Introduction

There are many factors involved in real estate pricing. In reality, it is often hard for us to tell which factors are more important and which factors are not. In this project, we are going to predict house prices of Ames, Iowa with supervised predictive modeling techniques. The dataset is from a Kaggle competition, it has a total number of 80 features for each observation. During this project, we learning the entire machine learning process from data cleaning to advanced model ensembling techniques like stacking.

 Data Cleaning

The data cleaning is always the first step of any data science project. After inspecting the dataset, the first thing we found out is that the target variable is not following the normal distribution.



The violation in the normality will cause the linear model to be influenced more by the tails, so we used the log data transformation to transform the target variable back to normal distribution.



After the first step, we started to work on exception handling. We found two obvious outliers after we checked the relationship between the target variable and other independent variables like ground living areas.



There are two observations that are obvious outliers, so we decided to remove them.



Processing of missing data is the next step of our data manipulation process. First, found out missing data. After inspection, we chose to separate the imputation into three groups. We used None to replace missingness at category group and utilized 0 for the numeric data. After these two groups, we found out several variables in these two groups are not meaningful impute with 0 or none. Then we chose other ways to handle missing data in these variables. One of these variables is LotFrontage variable means linear feet of street connected to the property. According to the meaning of this variable, 0 is not a good way to impute, so we decided to use the mean. For dealing with Other variables are all in the category group, we chose to fill the missing with the most frequent patterns.

Modeling and Feature engineering:

For the modeling, our team used linear models including Lasso, Ridge and ElasticNet to predict the price of selling houses. Lasso and Ridge are used to decrease the model complexity which is the number of predictors, and they both add a hyperparameter lambda to tell you how much your model weighs on the penalization. The difference between Lasso and Ridge is Ridge penalizes sum of the squared coefficients and Lasso penalizes the sum of their absolute value. Therefore, Lasso actually forced some coefficient estimates to be exactly zero. ElasticNet is the combination of Lasso and Ridge.

In order to get the best model for all of them, we used GridSearchCV with ten fords to find the optimal lambda for each of them. With the Alpha ranged in 0 to 0.001, we have the lambda for the best model equal to 0.0001, and it comes with the best CV score 0.9143.



For Ridge, with the Alpha ranged in 0.2 to 0.3, we have the lambda for the best model equal to 0.2474, and the best CV score comes with it equal to 0.9107.



For ElasticNet, our Alpha ranged in 0 to 0.001, the lambda for the best model equal to 0.0001, and L1r\_ratio = 0.5333 which means it place emphasis on Lasso slidely more, and the best CV score is equal to 0.9107.

Beside linear models, our team also tried Tree Based Model, including andom Forest, Gradient Boosting and Extreme Gradient Boosting. The idea of them is to use combination of learning models to increase the overall result. The difference is each tree is generated by using information from previous trees.

The result of our modeling:



After trying above six models, our team wants to see if we compare the feature importance in each model, and filter out some features that do not contribute to any of the model will it improve the performance of our models. The following are the top important features in each model after removing the duplication as following:

Lasso:

KitchenAbvGr, BsmtFinSF1, TotalBsmtSF, Foundation, BsmtFullBath, HeatingQC, Fireplaces, SaleType, SaleCondition, YearBuilt, YearRemodAdd, MSZoning

Ridge:

OverallQual, OverallCond, Neighborhood, PoolQC, RoofMatl, Exterior1st, Functional, CentralAir, HouseStyle, Condition1, Street, BsmtExposure, MasVnrType, LotConfig, MSSubClass, GarageType, Heating, Exterior2nd, HalfBath, BsmtCond

ElasticNet:

FullBath, 1stFirSF, GarageCars, BsmtQual, KitchenQual, LotArea, GrLivArea

Random Forest:

GarageQual, BsmtFinType1, Electrical, MiscVal, Condition2, RoofStyle, MiscFeature, FireplaceQu, LotShape, GarageFinish, BedroomAbvGr, PavedDrive, LandContour, ExterCond, Fence, BldgType, 3SsnPorch, LandSlope, GarageCond

Gradient Boosting:

Alley, BsmtFinType2, YrSold, 1stFlrSF, GarageArea

Extreme Gradient Boosting:

ScreenPorch, WoodDeckSF, GarageYrblt, BsmtUnfSF, TotRmsAbvGrd, OpenPorchSF, LotFrontage, 2ndFlrSF

After compare with all of the above model, we decided to eliminate the following features: BsmtFinSF2, BsmtHalfBath, EnclosedPorch, MasVnrArea, MoSold. And re-run all above six models with less features in the dataframe. Unfortunately, this does not help us with improving scores for any of the models, it ends up with decreasing them. The comparison can be found as bellow:



After the reduced feature modeling experiment, we decided not to reduce our features but creating more features. So we created the TotalSF that adding up the GrLivSF and BsmtSF, and the YrBtwRemod is the number of years between the last remodeling and the year sold of the house. These 2 extra features gave us another 10% improvement in the public leaderboard. We have created many other features and used the lasso and other tree models to test the feature importance but unfortunately, the first 2 features are the two with the strongest predicting power.

Model Stacking

By finishing up with the model selection and feature engineering, we were doing some extra experiment to further improve our model accuracy. By doing some research online, we have noticed this common Kaggle technique called stacking. It is basically by combining different models and get a better result. Our first attempt was taking the average between lasso and XGBoost which is our top 2 most accurate models, and it got us a test error of 0.11963 right away which is top 22% in the leaderboard. After that, we have also tried the regular stacking method, which is taking the prediction of other models as input and use a meta-model to predict base on the base model’s output. Unfortunately, the result is closed to the best base model but it did not beat the average of XGBoost and lasso.

The final result we were taking is the weighted average of lasso, GBM and GXBoost. The weighted assigned to each model is 0.5, 0.25, 0.25.

In summary, during this machine learning project, we have learned the basic general process of a machine learning project from data cleaning, missingness imputation to cross-validation and predictive modeling. Out key get away from this project is feature engineering is always the most important thing during the machine learning. The largest improvement we gained is by adding extra features, comparing with other methods like models selection and model ensemble. In the feature, feature engineering will definitely be one of the steps that we pay more attention to, in order to obtain a better predicting result.