House Price Prediction

1. Pre-Processing

1. I have combined train.csv and test.csv. The SalesPrice column, which is the set of labels or true prices is dropped from the combined table. This was done so that the values of test set could also undergo pre-processing so that no problems are encountered when finally predicting the price. After all the values were processed, the complete table was again split into train and test sets.

2.The columns having NaN values were found out and filled with the most frequent value. In case of columns having NaNs as most frequent value, missing values were imputed by 'None'.

3. In columns like LotFrontage the missing values were imputed by taking square root of corresponding LotArea values, and this step was verified by finding the correlation with SalesPrice which increased to 0.62.

2.Feature Engineering

1. The most correlated columns with SalesPrice were used to generate polynomial features. 2^{nd} and 3^{rd} degree polynomials that is columns with values raised to the power of 2 and 3 were created for topmost highly correlated columns. Also polynomials of the form x1*x2+x1*x3+x1*x4+x2*x3+x2*x4+x3*x4 were created. Usually it was through hit and trial, that I found out which columns among the highly correlated columns were to be used in generating polynomial features such that my model obtains a better score on Kaggle.

2. Encoding of categorical variables: Some columns had non-numeric values and hence each of the levels in these columns were encoded, such that all the values could be mapped into numeric numbers or levels. This command was executed through *pd.get_dummies()*

3.Normalization

1.I tried using standard normalization, but the model performed better when log(1+x) normalization was used.

2. Then after all these steps, the processed data was again split into train and test set.

4. Performing Regression

1. I tried 2 regression models, the first one being linear regression and the second one being linear regression with regularization which is called Ridge regression in sklearn modules.

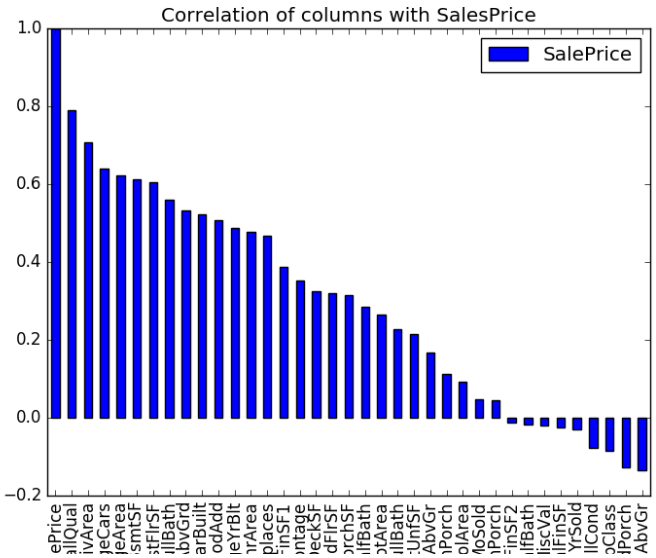
2. Under ridgre regression , the correct value of alpha, that is the tuning parameter was required. Somehow by plotting the cross-validation scores and the root mean square errors as a function of different alphas, the value of alpha under which the model performed the best was 5.

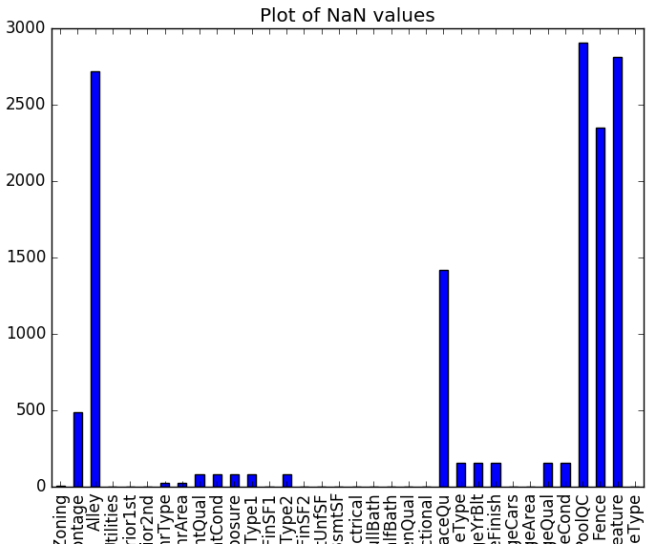
3. Increasing the number of iterations for predicting the model, by setting max_iter also improved the Kaggle score.

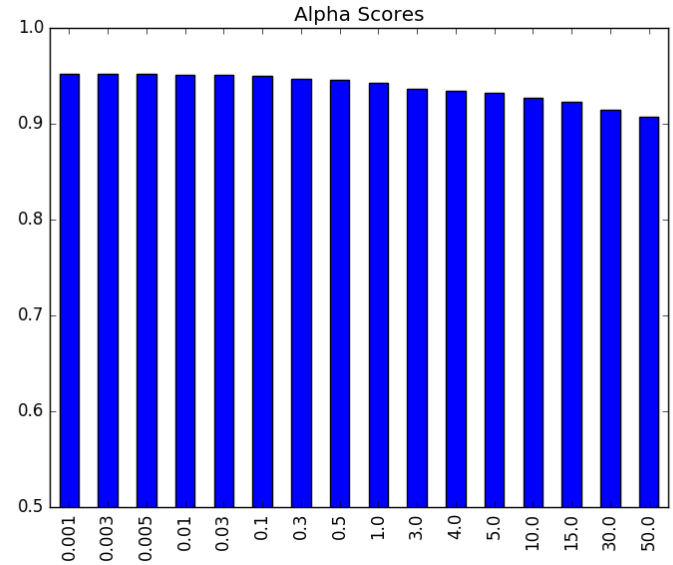
Things I noticed:
1.Instead of dropping the columns with large number of NaNs, such as Alley, PoolQC etc, the score on Kaggle was better if I just imputed None in all those NaN values and kept them in my model.

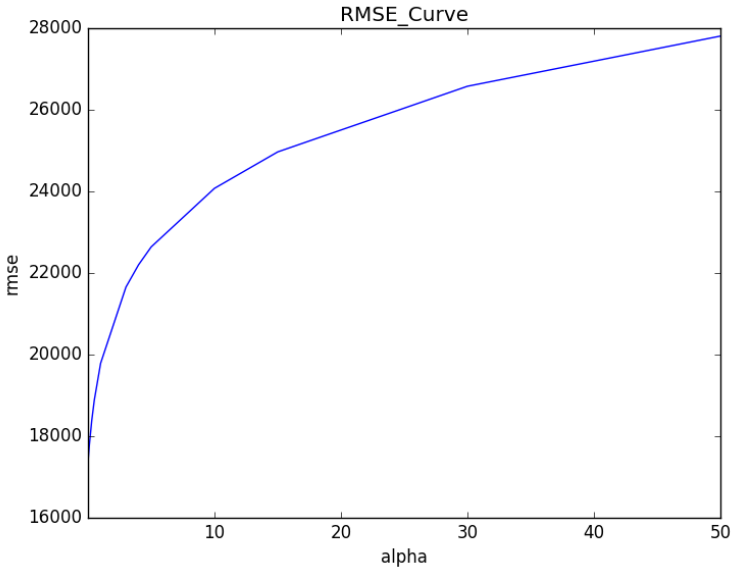
2. Having a higher R-squared score doesnt mean that the model will perform better, since at alpha = 5, the R-squared score was lower and still the model performed better.

Plots









Thank you
Regards
Siddhant Tandon
Matricola:1771650