**An Analysis of Classification Algorithms for Credit Card Fraud Detection Based on Time Complexity and Classification Accuracy**



**STUDENT NAME:** Gearoid Lacey

**STUDENT NUMBER:** C00183380

**COURSE NAME:** Masters in Data Science

**SUBJECT:** Data Analytics

**DEPARTMENT:** Department of Computing and Networking

**COURSE CODE:** CWS07

**DATE OF SUBMISSION:**

Table of Contents

[**1.** **Introduction** 3](#_Toc512268400)

[**2.** **Analysis of Literature** 5](#_Toc512268401)

[**3.** **Research Methodology** 7](#_Toc512268402)

[3.1 Dataset 7](#_Toc512268403)

[3.2 Exploratory Data Analysis 7](#_Toc512268404)

[**4.** **Algorithm Implementations** 11](#_Toc512268405)

[4.1 K Nearest Neighbour 11](#_Toc512268406)

[4.2 Logistic Regression 11](#_Toc512268407)

[4.3 Random Forest 12](#_Toc512268408)

[4.4 Cross Validation 13](#_Toc512268409)

[4.5 Analysis of Algorithms 13](#_Toc512268410)

[4.5.1 Evaluation Metrics 14](#_Toc512268411)

[4.5.2 Analysis of Results 14](#_Toc512268412)

[4.5.2.1 Algorithm Accuracy 15](#_Toc512268413)

[4.5.2.2 Algorithm Training Time 15](#_Toc512268414)

[4.6 Conclusions 16](#_Toc512268415)

[**References** 17](#_Toc512268416)

# **Introduction**

The purpose of this document is to analyse the effectiveness of three machine learning algorithms for credit card fraud classification. The analysis of these algorithms will focus on time complexity and classification accuracy. In this document a fraudulent transaction is any transaction made using illicit methods such as counterfeiting or stealing credit cards for example.

Advances in computing and networking have led to a rise in the occurrence of credit card fraud (Quah and Sriganesh, 2008). It is evident that the increase in online banking and online shopping has also led to an increase in credit card fraud. According to the Banking & Payments Federation of Ireland (BPFI) the total number of online and mobile banking payments in 2014 was approximately 64 million (Banking & Payments Federation of Ireland, 2015). In comparison, there were 44.9 million payments made through online banking services in 2012.

There are numerous ways fraud can happen. One instance where fraud can occur is through the use of social engineering (Atkins and Huang, 2013). Social engineering is the process of manipulating individuals into performing certain actions through deceit. Social engineering can take form in a number of different approaches. One example of this is when the fraudster calls the victim and claims to be someone else generally from a large reputable organisation and ascertains the victims payment information that way.

(Atkins and Huang, 2013) outline another form of social engineering called dumpster diving whereby the fraudster ciphers through the individual or organisations rubbish to extract information from mail for example. They also describe the situation where the fraudster gains entry to a premises and utilises the data from information boards. These are some of the simplest approaches to gaining information for deceitful purposes.

Another way credit card fraud can occur is through counterfeit cards (Ghosh and Reilly, 1994). The authors note that due to the difficulty in counterfeiting credit cards, this approach is more systematic. The authors also document a different systematic approach whereby the fraudster uses fraudulent credit card applications to gain the victims personal details which they then use to request a credit card.

Therefore, the purpose of this document is to analyse, in terms of accuracy and time complexity, the effectiveness of three machine learning algorithms for credit card fraud classification. The proposed algorithms to be implemented are K Nearest Neighbour (KNN), Logistic Regression (LR) and Random Forest (RF).

# **Analysis of Literature**

Many financial institutions now use data mining and machine learning to assist their existing measures for fraud the detection. (Ghosh and Reilly, 1994) state that financial institutions have, in the past, detected fraud simply by noting discrepancies in the fraudsters handwriting. In cases where the fraudster has access to a credit card these institutions rely on spotting transaction irregularities. Due to the increasing usage of online banking and payments as outlined previously, these financial institutions rely heavily on machine learning algorithms to alert them to any unusual transaction patterns.

(Ghosh and Reilly, 1994) designed a system which uses a P-RCE Artificial Neural Network (ANN) to classify fraudulent accounts for Mellon Bank. This ANN is a three layer feed forward neural network that makes two passes of the training data and then produces a fraud score.

