Report - Bonus Project 4ML3

Shriyans Sharma

Contents

Chapter 1

Abstract

Credit card fraud transactions have been increasing each year worldwide as a result of evolving technology and increase in credit card transactions. Small to large businesses and global industries are consequently exposed to significant financial losses while the common public has to deal with phishing calls and messages commonly used to steal credit card information. This paper aims to use a neural network approach using a hybrid sampling technique to address the severe class imbalance. Fraud transactions occur rarely in both, real life scenarios and the given dataset. Thus, the problem can be mathematically formulated as follows:

$$\mathcal{L}(y, \hat{y}) = \begin{cases} 0, & \text{if } \hat{y} = y, \\ C_{\text{FN}}, & \text{if } y = 1 \text{ and } \hat{y} = 0, \\ C_{\text{FP}}, & \text{if } y = 0 \text{ and } \hat{y} = 1. \end{cases}$$

where $C_{\text{FN}} >> C_{\text{FP}}$ i.e. the cost of flagging a fraud transaction as a normal transaction is significantly higher than the cost of flagging a normal transaction as a fraud transaction.

Also, the dataset has a total of 31 features. Of about 284,000 transactions, about 492 are fraud transactions. Some original features and their descriptions are not provided due to confidentiality issues. Thus, variables names from V1 – V28 are obtained via PCA i.e. a linear dimensionality technique used to reduce the number of variables while retaining vital information.
Dataset link: `https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud`

Chapter 2

Current related work and state-of-the-art

2.1 Pre-processing

Resampling techniques will be required since the dataset is highly imbalanced i.e. the ratio of normal to fraud transactions is approximately 10,000:17. Currently, random oversampling is commonly done using the SMOTE technique and random undersampling has become a standard practice in approaching this problem.
A research paper [Pyar 4], uses undersampling highlighting that it reduces overfitting, encourages faster training time and aims to balance the number of instances of the minority class and the majority class. The paper seems to have failed to account for the inability to deduce important vs. unimportant observations that must be kept or removed from the dataset which can heavily impact the generalization of the model.
This approach can be slightly modified to enhance feature conservation by removing instances of the majority class from regions in the dataset that are far from regions where fraud transaction instances occur since they would have no significant relevance to the fraud transaction itself. But, this approach may have a flaw where the relevance of a transaction from a distant region to a fraud transaction is assumed to be negligible.
Another research paper [Chawla et. al. 8], uses an oversampling technique called SMOTE i.e. Synthetic Minority Oversampling Technique as an approach highlighting that it increases the instances of the minority class by producing synthetic samples and improves the model's generalization capability. The paper fails to account for the underlying statistical distribution used for SMOTE resampling where all observations of the minority class are considered to be a part of a uniform distribution. Production of such synthetic samples using this approach cannot be evaluated to be statistically correct since each point may have a varying probability of occurrence within the minority class.
To fill the gap in the SMOTE technique, the k value should be as small as possible but not 1 since the generation of the synthetic samples using the formula, $X_{new} = X_0 + \lambda \times (X - X_0)$, will result in generation of collinear synthetic samples. The value of k should be minimized to less than 4 under the assumption that the nearest neighbors have the highest probability of occurrence in that region. This would provide more realistic samples while comparatively reducing noise in the dataset since each observation is not considered to have equal probability of occurrence. Yet, there is a nuance that it may introduce possible bias towards closest instances of the minority class in that region.

2.2 Model selection

Current state-of-the-art methods include modified SVMs, multi-layered perceptrons i.e. artificial neural network, and logistic regression. Other prominent techniques include: random forest, xgboost and decision trees.

A research paper [Zhang et. al. 7], uses a weighted SVM approach where a standard SVM was used with a varying penalty for every misclassified transaction. Furthermore, the weights assigned are proportional to the transaction amount. This approach takes the varying importance of each transaction into account and ensures assignment of asymmetric loss which emphasizes on selecting observations that carry more relevance to building a generalization of the dataset. However, it fails to account for the large computation costs even though the kernel trick can reduce the costs to a certain extent since the dataset itself is huge. In addition, the weighted SVM does guarantee convergence to a global optimum but it requires a large number of iterations and parameter updates are triggered primarily by misclassified samples.

The authors could have enhanced computation times by undersampling the majority class using the constrained undersampling approach described in section 2.1. However, the method itself is extremely sensitive to weight selection and requires careful threshold calibration to ensure optimal performance.

