Credit Card Fraud Detection

Shailesh Alluri
School of Computing
Clemson University
ralluri@clemson.edu

Bhavya Pulagam
School of Computing
Clemson University
vpulaga@clemson.edu

Abstract

Credit card fraud is a form of identity theft that involves an unauthorized access and usage of another person's credit card information to perform transactions without their knowledge. This project aims at detecting and flagging such credit card fraud using SVM(Support Vector Machines). Despite the emerging sophisticated security systems, the complexity with which fraudulent transactions occur through identity theft has been increasing. Prioritizing their user security, in order to mitigate risks, the banks that are dealing with these above mentioned issues would rather choose to flag a non-fraudulent transaction as a fraudulent one than the other way around.

1 Introduction

The nature of different datasets that we have discovered for fraudulent transactions is very imbalanced i.e., the ratio of the fraudulent to non-fraudulent transactions is extremely high. Because this dataset is imbalanced, a supervised classifier may behave inappropriately if the data are used as is. To put this into context, if a classifier were to train on this collection of data while attempting to reach the highest level of accuracy, it would probably classify every transaction as legitimate. To resolve this issue, we plan to utilize either the oversampling concept or the under- sampling principle.

Under-sampling techniques eliminate or combine cases in the majority class, whereas over-sampling techniques add additional synthetic examples to the minority class. Although it can be more effective when both sorts of approaches are used together, both types of re-sampling can be beneficial when employed separately. The under-sampling approach should only be used when it is certain that the few tuples selected (in this case, non-fraud) are representative of all non-fraud transactions in the data set.

Identity fraud has been on the rise for the past five years and significantly rose in 2021, according to a report from the Federal Trade Commission published in 2022. In 2021, there were 22% more identity thefts filed to the Federal Trade Commission than there were in 2020. The Federal Trade Commission only received approximately 1.7 million reports of identity theft in the United States in 2021. The most frequent complaint filed by consumers was identity theft, which made up 29.4% of all reports received by the FTC[8]. A bank's goal is to find as many frauds as possible! even if it necessitates classifying more non-fraudulent tuples as fraudulent activity. Therefore, we must reduce the frequency of false positives, or frauds that go undetected. In order to achieve this, even when the likelihood of a fraudulent transaction is low, we will classify samples that are near to the SVMs (support vector machines) soft margin.

2 Methodology

Principal Component Analysis (PCA) is a tried-and-true dimensionality reduction technique. It enables as much information as possible to be retained while describing the data X using a limited set (S) of uncorrelated variables (the main components). PCA is frequently used to summarize and visualize data as an exploratory technique. PCA is also sometimes thought of as a method for separating the signal from the noise, with the first S principal components being considered the signal and the rest the noise[7]. In order to analyze photographs, for example, or to pre-process data before using other techniques like clustering, PCA can be used as a denoising approach. In fact, it is anticipated that clustering will be more stable when used on noise-free data sets.

For PCA, we should maximise the variance of the projected data along the new components. Now let us suppose one such component is w then the projected data along w will be Xw where X is data of the order n x p and w is of the order p x 1. Now since X is centred covariance of X is given by (1/n)X^TX and assumed to be v. So the variation of the projected data along the new components can be given as

$$\sigma^2_{\vec{w}} = (1/n)(||Xw||^2_2) \tag{1}$$

$$= w^T vw \tag{2}$$

where X is the input matrix(data) and w is a vector along which the data is being projected.We need to maximise the above expression subject to the condition that wTw = 1.We can use a Lagrangian equation to solve this. We need to maximise the equation 2 subject to the condition that $w^T w = 1$

$$L(w, \lambda) = \sigma^2_{\vec{w}} - \lambda(w^T w - 1)$$

$$\frac{\partial L(w,\lambda)}{\partial \lambda} = w^T w - 1 \tag{3}$$

$$\frac{\partial L(w,\lambda)}{\partial w} = 2vw - 2\lambda w \tag{4}$$

Equating equations 3 and 4 to 0 we get the following.

$$w^T w = 1$$

and

$$vw = \lambda w$$

The solution to the above equations are unit eigen vectors with eigen values λ. Since X is a n x p matrix $X^T X$ has p eigen vectors. Let us suppose w_1 is one of the eigen vectors λ_1 of $X^T X$, then variance of the projected data along w_1 is as follows

$$\mathbf{X}_r = \mathbf{X}\phi_k \tag{5}$$

So to maximise variance along eigen vectors we have to choose eigen vector with maximum eigen value of first k eigen vectors with maxiumum eigen values.

