Credit card fraud detection using Keras NN

Daniel Dimanov
MSc Applied Data Analytics
Bournemouth University
Bournemouth,UK
i7461730@bournemouth.ac.uk

***Abstract*—Credit card frauds are a growing problem, costing companies billions of dollars annually. This paper introduces an approach using neural networks to tackle this issue. The model achieves over 94% confidence level. It is tested using AUC and ROC curve results for evaluating performance and discusses the process of creating the model and the encountered difficulties. The paper also concludes that the number of hidden layers does not prove to have an effect on the performance.**

***Index Terms*—FDS, Fraud Detection, Neural Networks, MLP, Keras**

I. Introduction

Credit card fraud has gained popularity in the last years. The increasing number of digitalised payments has made credit cards a preferred target for fraudsters [1]. In 2013 17% of the identity thefts reported to the Federal Trade Commission(FTC) were because of credit card frauds [2]. Attackers can steal a lot of money, very fast, while leaving encrypted and hard to follow traces. These types of frauds are the reason 128 million US dollars were reported to FTC to be stolen only in 2011 [4].Companies are losing billions of dollars from credit card fraud each year. $22.8 billion is the reported amount of losses in 2016 globally due to fraudulent transactions as stated by the Nilson Report [3].While there are models dating back to the late 90s [5], the problem is still present and various techniques are used to combat the increasing number of fraud attempts [6]. Designing a Fraud- Detection system(FDS) using neural networks is a complex process, involving a multitude of challenges. Imbalanced datasets and concept drift are some of the main challenges faced in the process [6]. The credit card transactions are naturally highly imbalanced, because there are lots of genuine transactions and only a minuscule amount are fraudulent [1]. Concept drift as described in Joào Gama et al. "refers to an online supervised learning scenario when the relation between the input data and the target variable changes over time." [7] In the context of credit card frauds the conceptual drift is introduced as a challenge, because of the unpredictable nature of human decisions for transactions. Facing these and many more challenges is making the process hard, but the necessity for such algorithm is rapidly growing [6]. This paper introduces a Multi Layer Perceptron classifier [1], which is trained on the ULB - Machine Learning group "Credit card fraud detection" dataset posted on Kaggle [2].

[1]https://www.kaggle.com/ddimanov/dmat2018-i7461730-daniel-dimanov/
[2]https://www.kaggle.com/mlg-ulb/creditcardfraud

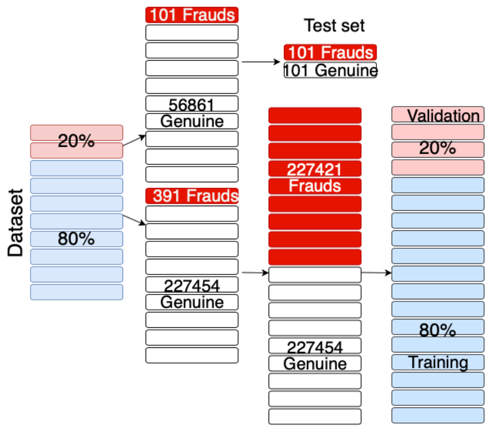


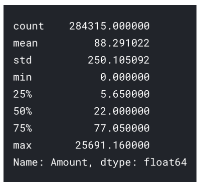
Fig. 1: Train-test-validation split

A. The Problem

The problem discussed in this paper is creating a neural network classifier, which is able to differentiate genuine from malicious(fraudulent) transactions. This involves dataset analysis, preprocessing of data, balancing the dataset, designing and training a Multi-layer Perceptron(MLP), experimenting with the design and finally analysing the performances of the classifiers.

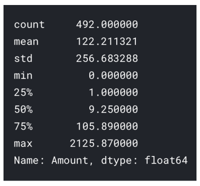
II. Method

This section describes the approach undertaken to tackle the described in I-A. First the dataset is statistically analysed using different tools (introduced in II-C) and visualisations shown in II-A. Then, because the dataset is oversampled to achieve best results while minimising loss of data. The process of oversampling (the chosen data balancing technique) is explained in II-B. After that a MLP is designed (more in II-D). For the purpose of testing the classifier with independent data, 20% of the dataset is kept away before oversampling. Undersampling was one of the considered strategies, but because of insufficient sample size for testing, instead ROC curve and AUC(area under the curve) were used. Another 20% of the the oversampled data is used for validation, but since there is some data leakage during validation and the results depend on the quality of oversampling algorithm, the test dataset is used for final testing Fig.1 .



```
count    284315.000000
mean         88.291022
std         250.105092
min           0.000000
25%           5.650000
50%          22.000000
75%          77.050000
max       25691.160000
Name: Amount, dtype: float64
```

