Milestone Report

Capstone Project 1

Table of Contents

Problem Statement

- Buying a house is an important juncture in life for the majority of the individuals, both emotionally and financially, most probably a one time investment.
- Hence, this decision goes through a lot of consideration with endless comparison of features a house has to offer vs the Selling price of the house.
- Often few of the critical factors are overlooked and one ends up paying a premium sum for an unworthy house, effectively making the whole exercise a painful experience.
- Through this project, We would try to address this pain point of the house buyers i.e feature wise comparison of house vs. the price offered for the house.
- We are aiming to build a model, which would compare all the major/ minor features of a house (e.g Total plot area, Garage availability etc.) and based on the analysis try predicting a reasonable price for the house i.e. Selling price of the house.
- In short, the end product of this Project would be a reasonable selling price of a house.

Dataset Description

- To train and test this model, we are using the **Ames Housing dataset**, a dataset of 79 explanatory variables which describes in detail about the various aspects of the residential homes in Ames, Iowa.
- The dataset is sourced from kaggle, made available as part of on-going Kaggle competition

Data Cleaning and Preparation

The below steps were taken to clean up the data.

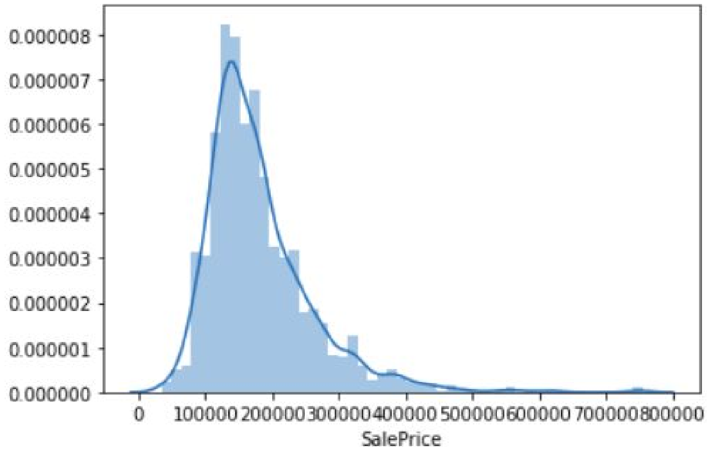
- First of all, we addressed the columns with missing values. All the columns were identified which consisted of even a single NULL/ NaN value.
- If the number of missing values exceed 50% of the total entries for a particular column, then we drop the column altogether.
- For columns containing categorical values, We replace the missing values with the mode value of the column.
- For columns with numerical variables, replace the missing values with the median value of the column.
- We also plotted a box plot for all the numerical variables, and in turn observed how the data is distributed across the range values.

- From the box plot above, We also got an understanding of the outliers present for each numerical column, which would be addressed during the EDA part.

Initial Findings from the Exploratory Data Analysis

Univariate Analysis

We analysed(distplot) the the target variable i.e. SalePrice under this.

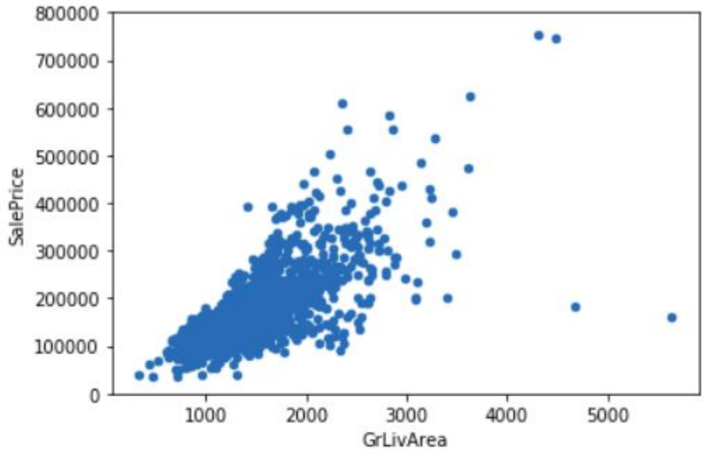


Observations

1. we observe that, the mean price of all the houses turns out to be a tad under $181k.
2. There is diversity in the house prices indicated by the standard deviation of around $79k.
3. The maximum price is listed at $755k, indicating a long right tail too.
4. The kurtosis value of 6.5 states that the dataset is not normally distributed and shows peakedness.

