Machine Learning Engineer Nanodegree **Luca Maiano**
Capstone Proposal Date: September 17th, 2018

Domain Background

As reported by BusinessInsider [1], last year, CreditCards.com found that credit card fraud was on the rise. Both number of frauds and types of credit card scams are more and more. These numbers seem unfortunately destined to grow even more in the future. For this reason, it is important to find a way to automatically recognise anomalies. A lot of research has been done in order to find a solution to this problem. References [6] and [7] propose solutions using Artificial Neural Networks.

Problem Statement

The aim of the fraud detection system is to detect fraud accurately and before fraud is committed. The goal is to detect least and accurate false fraud detection. The most commonly techniques used fraud detection methods are Nave Bayes (NB), Support Vector Machines (SVM) and K-Nearest Neighbor algorithms (KNN). Each transaction has a set of unique features, such as the value of the transaction, the time at which it occured, issuer and recipient, an id and other sensitive data. The problem will be structured as a classification problem, where each transaction could be classified as "Normal" or "Fraud".

Datasets and Inputs

In order to reproduce this kind of problem, I found a useful dataset available on Kaggle [4]. The datasets contains 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 and 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. Due to confidentiality issues, the authors 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, this feature can be used for example-dependant cost-senstive learning. Feature "Class" is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Solution Statement

Autoencoders are a particular class of neural networks that taken an input, "compress" that input down to core features and tries to reconstruct the original input from this squeezed representation. This kind of approach, is used for applications in which we want to recognise anomalies or "distortions" on data.

With this project I want to experiment with autoencoders in order to detect frauds and measure its results. The aim of this project is to measure the quality of this technique to make a comparison with others. The reason why I choose this model is that I want to play a little bit with an "alternative" and (maybe) wrong model, to measure how much does it differ from

most used implementations. The huge class imbalance is likely to cause huge problems for an autoencoder if I don't use some very clever data augmentation. Perhaps by integrating an adversarial factor into autoencoders (on the encoder end), results could be good enough to be compared with "standars" solutions.

Benchmark Model

In order to measure the quality of this experiment and to evaluate the results, we need some reference model. It seems really difficult to find others that conducted and shared the result of this kind of analysis. After some research, I found two other solutions on the exact same dataset that I selected. Reference [5] recognise frauds using logistic regression with 93.5% of accuracy while reference [6] uses KNN and Naive Bayes classifiers with 99.8% and 97.7% accuracy respectively. The best way to evaluate this project will be comparing the ROC curve that I will obtain with the ones below on Figure 1.

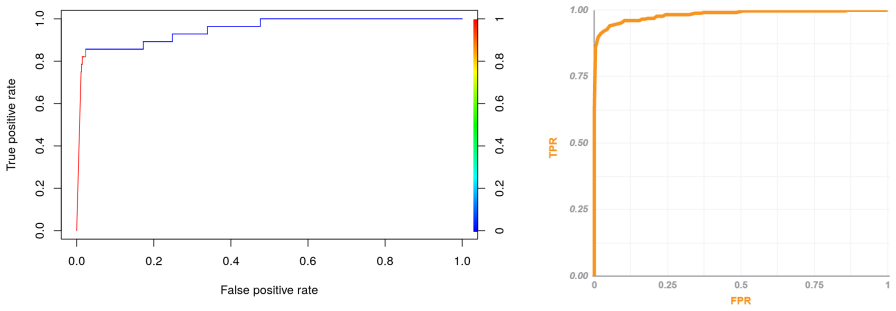

Figure 1: ROC curves from reference [5] and [6] respectively.

I will use those results as benchmark models. I do not expect to overcome the results achieved with KNN or Naive Bayes, but I think that the autoencoders will at least achieve the same results of model [5].

Evaluation Metrics

I will try to optimise the parameters of the Autoencoder in such way that the reconstruction error is minimised. ROC and Precision-Recall curves will be used in order to evaluate the results. A confusion matrix will help me to evaluate the results and compare them to the benchmark references.

Project Design

The project will be developed following those steps:

1. I will conduct at first an exploratory analysis to have a better understanding of the data;
2. the neural network will be trained and fixed with the right parameters;
3. at the end I will evaluate the results and I will compare them to the benchmark references.

Notice that in the dataset that I have chosen, "Time" is the seconds elapsed between each transaction and the first transaction in the dataset. It is unknown during what time of the day the transactions actually began. Then, the column can at most inform us how close the transactions were made in between 2 fraudulent ones. Since I want to predict frauds regardless of transaction time, this feature will be dropped earlier. Notice that time and amount have very different magnitudes in the dataset, which will likely result in the large magnitude value "washing out" the small magnitude value. It would be better to scale the data to similar magnitudes. Most of the data result from the product of a PCA analysis. I will do the same to the "amount" column (remember that time will be dropped!).

I need to face an other problem also. Fraudulent transactions are significantly lower than normal healthy transactions i.e. accounting it to around 1 or 2% of the total number of observations. With the autoencoder I will generate data for augmenting the dataset.

References

[1] Hillary Hoffower, BusinessInsider - *There's a good chance you're a victim of credit card scams and you don't even know it here's what to do.*

[2] Application of Credit Card Fraud Detection: Based on Bagging Ensemble Classifier, by Masoumeh Zareapoora and PouryaShamsolmoali.

[3] Credit Card Fraud Detection: A case study, by Ayushi Agrawal, Shiv Kumar and Amit Kumar Mishra.

[4] Credit Card Fraud Detection - Kaggle. Available at: https://www.kaggle.com/mlg-ulb/creditcardfraud.

[5] Case Study: How to Implement Credit Card Fraud Detection Using Java and Apache Spark. Available at: https://www.romexsoft.com/blog/implement-credit-card-fraud-detection/

[6] Credit Card Fraud Detection via KNN and Naive Bayes classifiers. Available at: https://www.kaggle.com/yuridias/credit-card-fraud-detection-knn-naive-bayes

[7] Neural data mining for credit card fraud detection. Available at: https://ieeexplore.ieee.org/abstract/document/809773/.

[8] Credit card fraud detection with a neural-network. Available at: https://ieeexplore.ieee.org/abstract/document/323314/.