ABSTRACT

Our final project is about predicting house prices using what we have learned from Machine Learning class. The dataset comes from" House Prices: Advanced Regression Techniques" Kaggle competition. The project contains 79 features and one target values. There are 1460 instances in training set and 1459 instances in test set. To get the best RMSE result of prediction, we tried different feature engineering and data processing method, and using models, like Lasso, XGBoost to predict the target value, the best result of our trying is 0.1079932 from ensemble model.

KEYWORDS

House Prices: Advanced Regression Techniques

ACM Reference format:

Kewei Li and Guojing Zhang. 2018. Data Mining Final Report: INFSCI 2160. In *Proceedings of ACM Woodstock conference (WOODSTOCK'18). ACM, New York, NY, USA, 2 pages.* https://doi.org/10.1145/1234567890

1 INTRODUCTION

The problem is how can we predict the house price using given information, also avoid under fitting and overfitting, and get the best test RMSE result. We have 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa. But the instances also contain shortcoming, like have missing values, these missing values caused a lot of trouble in building our model and testing our model.

The outlier also influences our prediction result.

1.1 Importance and Interesting Parts

Firstly, this dataset contains 79 features to predict the result, which is a challenging part. Based on the text data description, we are familiar with the features characteristics. The interactive feature could be much helpful for the performance.

Second interesting part about whether or not to get the dummy variables and how to get dummy variables. Also, the outlier instances and skewed features are important and interesting in our project.

This is a typical regression problem which can been used in many business fields. By processing this project, we have the deeper understanding about data processing and model selection. Also, this problem is very common in our real life. We can extend our understanding to many similar questions.

1.2 Solution

This problem is a typical regression problem, we have tried some related technique to solve this problem, like linear regression, KNN, regression tree and some advanced regression technique like boosting and bagging. Besides that, we did the feature engineering, including feature selection, adding interactive term and some new feature.

1.3 Key Idea

The key idea of our project is doing properly data processing and choose one best performance model. We go through the whole dataset, figuring out the shortcoming of the dataset, thinking about what we can do to deal with them. We tested will it improve or decrease our test accuracy by comparing the result with baseline model. For the model selection part, we use 5-fold cross validation to do the model selection and parameter tuning.

1.4 Contribution

Guojing mainly response for the feature part manipulation, and Kewei mainly response for the model part.

2 RELATED WORK

Our project topic is one typical regression problem, so we reviewed some related regression problem. Nguyen et al [3] has predicted author age from their input texts using regression models. This problem has been modeled as a classification problem at most time, because age has been investigated in terms of differences in distributions of characteristics between cohorts. But Nguyen et al used many regression models to male predications based on the combination of different features. Their results work well in three different corpora. Even a unigram only baseline already gives strong performance. Their paper shows the powerful of regression model and the influence of Feature analysis.

Another related work is Yeo et al [4] about predicting online purchase conversion for retargeting, it's regression problem as our project. They have analysis the factors influence in the retargeting. And using Naive Regression and Periodicity-aware Regression method to do prediction. We have learned we can evaluate performance of conversion prediction using different combinations of customer-level purchasing factors.

We also learned lots of thing from one article from Linghao [2], he illustrated some important factors when doing Kaggle competition. The first one is importance of feature engineering, the features decide the upper limit of your team, and the model decide how close you can get to this upper limit. So, we spend plenty of time on the features. Second is importance of a pipeline. To decrease the time spent on the new features fitting, it is necessarily built a pipeline for whole process. The pipeline should be able to modularize feature transformations automated ensemble selection. The last is we have learned some advanced regression technique, like stacking model, using some different base model training data and combine the result as the final result. It`s a good way to get the better result.

As we have mentioned above, XG boosting plays an important role in prediction. Since all of us are not very experienced with using it, we've collected one paper on understanding XG boosting. From

Chen and Guestrin [1], they have illustrated the scalability in all scenarios of XGBoost and the effectiveness of using XGBoost. We have tried the XGBoost in our project. It works second best among our prediction.

