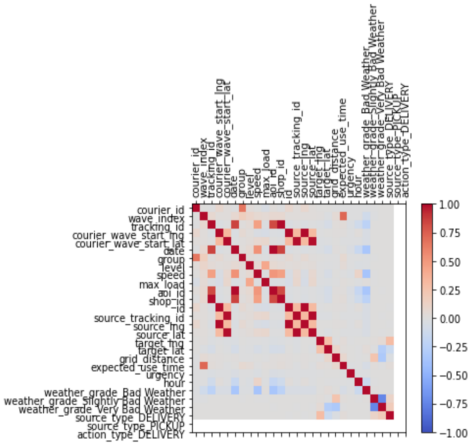
Machine Learning Kaggle Competition Report

1. Classification

1.1 Features Identification, Data Preprocessing & General Procedure

Before running any model, I pre-processed the data in the training dataset by getting dummies for variables 'weather_grade', 'source_type', and 'action_type'. I dropped the variables that are irrelevant to the model or those containing information that has been better presented by the other variables as could be observed from the plot showing the correlations between all the variables, such as 'courier_id', 'wave_index', 'tracking_id', 'group', 'id', 'aoi_id', 'shop_id', 'courier_wave_start_lng', 'courier_wave_start_lat', 'source_tracking_id', 'source_lng', 'source_lat', 'target_lng', 'target_lat', and 'expected_use_time'.

I set Y to be the 0-1 classified variable 'action_type_DELIVERY'. X contains the remaining 12 variables. I split the entire training dataset into 70% training subset and 30% testing subset with stratification based on Y. I planned to pre-process the data in the training dataset, train baseline models, conduct feature engineering, and fine-tune the parameters using the training subset and the testing subset. I would then select the model with the highest out-of-sample ROC-AUC score and use it to retrain the entire training dataset, and eventually, I will predict the testing dataset using the selected model.



1.2 Baseline Models, Feature Engineering & Parameters Hyper-tuning

1.2.1 Logistic Regression

I first trained a baseline Logistic Regression model on the training subset, setting the parameter C to be 1 and the threshold t to be 0.5 by default, and the corresponding out-of-sample ROC-AUC was 0.7191.

I then made a pipeline 'pca_lr' which contains 3 steps—StandardScaler, PCA, and Logistic Regression. PCA was utilized to minimize the model bias and avoid overfitting. After completing a manual GridSearchCV (training a series of models on the training subset and testing subset with the number of components for PCA ranging from 11 to 8), I observed that the out-of-sample ROC-AUC decreased as the number of components decreased. Therefore, the best Logistic Regression model was that with StandardScaler but without PCA, which reported the highest out-of-sample ROC-AUC = 0.8098. Using this model to train the entire training dataset, the reported in-sample ROC-AUC = 0.8103. Compared to the below models, Logistic Regression was not the model with the highest out-of-sample ROC-AUC.

1.2.2 K-Nearest Neighbors

I first Standard-Scaled and performed a GridSearchCV on the number of neighbors to select the best n_neighbors for a baseline KNN model using the training subset. From 10, 50, 100, 150, 200, the n_neighbors associated with the highest in-sample ROC-AUC =

0.8055 was 100.

I then trained a series of KNN models by performing a 3-steps pipeline — StandardScaler, PCA with n_components ranging from 12 to 8, and KNN with n_neighbors = 100. After a manual GridSearchCV, the best KNN model (n_neighbors = 100) with StandardScaler and PCA (n_components = 8) reported the highest out-of-sample ROC-AUC = 0.8595. Using this model to train the entire training dataset, the reported in-sample ROC-AUC = 0.8678.

1.2.3 Decision Tree Classifier

I first trained a baseline Decision Tree model without PCA, which reported in-sample ROC-AUC = 1 and out-of-sample ROC-AUC = 0.7630. I further trained a series of Decision Tree models with PCA n_components ranging from 11 to 6. The best Decision Tree model with PCA (n_components = 8) reported out-of-sample ROC-AUC = 0.7630. Because the out-of-sample was relatively low, the Decision Tree Classifier was not the best compared to the other models.

1.2.4 Random Forest Classifier

I first trained a baseline Random Forest model with the number of estimators = 100, 500, and 1000, respectively. Among the above, Random Forest with n_estimators = 1000 reported out-of-sample ROC-AUC only slightly higher than that with n_estimators = 500. Hence, 500 was an adequate number of estimators. I further trained a series of Random Forest models with various PCA n_components and various degrees of Polynomial Features. The best model among these was that with Polynomial Features (degree = 2), which reported in-sample ROC-AUC = 1 and out-of-sample ROC-AUC = 0.8634.

1.2.5 XGBT

After GridSearching for parameters, I first trained a baseline XGBT model setting colsample_bynode=0.8, learning_rate = 0.1, gamma = 0.001, max_depth = 10, n_estimators = 50, which reported in-sample ROC-AUC = 0.8899 and out-of-sample ROC-AUC = 0.8737. After applying PCA with various n_components, the baseline model without PCA performed the best. I use this model to train the entire training dataset, which reported in-sample ROC-AUC = 0.8863.

1.3 Retrain the Entire Training Set & Predict the Entire Testing Set

After testing out a series of models, I selected XGBT with colsample_bynode=0.8, learning_rate = 0.1, gamma = 0.001, max_depth = 10, n_estimators = 50. I retrained the entire Training dataset and predicted the entire Testing dataset using this model. However, the prediction was not satisfactory probably because I have deleted too many variables.

