Detecting Credit Card Fraud

Vinicius Caridá

ABSTRACT

Finance and banking services are very important in our day to day life, when almost everybody has to deal with financial transactions whether it is done physically or online. The productivity and profitability of both public and private sector has tremendously increased because of banking information system. Nowadays most of E-commerce application system transactions are done through credit card and online net banking. Credit card fraud is a term used for a type of fraud committed by a payment card like credit or debit card. Detecting an online credit fraud and preventing it from happening is one of the most important responsibilities of every company involved in financial sector. The main goal of the current work is to analyse the credit card fraud information, to pre-process them and finally to detect credit card fraud.

Data Set

The dataset used in this work is a real and anonymization data from a Brazilian bank

Introduction to the problem

In Machine Learning, problems like fraud detection are usually framed as classification problems which is predicting a discrete class label output given a data observation. The size of the dataset is relatively big with 150000 observations, it has 30 features, 27 of which have been anonymized and are labeled PP1 through PP28. The remaining three variables are the Sacado, Occorencia and a class label showing whether observation is fraudulent or not. The process of doing the current task was devided in several steps: a) explanatory data analysis where the main characteristics of the dataset is analysed and based on the results the subsequent tasks are defined; b) pre-processing data in order to deliver a clean dataset to the classifier; c) spot-checking which is the task of using several algorithms with different prospective (i.e. linear or non-linear, probability-based or distance-based, ...) in order to gain a better understanding about the nature of data and to measure and chose the one with the best classification result; d) the last step is to improve the results in terms of accuracy or reducing the size of features while getting almost as good result as before.

1 Explanatory data analysis

1.1 Missing values

First, the existence of missing values was examined. Luckily, there are no missing values and we, therefore, do not need to worry about missing value imputation.

1.2 Distribution of features and data transformation

Afterward, the distribution of features, considering the value of each that resulted in a fraud or not a fraud, was analyzed. As illustrated in Figure 1 although for the majority of features the distributions of the values for "fraud" and "not a fraud" are overlapped, for "PP4", "PP14", "PP10", "PP17", "Sacado" and "Ocorrencia" the distributions are fairly separable. Although this does not guarantee the importance of the feature, yet it can be a good indication that these features help the model in decision making. Between these features, Sacado has the most separable distributions. More analyses showed that the mean value of this feature for "fraud" (= - 135.3) is significantly lesser than the "not a fraud" class (= -88.5), which depending on the meaning of this feature, this can bring new insights. The skewness of the distribution the features was also analyzed. Despite the fact that the majority of features has not a high degree of skewness, it is high for "PP8", "PP23" and "Sacado".

In order to make the data distribution more Gaussian-like and soften the high degree of kurtosis (> 3), which is present in the majority of features, the *Box-cox* transformation also known as *power transformation* was applied on the data (it is applied before passing the data to the classifier. The data analysis is done on the original data). Normal distribution is favorable because it is simple to analyse and to understand. Also, it has important properties which makes working with it easier (i.e. sum of two normal distribution is also normal).

1.3 Correlation of features

It would be interesting to know if there are any significant correlations between our predictors, especially with regards to our class variable. It is important to mention that the correlation between features does not affect the classification performance per se. But means that the existence of one feature does not bring new information when a correlated feature with it is already on the dataset. Correlation of features however can affect the accuracy of the classifier negatively when there is a curse of dimensionality. One of the most visually appealing ways to determine the correlation is by using a heatmap.