Bivariate Analysis

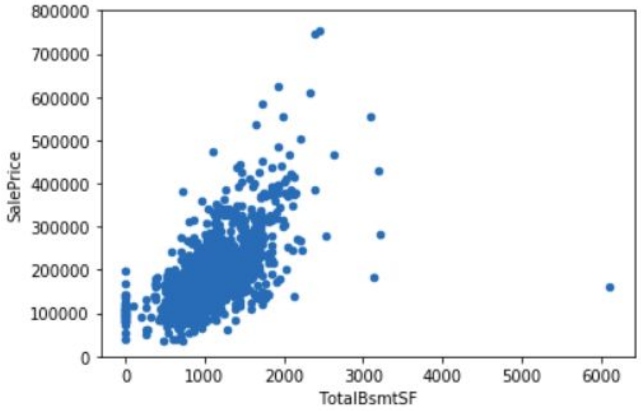
Case 1
Analysing "GrLivArea" vs. "SalePrice"



Observations

1. Most of the houses are in the range of 500 sq. feet to 2500 sq. feet.
2. "GrLivArea" shows a linear relationship i.e. an increase in the living area is marked by a proportional increase in the selling price. However, there are cases wherein this linearity is broken and the prices observed sudden jump/ dip.

Case 2
Analysing "TotalBsmtSF" vs. "SalePrice"



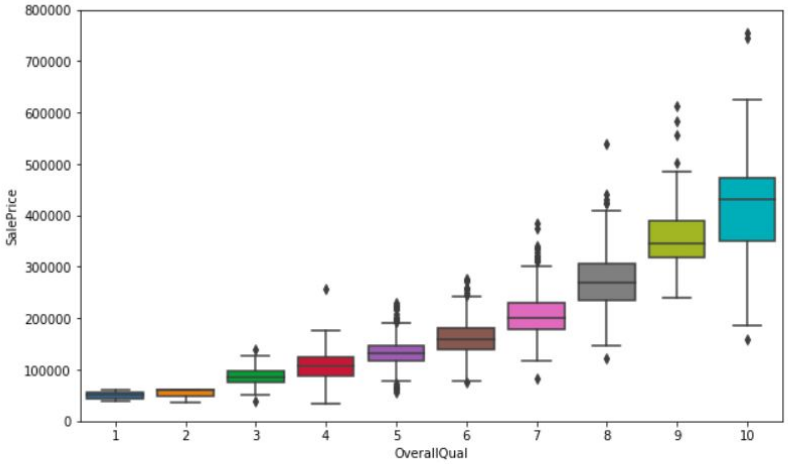
Observations

1. For most parts of the distribution, both the variables seem to be linearly related to each other with the exception of some outliers.

2. However, there are also a substantial number of cases, wherein the selling price doesn't seem to be affected by the basement area i.e entries found for houses with basement areas as zero, suggesting either the data unavailability or the basement absence in the house.

Case 3

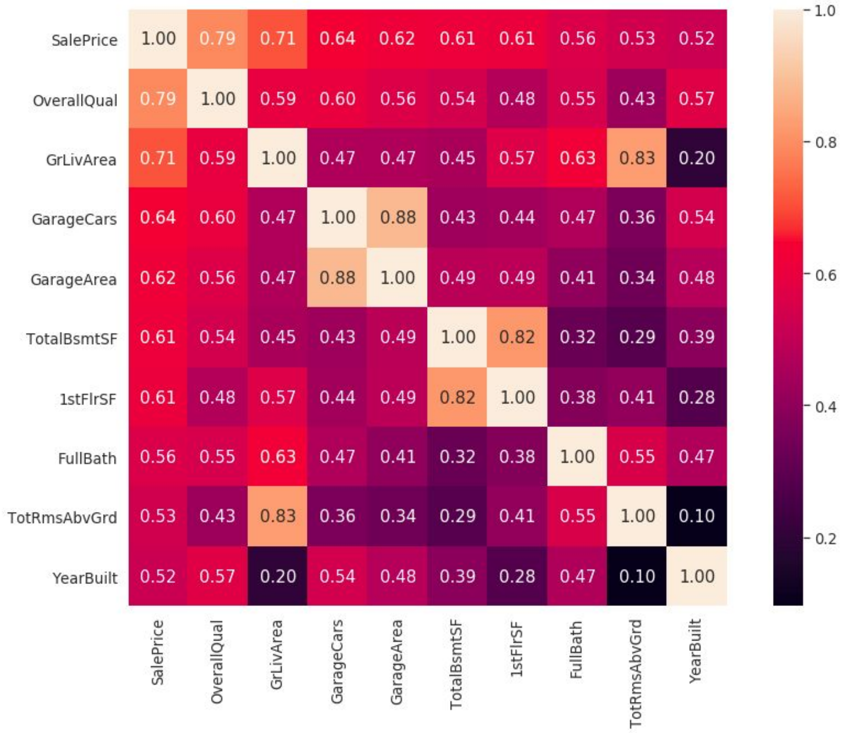
Analysing the "OverallQual" vs. "SalePrice"



