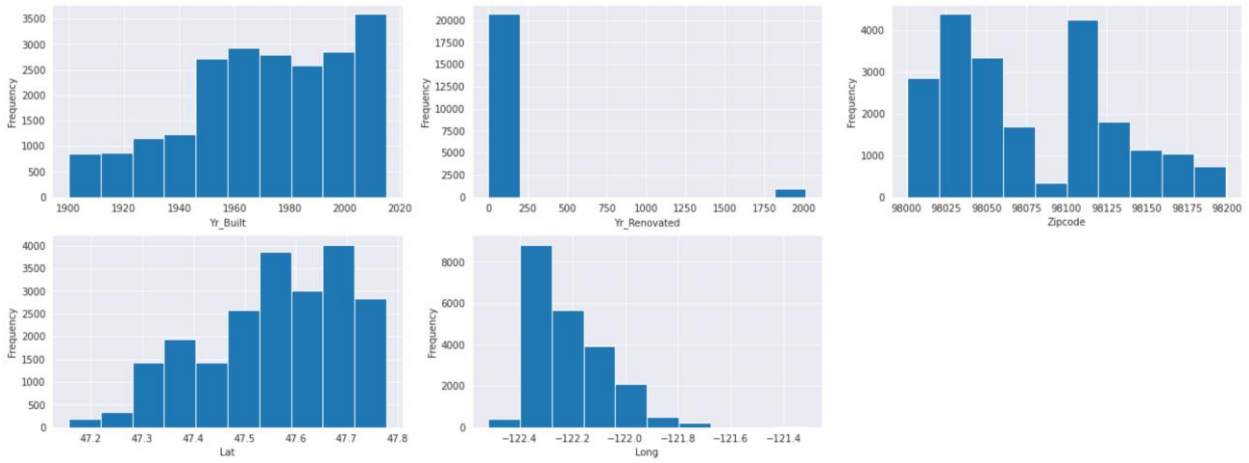
Predicting house prices using independent variables is a regression problem. To get a generalized model we followed below approach:

1. Implemented Basic regression Model once and Evaluated the model performance
2. Applied other nonlinear models to compare model performances to predict prices with using accuracy and mean square error parameters
3. Improved Model performance by applying feature engineering, feature enrichment, cross validation and hyper parameter tuning.
4. Deployed final generalized model using pickle and flask libraries

Salient features of data

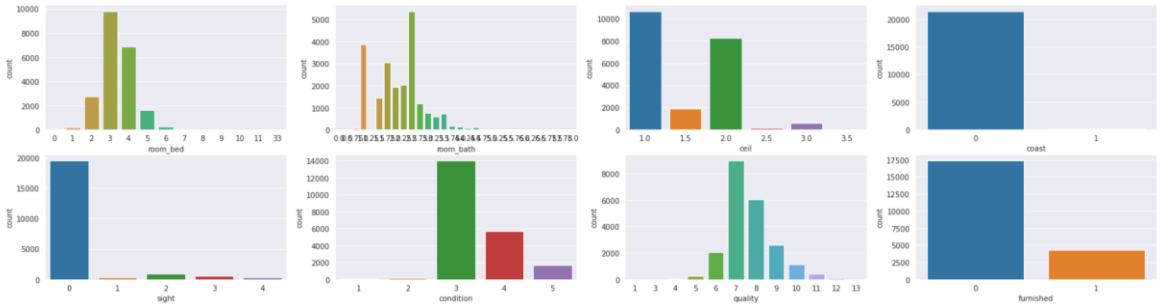
1. Univariate Analysis of Independent Continuous Variables:





- Overall 8 features have unimodal distribution and rest of the features are multimodal distribution (More than two peaks/distinct values).This means that for some features data is not normally distributed and having more than one peak.
- Visualization of distribution also indicates all the features have high frequency on the left tail of distribution (Right Skew) as found earlier using skew function.
- Range of year built of the house is between 1900 - 2015 which indicates that the given data set consists of houses which are more than 100 years old as well recently built houses. Approx. 50% of houses were built before 1975.
- Approx. 95% data points have '0' values in yr_renovated that means those houses were not renovated.