3 DATASET DESCRIPTION

There are 2919 records for the problem. The Kaggle has already split the record into train and test. The train dataset contains 1460 records with label "SalePrice", and test dataset contains 1459 records without label. The training data set included 79 attributes and 1460 houses with the sales price for each house from January 2006 to July 2010. There are 46 categorical variables including 23 nominals and 23 ordinal ones, and 33 numeric variables in the dataset. These explanatory variables describe almost every feature of residential houses. The target value named "SalePrice". Based on the text data description, we are familiar with the features characteristics to get a better understanding to feature and discussed how to deal with it.

3.1 Features Separation

we separate the features to three different kinds of number, categorical, ordinal and numerical. For some kind of categorical variables, like Neighborhood, it doesn't have distance between them. For ordinal, likeExterQual, it contains: Ex, Gd, TA,Fa, Po. There exist differences between them. Ex means excellent is better than Gd, good. For numerical variables, we get the mean, standard deviation and quartile numbers. We also draw the distribution graph to see if it's normal distribution. we will discuss how to data processing according to different features.

3.2 Missing Values

we count the missing value of each feature; the missing value problem will be solved on the basis of our understanding to features.

MSZoning	4
LotFrontage	486
Alley	2721
Utilities	2
Exterior1st	1
Exterior2nd	1
MasVnrType	24
MasVnrArea	23
BsmtQual	81
BsmtCond	82
BsmtExposure	82
BsmtFinType1	79
BsmtFinSF1	1
BsmtFinType2	80
BsmtFinSF2	1
BsmtUnfSF	1
TotalBsmtSF	1
Electrical	1
BsmtFullBath	2
BsmtHalfBath	2
KitchenQual	1
Functional	2
FireplaceQu	1420
GarageType	157
GarageYrBlt	159
GarageFinish	159
GarageCars	1
GarageArea	1
GarageQual	159
GarageCond	159
PoolQC	2909
Fence	2348
MiscFeature	2814
SaleType	1

Figure 1: The Missing value features and their corresponding number

3.3 Distribution

we have looking at the numerical features to see if these distributions. The skewed distribution will influence the model performance because many model, like Lasso assume target value is normal distribution, Named Assumptions of normality. in order to get the distributions of continuous variables, we build the box plot and Kernel Density Estimate plot for continuous variables to show if the variables and label is norm distribution. We also count the skewness of each features.

	Skew
MiscVal	21.947195
PoolArea	16.898328
LotArea	12.822431
LowQualFinSF	12.088761
3SsnPorch	11.376065
BsmtFinSF2	4.145323
EnclosedPorch	4.003891
ScreenPorch	3.946694
MasVnrArea	2.601240
OpenPorchSF	2.535114
WoodDeckSF	1.842433
LotFrontage	1.502351
1stFlrSF	1.469604
BsmtFinSF1	1.424989
GrLivArea	1.269358
TotalBsmtSF	1.162285
BsmtUnfSF	0.919351
2ndFlrSF	0.861675
GarageArea	0.241176

