ABSTRACT

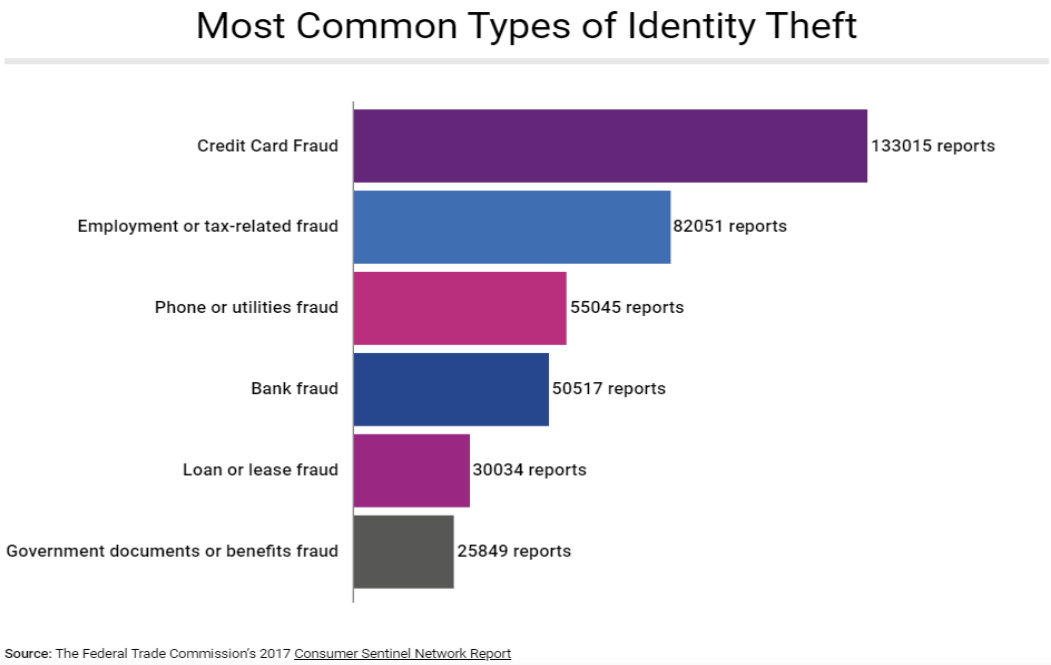
Cyber criminal activities are witnessed in huge proportions now more than ever, especially in banking arena and Machine learning algorithms help to mitigate its effect by making detection early in the stage using behavioral patterns.

In this research, we propose an approach using Artificial Neural Networks (ANN) with K-Means SMOTE resampling technique on the credit card (CC) transaction data set from Kaggle. We evaluated the model performance according to f1-score metric by tuning hyperparameters for both the resampling technique and ANN. The goal of this project is to train a model which performs well in detecting fraudulent CC transactions.

1. INTRODUCTION

According to the US Federal Trade Commission (Commission, 2019), reports of credit card fraud went up by 24% last year alone, with no signs of slowing down. It's increasing despite a steady advancement towards more secure cards and transaction methods, leading many of the world's largest merchants and credit card issuers to search for new solutions to the problem. That's why more industry players are turning to AI and machine learning techniques to limit or prevent fraudulent activity. In our project, we use our acquired acumen in Machine Learning course about neural networks to explore the possibility of coming up with a good forecasting model to predict fraudulent transactions using CC.

For this end, first, we performed a brief pre-processing of the dataset, then since the dataset is highly imbalanced, we implemented K-means SMOTE Resampling technique, and finally we fit and tune a feed-forward neural network in order to achieve high prediction performance.



2. DATASET

The datasets[1] contain transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data. Features V1, V2, ... V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

3. LITERATURE REVIEW

A lot of research related to credit card fraud detection have been published. Majority of them make use of Machine learning algorithms.

In "*Analysis on Credit Card Fraud Detection method",* the authors compared various techniques to detect credit card frauds. According to the results, ANN produced an output of 70% of True Positive (TP) Rate while the best model (Fuzzy Darwinian Detection) produced 100% of TP rate.(Raj & Portia, 2011).