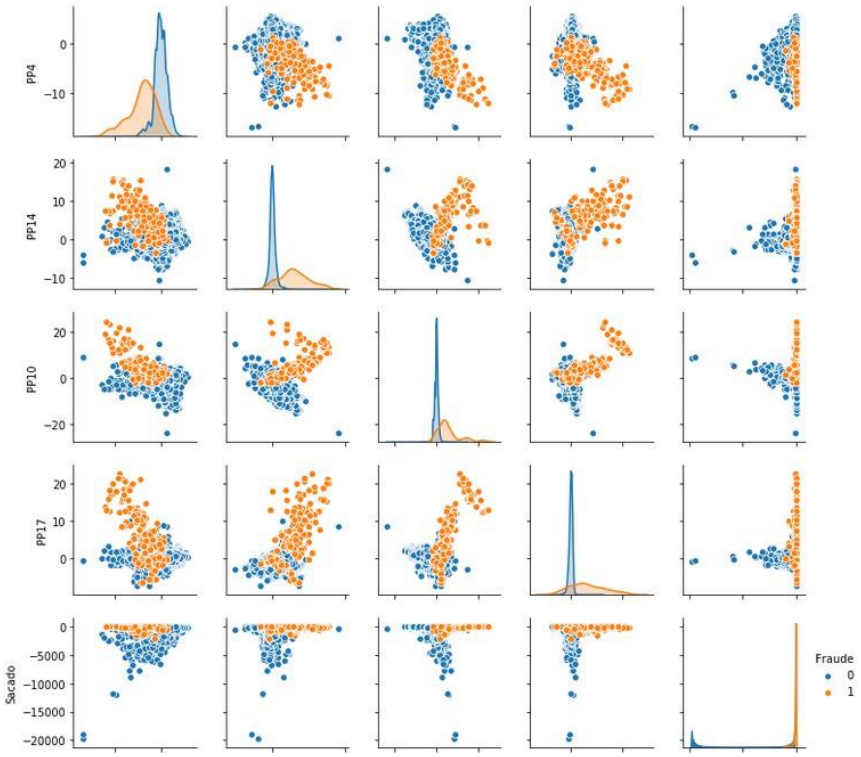

Figure 1. The distribution of features for fraud and not a fraud classes. These features show a higher degree of separability between the distributions of each class in comparison with the other features.

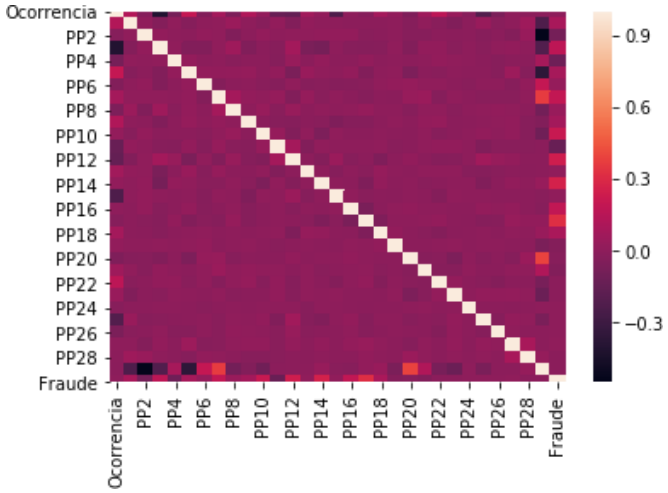

Figure 2. The correlation between the variables. As seen, the there are few variables with negative correlation. Nevertheless, this correlation is not significant.

As illustrated in Figure 2, some of our predictors seem to be correlated with others negatively. Nonetheless, seeing the heatmap there seem to be relatively little significant correlations. This can probably be associated with two factors: 1) The data was prepared using an algorithm like PCA where the eigenvectors are not correlated, and therefore the predictors are principal components; 2) The huge class imbalance might distort the importance of certain correlations with regards to our class variable. In order to understand this, the proportion of data related to "fraud" and "not a fraud" was analyzed.

1.4 Imbalanced classes and re-sampling

The proportion of observation associated with "fraud" and "not a fraud" was analyzed. As shown in Figure 3, the majority of observations (0.99%) are non-fraudulent.