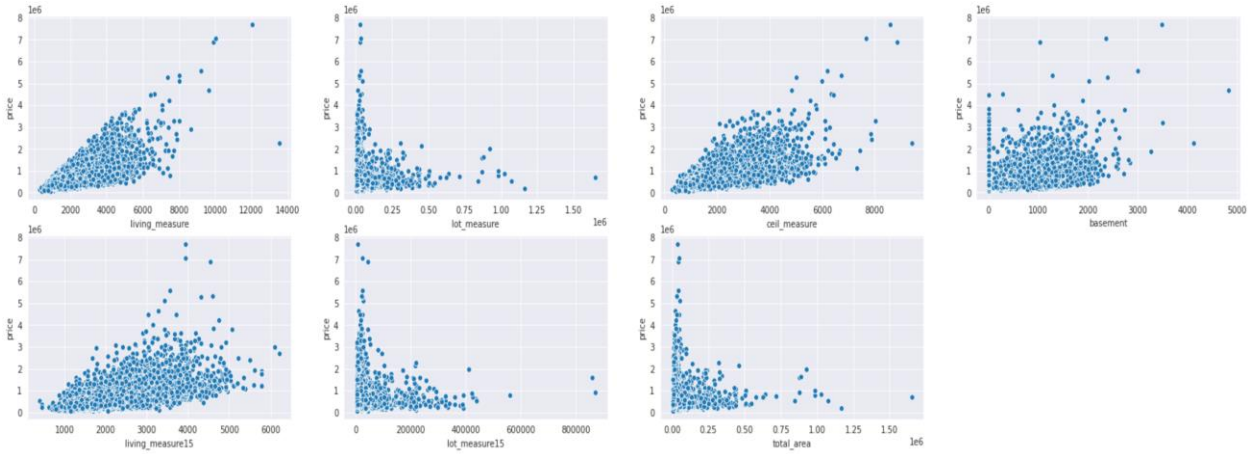
2. Univariate Analysis of Categorical Features:



- Approximately, 43% Houses have 3 bedrooms and 32% houses have 4 bedrooms. There is one house with 33 bedrooms- it might be an inn or resort. We'll keep it as there is only one such data point.
- Approx. 46% houses are 1 story building wherein 34% houses are two story buildings.

- Approx. 99% houses do not have coast/Waterview.
- 90% houses are not being viewed.
- At Least 65% houses are rated as in good condition.
- Only 20% houses are furnished

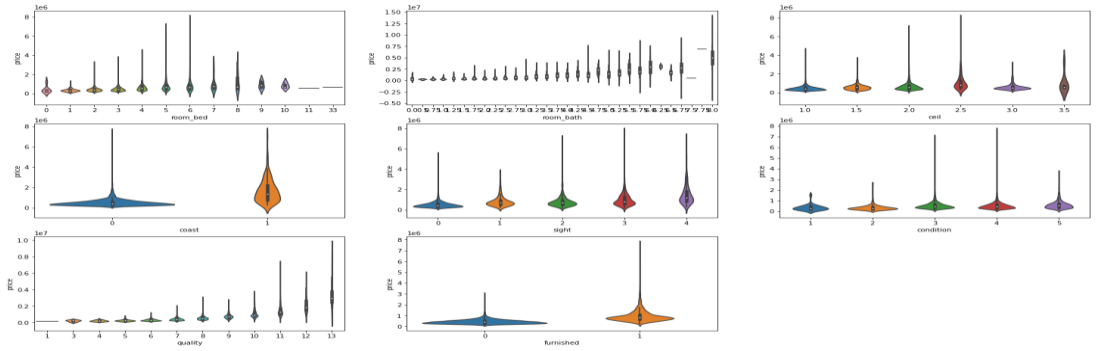
3. Bi-variate Analysis - Target with Continuous Features



- Living measure and ceiling measure have linear correlation with target respectively wherein total area is not linearly correlated with target.

- In bi-variate analysis we observed that features like Ceil measure, basement, total area can be clubbed together for feature enrichment.