Their system combined 50 variables to produce 20 features that were then passed to the ANN for account classification. As there were fewer fraudulent transactions than valid transactions, the authors ensure that all fraudulent transactions were included in the sample for training the ANN. When tested using two million transactions, the system vastly improved the fraud detection measures currently in place in Mellon Bank by only requiring a review of 50 accounts per day as opposed to reviewing 750.

Although fraudulent accounts and transactions can largely be classified by data mining and machine learning techniques, there is still a requirement for human verification (Quah and Sriganesh, 2008). Instead of reviewing hundreds or thousands of accounts or transactions, banking personnel only need to verify the accounts or transactions that are flagged as fraudulent by their data mining or machine learning model making the whole process more efficient.

(Quah and Sriganesh, 2008) outline a system for real time fraud detection that uses a neural network to cluster accounts as genuine or fraudulent. In essence the authors have implemented a clustering algorithm such as KNN using a form of ANN that allows for more dynamic relationships in the data.

(Ao and International Association of Engineers, 2011) provide an analysis of three machine learning algorithms for fraud detection. The algorithms analysed are Naïve Bayes, KNN and LR. The study was carried out on a dataset with approximately 280,000 transactions that was highly unbalanced. It is worth noting that the KNN algorithm was only .3% less accurate than Naïve Bayes. As the authors used several other methods to evaluate the effectiveness of each algorithm, KNN was actually the optimal algorithm. Noticeably LR provided quite a poor classification accuracy of approximately 54%.

Similarly (Shen et al., 2007) provide an analysis of three machine learning algorithms for credit card fraud detection. In contrast to the study done by (Ao and International Association of Engineers, 2011), the authors analyse decision trees, neural networks and LR also. In this study neural networks provided the best performance followed by LR and decision trees. Again, the dataset in this study suffers from a highly unbalanced set of classifications as only 0.07% of the training data was labelled as fraudulent.

(Duman et al., 2013) developed a fraud detection system for a Turkish bank which detects 97% of fraudulent transactions. In addition to the previously mentioned studies, this too suffered from highly unbalanced data where the authors had 978 fraudulent transactions and 22 million valid transactions. Therefore, the authors had to use a stratified sample of the data that included all the fraudulent transactions and only a fraction of the valid transactions. The authors tested multiple existing data mining algorithms as well as some ensembles of these algorithms to determine the best solution for fraud detection. From this analysis, the authors determined that the Migrating Birds Optimisation algorithm was the best option.

(Mahmud et al., 2016) provide a comprehensive overview of different types of machine learning algorithm to deduce the best approach to correctly detecting fraudulent transactions. The authors found that meta algorithms such as RotationForest, Bagging and RandomCommittee provided the highest accuracies for fraud detection. The accuracies for these algorithms ranged from 98.13% to 98.25%.

(Bhattacharyya et al., 2011) present an analysis of three machine learning algorithms for credit card fraud classification. The algorithms analysed by the authors are LR, Support Vector Machines and Random Forest (RF). Again, like previous studies, this also suffers from the effects of a highly unbalanced dataset. The authors tested each algorithm on samples with different rates of fraudulent transactions. All three algorithms reported impressive accuracies with RF providing the highest accuracies overall.

# **Research Methodology**

## 3.1 Dataset

The dataset for this analysis was sourced from Kaggle (Machine Learning Group - ULB, n.d.).

The dataset contains 284,807 transactions, 492 of which are fraudulent. The dataset consists of thirty-one features in total. Twenty-eight of these features are the result of a principal component analysis transformation and are labelled from V1 to V28. The other variables consist of time in seconds from the first transaction, the amount of the transaction and the class of the transaction, where 0 represents a valid transaction and 1 represents a fraudulent transaction.

## 3.2 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is one of the first steps to undertake when working with large quantities of data. More specifically the author aims to visualise the data to determine any relationships or outliers within the data itself. To perform EDA the author intends to use the R programming language (R: The R Project for Statistical Computing, n.d.). R was primarily designed for statistical analysis and provides a wide range of packages which make visualising and modelling the data an efficient process. Therefore each of the algorithms will be implemented and analysed in R. It is worth noting that the analysis could also have been undertaken in Python which is equally as efficient at performing statistical analysis.