Another research paper [Kasasbeh et. al. 4], uses an artificial neural network explaining that the approach can handle extremely complex and non-linear relationships, supports parallel processing allowing faster processing for large datasets and handles noise in datasets much better as compared to other approaches. Nonetheless, it fails to account for unreliable convergence towards global optimum i.e. there is a possibility that the model converges to a local minimizer, higher chances of overfitting since the model's primary focus is on capturing more complex relationships, higher computation power requirements, and suitable hyperparameter tuning since small changes in the hyperparameter can significantly affect results.

Authors may have overlooked performance improvement by using the mini-batch gradient descent algorithm where the size of the batch must be optimal to ensure efficiency since sgd i.e. stochastic gradient descent would be lead to noisy and unstable convergence whereas batch gradient will be too slow for such a large dataset and consumes a higher amount of memory. In addition, the L2 regularization penalty should be used for the model instead of L1 since L1 harshly penalizes smaller weights which could remove important information for the model to use while training. Furthermore, the number of layers and neurons in each layer should be chosen efficiently to balance the trade off between computational complexity vs processing power.

Finally, a research paper [Alenzi et. al. 8] uses the logistic regression approach citing that it is a good choice of a machine learning model because it is based on no assumptions on the distribution of the feature space, efficient to train and classifies unknown records. This paper overlooks critical flaws where logistic regression may be efficient but requires more iterations as compared to other standard approaches like the neural network. It may also fail to detect very complex non linear relationships because a polynomial classifier may be required where computations can be reduced to a certain extent using a kernel trick but it compounds over iterations, and updates the gradient based on every observation since it uses log loss i.e. cross entropy.

The performance of this approach is substandard compared to other techniques without refinements but an appropriate choice of learning rate, polynomial classifier and initialization of weights could match the performance with other algorithms like weighted SVMs and neural networks. Nevertheless, a highly optimized logistic regression may not be able to match the performance of other fine-tuned complex approaches like weighted SVMs and neural networks.

2.3 Performance metrics

Currently, most research papers focus performance analysis via metrics which primarily are: accuracy, precision, recall, f1 scores and confusion matrices. Two papers, [Date et. al. 4] and [Zhang et. al. 8] have used these metrics to analyze the performance while [Zhang et. al. 8] added an extra metric called financial recovery i.e. $\frac{amount\ in\ true\ positives}{amount\ in\ all\ positives}$. Other papers that used financial recovery as a metric also evaluate the amount of fraud transactions that the model was able to catch while giving more importance to an undetected transaction of \$5000 as compared to an undetected transaction of \$50. Another paper, [Date et. al. 4] uses the FPR i.e. False Positive Rate as a metric to evaluate the proportion of transactions that are flagged as fraud falsely to all normal transactions. There are two other metrics that are quite commonly used in this case: the AUPRC i.e. Area under Precision Recall Curve, and the AU ROC i.e. Area under Receiver Operating Characteristic Curve. The paper, [Date et. al. 4] uses the AU ROC curve by justifying that ROC AUC is better than accuracy to evaluate performance since it is independent of majority class dominance or skewed distribution in the dataset. This paper fails to account for the focus on the minority class of the dataset i.e. fraud transactions where using ROC AUC alone would not suffice so AUPRC could be a good addition since it is primarily used to evaluate the model performance in detecting positive cases which are very rare i.e. in this case $< 1\%$.

Chapter 3

Why is this approach the best choice?

Rather than formulating a new algorithm, this approach applies small, targeted modifications to pre-existing techniques to enhance performance, reduce processing time and computational complexity.

Several research papers and authors implement state-of-the-art methods adding specific refinements to optimize performance but they fail to account for small changes that could increase performance of the model drastically. Most approaches ideally use the standard technique instead of considering a hybrid approach combining the constrained SMOTE approach as discussed in section 2.1 with random under-sampling. In comparison to other papers, this will oversample the minority class by adding synthetic samples without increasing noise in the dataset and providing a more realistic dataset. And, using this approach without balancing the two classes at a ratio of 1:1 and maintaining a slight imbalance, the majority structure of the dataset is preserved for training of the neural network.

From all the approaches that are commonly used for an appropriate machine learning model, multi-layered perceptrons i.e. neural networks, weighted SVMs and other techniques, multi-layered perceptrons requires the most computational complexity but take lesser iterations to converge to the global optimum. Other research papers primarily focus on performance optimization by the number of neuron layers, but this will be focused on using an appropriate gradient descent method, optimizing the number of neuron layers, and managing the trade-off between performance vs processing time by using an apt batch size.