4. Bi-Variate Analysis - Target with Categorical Features:



- Cost and furnished both the variables are linearly correlated with target. If the house has a coastal view, prices are high and same for furnished as well.

- House price increases exponentially with quality.
- House prices are very high for 5-7 bedrooms but 40% houses have 3 bedrooms.
- Prices increase for houses having bath rooms till 6, the slightly decreases and again very high for 8.
- House prices are stagnant till condition 2, but it increases at 2-4 linearly and very high for quality.

Initial Data pre-processing

- In the given data set the Primary key attribute 'Cid' is an identifier variable for each record and does not contribute anything to the price of house property. We removed this unwanted variable as data preprocessing.
- Object type attribute dayhours is present in datetime format. We converted this to year to simplify the model building as other columns related to year data are also in year format.
- Features coast , furnished, yr_rennovated, sight have been dropped as 90% feature values are dominated by zero. This means 90%+ houses are not furnished, not coast facing and are not renovated.

Algorithms and Techniques used:

After comparing basic linear regression models and other nonlinear models we found that nonlinear models like Random Forest regressor , KNN regressor and Gradient Boost regressor giving better results. Where Random Forest regressor was overfitted we found Gradient Boost regressor to be the best fit model in our case.

Log transformation: We used log transformation to bring attribute (skewed) distribution close to gaussian which helped us to remove skewness.

IQR based Outliers treatment: We tried IQR technique to treat outliers present in our data set.

Cross validation : We've tried with RandomSearch CV to validate our model on different training and test data sets.

Hyper parameter tuning: For performance improvement we applied hyper parameter tuning to get best parameters for our best fit model.

STEPS LEADING TO THE 'FINAL SOLUTION'

In our model implementation we considered model accuracy and Mean Absolute Percentage Error on training and testing data set as evaluation parameters

1. Basic model implementation without outlier treatment: We built our first basic model using Ordinary least squares method without any preprocessing on data. We got accuracy of 70.2% in this step.

Using Ordinary Least Squares :

Before outlier treatment:

OLS Regression Results

Dep. Variable:	y_train	**R-squared:**	0.702
Model:	OLS	**Adj. R-squared:**	0.702
Method:	Least Squares	**F-statistic:**	1780.
Date:	Sun, 28 Jun 2020	**Prob (F-statistic):**	0.00
Time:	08:25:49	**Log-Likelihood:**	-2.0618e+05
No. Observations:	15129	**AIC:**	4.124e+05
Df Residuals:	15108	**BIC:**	4.126e+05
Df Model:	20		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	5.408e+05	1632.632	331.214	0.000	5.38e+05	5.44e+05
x_train[0]	-2081.5653	1654.328	-1.258	0.208	-5324.249	1161.118
x_train[1]	-3.027e+04	2060.775	-14.687	0.000	-3.43e+04	-2.62e+04
x_train[2]	3.129e+04	2990.001	10.464	0.000	2.54e+04	3.71e+04
x_train[3]	7.895e+04	1896.675	41.625	0.000	7.52e+04	8.27e+04
x_train[4]	1517.8508	1153.272	1.316	0.188	-742.702	3778.404
x_train[5]	4158.6805	2313.962	1.797	0.072	-376.965	8694.326
x_train[6]	5.294e+04	1765.497	29.986	0.000	4.95e+04	5.64e+04
x_train[7]	4.137e+04	1964.424	21.060	0.000	3.75e+04	4.52e+04
x_train[8]	1.822e+04	1817.477	10.026	0.000	1.47e+04	2.18e+04
x_train[9]	9.939e+04	3565.750	27.875	0.000	9.24e+04	1.06e+05
x_train[10]	7.346e+04	2038.296	36.038	0.000	6.95e+04	7.75e+04
x_train[11]	2.64e+04	1859.792	14.193	0.000	2.28e+04	3e+04
x_train[12]	-7.506e+04	2561.421	-29.302	0.000	-8.01e+04	-7e+04
x_train[13]	9598.4679	1717.357	5.589	0.000	6232.240	1.3e+04
x_train[14]	-3.328e+04	2107.741	-15.791	0.000	-3.74e+04	-2.92e+04
x_train[15]	8.332e+04	1773.960	46.970	0.000	7.98e+04	8.68e+04
x_train[16]	-3.228e+04	2221.413	-14.533	0.000	-3.66e+04	-2.79e+04
x_train[17]	1.617e+04	2799.811	5.776	0.000	1.07e+04	2.17e+04
x_train[18]	-1.122e+04	2270.199	-4.942	0.000	-1.57e+04	-6769.688
x_train[19]	1.821e+04	2748.175	6.627	0.000	1.28e+04	2.36e+04
x_train[20]	3255.2024	1146.007	2.840	0.005	1008.889	5501.516
x_train[21]	1.414e+04	1636.136	8.641	0.000	1.09e+04	1.73e+04