Figure 2: The skewed numerical features and their corresponding skewness

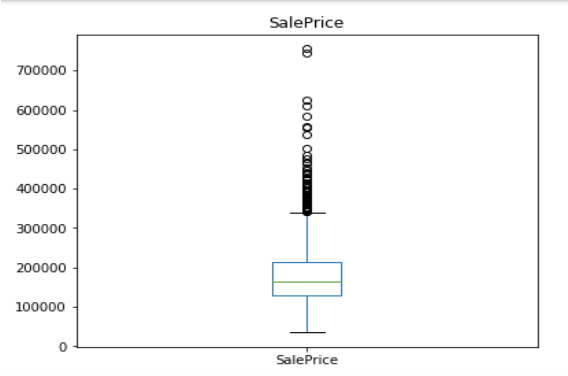

Figure 3: SalePrice Box Plot

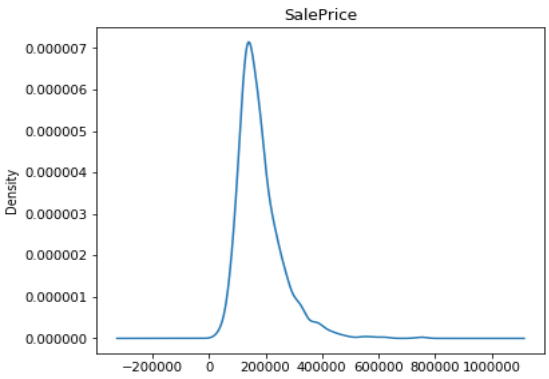

Figure 4: SalePrice KDE

3.4 Outlier

As we can see from the box plot above. Outliers are significant in the SalesPrice. Outliers are one of those statistical issues that everyone knows about, but most people aren't sure how to deal with. Most parametric statistics, like means, standard deviations, and correlations, and every statistic based on these, are highly sensitive to outliers. And since the assumptions of common statistical procedures, like linear regression and ANOVA, are also based on these statistics, outliers can really mess up our analysis.

3.5 Features Correlation:

For the correlation, we first build the correlation table between all features and label. And then we select the top 15 features to build the correlation matrix to check if there are some influential correlations within two features.

Based on the heat map, we can see that the top correlation features also have some kind of correlation between them. Therefore, we planning to need to add the interaction or maybe drop the high correlation features in order to reduce the probability of over fitting.

Beside the linear correlation, we also need to check if some of variables have some other relation with label, like polynomial relation or others. Therefore, we build scatter plot for the continuous features with label to visualize the relation of them.

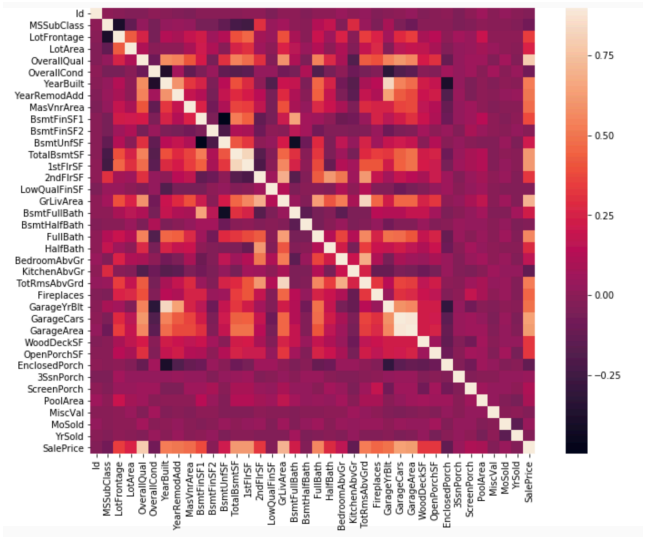

Figure 5: HeatMap for Feature Correlation

4 DATA PROCESSING

4.1 Missing Value

For different kinds of features, we have chosen different ways to deal with missing values:

For features like: Alley, MasVnrType, NA in those features means the house do not have it, it's also one kind of category of this feature, so we changed NA to "N" as a new category.

For features like: BsmtQual, BsmtCond, those features are ordinal values, the NA means zero, we changed NA to 0.

For feature like: LotFrontage, these features are numerical values, at the beginning, we changed NA to the median of this feature. Because we use baseline to measure median or mean value works better, and finally choosing median because of better R square score. But we finally group this feature by neighborhood and filled the missing value with their neighborhood' s LotFrontage median.

4.2 Drop and Add

As we have mentioned, the ID column is useless for our prediction, which is only one unique identifier for each house. So, we drop it. we also drop Utilities column, this column contains 1457 same values and 2 missing value, only one different value inside. It's The useful number of features now is 78.