Observations

1. As expected, The relationship between the Quality and the selling price is pretty straight forward, with both the variables being linearly related.
2. The Sale price shows an increasing trend for houses that are marked with better quality.

Correlation Matrix



	SalePrice	OverallQual	GrLivArea	GarageCars	GarageArea	TotalBsmtSF	1stFlrSF	FullBath	TotRmsAbvGrd	YearBuilt
SalePrice	1.00	0.79	0.71	0.64	0.62	0.61	0.61	0.56	0.53	0.52
OverallQual	0.79	1.00	0.59	0.60	0.56	0.54	0.48	0.55	0.43	0.57
GrLivArea	0.71	0.59	1.00	0.47	0.47	0.45	0.57	0.63	0.83	0.20
GarageCars	0.64	0.60	0.47	1.00	0.88	0.43	0.44	0.47	0.36	0.54
GarageArea	0.62	0.56	0.47	0.88	1.00	0.49	0.49	0.41	0.34	0.48
TotalBsmtSF	0.61	0.54	0.45	0.43	0.49	1.00	0.82	0.32	0.29	0.39
1stFlrSF	0.61	0.48	0.57	0.44	0.49	0.82	1.00	0.38	0.41	0.28
FullBath	0.56	0.55	0.63	0.47	0.41	0.32	0.38	1.00	0.55	0.47
TotRmsAbvGrd	0.53	0.43	0.83	0.36	0.34	0.29	0.41	0.55	1.00	0.10
YearBuilt	0.52	0.57	0.20	0.54	0.48	0.39	0.28	0.47	0.10	1.00

Observations

1. As we predicted and analysed before, the columns "OverallQual" (0.79), "GrLivArea"(0.71) and "TotalBsmtSF"(0.61) are amongst the variables showing strong correlation with the "SalePrice".
2. The columns "GarageCars"(0.64) and "GarageArea"(0.62) have a high correlation coefficient, but since the number of cars a Garage can accommodate, is a function of area of the Garage. Hence, We won't be distinguishing them as separate variables and would only be considering the "GarageCars" (high correlation coefficient amongst the two), in our further analysis.
3. Similarly, "TotalBsmtSF"(0.61) and "1stFlrSF"(0.61) show same degree of correlation. It also seem that the "1stFlrSF" i.e. area of first floor is a function of "TotalBsmt" i.e. basement area. Infact, they would be almost equal in size. Hence, we would only consider the "TotalBsmtSF" for further EDA, out of the two.
4. "TotRmsAbvGrd"(0.53) and "GrdLivArea"(0.71) also effectively mean the same thing i.e "GrLivArea" (living area in sq. ft.) is a function of / dependent on "TotRmsAbcGrd" (total rooms). Hence, out of these two, we would consider "GrLivArea" for further EDA.
5. “YearBuilt" column also shows a significant correlation with the "SalePrice", which would be subject for further analysis.

Hypothesis Testing

Case 1

Null Hypothesis

There is no change in the average house prices built before and after 1950..

Alternate Hypothesis

There is change in the average house prices for houses built before and after 1950..

Observations

We could observe from the t-tests, the p value was not significant and hence we rejected the Null hypothesis in our case.

This means, there were significant changes in the average house prices in the houses built before and after the 1950's.

Case 2

Null Hypothesis

There is no change in the average house prices with different numbers of bathrooms

Alternate Hypothesis

The average house price differs as the numbers of bathrooms change.

Observations

We compared the house prices for houses with different numbers of bathrooms (2 and 3 in this case).

From the t-tests, the p-value observed is pretty insignificant, which leads us to believe that the house prices differ for houses with different numbers of bathrooms.

In-depth Analysis using Machine Learning

Pre-requisites:-
We have already performed the EDA and hypothesis testing in the previous steps.
Now, We would try implementing machine learning models on the cleaned datasets.

Step 1:
The EDA was performed on the test dataset too, and the test data was cleaned up.
Post that, the test data was concatenated with the training dataset.