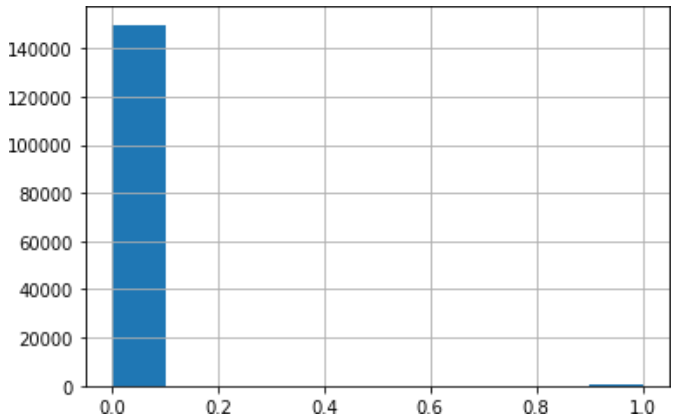

Figure 3. The histogram of data associated with Fraud and Not a Fraud. As illustrated, the classes are highly imbalanced.

When using accuracy, we attribute the same cost to false positives and false negatives. When the dataset is imbalanced (say it has 99% instances in one class and only 1% in the other) the problem begins when the actual costs we assign to each error are not the same. If we deal with credit default which is relatively a rare event, the cost of not predicting the risk of a fraud is extremely high when it happens.

As seen in the histogram of the class, the data related to Fraude (=1) are only 0.0015% of the entire data, showing that the dataset is extremely imbalanced. There are three ways to deal with this problem: 1) using different metrics like precision and recall in order to measure the performance of the classifier; 2) applying stratified K-fold in order to sampling the data considering the original proportion of the minority to majority class; 3) using re-sampling with replacement to create balanced classes.

Here I prefer using re-sampling in order to have a balanced data. To do so, the minority class is over-sampled using an improved version of *SMOTE* called *ADASYN*.

1.5 Anomaly detection

Most parametric statistics, like means, standard deviations, and correlations, and every algorithm based on these metrics, are highly sensitive to outliers. Despite all this, it is not acceptable to drop an observation just because it is an outlier. Being an outlier can simply mean being different. Then, outliers can be legitimate observations and are sometimes the most interesting ones. One way to deal with outliers in classification task is to remove them and then to measure the classification performance. If the performance is dropped, this means that although the outliers are different than the rest of the data, but they help the model in making good decisions. There are several ways to detect outlier. Here, I used IQR and Isolation Forest. After detecting and removing the anomaly observations, the performance of the model was evaluated. Since, the accuracy was dropped, it is an indication of the fact that these anomaly observations are important. Thereby, I decided not to drop them from the dataset.

1.6 Feature selection and model explainability

A benefit of using ensembles of decision tree methods like Gradient Boosting or Random Forest is that they can automatically provide estimates of feature importance from a trained predictive model. First, the feature importance with a threshold of 98% was calculated. It was found out that there is no zero-important features and 28 features is required for 98% cumulative importance (see Figure 4 - right).

Although classic feature importance methods are useful, they lack showing the contribution of features and their values in predicting the target class, hence not being able to explain the model. SHapley Additive exPlanations (SHAP) is a new game theoretic approach to explain the output of machine learning models. It connects optimal credit allocation with local explanations using the classic "Shapley values" from game theory and their related extensions. From here, I will explain the contribution of features using shaply values.

The importance of features was evaluated using XGBoost. As shown in Figure 4 - left, PP14, PP4, PP10 and Sacado are the most important predictors. This plot sorts features by the sum of SHAP value magnitudes over all samples, and uses SHAP values to show the distribution of the impacts of each feature on the model output. The red color and blue colors represent the

