Final Report: Detecting Credit Card Fraud

Mark Cohen

Introduction

It is estimated that fraud involving credit and debit cards causes losses of nearly $30 billion worldwide, as of 2019.[1] Such fraudulent transactions represent a major financial risk not just for consumers — who are generally protected from responsibility for such payments — but even more so for issuing banks and merchants.[2]

A central aspect of the problem of credit card fraud is *timing*. Once the customer notices an unauthorized purchase on their monthly bill or online account, the transaction will typically have already been completed several weeks in the past. The goods or services have been provided by the merchant, and the bank has transferred the funds. To effectively address the risk, it is essential to identify fraudulent transactions in near real-time so that the payment can be preemptively rejected or at least flagged for further confirmation.

The goal of this project is to employ machine learning to use the data available to the card issuer at the time of a transaction to predict whether it is at elevated risk of being fraudulent. The phrase "elevated risk" is used deliberately: the goal is not to definitively label a transaction as fraudulent but instead to provide a signal to initiate heightened security measures, such as requesting the customer to confirm the transaction via text message.

As such, the focus is on casting a wide net that will capture a large share of fraudulent transactions. Reducing false positives is an important criterion for choosing among alternative models, but the expectation should be that they will not be eliminated. In fact, to preview the results, a tuned XGBoost classifier model was able to correctly identify three-quarters of fraudulent transactions at the cost of 11 false positives for each case of fraud.

Data and Modeling Approach

The data used in this project comprise a simulation of over 20 million transactions by 2,000 U.S.-based card users over multiple decades generated by a team at IBM.[3] The advantage of synthetic data is that it can include information that would risk identifiability in real-world data and thus could not be publicly shared due to privacy concerns.

In addition to information on each transaction — including amount, location, merchant name and type — the dataset includes details about users and their cards — including home

[1] https://www.cnbc.com/2021/01/27/credit-card-fraud-is-on-the-rise-due-to-covid-pandemic.html

[2] Who Pays For Fraudulent Credit Card Transactions? (forbes.com)

[3] Erik R. Altman. 2019. "Synthesizing Credit Card Transactions." https://arxiv.org/abs/1910.03033

address, income, credit limits. Each transaction is labeled as either fraudulent or not.

The target labels are highly unbalanced: just over one tenth of one percent of the transactions are fraudulent. Thankfully, with so many transactions, it is possible to nonetheless generate a sizable balanced dataset without oversampling, while still holding back data for testing.

The procedure adopted was:

1. Randomly select 200 users (i.e. one-tenth) as a testing data set. This represented a total of 1,176,707 transaction records. *Users* were selected instead of transactions so as to avoid any contamination of user-level features between the training and testing data.
2. Pull out all transactions labeled as fraudulent for the remaining (1,800) users. This amounted to 13,850 transactions.
3. Randomly select a matching number of non-fraudulent transactions from the same users.

The original data included 16 features, plus the target, on each transaction record, 20 features on the user level, and 18 features on the card level. Some of these were not useful, or at least not on their own, and several categorical features needed to be recorded. Ultimately, as input fed into the model, the data had 42 features:

user index	whether the transaction occurred within the user’s home city	whether an incorrect expiration date was entered
card index	whether the transaction occurred within the user’s home state	whether the transaction used a chip
the amount of the transaction	whether the transaction occurred outside the U.S.	whether the transaction was online
whether the card had a chip	the user’s age	whether the transaction swiped the card
the total number of physical cards issued for a particular card account	whether the user was retired	whether the card brand was American Express
the longitude of the user’s residence	the time until the card expires	whether the card brand was Discover

the latitude of the user’s residence	the time since the account was opened	whether the card brand was Mastercard
the card’s credit limit	the average rate of fraud observed in the training data for the transaction’s Merchant Classification Code	whether the card brand was Visa
the per capita income of the user’s home zipcode	whether the transaction had a bad zip code error	whether the card was a credit card
the user’s annual income	whether the transaction had a technical glitch	whether the card was a debit card
the user’s total debt	whether the transaction had a bad CVV entered	whether the card was a prepaid debit card
the user’s fico score	whether the transaction had an incorrect credit card number entered	whether the user was female
the number of different credit cards held by the user	whether the transaction had an incorrect PIN entered	whether the user was male
whether the transaction occurred within the user’s home zip code	whether the transaction had insufficient balance	whether the transaction was fraudulent

Seven model families were chosen for comparison:

1. A naive “needle in the haystack” approach (i.e. randomly assigning a fraud probability to each transaction)
2. Logistic regression
3. Stochastic gradient descent
4. Random forest
5. AdaBoost
6. XGBoost