(a) Genuine transactions



```
count     492.000000
mean      122.211321
std       256.683288
min         0.000000
25%         1.000000
50%         9.250000
75%       105.890000
max      2125.870000
Name: Amount, dtype: float64
```

(b) Fraudulent transactions

Fig. 2: Statistics for genuine and fraudulent transactions

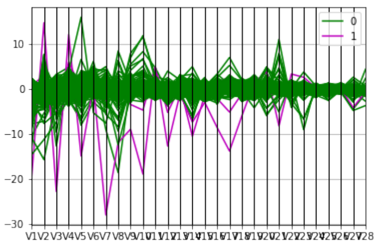


Fig. 4: Parallel plot of a sample from the dataset

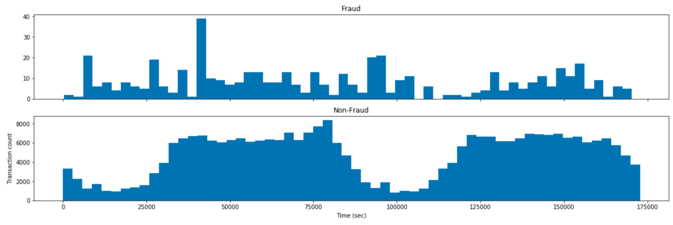
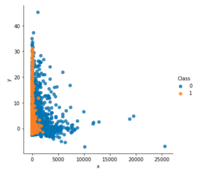
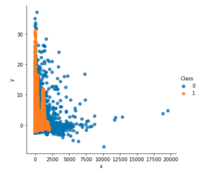


Fig. 3: Transaction times



(a) Before



(b) After

Fig. 5: Train and validation data before and after oversampling

A. *Dataset Analysis*

The used dataset contains 284,807 transactions (492 of which fraudulent) collected in the span of two days in September 2013 [8]. The data consists of 30 parameters with a 31st labelling the transaction as either fraudulent or genuine. 28 of the parameters are features of the transactions, which are encoded (using PCA Dimensionality reduction) to protect users private data [8]. The fraudulent transactions represent only 0,172% of the whole dataset, which will be tackled in II-B. Nevertheless, some conclusions can be drawn, after examining the data more closely. As evident from Fig.2 the amounts of the transactions are pretty similar. The fraudulent ones have a smaller maximum amount, but most importantly the data is highly imbalanced with only 492 fraudulent transactions and over 280 thousand genuine ones. In terms of the time when the frauds occurred compared to the genuine transactions there is certainly a trend that the density of fraudulent transactions is unnaturally high in off-peak hours as illustrated by Fig.3.

This requires time series and using recurrent neural networks as well as passing the data in a specific way [9]. While this may be a great idea, this paper does not focus on the time parameter and even drops it in the further training. When the data is pair-plotted and parallel plotted the contrast between the classes for other features (especially V1-V4 ,V7,V11,V17, V23 and V24) becomes more apparent (Fig.4 the pair plot can be found in the kaggle notebook) .

B. *Dataset Balancing*

For balancing the data this paper explores an approach using oversampling. The oversampling is done so that the classifier can be exposed to more fraudulent transactions and for the results to be significant. Both Synthetic Minority Oversampling Technique (SMOTE) [10] and Adaptive Synthetic (ADASYN) [11] were considered for the process and after achieving more satisfactory results, the ADASYN algorithm was chosen Fig.5.

C. *Technology used*

The project published on [3] uses Kaggle notebook as an environment and Python as a programming language. The libraries used in the project can be found in the notebook, but mainly Pandas are used for data processing (statistical analysis of data as well as some visualisations). Numpy is used for lists creation and linear operations on data. Seaborn is used for the more complex visualisations. Scikit-learn [4] is used for Principal component analysis(PCA), dimensionality reduction and splitting the dataset. The ADASYN is used for oversampling. Keras (which uses tensorflow in the backend) is used for the designing of the MLP, adding the different layers, training, testing and twitching the model. Keras is the library used to boost productivity in designing a neural network. Lastly, Scipy is used for results analysis and statistical operations and significance testing.

D. *Neural Network*

The neural network used for classification in this paper is build using the Keras library. It consists of 4 layers (input, 2 hidden and 1 output). More detailed information for the layers can be found in Table.I. The activations are rectified linear unit ReLU(because the data is linearly inseparable and Relu is breaking the linearity), followed by tanH, because it performs well when classifying two classes. [12] The last layer is softmax, which squishes the output between 0 and 1, thus outputting a probability for each class. The one with the more probability is chosen. There are two extra things implemented in this paper. First cross-validation is done with 5-fold validation, using Scikit-learn to split the data and feed it in parts for training and validation. Finally, AUC of a ROC curve is used as a metric to compute the performance.