$$\phi_k = U[:, 1 : k] \tag{6}$$

SVM(Support Vector machines) Support Vector Machines (SVMs) are supervised learning models that use spatial mapping to associate training instances with points in order to maximize the distance between the two categories. An optimal line or hyperplane for class separation is sought by the SVM training algorithm. The decision boundary and the nearest instances from each of the two classes are separated by a margin that optimizes the distance. By implicitly mapping their inputs into high-dimensional feature spaces, a technique known as the kernel trick, SVMs may effectively conduct non-linear classification in addition to doing linear classification.Unsupervised learning is necessary when the data are unlabeled since supervised learning is not available. This method looks for natural groupings of the data by clustering and then maps new data to these created groups. To categorize unlabeled data, the support vector clustering algorithm uses the support vector statistics created by the support vector machines algorithm. Support Vector Machine employs Hard Margin SVM by default. Only if our data can be linearly separated does it perform properly. Hard margin SVM prevents any misclassification from occurring.

Typically, the classes are not separable, even with data transforms. As such, the margin is softened to allow some points to appear on the wrong side of the decision boundary. This softening of the margin is controlled by a regularization hyperparameter referred to as the soft-margin parameter, lambda, or capital-C ("C").

The Hard Margin SVM won't create a hyperplane if our data cannot be separated or is nonlinear because it can't separate the data. Soft Margin SVM steps in to save the day as a result. The tight

SVM limitations are loosened by soft margin SVM, which permits some misclassification. To utilize Soft Margin SVM, use the Regularization parameter (C). Using the regularization parameter, we may decide how much misclassification we want to prevent (C). Hard margin SVM frequently has large values of C. In soft margin SVM, small values of C are often encountered. By agreeing on a trade-off between these two objectives for the non-separable scenario, we should maximize the SVM vanilla objective, where C > 0 is the trade-off parameter between margin-maximization and the slack penalty. For = (1,...,m), the primal problem is

$$min_{w,b,\epsilon}(1/2)||w||^2 + C\sum_{i=1}^{m}\epsilon_i^p \tag{7}$$

subject to $y_i(< w, x_i > +b) \geq 1-_i$ and $\xi_i \geq 0$ for all $i \in [m]$

The parameter C is determined by n-fold cross validation for a given dataset. As in the separable case, the equation (1) is a convex optimization problem since the constraints are affine and thus convex and since the objective function is convex for any p > 1. There are many possible choices for p leading to more or less aggressive penalizations of the slack terms. The choices p = 1 and p = 2 lead to the most straightforward solutions and analyses.

To weight the C value in accordance to the importance of each class is perhaps the simplest and most frequent addition to SVM for imbalanced classification. Particularly, each sample in the training dataset contains a unique penalty term (C value) that is employed in the SVM model's margin computation. The value of an example's C-value can be calculated as a weighting of the global C-value, where the weight is defined proportional to the class distribution[5].

$$C_i = weight_i * C$$

The margin can be made softer for the minority class by using a bigger weighting, while the margin can be made harsher and misclassified examples avoided by using a smaller weighting for the majority class. Small Weight: Smaller C value, larger penalty for misclassified examples[5].

Larger Weight: Larger C value, smaller penalty for misclassified examples.

As a result, the margin is encouraged to constrain the majority class with less flexibility while allowing the minority class to be flexible and, if necessary, misclassify samples from the majority class onto the minority class side.

3 Dataset

The files include credit card transactions done by European cardholders in September 2013. We have 492 frauds out of 284,807 transactions in our dataset of transactions that took place over the course of two days. The dataset is seriously out of balance, with frauds making up 0.172 percent of all transactions in the positive class. It only has numeric input variables that have undergone PCA transformation. Unfortunately, we are unable to offer the original characteristics and additional context for the data due to confidentiality concerns. The major components obtained with PCA are features V1, V2,..., V28. The only features that have not been changed with PCA are "Time" and "Amount." The seconds that passed between each transaction and the dataset's first transaction are listed in the feature "Time." The transaction amount is represented by the feature "Amount," which can be utilized for example-dependent, cost-sensitive learning. The response variable, feature "Class," has a value of 1 in cases of fraud and 0 in all other cases. We advise utilizing the Area Under the Precision-Recall Curve to measure accuracy given the class imbalance ratio (AUPRC). For categorization that is not balanced, confusion matrix accuracy is meaningless. The dataset was gathered and analyzed as part of a research cooperation on big data mining and fraud detection between Worldline and the Machine Learning Group at ULB.