Since area related features are very important to determine house prices, we add one more feature named TotalSF, which is the total area of basement, adding first and second floor areas of each house.TotalBsmtSF = 1stFlrSF+2ndFlrSF.

4.3 Different Features

For categorical features, we transform those features as dummy values. For ordinal features, like, BsmtCond, which has ex means

excellent, Gd means good, TA as typical, Fa as fair, Po as poor, and Na means no basement. We transformed this category to 5,4,3,2,1,0 separately.

For numerical variables, we examine their distribution as we mentioned before. So, we log transformation for numerical features to make them normal distribution. But after log transformation, comparing with the baseline model, the RMSE increased unexpectedly. So, we didn't use log transformation in our project. But we transformed target values.

4.4 Outlier

We have two plans for deal with outlier at the beginning, the first one is dropping all outlier because it influences the result greatly. Second way is transforming outliers to one certain values, like mode.

We finally decided to use PCA dropping the outlier. We dropped the outlier which x value is bigger than 20000 or y value is bigger 3000.

As we can see from the graph above, the instances are now more centralized.

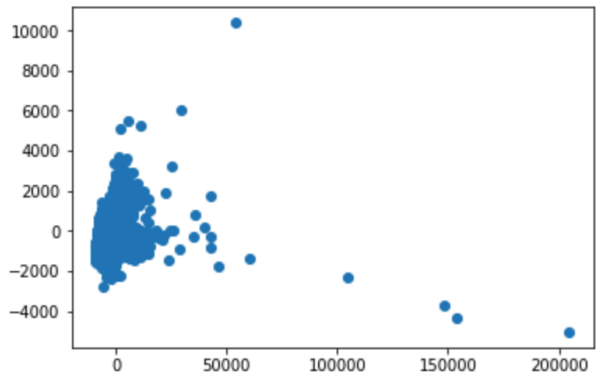

Figure 6: The Scatter Plot Before Drop Outlier

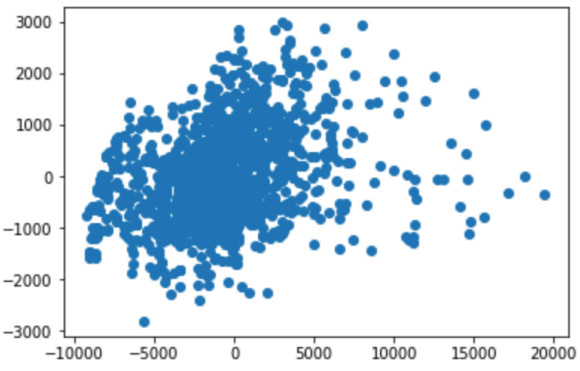

Figure 7: The Scatter Plot After Drop Outlier

As we can see from the graph above, the instances are now more centralized.

5 EVALUATION

Kaggle are evaluated on Root-Mean-Squared-Error (RMSE) between the logarithm of the predicted value and the logarithm of the observed sales price. (Taking logs means that errors in predicting expensive houses and cheap houses will affect the result equally.)

We know that the Kaggle uses Root Mean Squared Logarithmic Error (RMSLE) to calculate the score of the result of submission. Therefore, we decided to use the same score when we do the validation to bring into correspondence with Kaggle.

5.1 Validation Settings

We use 10-fold cross validation to do the model selection and parameter tuning.

First, we separated the train data from Kaggle to 2 parts, the first part is CV part and second part is test part. We keep the test part untouched to get the test RMSE. And separate the CV part to 1 part as validation set and other 4 as train parts to get the best model and parameters of that model. The CV and test separation ratio are 0.2.

5.2 Baseline Model

After necessary feature processing, like fill missing value and get dummy. We are using baseline model to evaluate the usefulness of data processing, is it improve or decrease the result.

At the beginning of this project, we choose XGBoost model as baseline for several reasons:

First, XGBoost is a tree-based model, which is not sensitive to outliers. The raw data contains some outliers, so this model is chosen as a baseline model.