In "*Credit card fraud detection using Bayesian and neural network*" article by Maes, Tuyls, Vanschoenwinkel, and Manderick (2002), comparison table states that Bayesian Belief Network (BBN) performs better than ANN used for fraud detection.

In "*Detecting credit card fraud by ANN and Logistic Regression*", by Sahin and Duman (2011), ANN and LR have been compared according to the accuracy and the true positive rate. The results illustrated that ANN models outperform logistic regression models. In addition, they substantiated that LR models overfit the training set more than ANN did.

In summary, there has been a lot of recent research, which aims to explore ANN performance in the context of detecting fraud in CC transactions. Therefore, it could be a good inspiration to use this model as our focus.

However, given the imbalance nature of the data used in credit card fraud detection it will be of importance to consider resampling techniques before training the model.

There are many resampling techniques such as oversampling, under sampling, Synthetic Minority Oversampling TEchnique (SMOTE), etc. Under sampling

[1] Dataset URL: https://www.tinyurl.com/vgwe4f6

techniques can lead to loss of important information whereas oversampling and SMOTE can lead to generation of noise[2].

To overcome these issues, Last, Douzas, and Bacao (2017) proposed a method in *"Oversampling for Imbalanced Learning Based on K-Means and SMOTE*" which combined K-Means clustering algorithm with SMOTE in order to increase the effectiveness of the resampling strategy.

Our goal in this project will be to combine ANN model with K-Means SMOTE and figure how best to fit the model to reach a good prediction performance.

4. METHODOLOGY

The following lines discuss the overall methods for our study.

- **Preprocessing –**

First, we scaled the features 'Time' and 'Amount' because their standard deviation is higher than the other features. Besides, we intend to use K-means algorithm for clustering step in this project, which depends on Euclidian distance between datapoints, making scaling meaningful.
We also split the dataset into training set (80%) and test set (20%).

- **Resampling Technique –**

K-Means Synthetic Minority Oversampling Technique (SMOTE) consists of three steps: clustering, filtering, and oversampling. Once the input space is clustered into K-groups using K-Means Algorithm, filtering allows to select clusters for oversampling, retaining those with a high proportion of minority class samples. It then distributes the number of synthetic samples to generate, assigning more samples to clusters where minority samples are sparsely distributed. Finally, in the oversampling step, SMOTE is applied in each selected cluster to achieve the target ratio of minority and majority instances. The algorithm is illustrated in figure 1.
SMOTE chooses a random minority observation A within the cluster, finds a random neighboring minority instance B of that point and determines a new sample 'x' by randomly interpolating A and B. In geometric terms, the new point 'x' is thus placed somewhere along a straight line from A to B. The process is repeated until the number of samples to be generated is reached.
SMOTE's hyperparameter k nearest neighbors, or k-nn, constitutes among how many neighboring minority samples of A the point B is randomly selected.