[3]https://www.kaggle.com/ddimanov/dmat2018-i7461730-daniel-dimanov
[4]https://scikit-learn.org/stable/

TABLE I: Characteristics of models: Control and Experimental arms, number of layers, number of neurons in each hidden layer and its activation function and the mean of its performance

	Layers	L1	L1 Act	L2	L2 Act	L3	L3 Act	AUC
Ctrl	2	29	ReLU	3	tanH	-	-	94,14
Exp	3	29	ReLU	16	ReLU	3	tanH	92,53

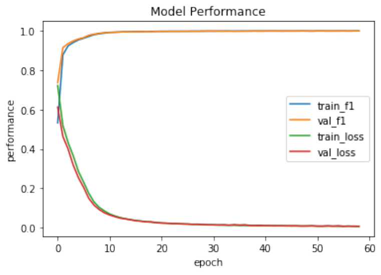
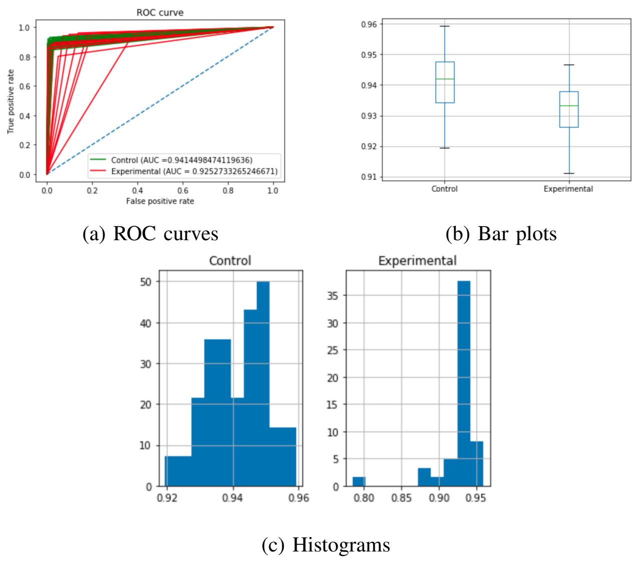


Fig. 6: Zoomed graph of the performance history of the model during training



(a) ROC curves

(b) Bar plots

(c) Histograms

Fig. 7: Visualisation of the results of the experiment

Hypothesis 1: The number of hidden layers affects the performance of a neural network classifier on the "Credit card fraud" dataset [5].
H^0: Null hypothesis: results suggest there is no relationship between the number of layers and the performance.
H^1: Alternate hypothesis: results suggest there is a relationship between the number of layers and the performance.

III. Numerical Results

A. Experimental Setup

The model created in this paper follows the structure displayed in Table.I under "Control Arm" and II-D. It follows some general conventions like having at least the same number of neurons in the hidden layer and the input layer [12] , using 2 output neurons on the output layer, which represent the fraudulent and genuine transaction classes and using softmax function to calculate the probability for each output. The model uses Adam optimiser, which uses stochastic gradient descent, but also keeps the learning rate per neuron weight according to how the performance changes. It is efficient and adjustable. It uses binary cross entropy for loss function, because there are only two classes and can be treated as positive and negative. The metric used for computing the on batch performance is f1-score, but because the data is oversampled the accuracy is similar. The model is trained with a batch size of 5000 transactions and converges around 10th and 20th epoch. (Please refer to Fig. 6) The number of epochs for the model were decided to be 50 to ensure maximum performance gain, while conserving computational power and not overfitting. The experimental design to test the hypothesis differs from the control arm by having an extra ReLU hidden layer with 16 neurons (Refer to Table.I).