Once the dataset was loaded into the R environment, a plot was generated with Time on the x axis, the transaction amount on the y axis and then each observation was coloured based on its class, as shown in figure 1. As you can see from the plot, it is evident that the data set is highly unbalanced as it is quite difficult to locate any green data points which indicate a fraudulent transaction. In producing this plot, the alpha level which refers to the transparency of the data point was dynamically set based on the class of the observation which enabled the author to roughly locate fraudulent transactions beneath all the valid transaction data points. When making the plot in figure 1, the author was alerted that one row of data was not considered for the plot as it contained a null value.



**Figure 1.** Transaction Amount vs. Time

Following this the author retrieved summary statistics for the fraudulent transactions by filtering the raw data so it only included the fraudulent transactions. The fraudulent transaction amounts had a mean of €122.21 and a maximum of €2125.87. Noticeably the minimum fraudulent transaction amount was €0.00. This could be an error in the dataset or the result of a fraudulent transaction that was less than 1 cent. Based on this, the fraudulent data was then filtered to see how many of the transactions were less than fifty cent. This resulted in thirty four transactions being classified as fraudulent where the transaction amount was fifty cent or less. This seemed quite high number of transactions for quite an insignificant amount of money. This prompted the author to look at how many fraudulent transactions had a value of ten cent or less which resulted in thirty two transactions.

Considering the maximum value of the fraudulent transactions and the number of transactions that were less than fifty cent, the values of the fraudulent transactions were plotted against the time of the transaction. Figure 2 shows the dispersion of the fraudulent transactions.



**Figure 2.** Dispersion of fraudulent transactions

As you can see in Figure 2, there is no distinct relationship between the fraudulent transactions and the time. It is also noticeable that the majority of the fraudulent transactions are below €250 and more specifically they are closer to zero. To see exactly where these points lie, two plots are created which essentially zoom in on the y axis. The first plot looks at the all the transactions that have a value of less than €10. Again there appears to be no obvious relationship in this plot but it is noticeable that many of the data points have transaction amounts between €0 and €2.50. The second plot looks at all the transactions above €10. Similarly, there appears to be no obvious relationship in the data.

The second plot also shows that the data points become even more dispersed once the transaction amount becomes greater than €500. To confirm that the relationship between all fraudulent transactions and time is not linear the *cor()* function was used to get the correlation between both. This resulted in a value of 0.04873 which confirms the lack of a relationship between the data.



**Figure 3.** Dispersion of transactions with limits on the y axis.

As the majority of the variables are the result of a principle component analysis transformation, the EDA is quite limited as there is no description of what variables V1 to V28 represent making them much harder to work with. This is in direct contrast to the studies outlined previously as the authors had a more thorough description of the datasets they were working with.

# **Algorithm Implementations**

As previously stated the algorithms that will be used as part of this analysis are K Nearest Neighbour, Logistic Regression and Random Forest. A description of each algorithm is provided in the following sections. Algorithms are fundamentally judged on execution time and accuracy which is the primary aim of the analysis of these algorithms.

## 4.1 K Nearest Neighbour

KNN is a popular classification algorithm due its simplicity in implementation and comprehension. These were the two main reasons for choosing this algorithm. KNN operates by providing the algorithm a training set where each observation has a classification label assigned to it. Where most models require training, KNN simply stores the training set and then finds the closest K neighbours to an unseen observation. The algorithm determines the class of the new observation by finding the most popular class in the K nearest neighbours. For this reason the parameter K should always be set to an odd number to avoid a situation where the number of classifications for the K nearest neighbours is the same.

KNN uses a distance formula to determine the closest K neighbours. By default the algorithm uses the Euclidean distance formula which finds the straight line distance between two points.

Therefore the main parameters for this model are the number of neighbours to look for (K) and the distance formula to use. Low values for K tend to work quite well. When K = 1 the algorithm tends to over fit the data so K = 3 or K = 5 are better selections.