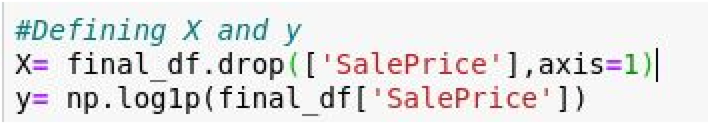
Step 2:
The given dataset contains both, categorical and numerical features.
The ML models won't accept the categorical features, hence we have to convert the categorical features into the numerical features.

Hence, We extracted all the categorical features present in the dataset and applied the **one-hot encoding** method to convert the categorical features into numerical ones.

Step 3:
During the above steps, we have had columns duplicated in the datasets.
Here, we remove the duplicated columns.

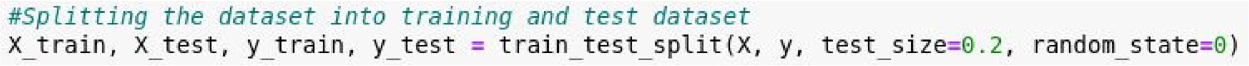
Step 4:
Now, we are ready to define the "X" and "y" variable i.e. the input variables/ features and the target variable respectively.

For the target variable, since the target variable is the price and has a higher variance, we take the log of the target variable i.e. log value of SalePrice value.



```
#Defining X and y
X= final_df.drop(['SalePrice'],axis=1)
y= np.log1p(final_df['SalePrice'])
```

Step 5:
We split the given dataset into training and testing data using the scikit-learn's train_test_split function. The test dataset is 20% of the total available data.

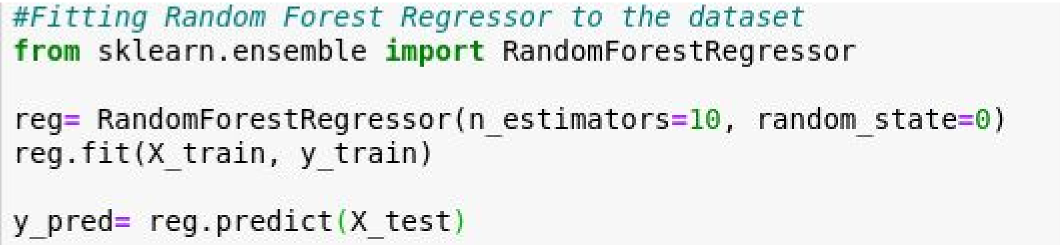


```
#Splitting the dataset into training and test dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

Step 5:
Model Selection: The dataset with us is non-linear in nature as we had many categorical variables that were converted into numerical variables by one-hot encoding method.

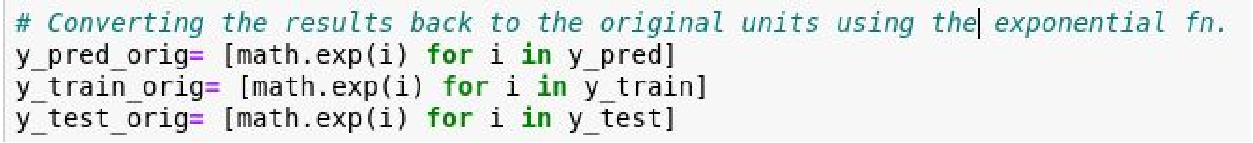
Case 1:
Due to this limitation the simple linear regression model won't be efficient and produce good results here.
We also tried implementing the same but met with astronomically high error rates, making linear regression an unviable proposition.

Case 2:
Further keeping in mind the complex nature of the dataset, we decided to proceed with the "**Random forest regression**" method.
This method would take care of the non-linearity in the data and treat the variables accordingly.



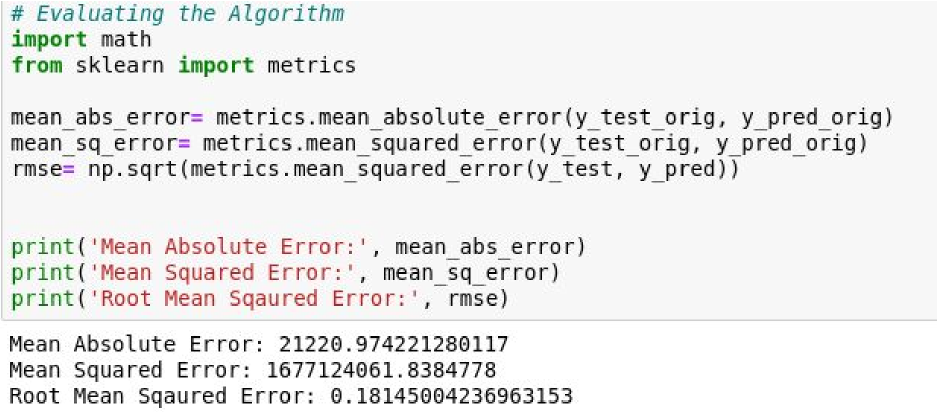
```
#Fitting Random Forest Regressor to the dataset
from sklearn.ensemble import RandomForestRegressor