Omnibus:	12495.137	**Durbin-Watson:**	2.002
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1100264.208
Skew:	3.437	**Prob(JB):**	0.00
Kurtosis:	44.209	**Cond. No.**	1.03e+16

2. Implemented Linear regression : We tried implementing a Linear regression model and found out the sum out the sum of squared errors were high on both training and test data set.

3. Regularization of linear models : After implementing Regularization models using ridge and lasso methods, these methods made our model under fitted hence we are not going to regularize our model.

4. Implementation of Nonlinear models : To compare results of linear model with nonlinear models we implemented Decision Tree Regressor, Kneighbors Regressor, Ada Boost Regressor, Random Forest Regressor .

	Model	Training_Accuracy	Test_Accuracy	Training_MAPE	Test_MAPE
0	Decision Tree	0.90	0.81	10.82	14.23
1	KNN	0.85	0.81	13.06	14.68
2	Ada Boost	0.76	0.75	20.01	20.10
3	Random Forest	0.98	0.88	4.35	11.30
4	Gradient Boost	0.91	0.89	10.13	11.28

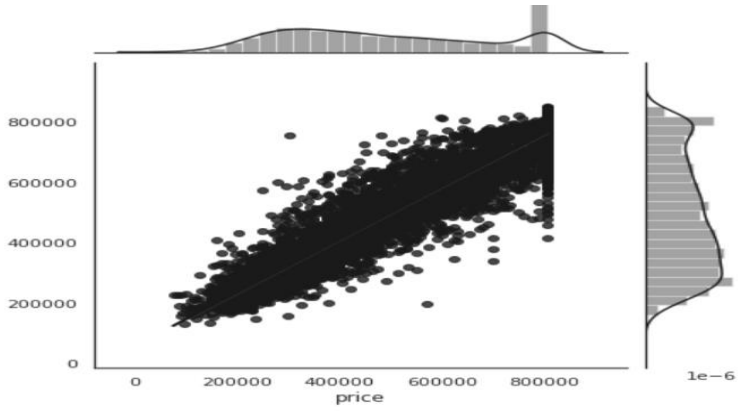
5. Evaluation of Nonlinear models :

In random forest feature selection, the trees are more independent (Non Linear) of each other compared to other models, which resulted in better predictive performance due to better variance.

In the above table, we can see that nonlinear models random forest and gradient boost regression have least sum of squared errors as compared to other models.

6. Best Fit Model selection:

By comparing all linear and nonlinear models we selected Gradient boost regressor as the best fit model as other nonlinear models were found to be over fitted. Gradient boost regressor is giving accuracy of 91 % on training data and 89 % on testing data.



7. Feature engineering / Performance improvement :

a. Label Encoding : we applied label encoding on Features 'room_bath', 'ceil', 'basement', 'yr_built', 'zipcode', 'lat', 'long' ,'yr_sold' to use these variables in our model building with encoded data rather than raw values.

b. Feature Selection: we found five features (living_measure, quality, lat, long, yr_built) are covering 95% of variance. We built out selected best fit line model keeping only these 5 variables

c. Hyper Parameter Tuning : We implemented hyper parameter tuning for our best fit model gradient boost regressor to get best parameters . With this we got a 1% increase in accuracy on the test data set.

