Ming Xian

Professor Renyu Zhang

BUSF-SHU 210

12 April, 2021

Kaggle Project Report

Classification

·Features Selection

I selected these variables 'speed', 'max_load', 'weather_grade', 'source_type', 'grid_distance', 'urgency', 'hour' as predictors, based on their relevance to affect the delivery action. To be more specific, variables of speed, max_load and weather_grade, source_type and urgency are unique to different delivery men under different starting situations. The grid_distance is obviously useful in prediction because it provides the distance they the delivery man needs to travel. The hour is also nonnegligible in predicting which action to take, as there should be difference between rush hours and normal hours. On the contrary, irrelevant variables like some specific ids and direct demographic information are removed from both the training and testing sets. The y variable in the training set is 'action_type', and the y variable we are interested in prediction in the testing set is 'action_type_DELIVERY', with 1 representing the delivery action. Also, I got the dummy variables for 'weather_grade', 'source_type' and 'hour'.

·Model Selection

Note that for both classification and regression problem, the general steps for examining their prediction scores are: compare in-sample scores >> figure out the one having the best in-sample score and perform out-of-sample score to double check whether this model is good enough. For the first step of comparing in-sample scores,

sometimes I used down-samples because running the original dataset would require too much time (more than 6 hours).

I tried 4 models: logistic regression, random forest, kNN and decision tree, and ended up choosing the decision tree as the best model, as it gives the best AUC.

1. For the logistic regression model, I subtracted L2(ridge) and L1(lasso) penalties to regularize, in order to address the overfitting issue. Then, I used cross-validation to fine tune the regularization parameter C, with the number of folds equal to 5. For the L2 case, the best C is 0.001 with an associated AUC of 0.8061. For the L1 case, the best C is 10 with an associated AUC of 0.7140. Note that for L1, I split the data into two parts and ran the train set of 30%, because running all the original training set takes too much time (more than 8 hours for my computer). The AUC computed for L1 is in-sample as well, so as to compare apples to apples.
2. For the random forest model, I also cross-validate to fine tune the regularization parameters, getting the AUC of 0.7820, which is worse than the L2 logistic regression model and better than the L1 logistic regression model.
3. For the kNN model, I used cross-validation to fine tune the regularization parameter, getting an AUC of 0.8340. This score is the best among all models so far.
4. For the decision tree model, I used cross-validation to fine tune the regularization parameters, getting an AUC of 0.8503. The overall testing accuracy is 0.7787, which is also reasonably good. I've computed the Mean-F1-Score as well, which is 0.8046, reassuring that this model is good enough. These two pretty good scores enabled me to choose the decision tree model as the best model in prediction.

·Handling Outliers

When running the kNN model, I standardized the features of the data points both in the training set and testing sets, in order to mitigate some outlier effects.

Regression

·Features Selection

I selected the same predicting variables 'speed', 'max_load', 'weather_grade', 'source_type', 'grid_distance', 'urgency' and 'hour' as predictors as what I did in classification, based on their relevance to affect the delivery action (the reasons are same as what I've explained before). I also transformed some into dummy variables same as classification. The target y variable should be assigned to 'expected_use_time'.

·Model Selection

I tried 4 models: linear regression, decision tree, XGBT and random forest, and ended up choosing random forest as the best model, as it gives the best R^2.

1. For the linear regression model, I applied elastic net linear regression in regularization to address overfitting. Then I used cross-validation to fine tune the regularization parameter alpha, with the number of folds equal to 5. The best alpha yields to 0.02 with an R^2 of 0.2509.
2. For the decision tree model, I used cross-validation to fine tune the regularization parameters, getting the score of R^2 -- 0.3518, which is better than the linear regression model.
3. For the XGBT model and random forest, surprisingly they had the exact same R^2 of 0.3812, which is the best score so far. I continued to compare their out-of-sample R^2, it's 0.3980 for XGBT and 0.3969 for random forest. We can see their difference is very tiny, so there's no strong evidence for us to choose the model having a tiny bigger R^2. Usually we tend to choose the simpler model, yet based on what we learnt so far, we are not equipped with enough knowledge to tell which model is simpler. Hence I randomly chose the random

forest model as my best model. After that, I tried clustering to see if it can improve the score, expecting to see the delivery men of similar behavior can be explained by models individually better, yet it turned out to be worse. So, I used the simple random forest model in the final prediction.

·Handling Outliers

I tried PCA with 6 components to see whether dimensionality reduction through PCA could help improve the classification accuracy and remove outliers as data noise. However, it turned out PCA does not make improvement. Actually, there's no need to even try PCA since these features are not linearly correlated.