Credit Card Fraud Prevention Using Machine Learning

Rundall, Philip

April 23, 2020

Abstract

Credit card fraud is a multi-billion dollar industry. With the rise of technology in financial transactions finding solutions is only growing in importance. Using Gaussian Naive Bayes Classification algorithm we attempt to illustrate a solution generated by machine learning on a dataset from ULB group. Our method consists of selecting relevant features, SMOTE resampling of training data and cross validating the model on unseen data. Our resultant is an array of robust metrics that can be selected to fit the users preference for limiting false positives or false negatives. Overall the confidence interval for the AUM of our model is [0.94,1.00].

1 Introduction

Credit card fraud is a convoluted game where over accusation can lead to customers feeling mistreated by Credit lending companies, but moreover a complete disregard for audit of credit transactions will lead to huge losses for banks. The Nilson Report estimated in 2016 alone, $22.8 billion in losses were attributed to credit card fraud. Without proper scrutiny of transactions in place thieves may be more enticed to commit fraud if unnoticed. The balance of eliminating Moral Hazard and customer satisfaction is the problem at hand.

Fortunately enough we have access to a quarter million credit transactions that are marked with 29 characteristics and whether they have been identified as fraud or not. This set up lends itself to the Gaussian Naive Bayes algorithm to identify state-spaces of factors that are highly likely to be associated with a fraudulent transaction. These correlations can be used in the future to identify transactions that are not previously identified as fraudulent or genuine. We split the data conventionally to simulate seen and unseen data as a proxy for real world evaluation.

2 Data

2.1 Section

The dataset for this analysis contains information for 284,807 credit card transactions, including 492, or 0.172%, fraudulent occurrences and was sourced from Kaggle. Due to confidentiality reasons, the dataset

Figure 1:

`/Users/mosh/Downloads/Screen Shot 2020-04-22 at 9.57.32 AM.png`

contains modified input features V1, V2, . . . , V28, which have undergone principal component analysis (PCA) transformation. The PCA method can be described as a dimensionality reduction technique. The other input variables are "Time" and "Amount" which represent the transaction time and transaction amounts, respectively. Furthermore, we assume that the dataset feature values have been previously scaled. The dataset also contains values for "Class" where fraud is indicated by a value of 1 and is 0 otherwise.

2.2 Data Splitting

The data is split into training and testing datasets: two-thirds of the data are assigned to the training set and one-third of the data are assigned to the testing set. The training set is used to develop the model while the testing set is used to evaluate the model performance.

2.3 Feature Selection

Since this model includes 30 input features, including V1,...,V28, which have undergone PCA transformation, we must further examine factor relevance. Since the V1-V28 variables have been anonymized, we cannot determine any economic significance in any of them. To determine which of these features we will keep, we will look at the distributions of Genuine vs Fraudulent for each feature. If the distributions overlap too much, we will not include the feature in our model because the Naive Bayes Classifier will not be able to extract any useful information from it. This will help reduce model complexity and overfitting of our model. The following input features will be included in our model: Time, Amount, V1, V2, V3, V4, V9, V10, V11, V12, V14, V16, V17, V18, V19, V26, V28.

Figure 2:

`/Users/mosh/Desktop/Screen Shot 2020-04-23 at 12.26.02 PM.png`

The following correlation plot shows that there is no strong correlation between the remaining V input factors. Also there is at least some correlation between most input variables and the fraud class.

Figure 3:

/Users/mosh/Desktop/Screen Shot 2020-04-23 at 12.27.44 PM.png

2.4 Resampling

Due to our dataset being imbalanced and having significantly more genuine than fraudulent charges, we must resample the data to make it more balanced. There are a few options when resampling our data, we could either oversample from our fraudulent charges, undersample from our genuine charges, or perform Synthetic Minority Oversampling Technique (SMOTE). We used SMOTE to resample our data because we have so few observations classified as fraudulent charges. If we chose to under sample from our genuine charges, our testing dataset would be extremely small, and if we chose to oversample from our fraudulent charges, we would not be adding any new information to our model.

SMOTE works by first taking a minority class observation, a fraudulent charge, and first selects the K nearest neighbors within the feature space. K is usually set to 5. Then one of the nearest neighbors is randomly selected. Finally, a point between the observation and randomly selected neighbors within the feature space is selected. This point becomes a new observation for the minority class. By synthesizing this new observation, we create a “plausible” fraudulent charge observation.

Figure 4:

/Users/mosh/Desktop/Screen Shot 2020-04-23 at 12.29.53 PM.png

In the Figure 4, the green observation would represent our fraudulent charge. The 5 blue observations would represent the 5 nearest neighbors to the green point, and the red observation would represent our new synthesized observation.

While this example does this algorithm in 2-Dimensions our state-space consists of 17. This method generated an additional 190,134 “plausible” fraudulent observations to make the ratio of fraudulent to Genuine in the training data 1:1. By synthesizing these data points instead of resampling from our fraudulent class, we are able to add new information to our model which will improve its ability to correctly classify fraudulent charges.

3 Methods

This study employs the Naive Bayes method.