Standard metrics: accuracy, precision and fpr i.e. false positive rate will be required to evaluate the general performance of the model. And, AUPRC will be required to evaluate the minority class i.e. fraud transactions. Although, accuracy can be replaced by AUC ROC to evaluate the performance of the model across multiple thresholds since the dataset is highly skewed, it is not necessary for this specific case since the primary focus is on detection of fraud transactions.

Chapter 4

Final method

4.1 Preprocessing

Firstly, the data will be split into training and testing sets. Features of the training dataset must be scaled using RobustScaler to ensure robustness against outliers but preserve the pattern while ensuring that variables with larger magnitudes do not dominate the model calculations. Then, the class imbalance issue will be resolved using standard random undersampling with the constrained SMOTE technique discussed in section 2.1. This will reduce the observations of the majority class and add synthetic samples of the minority class without adding noise to the data.

4.2 Model selection

A multi-layered perceptron i.e. neural network will be used with an L2 regularization penalty, mini-batch gradient descent algorithm. Furthermore, two experiments will be conducted to determine optimal batch size and the number of neuron layers required. And, the adam optimizer will be used for an adaptable learning rate rather than having a constant learning rate which would lead to a better gradient descent.

4.3 Loss function

The binary focal loss function will be implemented which is formulated as:

$$\mathcal{L}_{\text{focal}}(y, \hat{y}) = -\alpha \, y \, (1 - \hat{y})^{\gamma} \log(\hat{y}) - (1 - \alpha) \, (1 - y) \, \hat{y}^{\gamma} \log(1 - \hat{y})$$

This loss function as compared to the standard binary cross entropy function primarily focuses on training harder to classify samples and down-weights the easier classification samples. $\gamma = 0$ results in the standard binary cross entropy formulation whereas $\gamma > 0$ results in higher contributions from harder training samples, i.e. samples that are classified with extremely low confidence, while it takes lower contributions from easier training samples. $\alpha = 0.5$ makes the loss function symmetric whereas an $\alpha > 0.5$ results in an asymmetric loss function formulation as described in the abstract. Thus, a value of $\gamma > 0$ and $\alpha > 0.5$ will be used for this methodology to make an asymmetric loss function which focuses on harder to classify training samples.

4.4 Performance analysis

Accuracy, precision, fpr i.e. false positive rate, AUPRC and financial recovery will be used to evaluate the model's performance while primarily focusing on the minority class i.e. fraud transaction cases.

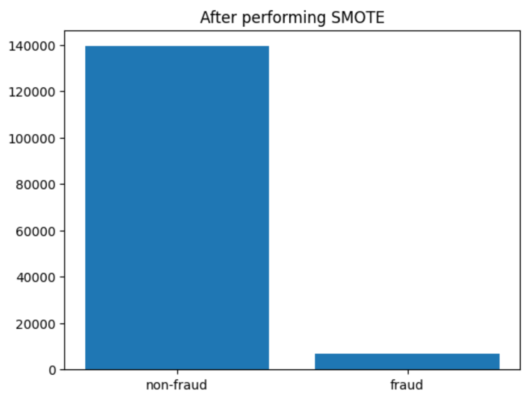
Chapter 5

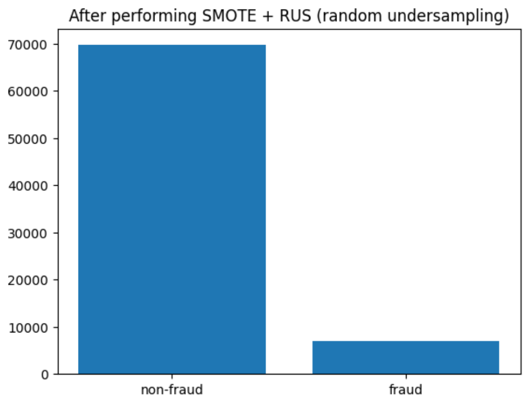
Results and Limitations

5.1 Results

5.1.1 Pre-processing

The data was pre-processed by being split into training and testing sets. And, the training set was further split into a training and validation set. Then, the hybrid technique, SMOTE + RUS was performed on the training set to increase but preserve the ratio instead of balancing out the two classes.





5.1.2 Experiment 1

Afterwards, an experiment was performed to find the optimal batch size between 64, 128 and 256. The model was trained with the three batch sizes in 10 iterations and the auprc was calculated for each of the three batch sizes in each iteration. Finally, it was observable that the model had the highest auprc with a batch size of 256. A larger batch size such as 1024 was not used in the experiment since it would produce smoother boundaries leading to neglect of rare, unique edge cases of fraud transactions.