[2] https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/

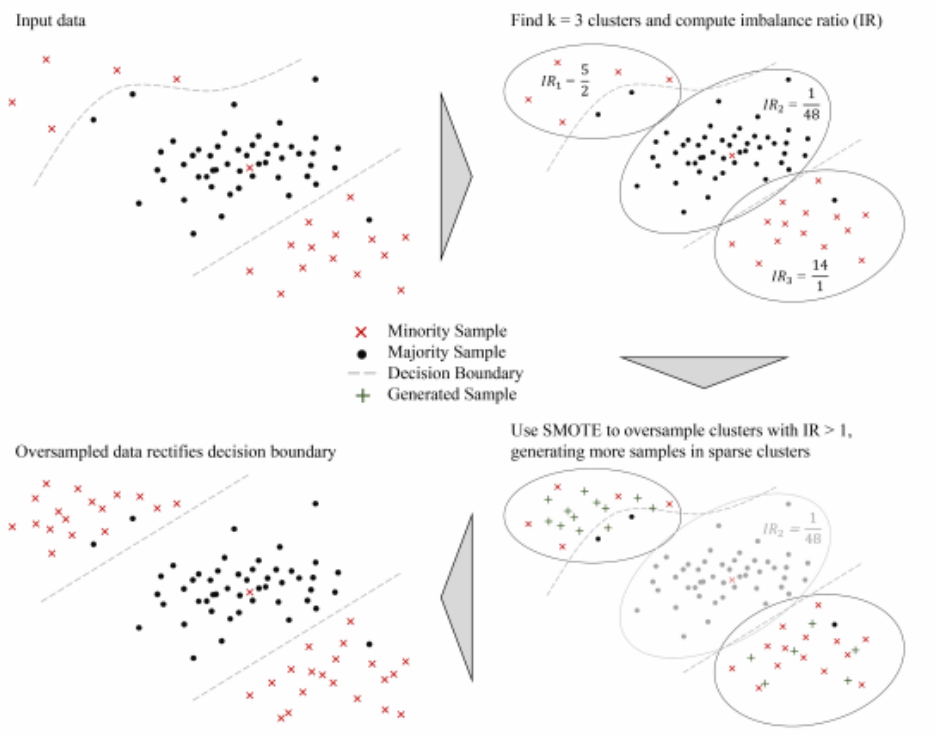


Figure 1: K-means SMOTE steps
[image from: *Oversampling for Imbalanced Learning Based on K-Means and SMOTE,* Felix Last & al.]

Here, as a starting point, we initiated K-Means SMOTE algorithm with the following parameters:

- Sampling Strategy = 0.333 (#minority instances / #majority instances)
- K-neighbours = 5
- random_state = 1234
- Kmeans_estimator = MiniBatchKMeans (random_state = None)
- cluster_balance_threshold = auto

The sampling strategy was fixed in order to reach a ratio of 25% of minority instances in the training set.
Note that the resampling technique has been applied only on the training set. The purpose will be to evaluate, on the test set, changes in F1-score between the model trained with K-means SMOTE and the model trained without.

- **ANN –**

We initiated with a fully connected feed-forward neural network with 1 hidden layer containing 5 neurons and an output layer with a single neuron. Gradually, we increased the depth and the width of the network in order to enhance the prediction performance. We also added regularization and tuning parameters (such as batch_size = 32, epochs = 10, validation_split = 0.2, verbose = 2, shuffle = True) on the final model.

We used ‘relu’ activation function for hidden layers and ‘sigmoid’ activation function for the output layer since the output is a single binary classifier.

Figure 2 illustrates the architecture of the ANN.

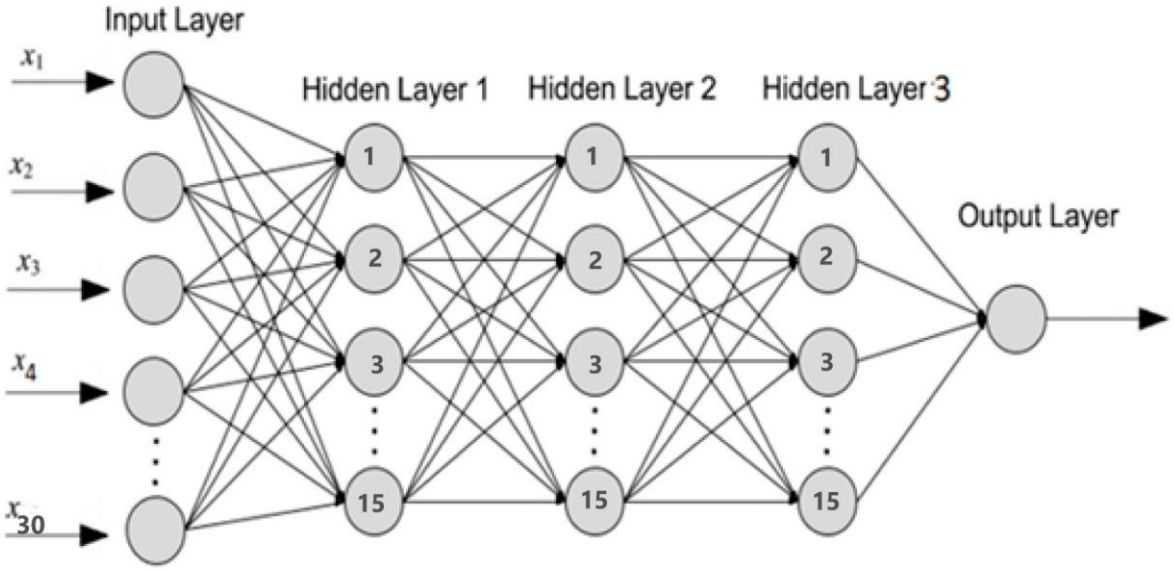


Figure 2: ANN architecture

- **Performance Metrics –**

When the True Positive Rate (TPR) and True Negative Rate (TNR) are important, Accuracy is a good performance metric. But when False Negative Rate (FNR) and False Positive Rate (FPR) are crucial, F1-Score is a better metric.