reg= RandomForestRegressor(n_estimators=10, random_state=0)
reg.fit(X_train, y_train)

y_pred= reg.predict(X_test)
```

Step 6:
Converting the results back to the original units (using exp function)



```
# Converting the results back to the original units using the exponential fn.
y_pred_orig= [math.exp(i) for i in y_pred]
y_train_orig= [math.exp(i) for i in y_train]
y_test_orig= [math.exp(i) for i in y_test]
```

Step 7:
Calculating the error



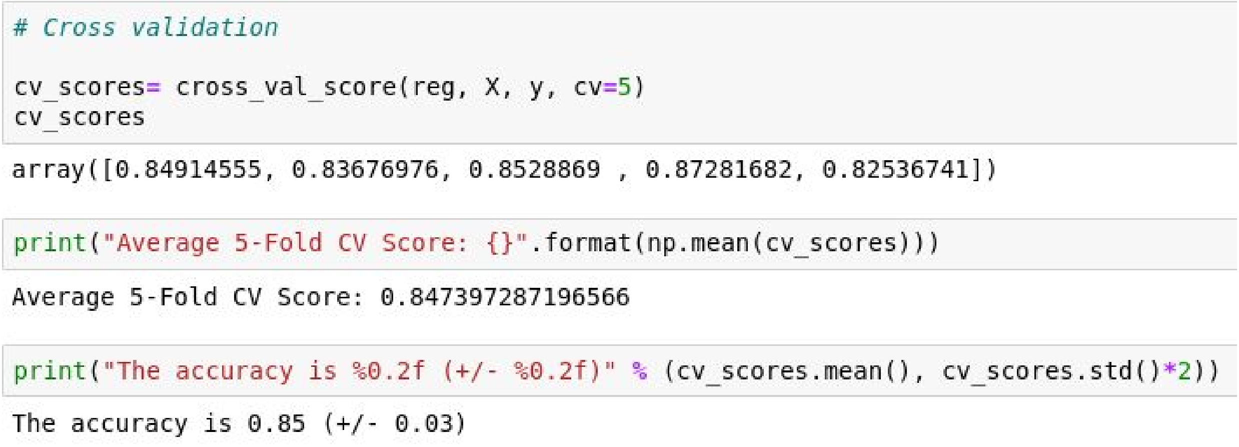
```
# Evaluating the Algorithm
import math
from sklearn import metrics

mean_abs_error= metrics.mean_absolute_error(y_test_orig, y_pred_orig)
mean_sq_error= metrics.mean_squared_error(y_test_orig, y_pred_orig)
rmse= np.sqrt(metrics.mean_squared_error(y_test, y_pred))


print('Mean Absolute Error:', mean_abs_error)
print('Mean Squared Error:', mean_sq_error)
print('Root Mean Sqaured Error:', rmse)
```

```
Mean Absolute Error: 21220.974221280117
Mean Squared Error: 1677124061.8384778
Root Mean Sqaured Error: 0.18145004236963153
```

The mean absolute error is 21220, 8.61 % of the mean house prices i.e. 182899.
With 10 trees, the RMSE is 0.181

We have played around with increasing the estimator nodes, the RMSE value keeps hovering at this range.

Step 8:
Applying cross validation and checking the mean score.



```
# Cross validation

cv_scores= cross_val_score(reg, X, y, cv=5)
cv_scores
```

```
array([0.84914555, 0.83676976, 0.8528869 , 0.87281682, 0.82536741])
```

```
print("Average 5-Fold CV Score: {}".format(np.mean(cv_scores)))
```

```
Average 5-Fold CV Score: 0.847397287196566
```

```
print("The accuracy is %0.2f (+/- %0.2f)" % (cv_scores.mean(), cv_scores.std()*2))
```

```
The accuracy is 0.85 (+/- 0.03)
```

As seen above, We have around 85% accuracy with this model.

Hence, we would be accepting this model.