4 Experiments

Python programming language was used to implement this project.

We used PCA(Principal Component Analysis) to reduce the dimension of the input dataset. We chose the first 10 significant features from the 28 as they retain 96.6% variation of the whole data. The rest of the features are dropped as they do not contain much significant information.

4.0.1 Approach-1

We try fitting a hyperplane(line) that best classifies the fraud and non-fraud data points giving equal weight to each class for the under sampled data. The non-fraud data is under sampled here[1]. The training data set we use has 293 fraud transactions and 300 non fraud transactions that are randomly sampled from the data. We take these 300 non fraud transactions to be the representative of all the non-fraud data points we have.

4.0.2 Approach-2

In the second case we try to fit a hyperplane(line) that best classifies the fraud and non-fraud data points giving unequal weights to each class[3]. The weights we give to each class is the reciprocal of the ratio of the respective classes in the training data set. Here we are changing penalty constant(C) for each class. The penalty constant C can be interpreted as the importance we give to each to class. Higher the importance, lower the chance of being misclassified. Since our goal is to minimise False neagtives cases we tend to give a bit higher importance to the fraud class[2].

4.0.3 Approach-3

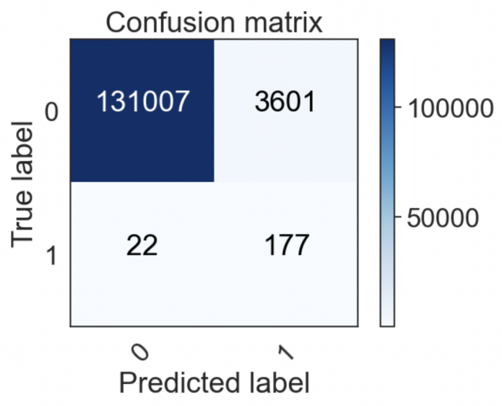
In addition second case, in the third case we try to shift the hyperplane(that we fit in second case) by "buffer/$||w||^2$" towards the non fraud class. By shifting the position of the hyperplane towards the non fraud class, we are assigning any suspicious transactions that were tagged non fraud as fraud. This approach allows the banks or clients to follow their "Better safe than sorry policy". This approach will tag transactions as fraud even if they have a low probability of being a fraud and would usually be classified as non fraud. Now the challenge that banks or clients face is to choose an appropriate buffer. The choice of the buffer usually depends on the amount of resources they are willing to dedicate to fraud detection. The amount of resources is directly proportional to the number of transactions they inspect. The total number of transactions they inspect is the sum of False positives and True positives. In other words every transaction that our model flags as "fraud" banks inspect them. Increasing the buffer increases the False positives but also decreases the number of false negatives, which is the ultimate goal of the banks. But in doing so, banks have to pay a price, the price here is the total number of transactions they are going to inspect. Choosing a buffer is a trade off between false positive and false negative cases.

4.1 Evaluation

We evaluate the performance of the model from confusion matrix. We can observe from the Figure 1 below that there are 22 fraudulent transactions that are classified as non-fraud. These are the cases that the banks want to avoid the most. There are also 3601 non fraudulent transactions that are flagged as fraud, the banks are not that concerned about this number unless it is too high.

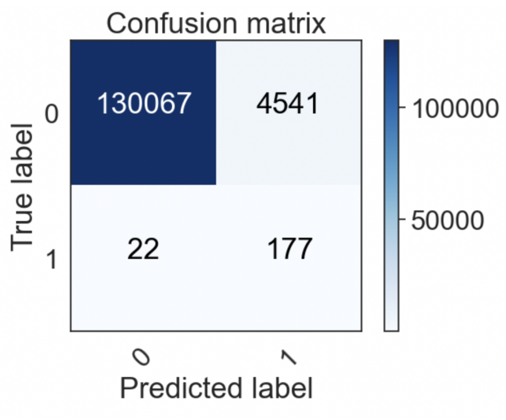
In this case, the weight we gave to fraud class is 0.6 and non-fraud is 0.4. We can see the resultant confusion matrix in the Figure 2 below. We see a decrease in the False negative cases from 22 to 20. At the same time we see an increase in the false positive cases from 4541 to 6229 this is a trade-off between FN and FP cases. We need to choose an ideal model that serves our purpose.