Secondly, this model contains built-in functions to handle the missing values, even though these functions cannot handle missing values perfectly. But it's enough to work as a baseline model. By running the baseline model, we got a result to measure how to deal with missing values works better, and we can choose the best one to handle missing values with the help of baseline model.

Third, XGBoost is very fast to get the result. The implementation of XGBoost allow user accelerate computing by GPU. Even without accelerating by GPU, it is still very fast because of parallel computing. This is very useful for parameter tuning step because parameter tuning will spend lots of time.

But we have changed the baseline model from XGBoost to LASSO for several reasons:

First, XGBoost is not sensitive to our feature engineering, it cannot test our data processing is sufficient or not. Also, the result of XGBoost is not good enough for us. Compared with XGBoost, LASSO is more sensitive to feature engineering.

Secondly, LASSO is a standard linear model, we didn't do any feature selection to baseline model, but LASSO shrink unimportant feature coefficient to zero, it equal to feature selection. The local cross-validation training RMSE is 0.1984370, the test RMSE is 0.1446901.

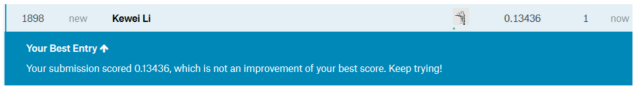

Figure 8: The Kaggle Result of Baseline Model

5.3 Result

After log transformation to make target value normal distribution, we got the second result: The local cross-validation training RMSE is 0.119749, the test RMSE is 0.1125465.

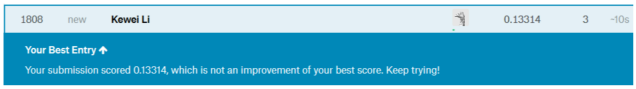

Figure 9: The Kaggle Result of Baseline Model After Log Transformation

After tuning parameter, we got alpha = 1, which satisfy what we choose at the beginning. the result shows same as the baseline line model without tuning parameter, as figure 8 shows.

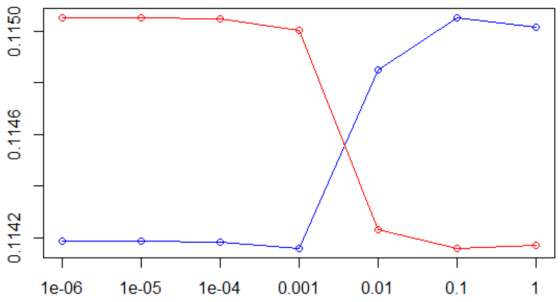

Figure 10: The Correlation between RMSE and Alpha. The Blue Line is the CV RMSE, the Red Line is Test RMSE, the X-axis Means the Alpha.

We first tied scaling the data, but the result didn't change, the local cross-validation training RMSE is 0.1201737, the test RMSE is 0.1254639. The reason may be the outlier didn't been removed. So, after scaling the data and removing the outlier instances: The local cross-validation training RMSE is 0.1158470, the test RMSE is 0.1186459.

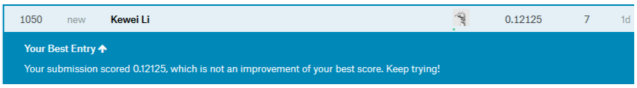

Figure 11: The Kaggle Result of Baseline Model After Scaling and Removing Outlier

Also, we log transformation to all skewed numerical features to make them normal distribution, we got the result: The local cross-validation training RMSE is 0.1133911, the test RMSE is 0.1141733.