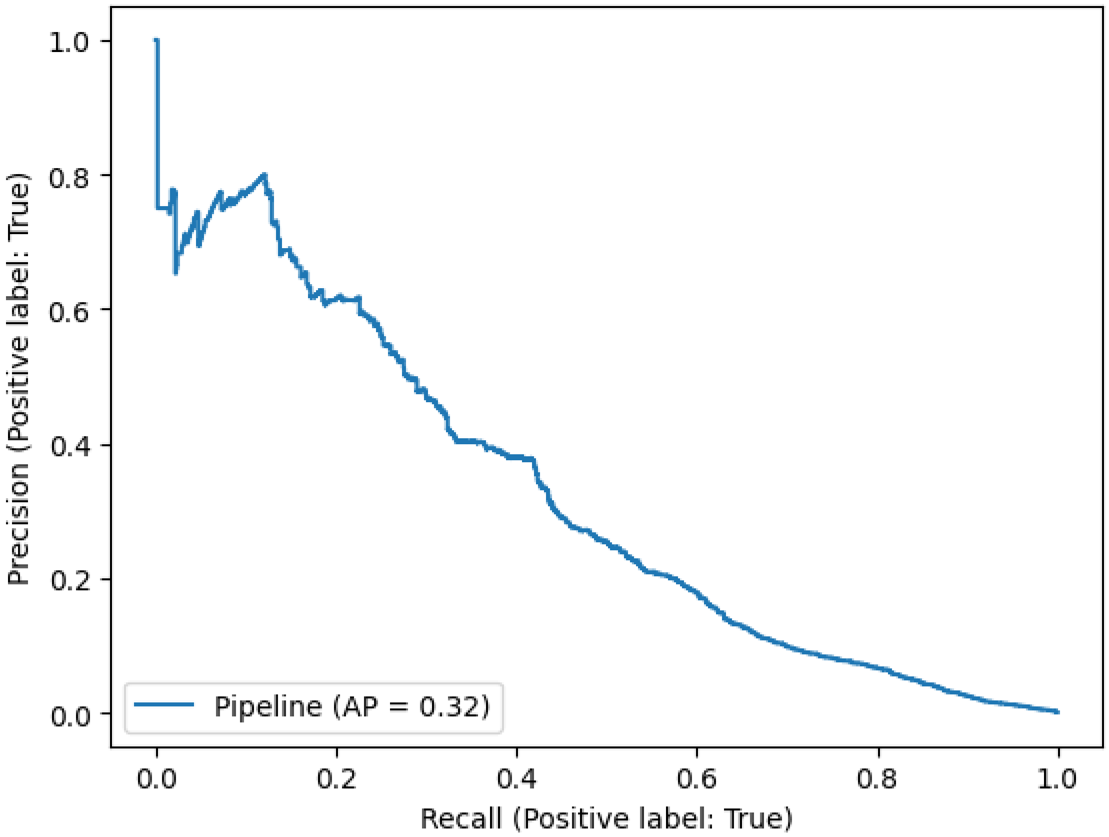
The hyperparameters of each model type were tuned using, separately grid or random search and Bayesian optimization. The performance of the models for tuning was measured using the F1 score, and the performance of the tuned models on the testing data was measured by the positive case precision with recall of at least 75%.

Findings

In general the tree-based methods out-performed the vector-based methods. The following table reports the number of false positives for each truly fraudulent transaction identified necessary to catch at least 75% of fraudulent transactions:

Model family	Tuning strategy	False positives per true
XGBoost	Bayesian	11
XGBoost	Search	13
Random Forest	Search	19
Random Forest	Bayesian	25
AdaBoost	Search	26
AdaBoost	Bayesian	26
Stochastic Gradient Descent	Bayesian	43
Logistic Regression	Search	45
Logistic Regression	Bayesian	45
Stochastic Gradient Descent	Search	53
Needle in the haystack	N/A	808

XGBoost represents by far the best model. To get a picture of the overall tradeoff between precisely identifying cases of fraud and capturing as many cases as possible, the chart below shows the precision-recall curve. Each point represents a different confidence threshold for identifying a transaction a fraudulent:



This shows that as an alternative to the strategy proposed here, the model could also be used to catch something over a tenth of fraudulent transactions with a moderately high degree of confidence (~80%).

The XGBoost model also captures the contribution of different features to its predictions. According to this, the eight most informative pieces of information concerns either:

1. The location of the transaction: whether it occurs in the user's home area or whether it is overseas, and the rate of fraud observed in the training data for the MCC
2. The mode of transaction: whether it was online, chip, or swipe

Conclusion

The model generated by this project can be used as a real-time warning tool by card issuing institutions, flagging potentially fraudulent transactions for further confirmation by cardholders. This admittedly represents a tradeoff: on one side, catching fraud and on the other, the inconvenience to customers of having to confirm and perhaps repeat a transaction. While customers are generally not financially liable for fraudulent transactions, identifying and disputing such transactions also represents a nuisance for them. So, cutting down fraud potentially by 75% at the cost of around 11 false positives seems like a fair trade. To put this another way, this averages out to just over 1% of transactions being flagged as fraudulent.

This project suggests two further avenues of investigation. First is to consider modeling methods that were not employed here due to lack of resources, namely deep learning. To be more precise, this is a problem that could be fit for a recurrent neural network. The approaches used here do not exploit one crucial piece of information that would be available at the time of a given transaction: the previous transactions of *that particular* card and user. That is to say, the data has a sequential character that approaches like XGBoost cannot capture. Something like, perhaps, a Long Term Short Term Memory model could capture not just whether a transaction is suspicious based on its immediate characteristics but whether it is unusual *in light of* the user's transaction pattern.

A second avenue would be to adapt the modeling approach developed here for use by not just card issuers but also card-accepting merchants. As noted above, a challenge here is that data on financial transactions are highly privacy sensitive. The synthetic data used in this project represents a view of information held only by card issuers. Merchants will often have a more limited scope of vision, lacking for example information on customer's full credit history and income. However, one upshot of the analysis here is that the most impactful features are things that merchants already know (the type of transaction, the amount) or could be shared by the card issuers or networks without major privacy concerns (whether the customer is in their home area, the risk associated with the merchant's MCC). Thus, in addition to card issuer's own anti-fraud efforts, merchants (especially online merchants) could deploy their own complementary measures.