We tried fitting hyper-planes for different buffers = 0.3,0.5, 0.7. As we can see from Figure 3, the following confusion matrices for each buffer, the number of False positives are increasing as we increase the buffer and the number of false negatives is decreasing as we increase the buffer. As we mentioned earlier it is always a trade off between false positives and false negatives. As we try to increase one quantity by regulating the buffer the other quantity decreases. We see that increasing the buffer from 0.3 to 0.7 increases the false positives from 7760 to 17402 and false negatives decrease from 17 to 11. For a decrease of 6 false negatives false positives increased by almost 10000. In real life we notice that as we keep increasing the buffer the change in false negatives relative to false positives keeps on decreasing. The banks or clients try different buffers and choose a buffer that better suits their needs considering the financial resources they are willing to spare for the fraud detection.

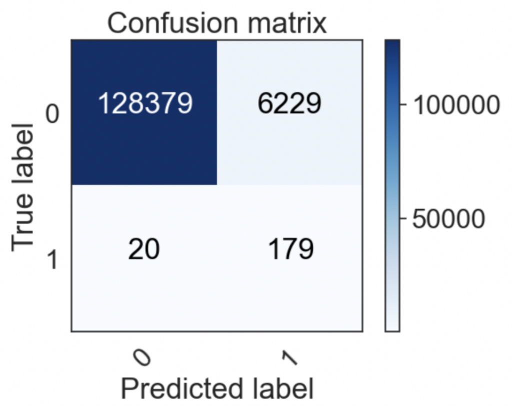


(a)

Figure 1: Confusion matrix for approach-1

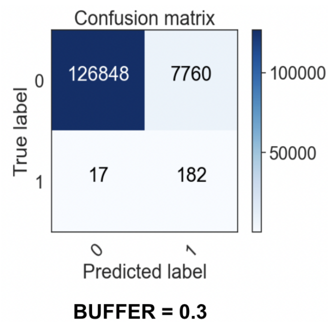


(a)

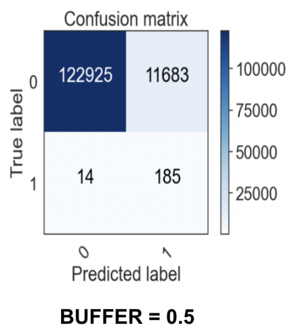


(b)

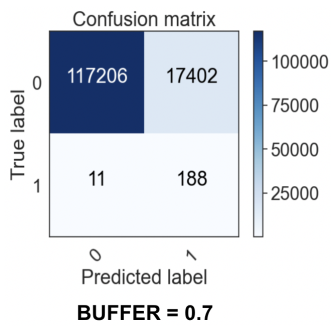
Figure 2: Confusion matrices for approach-2



(a)



(b)



(c)

Figure 3: Confusion matrices for approach-3

References

[1] Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson and Gianluca Bontempi. Calibrating Probability with Undersampling for Unbalanced Classification. In Symposium on Computational Intelligence and Data Mining (CIDM), IEEE, 2015

[2] Max Kuhn, Kjell Johnson, Applied Predictive Modeling, pages 343-344, 2013 [3] Fernández , Learning from Imbalanced Data Sets, pages 125-126,130, 2018.

[3] Fernández , Learning from Imbalanced Data Sets, pages 125-126,130, 2018.

[4 Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, An Introduction to Statistical Learning: with Applications in R, page 347, 2013.

[5] Yunqian Ma , Haibo He, Imbalanced Learning: Foundations, Algorithms, and Applications,pages 86,89, 2013.

[6] Xulei Yang, Qing Song and Yue Wang (2007). A Weighted Support Vector Machine for Data Classification, https://www.worldscientific.com/doi/abs/10.1142/S0218001407005703

[7]F. Song, Z. Guo and D. Mei, "Feature Selection Using Principal Component Analysis," 2010 International Conference on System Science, Engineering Design and Manufacturing Informatization, 2010, pp. 27-30, doi: 10.1109/ICSEM.2010.14.

[8]Jay Mayfield, 2022, https://www.ftc.gov/news-events/news/press-releases/2022/02/new-data-shows-ftc-received-28-million-fraud-reports-consumers-2021-0