	Skew
PoolArea	15.119426
3SsnPorch	8.924822
LowQualFinSF	8.744143
MiscVal	5.597060
ScreenPorch	2.978396
BsmtFinSF2	2.564481
EnclosedPorch	2.025461
MasVnrArea	0.636361
2ndFlrSF	0.327362
1stFlrSF	0.223905
WoodDeckSF	0.222631
LotArea	0.210453
GrLivArea	0.175316
OpenPorchSF	0.100164
BsmtFinSF1	-0.486920
LotFrontage	-0.737114
BsmtUnfSF	-1.538259
GarageArea	-3.042465
TotalBsmtSF	-3.961274

Figure 12: The Skewness after Log Transformation

6 MODEL SELECTION

6.1 Candidate Models

At the beginning, we choose different models: XGBoost, MLP, Adaboosting, RandomForest, KNeighborsRegressor, there results show above:

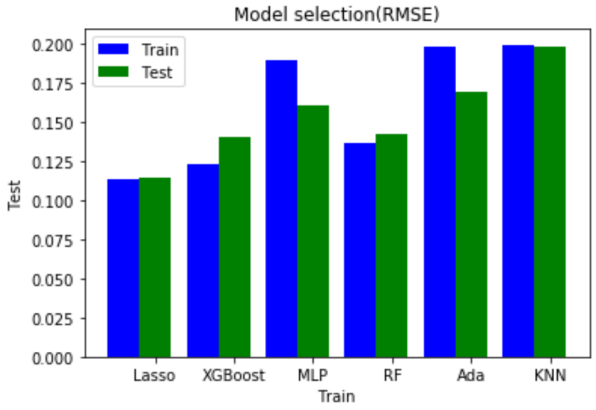

Figure 13: The Results of Training and Testing RMSE Among LASSO，XGBoost, MLP, RandomForest, AdaBoost, KNN Models

By reviewing all 5 candidate models, the XGBoost is the best one. But we still not satisfied with the result, it even worse than the

baseline model. We tried the ensemble model which will illustrate below.

6.2 Ensemble model

We have chosen two different ensemble models.

Firstly, we try to use simple ensemble model, which only calculate the average result of XGBoost and Lasso model. The local test RMSE is the best one, it's 0.1079932.

But the leaderboard score is similar to our best score.

The second we made stacking model which contains LASSO, XGBoost and RandomForest model. We got the lowest test RMSE until now.

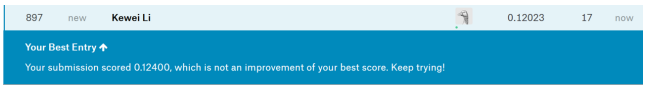

Figure 14: The Result of Stacking Model

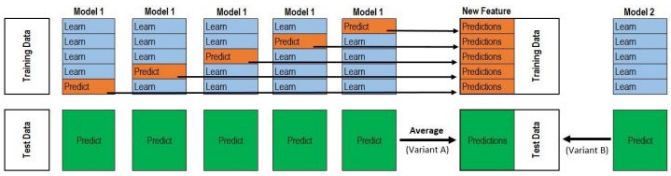

Figure 15: How do we Implement the Stack Model

The step of stacking is following this picture. At first, we choose a model as base model, by training this model with training dataset, we can get trained model M1. We use this model to predict training dataset and testing dataset respectively, then we get P1 and T1. If we have three base models, then we have P1, P2, P3 and T1, T2, T3 like the picture below, these will be our new training dataset and testing dataset.

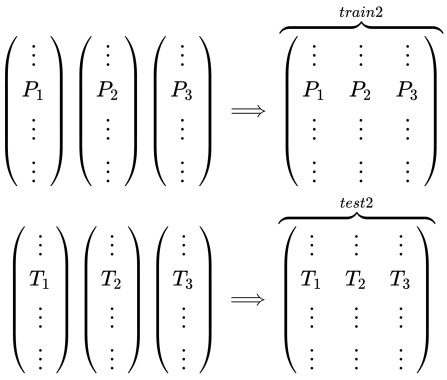

Figure 16: The Step of Stack Model

After that, we choose a model (could be one of base model or a new model) as the ensemble model in second layer, we called it "Stacker". Then we train this stacker by using new training dataset and get the final prediction by using new testing dataset.