3.1 Naive Bayes

The Naive Bayes method assumes independence of predictor variables. The formula is given as,

$$P(C_1|x_1,...,x_p) = \frac{P(C_1)[P(x_1|C_1)...P(x_p|C_p)]}{P(C_1)[P(x_1|C_1)] + ... + P(C_m)[P(x_1|C_m)...P(x_p|C_m)]}$$

where x is the set of predictor values $x_1, ..., x_p$ and C is the set of classes.

The resultant probability in our case, is the probability of a fraudulent transaction given the set of characteristics about the transaction. A threshold for the probability is selected for when to mark the claim as fraudulent or deny. In subsection 3.3 we discuss the optimal selection of this threshold with respect to sensitivity to false positives and false negatives.

3.2 Guassian Naive Bayes

The Gaussian Naive Bayes method is an extension of the previous method, where the likelihood of the features is assumed to be:

$$P(x_i|C) = \frac{1}{\sqrt{2\pi_C^2}} exp(-\frac{(x_i - \mu_C)^2}{2\sigma_C^2})$$

It involves prior and posterior probability.

3.3 ROC and AUC

As mentioned in 3.1 Receiver Operating Characteristic (ROC) is a measure of thresholds effect on false positives. For each threshold it parametrically calculates the vector (True positive rate, False Positive rate). This is an incredibly valuable metric to this problem as selecting a threshold that reduces accusations of fraud can improve client satisfaction and retention.

When the ROC vectors are plotted they form a concave line with respect to the False Positive rate axis. The Area Under the Curve (AUC) is an evaluation metric of the models success in separating the distributions of TPR and FPR. An AUC = 1 implies the model has perfectly separated True Positives from False positives with no ambiguity to the results. As AUC approaches .5 the model deteriorates into complete ignorance of the true classification.

3.4 Stratified k-Fold Cross Validation

Stratified k-Fold cross validation is used to test the robustness of a model. The process is to split the data into "k" equal parts with similar fraud to genuine ratios and recursively use k-1 sections of training data to predict the non-selected section. Cross fold acts as a generation of "unseen" data in an attempt to simulate new data. A consistent evaluation metric across many folds insinuates the metric can be expected on future data and over-fitting is not present.

Figure 5:

/Users/mosh/Desktop/Screen Shot 2020-04-23 at 12.54.21 PM.png

The large size of the dataset allows us to perform cross validation up to k=15 and report a confidence interval for FOC and AUC.

4 Results

Most classification problems use Accuracy, Recall, or Precision to rate the performance of a model. However, because our dataset is so imbalanced with relatively few fraudulent transactions, these will not give a true indication of how our model is performing. The numerator of these metrics include True Negatives, which is over 99% of the data therefore these values will be very high and overly inflate the performance of our model. This is why we will use ROC and AUC to judge our model's performance.

4.1 Confusion Matrix

We evaluate an initial Naive Bayes model. The confusion matrix as a whole still provides some valuable intuition into our models performance, despite the imbalance of the dataset. The confusion matrix for the testing data is as follows:

This initial model captures 79.2% of fraud. It is clear that the common metrics evaluating this confusion matrix does not accurately describe our results. The model raises the red flag on charges far too often yet our accuracy is still above 99%. Most people would stop here but, we move on to the realm of AUM to improve on this substantially by iterating through different thresholds and subsets of the data.

4.2 Robustness

In figure 6 we see the mean of our stratified 15-fold cross validation ROC in thick blue. It is clear that over a 80% True Positive rate can be reached and maintained with very minimal False Positive Rate (~1%).

On robustness, The average AUC from each 15 folds is so concentrated around the total mean our confidence interval, seen in gray, and it ranges from [94%,100%]. Cross-validation provides a level of robustness that could only be generated by a dataset orders of magnitude larger. Our evaluation metric AUC = 0.97 is even more imposing when compared to the chance AUC, represented by the dotted red line. If Naive Bayes found no relevance in our variables our AUC would line with this "chance" line. We can conclude that the factors chosen are highly useful in the identification of fraud.

Figure 6:

/Users/mosh/Desktop/Screen Shot 2020-04-23 at 12.58.39 PM.png

Conclusion

Using the Guassian Bayes method, we have successfully proved the value of machine learning in mitigating the harmful effects of credit card fraud judgement. The two sides of the coin to be balanced here are customer satisfaction and limiting loss. When a lender puts a value to these two objectives, a threshold can be easily assigned to maximize the lender's utility. Despite the robustness of our conclusions and the ability to resample data, we must recall that input factors for this datasat have undergone PCA transformations for confidentiality and other reasons. Therefore, it may be difficult to replicate our model and results based on different datasets.

Future Works and Improvements

The main way we could improve the results of this project is to increase our dataset, specifically to include more fraudulent observations. This would allow us to not need to resample and balance our dataset. Another way we could improve our performance is by using Adaptive Boosting. Adaptive Boosting is a boosting method used to improve results if a dataset is imbalanced. It works by placing an emphasis on correctly classifying previously incorrectly classified examples.

Having evaluated a robust metric for AUC this model can be easily compared to other machine learning techniques. Some that are usually applied to classification problems include Support Vector Machines, Logistic Regressions and Random Forests. The same process of k-fold cross validating an AUC can be applied to the listed algorithms in an attempt to beat our model; However our results will be tough to beat.

References

[1] https://www.kaggle.com/mlg-ulb/creditcardfraud

[2] https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc_crossval.html