“The F1-score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal.”[3]

The formula for the F1 score is:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

where,

$$\text{the precision is: } \frac{\text{true positives}}{\text{false positives} + \text{true positives}}$$

$$\text{and the recall is } \frac{\text{true positives}}{\text{false negatives} + \text{true positives}}$$

[3] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

Due to the imbalance nature of the dataset, the accuracy metric is not appropriate in this case. That is why we chose to focus on the f1-score which accounts for false positive rate and false negative rate.
Moreover, AUC - ROC curve is one such performance measurement used in classification problem. It defines the degree of separability. It signifies how much a model is capable of distinguishing between classes. In other words, higher AUC indicates the model is better at predicting 0s and 1s exactly as it is. And when AUC is 0.5, it means model has no class separation capacity whatsoever.

5. RESULTS

The following is the table specifying the minority and majority class instances before and after applying K-Means SMOTE –

	Number of instances	
	Class 0	Class 1
Without KMS	227451	394
With KMS (ratio=0.333)	227451	75741

Table 1: K-Means SMOTE outcome

The following table mentions the change in f1 score and accuracy score with ANN (with/o K-Means SMOTE) –

Model	Parameters	F1-Score	Accuracy
Model1	(30,5,1-NA)	0.00	0.99
Model2	(30,5,1-0.333)	0.77	0.99
Model3	(30,10,1-NA)	0.00	0.99
Model4	(30,10,1-0.333)	0.79	0.99
Model5	(30,5,5,1-NA)	0.00	0.99
Model6	(30,5,5,1-0.333)	0.77	0.99
Model7	(30,10,10,1-NA)	0.76	0.99
Model8	(30,10,10,1-0.333)	0.77	0.99
Model9	(30,15,15,1-NA)	0.78	0.99
Model10	(30,15,15,1-0.333)	0.77	0.99
Model11	(30,15,15,15,1-NA)	0.79	0.99
Model12	(30,15,15,15,1-0.333)	0.76	0.99
Model13	(30,15,15,15,1-0.538)	0.78	0.99
Model14	(30,15,15,15,1-0.176)	0.77	0.99

Table 2: Different models performance

Where, Parameters is a list containing the structure of the model,

E.g.: for ANN with 0.5 K-Means SMOTE ratio with hidden layer =1, the parameter list is (30,5,1-0.5)

The two best models are highlighted in red in the above table.

First, we can observe that ANN with 5 neurons per hidden layer does not perform well in terms of f1 score; though ANN with K-Means resampling with the same parameters generates almost the best f1 score (77%).

Then, when ANN itself is well-tuned (Model 11 with parameters: batch_size = 32, epochs = 10, validation_split = 0.2, verbose = 2, shuffle = True), we reach the best f1 score(79%) and the KMS does not increase the prediction performance anymore.

Finally, once we attain an acceptable structure for ANN, exploring different sampling strategy does not provide remarkable increase in f1-score (Model 12, 13 and 14).

The ROC curve for one of the best models (Model 11) is shown in the following figure. Area under the ROC-curve makes a trade off between the True positive rate and the false positive rate. Since the curve here is near the left upper corner, we can say that the prediction performance of the model is good.