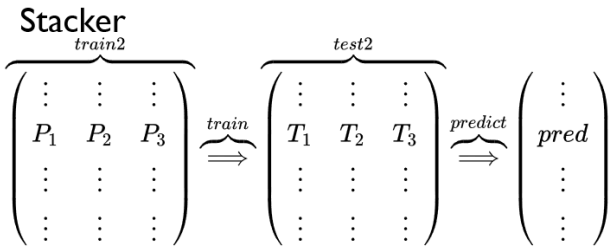

Figure 17: The Step of Stack Model

The problem is we train the base model from training dataset and we use it to predict training dataset, it must be overfitting. To solve this problem, we use cross validation to train our base model. Taking 2-folds cross validation as an example, we split the training dataset into 2 parts. Training the first part to predict the second part and training the second part to predict the first part. The combination will be the correct P1.

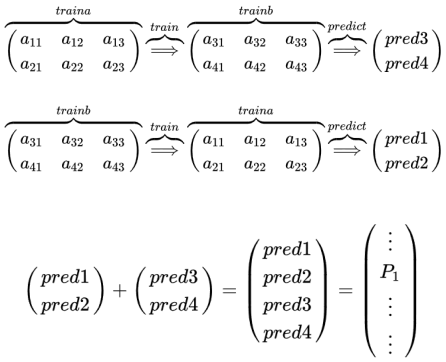

Figure 18: The Step of Stack Model

7 DISCUSSION

As we have discussed in the related work part, many understanding efforts based on the deep analysis and understanding of features. We have put many efforts on features analysis. And we deal with the missing value based on our understanding and analysis, but simply fill them with 0 or N.

We also tried many new models which we have not studied in the class, like XGBoost and ensemble model. They are having better performance or quicker running times.

But they also exist some limitation of our work, we can try more powerful models which we seen in the related work. And for ensemble model parts, we only use R to implement the simpler one, which is the average of LASSO and KNN, LASSO and XGBoost. We didn't use R to implement the stacking model. Last limitation part is, we use our stacking result as final result and upload it to Kaggle, we got 893 ranking of 4330 teams. It's good but not best, if we have more time, we will try use stacking result as part of our final result, we will try combine the stacking result with more different models. It may get better performance.

For the model selection, there are still some new powerful models we can choose. Elastic Net, which combine the Ridge and Lasso, maybe do the better feature selection and result than Lasso. Another powerful tree model is called light GBM, which can improve the speed and accuracy from XGBoost.

8 CONCLUSION

From this competition, we learn a lot. In feature treatment part, we need to read the description file to understand the meaning of feature and the meaning of missing value of each feature. By understanding these, we can transform the row file to the data that can do the prediction, like fill the missing value, get dummy variables and transfer string data to integer value. In the model part, we test 6 different models, which includes linear model, Lasso, and non-linear model, XGBoost, which is a tree-based model. And based on our result, when we do the complicated model, the non-linear model is always out-performing the linear model. We also tried ensemble model, which got the best performance of our prediction. If we have more time, we can improve the performance by trying combine the stacking result with more different models.

ACKNOWLEDGMENTS

The authors would like to thank Luda Wang for providing supports. Two authors of the current study have equal contributions.

REFERENCES

[1] Chen, T. , & Guestrin, C. . (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM.

[2] How to Rank 10% in Your First Kaggle Competition:2018. *https://www.kdnuggets.com/2016/11/rank-ten-precent-first-kaggle-competition.html.* Accessed: 2018- 12- 12.

[3] Nguyen, D. , & Smith, N. A. . (2011). Author age prediction from text using linear regression. *Acl-hlt Workshop on Language Technology for Cultural Heritage*. Association for Computational Linguistics.

[4] Yeo, J. , Kim, S. , Koh, E. , Hwang, S. W. , & Lipka, N. . (2017). Predicting online purchase conversion for retargeting.