FINAL MODEL EVALUATION :

After doing all the feature engineering and data processing we found out Gradient boost regressor with parameters (min_samples_leaf= 5, max_features='auto', max_depth=6, learning_rate =0.1) giving best results on training and test data sets.

Model	**Training Accuracy**	**Testing Accuracy**	**Training MAPE**	**Testing MAPE**
Gradient Boost With feature selection	0.89	0.87	11.20	12.27
Gradient Boost With feature selection After tuning	0.91	0.88	10.27	11.88

In the above summary, we can notice the scores of our best fit model i.e. gradient boost model.

In the Basic model, we used 17 features after outlier treatment and built a gradient boost regressor model which gives 90 % of accuracy in training data and 89% of accuracy in test data.

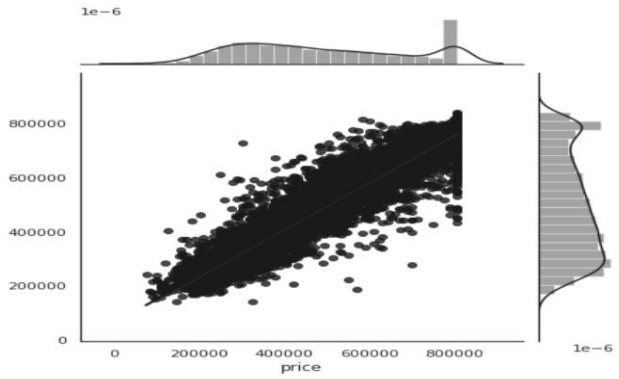
In Feature Selection technique, we kept only 5 features (Most Important : Living Measure, quality, lat, long, yr_built) and built gradient boost regressor model which gave 91 % of accuracy in training data and 88% of accuracy in test data.

Hyper parameter tuning on Best fit model : To get best parameters for gradient boost model we implemented hyper parameter tuning keeping selected important features only and able to improve training accuracy from 89% to 91% and testing accuracy 87% to 88%.

Evaluation Of Final Model : To validate our best fit model we implemented random search cross validation.
Comparison with Earlier EDA: If we go back and check the correlation of the independent variables with price we found that important features used in our final model had high correlation values as shown below : Correlation between price and

Living Measure	Quality	lat	long	Yr_built
0.7	0.67	0.31	0.022	0.054

Visualization of prediction using best fit final model:
We can see from below visualization that our model is fit to the data with minimum outliers



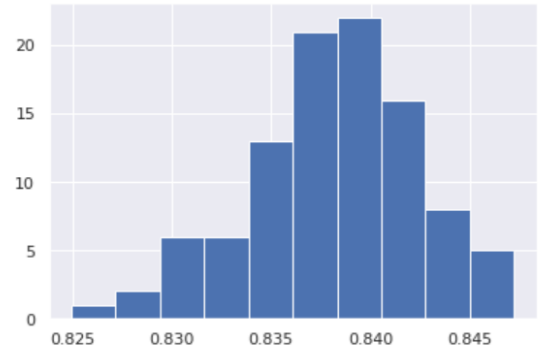
Accuracy Improvement from the benchmark model

We have accuracy gain from 70.2% to 88% from our 1st iteration of basic regression model to our final model of using Gradient boost regression with best parameters.

After Feature Selection technique with keeping 5 Most Important features, gradient boost regressor model gave 91% of accuracy in training data and 88% of accuracy in test data.

IMPLICATIONS FROM FINAL MODEL:

To get the implication of our model we used bootstrap technique to get an idea of accuracy with confidence interval.



95.0 confidence interval 82.9% and 84.5%

If we put our best fit Gradient Boost Regressor model into a production, then we can expect a maximum of **85% accuracy at 95% of confidence interval.**

Here we considered only 100 no. of bootstrap iterations and 5% of records in a dataset, in order to reduce the computation time. As long as we increase the no. of iterations and consider all the records, we may get a better accuracy than the above bootstrap model. The same can be expected in production as well.