## 4.2 Logistic Regression

LR, despite its name, is a classification algorithm. The reason LR is often considered a regression algorithm is it returns the probability of an observation belonging to a particular class. As the probability is used to determine the classification, it is therefore considered a classification algorithm. The basic LR algorithm is best used for binary classification but it can be extended for multinomial regression. The logistic function is as follows (James et al., 2013):

where β0 represents the log odds and B1 represents the coefficient for the predictor X. LR uses maximum likelihood to determine the best coefficients for classification. In essence, maximum likelihood aims to find the optimal coefficients that will allow the logistic function to produce a predicted class probability for an observation that matches the observations real classification.

The logistic function produces and an S shaped curve with values ranging from 0 to 1 on the y axis when plotted. The reason the values range from 0 to 1 is due to the previously mentioned point, that logistic regression returns a class probability as opposed to a class label. If the coefficient of a variable is positive then increasing that variable causes an increase in the output probability. If the coefficient is negative then an increase in that variable causes a decrease in the probability.

An important aspect of machine learning algorithms is regularisation. Regularisation is the process of restricting the model to avoid overfitting of unseen data. There are two types of regularisation that can be used with LR, L1 regularisation and L2 regularisation. By default LR uses L2 regularisation which forces the coefficients of each feature to be as close to 0 as possible. This means that no particular feature has a significant effect on the outcome. LR can also be used with L1 regularisation which also forces the coefficients to be close to 0 also. L1 regularisation allows some of the coefficient to be exactly 0, meaning they are ignored in the predication process. Therefore L1 regularisation is a form of feature selection.

## 4.3 Random Forest

RF is an ensemble algorithm that improves upon the Bagging machine learning algorithm (James et al., 2013). The Bagging algorithm functions by taking a certain amount of bootstrap samples from the dataset and then builds a decision tree for each bootstrap sample. Bootstrapping is a resampling method where samples are generated at random with replacement from the dataset. The main difference between both algorithms is the number of features the tree’s utilise. In Bagging each tree is built using all the available features in the sample. In comparison, RF randomly selects a subset of these features for each tree, and then builds a tree on these features only. This randomness in tree generation provides a significant improvement as in a scenario where your dataset contains a strong predictor, Bagging will likely generate very similar tree’s for each sample as most will start by splitting on the strongest predictor. As RF is random in terms of the features it looks at, it will avoid this scenario.

## 4.4 Cross Validation

Cross validation is the process of repeatedly training a model on random sets of the data (James *et al.*, 2013). There are multiple forms of validation for machine learning algorithms such as the validation set approach where the data is split into a training and test set, usually in the ratio of 75:25. Another validation approach is Leave One Out Cross Validation (LOOCV). LOOCV operates by using one observation from the data as a test observation. The other n -1 observations are then used to train the model. This is implanted n times so each point in the data is used as a test observation. The model is then evaluated based on the Mean Standard Error (MSE) for each of the trained models. The main issue with this model is it is computationally expensive as the model has to be trained n times. Therefore it will not be used to validate the models in this paper.

The validation method that will be used for each of the previously described algorithms will be repeated K-Fold Cross Validation. This validation method functions by randomly dividing the data into K equally sized segments. The first segment of data is then used as a test set and the rest is used as the training set. The model is then trained and tested using the training and test sets. This approach is then repeated for a specified number of iterations. By default K is usually set to 10.

Repeated K-Fold Cross Validation is therefore used to find the best model that can generalise well on unseen data. When used with KNN, repeated K-Fold Cross Validation finds the optimal value for k (the number of nearest neighbours).

## 4.5 Analysis of Algorithms

As a form of experiment control, all three algorithms were trained and tested on the same sample of data. This sample consisted of approximately 28,000 rows of data. It is worth noting that this sample was in fact a stratified sample as it included all 492 fraudulent transactions. All three algorithms were initially trained and tested without using any form of cross validation to see how they would score initially. Following this each model was again trained and tested using repeated K-Fold Cross Validation where K was set to ten and the cross validation was repeated five times. Interestingly each algorithm worked extremely well without cross validation with some not even improving their accuracy score on the cross validated model. The author suspects that the cross validated models did not improve the accuracy of the standard models as the original accuracies were already extremely high. Table 1 displays a comparison of the results from training and testing the basic and cross validated models for each of the algorithms.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Algorithm | Basic Model Accuracy | Basic Model Training Time | Cross Validated Model Accuracy | Cross Validated Model Training Time | Basic Model on Independent Test set | Cross Validated Model on Independent Test set |
| K Nearest Neighbour | 99.94% | 174(s) | 99.94% | ~89(s) | 99.53% | 99.54% |
| Logistic Regression | 99.92% | 15(s) | 99.92% | ~21(s) | 99.53% | 99.53% |
| Random Forest | 99.97% | 5.36(s) | 99.96% | ~96(s) | 99.81% | 99.76% |

