Tarik Koric netid: koric1
Pranav Velamakanni netid: pranavv2

Introduction and Explanation of dataset

The goal of the project is to build models that can predict the price of a home using variables defined in the dataset that best describe the property.

The Ames housing dataset used in his project is a collection of sales data of residential property in Ames, Iowa between 2006 and 2010. This dataset has been compiled by Dean De Cock. The dataset contains 2930 rows and 80 columns with 23 nominal, 23 ordinal, 14 discrete and 20 continuous variables.

Pre-processing procedure

The dataset is split into 2 sets, train and test which are saved as CSV files. The data split was performed using R and the rest of the pre-processing steps were performed using Python.This data is read into 2 Pandas data frames, one representing the train dataset and the other representing the test dataset. The PID (Parcel identification number) column is saved to a NumPy array for later use.

Several columns ('PID','Street', 'Utilities', 'Condition_2', 'Roof_Matl', 'Heating', 'Pool_QC', 'Misc_Feature', 'Low_Qual_Fin_SF', 'Pool_Area', 'Longitude','Latitude') are dropped from the training and test datasets since they may not help in predicting the price of the house. Columns like Heating and Pool_area do provide information about the features of the house, the dataset contains several other fundamental variables that can prove to be better indicators for predicting the price of the house.

Winsorization is a process which involves transforming values to limit or reduce the effect of extreme values in the dataset. This step is performed on several columns ("Lot_Frontage", "Lot_Area", "Mas_Vnr_Area", "BsmtFin_SF_2", "Bsmt_Unf_SF", "Total_Bsmt_SF", "Second_Flr_SF", 'First_Flr_SF', "Gr_Liv_Area", "Garage_Area", "Wood_Deck_SF", "Open_Porch_SF", "Enclosed_Porch", "Three_season_porch", "Screen_Porch", "Misc_Val") that have outliers or extreme values that can skew the results and lower the accuracy of the model. The values were clipped between 0 and the 95th percentile value for the respective column.

NaN values in the dataset need to be removed or replaced to train the model. To ensure consistency across the dataset and to avoid removing data points, NaN values for the Garage_Yr_Blt column have been replaced with values from the Year_Build column which would most be the same in most cases. Log of the Sale_Price column is taken and saved as y_train before removing it from the x_train data frame.

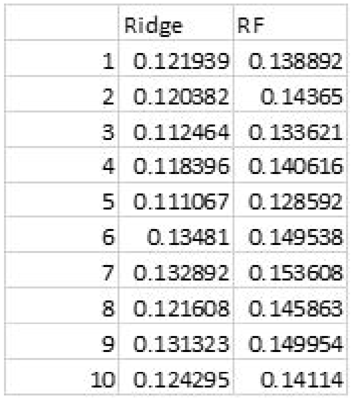
To convert categorical values to numeric values, one hot encoding was used on columns ('MS_SubClass', 'MS_Zoning', 'Alley', 'Lot_Shape', 'Land_Contour', 'Lot_Config', 'Land_Slope', 'Neighborhood', 'Condition_1', 'Bldg_Type', 'House_Style', 'Overall_Qual', 'Overall_Cond', 'Roof_Style', 'Exterior_1st', 'Exterior_2nd', 'Mas_Vnr_Type', 'Exter_Qual', 'Exter_Cond', 'Foundation', 'Bsmt_Qual', 'Bsmt_Cond', 'Bsmt_Exposure', 'BsmtFin_Type_1', 'BsmtFin_Type_2', 'Heating_QC', 'Central_Air', 'Electrical', 'Kitchen_Qual', 'Functional', 'Fireplace_Qu', 'Garage_Type', 'Garage_Finish', 'Garage_Qual', 'Garage_Cond', 'Paved_Drive', 'Fence', 'Sale_Type', 'Sale_Condition') containing categorical values. One hot encoding converts each unique value in a column to a new column and fills it with a 1 if that value exists for that row and 0 if it does not. This conversion helps train the model effectively.

This transformed data is used to train a Ridge regression model and a Random Forest regression model.

Models and Results

Training the data using ridge regression reduces the model complexity by coefficient shrinkage to get a more accurate model and uses the L2 regularization technique. Ridge regression is similar to linear regression but penalizes those terms with higher variance. Random Forest is similar to that of a decision tree model but minimizes the risk of overfitting by using a bagging technique and creating many trees from the data and using a voting like structure to determine the output.

The Ridge model uses an alpha parameter to tune the model. Once the data is cleaned the alpha was determined based off of the testing data by testing values between 0 to 1 in .1 increments. Once all tests passed the minimum RMSE alpha value was recorded. Similar tests were done for the Random forest parameters by using a brute force method of tuning the number of trees, max depth, and minimum number of samples to determine the best parameters to use.



	Ridge	RF
1	0.121939	0.138892
2	0.120382	0.14365
3	0.112464	0.133621
4	0.118396	0.140616
5	0.111067	0.128592
6	0.13481	0.149538
7	0.132892	0.153608
8	0.121608	0.145863
9	0.131323	0.149954
10	0.124295	0.14114

The results of the tests of the ten datasets are shown to the right with both the Ridge regression results and results from the Random Forest regression. What is interesting is that for every test dataset Ridge regression did better than Random forest. Random forest was unable to perform results under the given RMSE. The linear nature of the data is the reason that Ridge regression performed better in for the housing data. The tech specifications of the computer that the train and test data were run: intel(R) Core(TM) i7-6500U CPU @2.5 GHz 2.6GHz 16.0 GB Ram. The time it took to run all training and test data on the 10 datasets was 91.55 seconds.

What seemed to be the most important aspect to getting better results for the error was the preprocessing step of the data. When running just the raw data with very little processing one does not get anywhere close to these results,even when tuning the actual parameters in the model like alpha in ridge regression. Being able to identify what features are useful is very important in getting the results needed to pass the test.