Credit Card Fraud as Outlier Detection

Peter Meleney

April 2020

1 Introduction

In 2018 US credit card companies lost $9.36B to fraud, the highest reported loss for any country in the world.[2] Identifying credit card fraud is essential for protecting customers and companies. Companies with lower rates of fraud can protect earnings and pass savings on to customers, potentially leveraging a market advantage.

In 2018 Machine Learning Group - ULB provided a data set on Kaggle which contained transactions made by credit card over two days in September 2013 by European cardholders.[1] The objective of the exercise is to identify as many of the 492 frauds as possible, while excluding the 284,315 legitimate transactions.

2 Approach

I will approach this problem as outlier identification. The implicit hypothesis is that fraudulent purchases are dissimilar to ”regular” non-fraudulent purchasing habits, and therefore constitute a disproportionate number of the outliers. The advantage of this approach is that we will not need to exclude data or ”balance” the highly unbalanced data set, we will identify and flag outliers directly which, by their nature, constitute a small percentage of examples.

3 Data

The data set contains transactions made by credit card in September 2013 by European cardholders over a two day period. The majority of input features are the result of PCA transformation. This is to protect the identity of customers.

The features numbered V1, V2, V3,...,V28, are the principal components obtained by PCA, the other variables, Time and Amount, have not been altered. Time is the time in seconds since the first transaction in the data set, and the Amount is the value of the transaction (I assume in USD equivalent). The Class column is the output vector, 1 if fraud, 0 otherwise.

First I use a Yeo-Johnson power transformation to normalize the PCA components and Amount variable. Next I normalize the transformed PCA components using a MinMaxScalar to set the range to between [-1, 1]. Finally, I

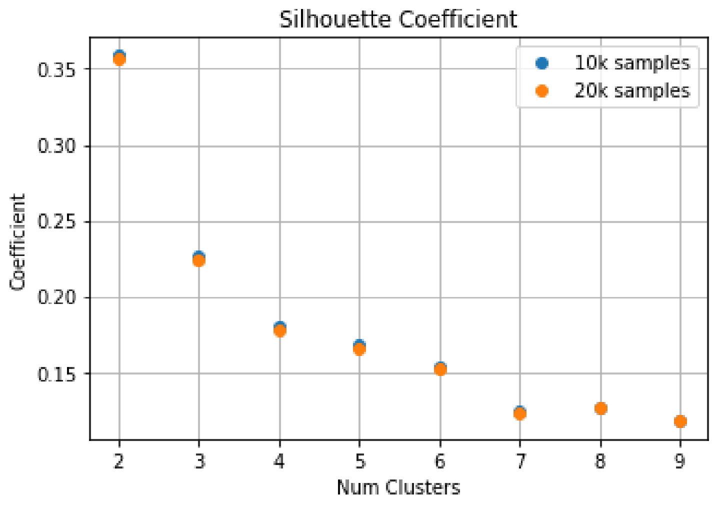

Figure 1: Silhouette method on a 10k and 20k subset of the data. k=2 is the consensus number of clusters.

transform the time variable by subtracting 1 day's worth of seconds from timestamps in the second day, and then executing the function:

$$time_{transform} = sin\left(\frac{2\pi}{24 * 3600} * time\right) \tag{1}$$

Unfortunately, the nature of the PCA vectorization does not allow for intelligent feature creation, so this step will be skipped.

4 Model

I assign a number of clusters to the data according to the silhouette method. These represent different kinds of "normal" purchasing behaviors that customers participate in.

Then I assume that each of these clusters is well-represented by a mixture of Gaussian distributions with n_components = 1, 2, 3, or 4. Finally, I find those data which fall furthest outside the range of any of the clusters, i.e. they are outliers to both clusters. This approach gives us a simple tuning parameter, the threshold beyond which a datum will be considered an outlier. By changing this parameter we can create an ROC curve and calculate the AUC of the model. The best performing model has n_components = 3, and an validation AUC of 0.943.

