Fraud Detection Analysis

Haochun Niu

12/26/2021

Executive Summary

As online shopping becomes more and more frequent, credit card frauds have been one of the most annoying problem for both bank and customers. In the past, I also have encountered several differnt credit card frauds. So, even though I have not lost any asset or money in the past, I become really concerned about this issue and do not really trust any online transaction any more.

As an enthusiastic data scientist, I decide to do something to make the world better. In this project, I create a highly accurate XGBoost Classification model to detect credit card frauds, based on the online transaction simulation data generated by Kartik Shenoy. The data was generated using Sparkov Data Generation | Github tool created by Brandon Harris. This simulation was run for the duration - 1 Jan 2019 to 31 Dec 2020.(https://www.kaggle.com/kartik2112/fraud-detection)

1.Data Description

Before we jump into the process of modeling, let's take a look at the data. Among all the variables, totally 18 variable used, and the output variable is the is_fraud variable. The table below shows the name, data type, and interpretation of each feature.

Features	*Data Type*	*Definition*
is_fraud (Y)	Categorical	A binary indicator of fraud transaction (1: Yes, 0: No)
amt	Numeric	The amount of transaction
gender	Categorical	The gender of card holder
city	Categorical	The city card holder locate
state	Categorical	The state card holder locate
zip	Categorical	The zip code of the location the card holder locate
category	Categorical	The category of the purchased item
lat	Numeric	The latitude of card holder's location
long	Numeric	The longtitude of card holder's location
city_pop	Numeric	The population of the city the card holder locate
job	Categorical	The card holder's job
merch_lat	Numeric	The latitude of the location of the transaction
merch_long	Numeric	The longtitude of the location of the transaction
distance	Numeric	The distance between locations of transaction and card holder
weekday	Categorical	The weekday of the transaction datetime (ex:Mon, Tue, ...)
hour	Numeric	The hour of the transaciton datetime (0~23)
day	Numeric	The day of the transaction datetime (1~31)
month	Numeric	The month of the transaction date time (1~12)
age	Numeric	The age of the card holder

2.Data Visualization

In order to gain a much deeper understanding of the data, I visualize the data with Tableau. You can access the dashboard via the link (https://public.tableau.com/app/profile/hao.chun.niu/viz/FraudDetectionEDA/Dashboard).

According to the visualization, I find out that hour might be the most effective feature. Apparently, a lot more frauds are done during midnight and early morning. As for gender, fraud tends to happen more on male card holder. However, surprisingly, it seems like the age of the card holder and the distance between the transaction and card holder do not have very obvious effects on the appearance of fraud

3.Data Split

Before modeling, to avoid overfiting, I split the data into train, validation, and test data. I will use the train dataset to fit the models, the validation to select the best model, and the test dataset to evaluate final model's performance.

Dataset	*Number of Rows*	*Percentage*
train	972507	52.5%
validation	324168	17.5%
test	555719	30%

4.Imbalance Data

In the orginal train data, the fraud cases are only accounted for 0.58% of data. Hence, we need to solve the imbalance data issue before modeling. Here, I use SMOTE to do the over-sampling and under-sampling. Eventually, by **over-sampling the minority 1200% and under-sampling the majority 150%**, I solved the imbalance data issue. Right now the minorities (Fraud cases) are accounted for about **40%** of the train data.

Is Fraud	*Before SMOTE*	*After SMOTE*
True (1)	0.58%	41.94%
False (0)	99.42%	58.06%

5.Evaluation Metrix

Given that identifying as many fraud cases as possible is the top priority and that the consequence of misclassifing non-fraud cases into fraud cases is realtively small, I will use the 'Recall' performance metric as my evaluation method. However, I will not irrationally misclassify non-fraud cases into fraud cases just to get higher recall. Therefore, I will not mannualy change the cut-off threshold, meaning that I will still use 0.5 as the cut-off threshold for classification. The table below shows the interpretation and definition of Recall.

Recall	*Data Type*	*Definition*
Recall_positive	**(Captured Positive)/(All Positive)**	What percentage of positive cases are captured by the model
Recall_negative	**(Captured Negative)/(All Negative)**	What percentage of negative cases are captured by the model

6.Modeling

Throughout the entire process, I tried several methods, such as SVM, Random Forest, and XGBoost. Eventually, the best model is the **XGBoost model with eta = 0.1 and max_depth =10**.

```
## ##### xgb.Booster
## Handle is invalid! Suggest using xgb.Booster.complete
## raw: 2.4 Mb
## call:
##   xgb.train(params = params, data = dtrain, nrounds = nrounds,
##     watchlist = watchlist, verbose = verbose, print_every_n = print_every_n,
##     early_stopping_rounds = early_stopping_rounds, maximize = maximize,
##     save_period = save_period, save_name = save_name, xgb_model = xgb_model,
##     callbacks = callbacks, eval_metric = "merror", num_class = 2,
##     eta = 0.1, max_depth = 10)
## params (as set within xgb.train):
##   eval_metric = "merror", num_class = "2", eta = "0.1", max_depth = "10", validate_parameters = "TRU
## callbacks:
##   cb.print.evaluation(period = print_every_n)
##   cb.evaluation.log()
## # of features: 18
## niter: 25
## nfeatures : 18
## evaluation_log:
##     iter train_merror
##        1     0.052375
##        2     0.049098
## ---
##       24     0.028305
##       25     0.028070
```

The model also has an outstanding performance on the test dataset. Even though my model misclassifies many non-fraud cases into fraud, my model succeeds to capture more than **90%** of the fraud cases, meaning that we could capture most of the frauds with my model.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction      0      1
##          0 546939    190
##          1   6635   1955
##
##                Accuracy : 0.9877
##                  95% CI : (0.9874, 0.988)
##     No Information Rate : 0.9961
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3603
##
##  Mcnemar's Test P-Value : <2e-16
##
##               Precision : 0.227590
##                  Recall : 0.911422
```

```
##                      F1 : 0.364229
##             Prevalence : 0.003860
##         Detection Rate : 0.003518
##   Detection Prevalence : 0.015457
##      Balanced Accuracy : 0.949718
##
##       'Positive' Class : 1
##
```

7.Feature Importance

Among the 18 features, according to information gain, **the transaction amount** and **the hour of the transaction** have way more significant effects than the rest of the features. The bar below shows the importance of each feature.