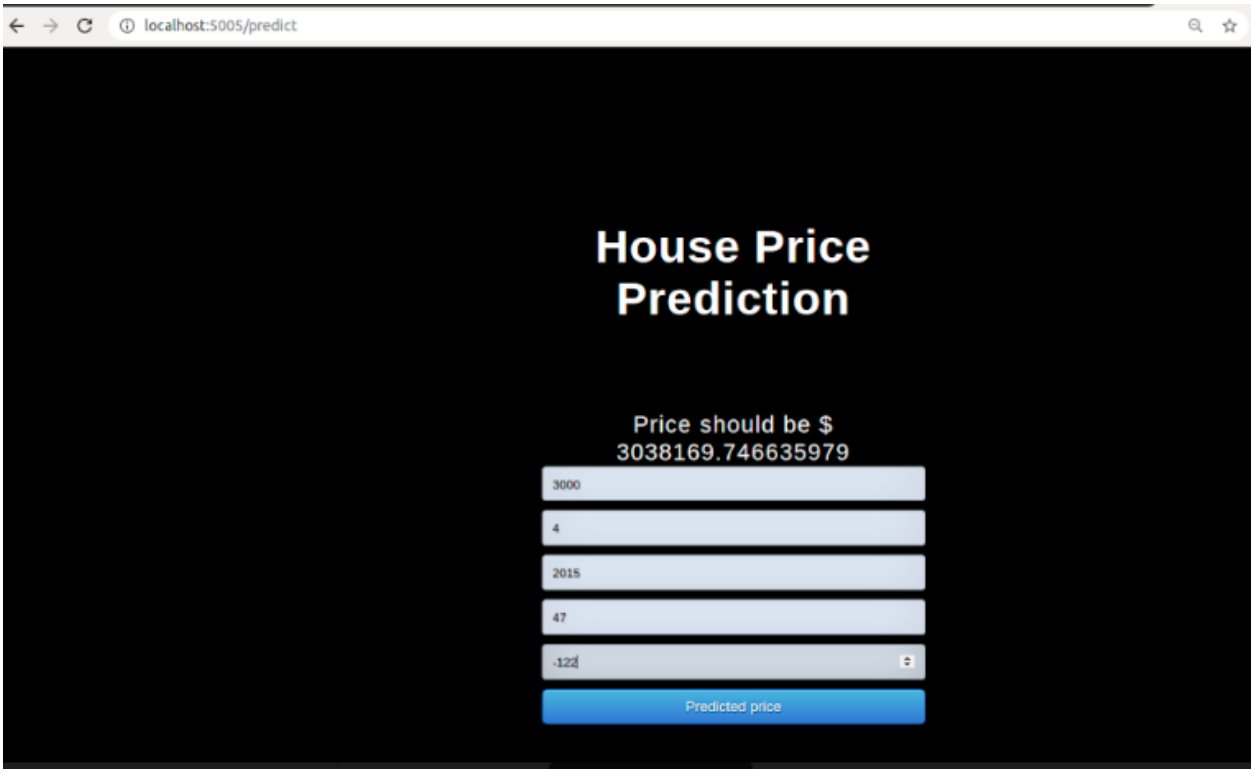
Limitation and Assumptions

- In our model we considered only 5 important features , if we get samples from production data to predict prices where features which we didn't use in our final model building are more dependent on price, our model will fall short and will not give correct predictions.

- As given data is only for one particular location i.e. Seattle , if we need to predict prices from different locations we will again need to build the model .

- Since the given data set had so many outliers , if we get a test data point which is an actual outlier ,our model won't be able to predict its price accurately. Example: If we get a test data point with a coast facing house furnished , our model won't be able to predict its price with good accuracy.

Model Deployment Result:



PROJECT LEARNINGS

- Inferences are important while doing EDA of any dataset to get the actual meaning of the findings and relate these findings with domain or business problems.
- Detecting and treating outliers in the data set is important and knowledge of using various techniques to do this is also important.
- Model building should be done in a step by step process, in each step data/feature operation ,comparison of result and inference of results are important to reach to the final model.
- Deployment the final model using pickle and flask library to check end to end flow for the test value prediction.
- EDA and Feature engineering are very important steps and next time we would like to give more time in these steps.
- We would try more techniques and methods available for data processing or feature engineering in our model building.