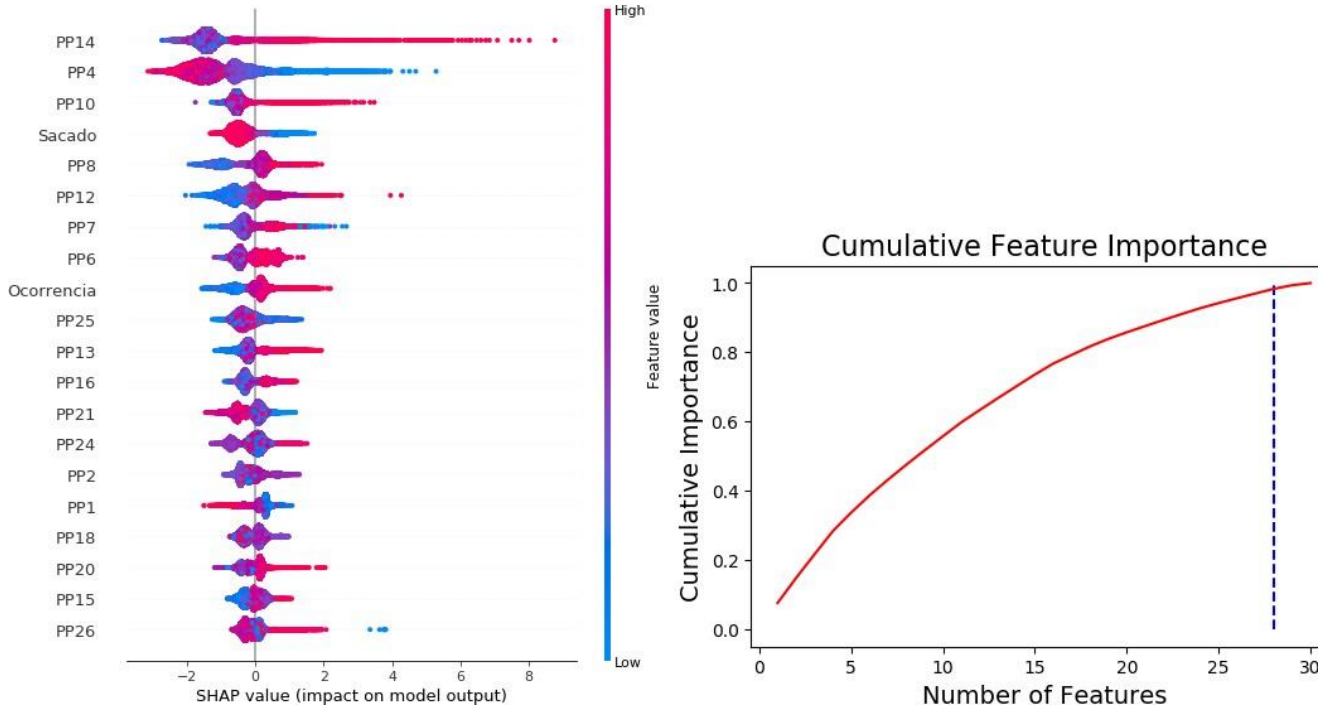

Figure 4. Left) the importance of features and the contribution of their values in fraud prediction; Right) As shown, 28 features is required for 98% cumulative importance.

high and low values of each feature, respectively. Interestingly, considering the separability of the distribution of these features (see Figure 1), it was predictable that these variables have the potential to be among the most important features. Figure 4 - left also shows the contribution of the values of each predictor in predicting the expected value of class given each observation. We can see that for PP14 the higher values push the model toward predicting the expected class (also called as *base value*), while for PP4 the lower values the model's output toward 1 (Fraud) and higher values. This is in accordance with the distribution of feature values showed in Figure 1. On the other hand, for features like PP18 and PP24 both high and low values influence the model toward a better prediction.