Figure 2 is a set of 4 ROC curves, each for a different number of n_components assigned in the Gaussian mixture model. The two best performing models have n_components = 2 or 3 depending on where in the curve we decide to place out

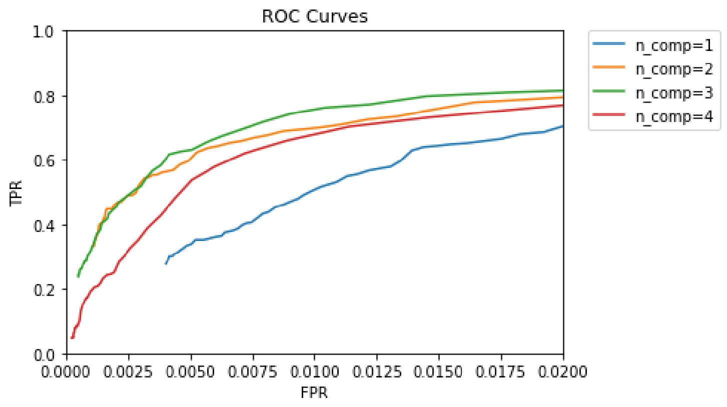

Figure 2: Validation ROC curves. Note the very zoomed-in axes, this is to give the viewer a better understanding of the performance of the models near their point of highest curvature, where a decision point will probably be set.

decision boundary. For this exercise I have chosen to use the n_components = 3 decision boundary because it is the best performing ROC curve near the area of greatest curvature.

5 Implementation

The average cost of a fraud is approximately \$122.21, so I will use that as the cost metric for this analysis. There is also a cost associated with alerting a customer, though it is much harder to quantify. The cost is the added friction to the buying process if the card is initially rejected and an alert sent to a customer's phone. We will therefore perform a very rough sensitivity analysis on the cost of a false positive, varying the cost between \$2.00 and \$10.00. However, if this happens only very occasionally I could imagine a positive effect since the card holder would then know that they were being actively protected from fraudulent transactions.

We can see from Figure 3 that, assuming the two days sampled here are representative of the average annual card activity in terms of fraudulent and non-fraudulent purchasing, the credit card company in question can expect to save between \$1.9M and \$6.4M depending on the actual cost of a false positive, assuming an optimal selection of threshold value.

6 Additional Work

This approach worked out very well. The assumption that fraudulent transactions are not like non-fraudulent transactions seems to hold in most cases.

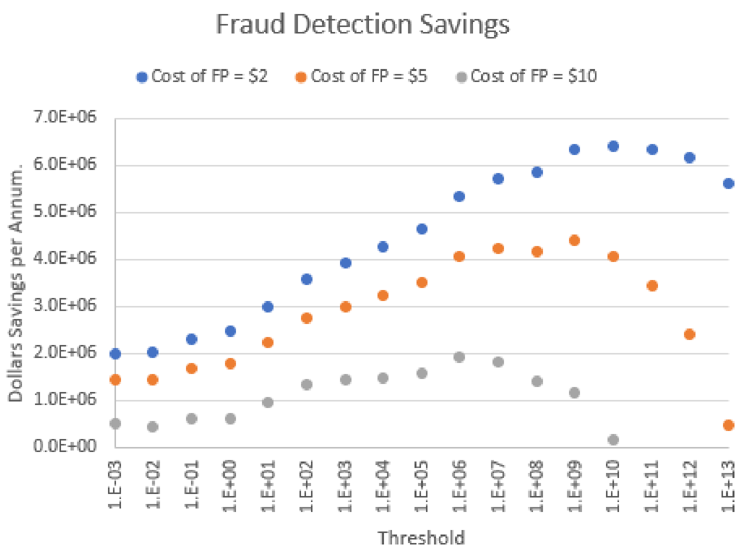

Figure 3: Projected savings to company assuming different costs associated with false positive errors.

However I believe that this model could be improved further by the implementation of a ensemble. The idea would be that the data are first run through this ”outlier detection” model and transactions with model output values beyond the threshold are flagged as potentially fraudulent. These represent the outlier fraud examples, and would make up the bulk of the flagged cases. I would then pass the un-flagged data to a second ”inlier detection” model, probably a highly flexible, non-parametric model like a neural network, which would attempt to identify those transactions which are not outliers, but are still fraudulent. Only if an input transaction passed both models would it be allowed to complete without alerting the customer.

References

[1] Machine Learning Group-ULB. Credit card fraud detection. https://www.kaggle.com/mlg-ulb/creditcardfraud, 2018. Accessed: 04/13/2020.

[2] Shift Processing. Credit card fraud statistics. https://shiftprocessing.com/credit-card-fraud-statistics/, 2020. Accessed: 04/14/2020.