1.4 Reidentify Variables & Improve the XGBT Model

I included variables 'courier_wave_start_lng', 'courier_wave_start_lat', 'source_tracking_id', 'source_lng', 'source_lat', 'target_lng', and 'target_lat'. After a series of GridSearchCV, the parameters associated with the best model were max_depth = 10 and n_estimators = 300. The reported in-sample ROC-AUC was 0.9991, the out-of-sample ROC-AUC was 0.9863, which could predict the testing dataset better than the previous models.

2. Regression

2.1 Features Identification, Data Preprocessing & General Procedure

Before running any model, I pre-processed the data in the training dataset by getting dummies for variables “weather_grade” and “source_type”. I dropped the variables that are irrelevant to the model or those containing information that has been better presented by the other variables, including 'courier_id', 'wave_index', 'tracking_id', 'group', 'id', 'aoi_id', 'shop_id', 'courier_wave_start_lng', 'courier_wave_start_lat', 'source_tracking_id', 'source_lng', 'source_lat', 'target_lng', 'target_lat', and ‘action_type’.

I set Y to be the outcome variable 'expected_use_time'. X contains the remaining 12 variables. I split the entire training dataset into 70% training subset and 30% testing subset. I planned to pre-process the data in the training dataset, train baseline models, conduct feature engineering, and fine-tune the parameters using the training subset and the testing subset. I would then select the model with the highest out-of-sample ROC-AUC score and use it to retrain the entire training dataset, and eventually, I will predict the testing dataset using the selected model.

2.2 Baseline Models, Feature Engineering & Parameters Hyper-tuning

2.2.1 Linear Regression

I first trained a baseline Linear Regression model with StandardScaler on the training subset, and the corresponding out-of-sample R-squared was 0.2955.

After completing a manual GridSearchCV (training a series of models on the training subset and testing subset with the number of components for PCA ranging from 11 to 8), I observed that the best Linear Regression model was the baseline model.

I further applied polynomial features by making a pipeline consisting of StandardScaler, Polynomial Features (degree = 2 to 4), and Linear Regression. The best model was that with Polynomial Features (degree = 3) with out-of-sample R-squared = 0.3735. Using this model to train the entire training dataset, the reported in-sample R-squared = 0.3758.

2.2.2 K-Nearest Neighbors Regressor

I first performed a GridSearchCV on the number of neighbors to select the best n_neighbors for a baseline KNN model using the training subset. From 10, 50, 100, 150, 200, the n_neighbors associated with the highest in-sample R-squared = 0.2315 was 150.

I then trained a series of KNN models by performing a 3-steps pipeline—StandardScaler, PCA with n_components ranging from 11 to 8, and KNN with n_neighbors = 150. After the manual GridSearchCV, the best KNN model (n_neighbors = 150) with StandardScaler and PCA (n_components = 8) reported the highest out-of-sample R-squared = 0.3502. As its out-of-sample R-squared was relatively low, KNN Regressor was not the best fit for the data.

2.2.3 Decision Tree Regressor

I first trained a baseline Decision Tree model (ccp_alpha = 0.01, max_depth=10) without PCA, which reported in-sample R-squared = 0.4018 and out-of-sample R-squared = 0.3669. I further trained a series of Decision Tree models with PCA n_components ranging from 11 to 8. The best Decision Tree model was the baseline model. I further applied polynomial features by making a pipeline consisting of Polynomial Features (degree = 2 to 4), and Decision Tree Regressor. The best model did not outperform the baseline model. As its out-of-sample R-squared was relatively low, the Decision Tree

regressor was not the best fit for the data.

2.2.4 Random Forest Regressor

I first trained a baseline Random Forest model with the number of estimators = 50, 100, and 500, respectively. Among the above, Random Forest with n_estimators = 500 reported in-sample R-squared = 0.9126 and the highest out-of-sample R-squared =0.3606. As its out-of-sample R-squared was relatively low, Random Forest Regressor was not the best fit for the data.

2.2.5 XGBT

After GridSearching for parameters, I first trained a baseline XGBT model setting colsample_bynode=0.8, learning_rate = 0.1, gamma = 0.001, max_depth = 10, n_estimators = 50, which reported in-sample R-squared = 0.4657 and out-of-sample R-squared = 0.3893. After testing PCA with various n_components and various degrees of polynomial features, the baseline model without PCA performed the best. I used this model to train the entire training dataset, which reported in-sample ROC-AUC = 0.4506.

2.3 Retrain the entire Training set and predict the entire Testing set

After testing out a series of models, I selected XGBT model with colsample_bynode = 0.8, learning_rate = 0.1, gamma = 0.001, max_depth = 10, n_estimators = 50. I retrained the entire Training dataset and predicted the entire Testing dataset using this model. However, as the classification problem, the prediction was not satisfactory probably because I have deleted too many variables.

2.4 Reidentify Variables & Improve the XGBT Model

I included variables 'courier_wave_start_lng', 'courier_wave_start_lat', 'source_tracking_id', 'source_lng', 'source_lat', 'target_lng', and 'target_lat'. After a series of GridSearchCV, the parameters associated with the best model were: max_depth = 10, n_estimators = 300. The reported in-sample R-squared was 0.8241, the out-of-sample R-squared was 0.8241, which could predict the testing dataset better than the previous models.