**Table 1.** Comparison of Algorithms on Validation Set

Note, by default RF creates 500 tree’s and each tree looks at a random sample of √N (where N=29 in this scenario) features which leads to the increase in training time. To avoid an extremely large training time for the cross validated model it was decided that the number of tree’s created in both models be set to 100. Although there is a significant increase in training time, there is also a marginal increase in accuracy.

## Evaluation Metrics

Traditional accuracy scores are generally considered quite a poor evaluation metric for classification problems as they only tell you the percentage of observations that were correctly classified (C. Müller and Guido, 2016) (Géron, 2017). There are multiple metrics for evaluating classification models such as confusion matrices, precision, recall (True Positive Rate), F1-score and Area Under the ROC (AUROC) curve for example. Each metric provides a different evaluation viewpoint. For the purposes of this experiment confusion matrices (which are used to calculate accuracies) were the primary source of evaluation as they give a detailed description of the correctly and incorrectly classified observations in the form of True Positives, True Negatives, False Positives and False Negatives.

## Analysis of Results

## Algorithm Accuracy

As previously alluded to, the accuracy metric is based on the results of a confusion matrix. Therefore, the confusion matrices from the output of the accompanying .Rmd file, will be used to calculate the accuracies by adding the number of True Positives and True Negatives and dividing by the number of observations in the validation set.

From Table 1 we can see that the calculated accuracies for all three basic models are quite similar, but the RF model is marginally more accurate. All three models are generally quite good even with default parameters as is the case here. This could also be caused by the relatively small sample size of 28,000 observations which is then divided into portions of 50%, 25% and 25% for training, validation and test sets.

In terms of accuracies for the cross validated models there is no accuracy change for KNN or LR, but RF decreases by .01%. Despite this decrease, RF remains the best overall model when validated on the independent test set. The reason for small or no increase in accuracy is primarily to do with the basic accuracies being extremely good. In general, as model accuracies get close to 100% the increases tend to be quite small.

## Algorithm Training Time

Again, from Table 1 we can see that all three models are relatively quick to train with the basic RF significantly faster than the rest. The reason RF is so fast is possibly due to the fact that of the 10 tree’s the model creates, each tree only looks at a small subset of the features. Generally linear models are often significantly faster than non-linear models which is why KNN is the slowest.

When looking at the cross validated models LR becomes significantly faster than the other two models as it’s a generalised linear model. Noticeably RF is now the worst in terms of training time. The reason for this is RF builds 10 trees for each round of the cross validation. This can become quite time consuming especially when you have a large number of features. One way to decrease the time complexity would be to limit the depth of each tree but this could cause adverse effects on the accuracy.

As previously mentioned, KNN does not perform well in terms of training time on large quantities of data with a large number of features. Hence, in both the basic and cross validated models the training time is quite high.

## GDPR Implications

As the data used in this project consists of real credit card transactions from a European bank, there is a possibility that GDPR laws could easily be violated. Fortunately, it is worth noting that there is no way to personally identify any person to a particular transaction due to the fact that the majority of the features have been transformed using a PCA transformation.

## Conclusions

The data in this dataset consisted entirely of homogenous data, meaning an artificial neural network would also be suited to the problem. Despite this, all three algorithms used in this experiment worked extremely well in this scenario. Although the training sample was relatively small due to hardware restrictions, it is expected that all three models would report lower scores if they had been trained in their current state on a larger sample of data. In a scenario where more data was being used in training both feature engineering and model tuning would make the models generalise to unseen data better.

