Fraud Detection with Bank Transaction Data

Ben Liu

SharpestMinds

Contents

Introduction

- Background: Fraud is a significant problem for any bank. Fraud can take many forms, which involves large amount of variables.
- Data: This project is based on a credit card transaction dataset that contains information about 800,000 records and 29 features from a bank in Canada.
- Objective: The goal is to build a predictive model to determine whether a given transaction will be fraudulent or not.

Methodology

- Since the fraud detection is a binary classification problem, I applied several classic supervised classification machine learning models, such as Logistic regression, XGBoost, and LightGBM.
- The primary metric would be ROC_AUC.

Data Preparation

The original dataset is in line-delimited JSON format:

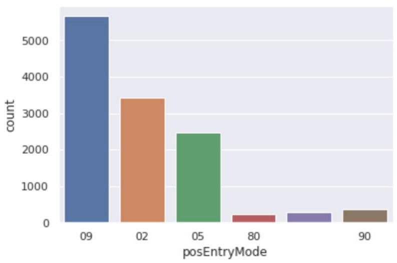
```
{"accountNumber": "737265056", "customerId": "737265056", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-08-13T14:27:32", "
{"accountNumber": "737265056", "customerId": "737265056", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-10-11T05:05:54", "
{"accountNumber": "737265056", "customerId": "737265056", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-11-08T09:18:39", "
{"accountNumber": "737265056", "customerId": "737265056", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-12-10T02:14:50", "
{"accountNumber": "830329091", "customerId": "830329091", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-03-24T21:04:46", "
{"accountNumber": "830329091", "customerId": "830329091", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-04-19T16:24:27", "
{"accountNumber": "830329091", "customerId": "830329091", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-05-21T14:50:35", "
{"accountNumber": "830329091", "customerId": "830329091", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-06-03T00:31:21", "
{"accountNumber": "830329091", "customerId": "830329091", "creditLimit": 5000.0, "availableMoney": 4990.63, "currentBalance": 9.37, "transactionDateTime":
{"accountNumber": "830329091", "customerId": "830329091", "creditLimit": 5000.0, "availableMoney": 5000.0, "transactionDateTime": "2016-07-11T10:47:16", "
```

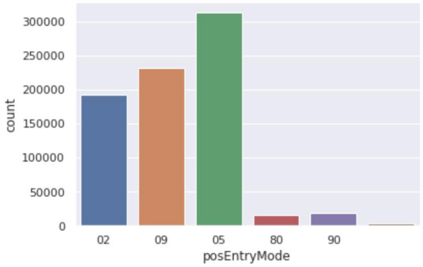
Transform it to a data frame:

	accountNumber	customerId	creditLimit	availableMoney	transactionDateTime	transactionAmount	merchantName	acqCountry	merchantCountryCode	posEntryMode
0	737265056	737265056	5000.0	5000.0	2016-08-13T14:27:32	98.55	Uber	US	US	02
1	737265056	737265056	5000.0	5000.0	2016-10-11T05:05:54	74.51	AMC #191138	US	US	09
2	737265056	737265056	5000.0	5000.0	2016-11-08T09:18:39	7.47	Play Store	US	US	09
3	737265056	737265056	5000.0	5000.0	2016-12-10T02:14:50	7.47	Play Store	US	US	09
4	830329091	830329091	5000.0	5000.0	2016-03-24T21:04:46	71.18	Tim Hortons #947751	US	US	02

EDA

Through exploratory data analysis, I selected a few features as explanatory variables. Take the "postEntryMode" for example:





The left plot was the "postEntryMode" distribution in the fraudulent transations, while the right plot showed that in the not fraudulent transactions. It seemed like "posEntryMode" had an influence on if it's a fraud. So this feature would be added to my model.

Data Cleansing

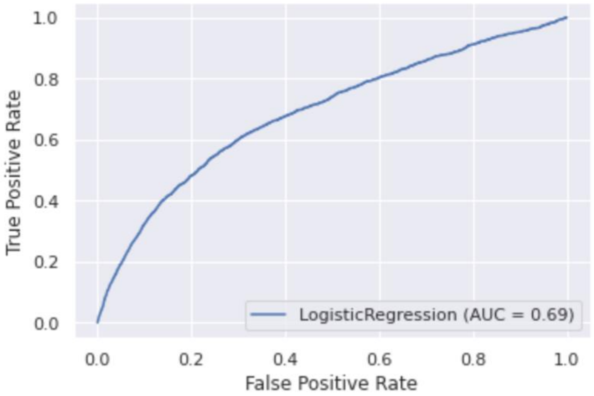
- Impute missing values.
- Encode categorical variables to dummy variables.
- Deal with outliers.