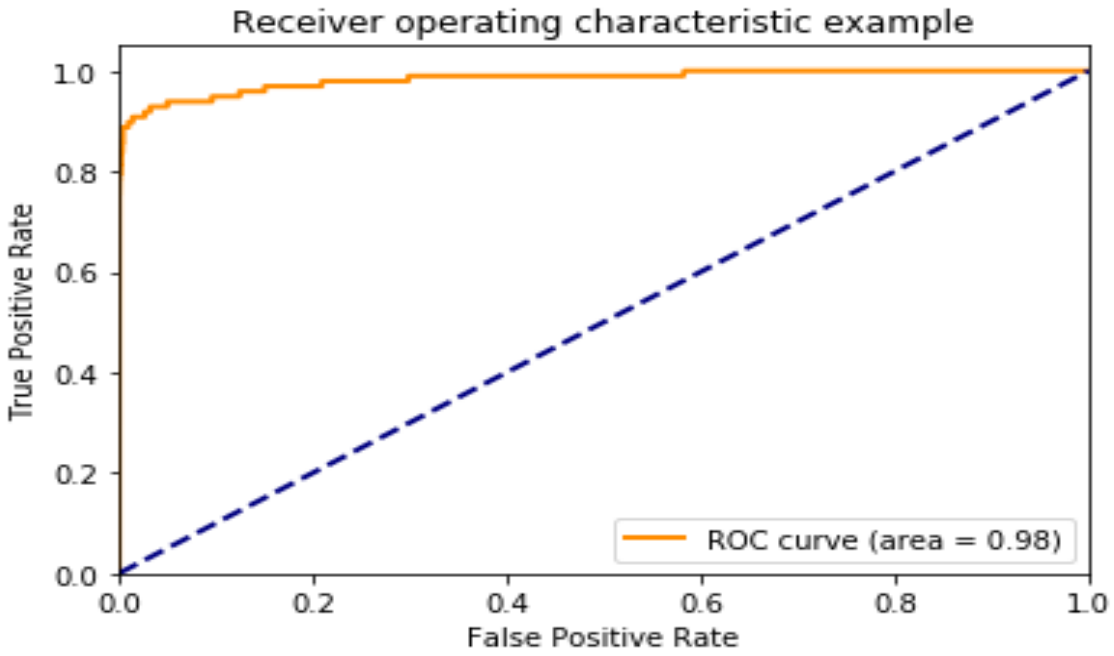


Figure 3: ROC curve

In correspondence, the above model with higher AUC can distinguish between credit card fraud and no fraud.

6. DISCUSSION & CONCLUSION

In this project, we targeted to evaluate the prediction performance of fraudulent Credit Card transactions with a feed forward neural network on a highly imbalanced dataset. To overcome the imbalance nature of the dataset, we have used the K-Means SMOTE resampling technique and compared performance according to the f1 score. We discovered that an ANN without resampling technique required a greater number of parameters (like hidden layers) to reach approximately the same performance as an ANN with K-Means SMOTE resampling technique and less parameters. The final model in the first case was an ANN with 3 hidden layers and 15 neurons per layer which provided an f1 score of 0.79. In the second case, the final model was an ANN with K-means SMOTE consisting of 1 hidden layer and 15 neurons yielded an f1-score of 0.79 as well.

Since in real world, we don't want a model to misclassify a fraudulent transaction, therefore with much more time, we could go further to minimize the False Negative rate by tuning the hyperparameters of the model and testing many other resampling techniques like Gaussian Mixture Model instead of K-Means for clustering step in the process of resampling.

Also, here since most of the features were masked due to confidentiality issues, finding an appropriate link between the target variables and features would have probably enabled us to increase the model performance.

Finally, it would be feasible to explore different classification algorithm to find out which ones suit the best to this imbalance nature of Machine Learning Problem.

REFERENCES

Commission, F. T. (2019). Consume Senttinel Network. 5-6.

Last, F., Douzas, G., & Bacao, F. (2017). Oversampling for Imbalanced Learning Based on K-Means and SMOTE. *arXiv preprint arXiv:1711.00837*.

Maes, S., Tuyls, K., Vanschoenwinkel, B., & Manderick, B. (2002). *Credit card fraud detection using Bayesian and neural networks.* Paper presented at the Proceedings of the 1st international naiso congress on neuro fuzzy technologies.

Raj, S. B. E., & Portia, A. A. (2011). *Analysis on credit card fraud detection methods.* Paper presented at the 2011 International Conference on Computer, Communication and Electrical Technology (ICCCET).

Sahin, Y., & Duman, E. (2011). *Detecting credit card fraud by ANN and logistic regression.* Paper presented at the 2011 International Symposium on Innovations in Intelligent Systems and Applications.