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

B. Experimental results

Both models use 5-fold cross validation using the Stratified K-Fold function from the Scikit-learn library. The way both models are tested for results is that each model runs for 7 iterations of 5 fold cross validation and all of the 35 performances of each individual model are kept for further analysis(which is considered to be a sufficient sample size [12]). The way the performance is tested is using the 20% of the dataset, which were kept in the beginning and are never exposed to the model. The reasons for not using the cross-validation results are that there is data-leakage during cross-validation and bias is introduced from the oversampling. Which means that the results will be highly inaccurate and dependent on the quality of the oversampling technique. Instead, the "pure" 20% are used. The problem is that the data is imbalanced, so first under-sampling of the test data was attempted. It worked well, but the sample size for testing was 202, which was insufficient sample size, so the data is used in its raw format, but because it is highly imbalanced the metric chosen for its evaluation is area under the curve(AUC) using Receiver operating characteristic(ROC) curve. The results were relatively satisfactory with average AUC scores of 94,14% and 92,53% (Table I). The ROC curves shows that the experimental arm data is a bit more unstable, so maybe a bigger sample size can be considered for future work or more epochs, because it may have not converged at 50. (Fig. 7) More detailed statistics of the results may be found in Table. II The fact that the results from the experimental arm do not come from a normal distribution makes it impossible to conclude with certainty that one of the models is better than the other. A Pearson correlation test was done to examine if there is any relationship between the results of both arms, but with a p score of 31,56% it revealed that the results are indeed not related, which means that the null hypothesis has to be accepted.

TABLE II: Results of control and experimental arms.

	Mean ROC AUC	STD	Normal Distribution	Min ROC AUC	Max ROC AUC
Control	94,15%	0,0092	74,81 %	91,94%	95,92%
Experimental	93,42%	0,030	7,67 x 10^{-10}%	78,49%	95,94%

IV. Conclusions

As credit card frauds are becoming more popular, this paper introduces a neural network classifier, which can identify fraudulent transactions with over 94% confidence. The process of designing the classifier involved a lot of data preprocessing mainly because the dataset is highly imbalanced. This problem was tackled using oversampling techniques and training the classifier with a combination of synthetic and real data. Even though, results suggest that the number of hidden layers has no significant effect on the performance of the model, future work should explore the possibility of reaching alternative hypothesis.

V. Future work

Aspects of the overall process as well as specific parts of this project may be improved. Time series and the ”time” parameter can be introduced to the classifier. Sample size sufficiency of the test set may be improved and a larger set of actual fraudulent transactions may be considered. The approach for building a test set with undersampled data can be improved, thus making results even more realistic. Finally, PR curve may achieve better results than ROC curve, because the data is extremely imbalanced [13].

References

[1] M. Zareapoor and P. Shamsolmoali, ”Application of Credit Card Fraud Detection: Based on Bagging Ensemble Classifier”, Procedia Computer Science, vol. 48, pp. 679-685, 2015. Available: 10.1016/j.procs.2015.04.201.

[2] National Criminal Justice Reference Service, ”Identity Theft - Facts and Figures”, 2015.

[3] ”The Nilson Report”, Nilsonreport.com, 2017. [Online]. Available: https://nilsonreport.com/publication_newsletter_archive_issue.php?issue=1118. [Accessed: 03- Jan- 2019].

[4] Federal Trade Commission, ”Consumer Sentinel Network Data Book”, 2014.

[5] E. Aleskerov, B. Freisleben, and B. Rao. Cardwatch: A neural network based database mining system for credit card fraud detection. In Computational Intelligence for Financial Engineering, pages 220–226. IEEE/IAFE, 1997.

[6] A. Pozzolo, G. Boracchi, O. Caelen, C. Alippi and G. Bontempi, ”Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy”, IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 8, pp. 3784-3797, 2018. Available: 10.1109/tnnls.2017.2736643.

[7] J. Gama, I. Z liobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. ACM Computing Surveys (CSUR), 46(4):44, 2014.

[8] A. Pozzolo, O. Caelen, R. Johnson and G. Bontempi. Calibrating Probability with Undersampling for Unbalanced Classification. In Symposium on Computational Intelligence and Data Mining (CIDM), IEEE, 2015

[9] J. Connor, R. Martin and L. Atlas, ”Recurrent neural networks and robust time series prediction”, IEEE Transactions on Neural Networks, vol. 5, no. 2, pp. 240-254, 1994. Available: 10.1109/72.279188.

[10] N. Chawla, K. Bowyer, L. Hall and W. Kegelmeyer, ”SMOTE: Synthetic Minority Over-sampling Technique”, Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002. Available: 10.1613/jair.953.

[11] He, Haibo, Yang Bai, Edwardo A. Garcia, and Shutao Li. “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” In IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), pp. 1322-1328, 2008.

[12] S. Rostami, ”Machine Learning with Kaggle Kernels – Part 2 : Part 2 – Experimental design”, Shahin Rostami Research, 2018. .

[13] T. Saito and M. Rehmsmeier, ”The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets”, PLOS ONE, vol. 10, no. 3, p. e0118432, 2015. Available: 10.1371/journal.pone.0118432 [Accessed 5 January 2019].