Feature Engineering

Explanatory variables (3 cases):

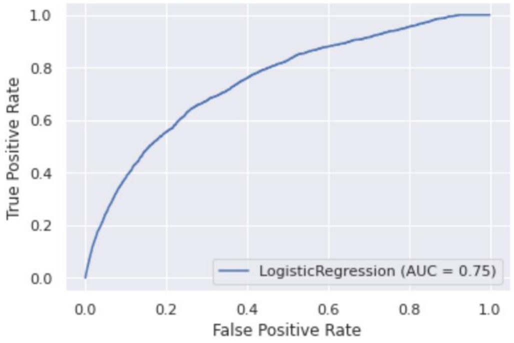
- 'cvvNotSame', 'amountOver', 'posEM_new', 'hour', 'transactionAmount', 'availableMoney', 'cardPresent'.
- 'cvvNotSame', 'amountOver', 'posEM_new', 'hour', 'transactionAmount', 'availableMoney', 'cardPresent', 'merchantCategoryCode'.
- 'cvvNotSame', 'amountOver', 'posEM-new', 'hour', 'transactionAmount', 'availableMoney', 'cardPresent', 'transactionType'.

Model1 - Logistic regression (Case 2)



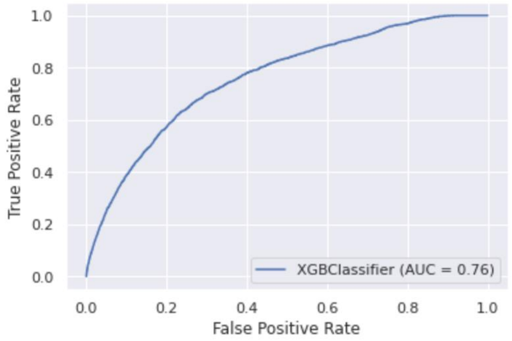
After train-test split, cross validation, model fitting, the ROC_AUC was 0.69.

Model1 - Logistic regression after scaling (Case 2)



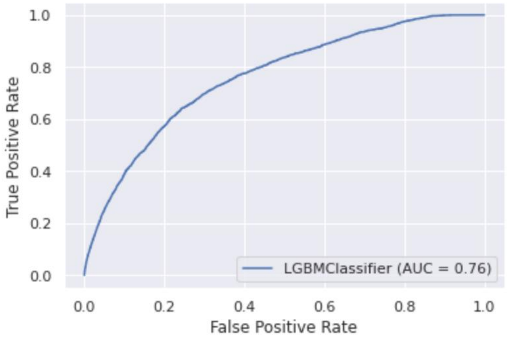
Since MinMax scaling before applying the model increased the ROC_AUC to 0.75, the scaled data frame would be used in the following modeling experiments.

Model2 - XGBoost (Case 2)



For the XGBoost model, I utilized the Grid Search method to conduct the hyperparameter tuning (learning rate, number of estimators), selected the optimal combination of parameters, and the related metric was 0.76.

Model3 - LightGBM (Case 2)



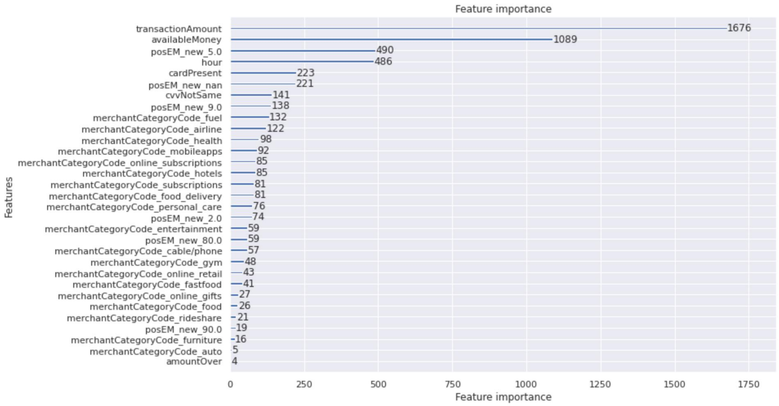
Similarly, Grid Search approach was used to conduct the hyperparameter tuning (learning rate, number of estimators, max_depth) for the lightGBM model. And the ROC_AUC of the optimal model was also 0.76.

Result

ROC_AUC	Case 1	Case 2	Case 3
Logistic regression (raw)	0.70	0.69	0.72
Logistic regression (scaling)	0.72	0.75	0.72
XGBoost (scaling)	0.74	0.76	0.74
LightGBM (scaling)	0.74	0.76	0.74

According to the ROC_AUC of 3 models in 3 cases, both XGBoost and LightGBM with the feature combination Case 2, achieved the highest ROC_AUC as 0.76, increasing the base value by 7%.

Feature Importance



It's straightforward that "transactionAmount", "availableMoney", "posEM_new" were the key factors that determined whether a transaction was a fraud or not.

Conclusion

- Accuracy: XGBoost and LightGBM both achieved the highest ROC_AUC as 0.76.
- Efficiency: LightGBM was computationally faster than XGBoost.
- Feature importance: The feature "transactionAmount" was the most significant factor to determine if it's a fraud.

Future

- Different kinds of feature combination.
- More machine learning algorithms.
- Various types of hyperparameter tuning.