As the aim of this document was to analyse the algorithms in terms of accuracy and time complexity it must be stated that there is no one algorithm that can be considered the best. When working on a project it is critical that a priority between time complexity and accuracy be chosen. Following this an algorithm can be implemented that suits the outlined priorities. In terms of credit card fraud classification, it is expected that a higher rate of false positives and lower rate of false negatives would be ideal. Therefore, a high True Positive Rate would be critical.

# **References**

Ao, S. I. and International Association of Engineers (eds) (2011) *The 2011 IAENG International Conference on Artificial Intelligence and Applications, the 2011 IAENG International Conference on Bioinformatics, the 2011 IAENG International Conference on Computer Science, the 2011 IAENG International Conference on Data Mining and Applications, the 2011 IAENG International Conference on Imaging Engineering, the 2011 IAENG International Conference on Communication Systems and Applications, the 2011 IAENG International Conference on Internet Computing and Web Services, the 2011 IAENG International Conference on Software Engineering*. Hong Kong: IAENG (International MultiConference of Engineers and Computer Scientists, IMECS 2011, 16-18 March, 2011, the Royal Garden Hotel, Kowloon, Hong Kong / [Eds.: S. I. Ao ...] ; Vol. 1).

Atkins, B. and Huang, W. (2013) ‘A Study of Social Engineering in Online Frauds’, *Open Journal of Social Sciences*, 01(03), pp. 23–32. doi: 10.4236/jss.2013.13004.

Banking & Payments Federation of Ireland (2015) *Online and Mobile Banking Report Full Year 2014 and Q4 2014*. Banking & Payments Federation of Ireland, pp. 1–3. Available at: https://www.bpfi.ie/wp-content/uploads/2015/05/BPFI-Online-and-Mobile-Banking-Q4-2014-FINAL.pdf [Accessed: 21 February 2018].

Bhattacharyya, S. *et al.* (2011) ‘Data mining for credit card fraud: A comparative study’, *Decision Support Systems*, 50(3), pp. 602–613. doi: 10.1016/j.dss.2010.08.008.

C. Müller, A. and Guido, S. (2016) *Introduction to Machine Learning with Python. A Guide For Data Scientists.* First. O’Reilly.

Duman, E., Buyukkaya, A. and Elikucuk, I. (2013) ‘A Novel and Successful Credit Card Fraud Detection System Implemented in a Turkish Bank’, in. IEEE, pp. 162–171. doi: 10.1109/ICDMW.2013.168.

Géron, A. (2017) *Hands-On Machine Learning with Scikit-Learn & TensorFlow.* First. O’Reilly.

Ghosh, S. and Reilly, D. L. (1994) ‘Credit card fraud detection with a neural-network’, in *System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on*. IEEE, pp. 621–630.

Ireland, D. P. C. of (no date) *A Guide for Data Contollers - Data Protection Commissioner - Ireland*. Available at: https://www.dataprotection.ie/docs/A-Guide-for-Data-Contollers/y/696.htm [Accessed: 23 April 2018].

James, G. *et al.* (2013) *An Introduction to Statistical Learning*. New York, NY: Springer New York (Springer Texts in Statistics). doi: 10.1007/978-1-4614-7138-7.

Machine Learning Group - ULB (no date) *Credit Card Fraud Detection*, *Kaggle*. Available at: https://www.kaggle.com/mlg-ulb/creditcardfraud [Accessed: 24 February 2018].

Mahmud, M. S., Meesad, P. and Sodsee, S. (2016) ‘An evaluation of computational intelligence in credit card fraud detection’, in *Computer Science and Engineering Conference (ICSEC), 2016 International*. IEEE, pp. 1–6.

Quah, J. T. S. and Sriganesh, M. (2008) ‘Real-time credit card fraud detection using computational intelligence’, *Expert Systems with Applications*, 35(4), pp. 1721–1732. doi: 10.1016/j.eswa.2007.08.093.

*R: The R Project for Statistical Computing* (no date). Available at: https://www.r-project.org/ [Accessed: 24 February 2018].

Shen, A., Tong, R. and Deng, Y. (2007) ‘Application of classification models on credit card fraud detection’, in *Service Systems and Service Management, 2007 International Conference on*. IEEE, pp. 1–4.