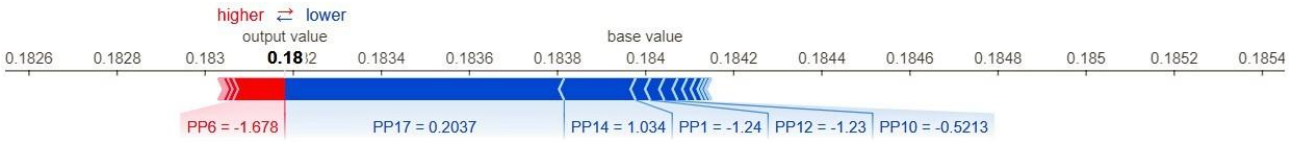

Figure 5

We can also analyse each observation based on the model's output. In Figure 6, the contribution of the most important features for a single observation is shown. Here the *base value* is the average model output over the training dataset and the *output value* is the predicted value for this observation. All the features which push the class label higher is shown in red and those that try pushing the predicted class label lower than the base value is shown by blue. We can see that for the first observation the PP14 , as an important feature, tries to lower the predicted class label (which considering the fact that the real class for this observation is 0, is a right thing to do). On the other hand, PP6 is pushing the result toward a wrong answer. Here, knowing the meaning of each feature is crucial for interpreting this behavior. As an example, the number of times an individual committed fraud in the past can act like either PP6 or PP14 features. If the user has never committed a fraud, this can push the model toward predicting a value near zero.

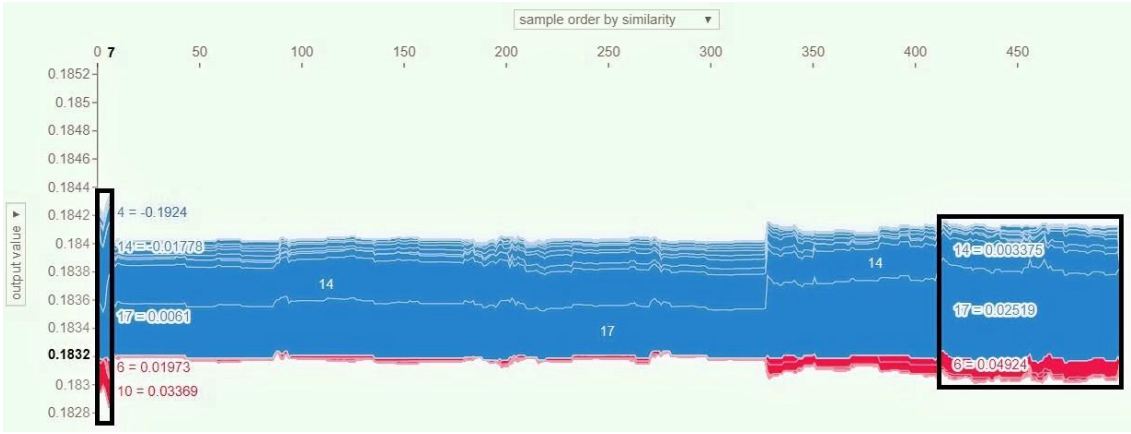

Figure 6

In order to find the observations with similar characteristics, the collective behavior of observations was analyzed. In Figure 6, each data in X axis is a single observation ordered based on the similarity of features and the Y axis represents the predicted values. Two example of clusters of similar observations (the black clusters) are illustrated in this figure. As you can see, for all the observations in the first group, the features PP17, PP4, PP14, PP6 and PP10 influence the model prediction. On the other hand, features PP17, PP14 and PP6 have the highest impact on the observations of the second group. Lets clarify this by an example. The first group may represent the activities/ transactions of students with the similar value for the some features (i.e similar amount of investments, salary and amount of the transactions) and the second group could be the observations related to individuals with high salary.

The results of this analysis prove that the current dataset has a stable behavior where hundreds of observations can be clustered in a single group (see the second cluster). For a dataset with random behavior, the above figure would have shown hundreds or thousands of mountains and valleys (high and downs). This is a great indication that by having the real name /meaning of the current features (because the current dataset is anonymized), it is possible to discover some latent relationship between the observations and to have a better understanding of the underlying conditions when suspicious observations happen (by knowing which features are important in which clusters and why).

1.7 Dimensionality reduction

Principal Component Analysis (PCA) was used to reduce the dimensionality of data. After finding the principal components, the elbow method was used in order to find the best components which maximize variance. As you can see in Figure 7, 21 components are responsible for 99.1% of the variance.