5.1.3 Experiment 2

Then, an experiment was performed to find the optimal neuron layer setup between $128 \rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 1$, $64 \rightarrow 32 \rightarrow 16 \rightarrow 1$, and $32 \rightarrow 16 \rightarrow 1$. Over ten iterations, the auprc was calculated for each neuron layer setup for each iteration. Finally, it was observable that the model with the neuron layer setup, $64 \to 32 \to 16 \to 1$, had the highest auprc concluding that it performed the best among the models.

5.1.4 Model statistics and explanations

The final model that was deduced as a result of the experiments had the following statistics:

$$\begin{aligned}
Optimal\ Threshold &= 0.7929 \\
Accuracy &= 0.9993 \\
Precision &= 0.8485 \\
False\ Positive\ Rate &= 0.0003 \\
AUPRC &= 0.7683
\end{aligned}$$

$$\begin{aligned}
Fraud\ Amount\ Recovered &= 12{,}307.24 \\
Total\ Fraud\ Amount &= 19{,}311.89 \\
Financial\ Recovery &= 0.63
\end{aligned}$$

In severely skewed datasets, accuracy is often misleading since misclassifications of the minority class have little effect, explaining the high value, 0.9993. Furthermore, the low false positive rate suggests that the model is able to classify normal transactions extremely well. And, the financial recovery of 0.63 suggests that the model was able to detect larger value transactions but failed to detect edge case transactions possibly due to the hybrid approach, i.e. SMOTE + RUS (random undersampling), which can introduce synthetic noise using SMOTE and remove important observations using the RUS. Overall, the metrics such as the auprc indicate the model is able to perform well and catch most of the fraud transactions well while failing to detect the rare and unique edge cases.

5.2 Limitations

All theoretical limitations discussed in section 2.2 become apparent while implementing the neural network. The training of the neural network is noisy and unstable due to the class imbalance of the dataset. Furthermore, the performance metrics may change during recompilations of the program. Main performance metrics like the AUPRC and Financial Recovery can usually vary approximately between ± 0.04 and ± 0.10 respectively.

Chapter 6

References

Kyi Pyar. (2024).
Improvement The Accuracy of Convolutional Neural Network with Using Undersampling Method on Unbalanced Credit Card Dataset.
`https://ijadis.org/index.php/ijadis/article/view/1333/77`

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer. (2002).
SMOTE: Synthetic Minority Over-sampling Technique.
`https://arxiv.org/abs/1106.1813`

Dongfang Zhang, Basu Bhandari, Dennis Black. (2020).
Credit Card Fraud Detection Using Weighted Support Vector Machine.
`https://doi.org/10.4236/am.2020.1112087`

Bassam Kasasbeh, Balqees Aldabaybah, Hadeel Ahmad, (2022).
Multilayer perceptron artificial neural networks-based model for credit card fraud detection.
`https://ijeecs.iaescore.com/index.php/IJEECS/article/view/26201/16267`

Hala Z Alenzi, Nojood O Aljehane. (2020).
Fraud Detection in Credit Cards using Logistic Regression.
`https://thesai.org/Downloads/Volume11No12/Paper_65-Fraud_Detection_in_Credit_Cards.pdf`

Disha A. Date , Rugvedi Y. More , Rutuja P. Harne , Mr. Dilip M. Dalgade. (2022).
Credit Card Fraud Detection Using Machine Learning.
`https://www.ijraset.com/best-journal/credit-card-fraud-detection-using-machine-learning-231`

Chapter 7

Explanation of packages and important parts of code

7.1 Packages

- Pandas - used for reading the csv file and storing x,y
- Numpy - used for finding the index for the max. value along a specified axis of an array
- Tensorflow - used to create the neural network i.e. multi-layered perceptron model, regularizers (l2) and Binary Focal Cross Entropy loss
- Scikit - used for performance metrics, train/test splitting of the dataset and scaling the data
- Imblearn - used for preprocessing techniques: SMOTE and RUS i.e. random undersampling
- Matplotlib - used for making bar plot and PRC i.e. Precision Recall Curve graph

7.2 Part of code

- k_neighbors (SMOTE), imblearn - to compute the synthetic point using the k nearest neighbors as discussed under section 2.1
- stratify, sklearn - used to ensure that the proportion of classes in the training and testing sets is the same as the proportion in the original dataset