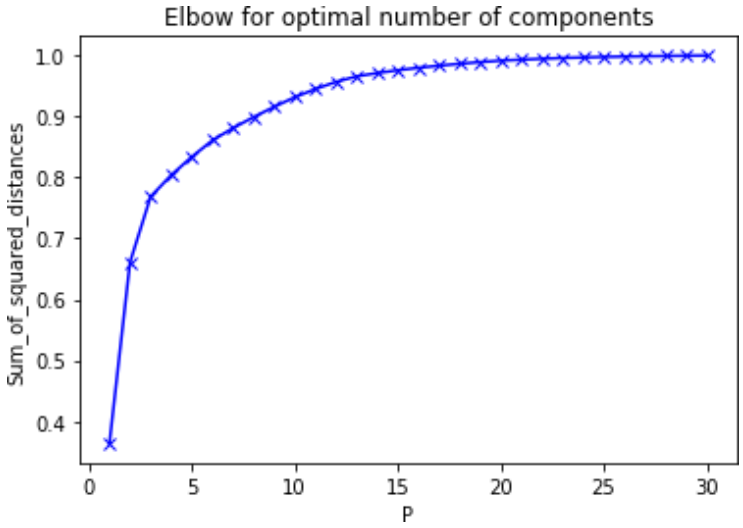

Figure 7. The elbow curve for deciding the optimal number of principal components

1.8 Classification

Five algorithms including Logistic Regression (with L2 norm as it penalty and C = 1), K-Nearest Neighbors (with K equals to 5), Gaussian Naive Bayes, Gradient Boosting Classifier (with learning rate = 0.7 , number of estimators = 200 and max depth = 8) and Decision Tree (with Gini criterion) were chosen in classification task. The algorithms were chosen among distance-based, Bayesian, ensemble, tree-based and generalized linear methods. In order to be able to test the performance of the models, a 80/20 train-test split was performed, splitting the balanced data set into two test and validation sets. The tests were carried out for dataset before and after applying PCA and choosing 21 principal components. The results show that by reducing the dimensionality from 30 to 21, the classification performance was decreased by only 1% while a significant number of variables were reduced. Since the classes are binary and balanced (after applying re-sampling), area under the curve (AUC) was used to measure the performance of the algorithms. As seen in Figure 8, Logistic Regression and Naive Bayes with 93% AUC have the best performance among the others. Since Naive Bayes manages to classify the data with high performance, we may assume that the probability independence is valid for the predictors.

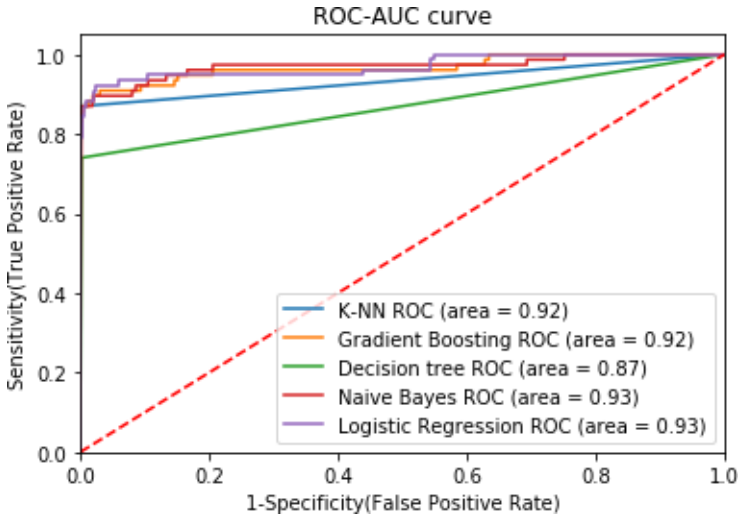

Figure 8. The area under the curve for K-NN, Gradient Boosting, Decision Trees, Naive Bayes and Logistic Regression.