Research Paper Business Analytics

Fraud Detection Using Machine Learning Techniques

Jurriaan Besenbruch

Supervised by
Prof. dr. Sandjai Bhulai

Vrije Universiteit Amsterdam
Faculty of Science
Business Analytics
De Boelelaan 1081a
1081 HV Amsterdam

August 20, 2018

Abstract

When using machine learning algorithms to detect financial fraud in general, or more specific credit card or similar fraud techniques, some flaws exist that make this hard to detect automatically. Most algorithms require a fair distribution between fraud and non-fraud cases. In real life, however, there no such balance between cases. In general, there is one majority class (non-fraud), while the minority class (fraud) happens less frequently. This paper uses simulated transaction data where the minority class occurs less than 1 out of 750 times.

Research Goal

Imbalanced data creates a significant drawback in classification performance with standard classifiers. The goal of this paper is to review several methods that could improve the classification performance in case of imbalanced data. The focus is on sampling techniques, distance-based methods, and deep learning. The first part of this paper consists of a literature study conducted on two related subjects. The first part of the research involves finding methods for each approach to handle imbalanced data. In subsequent research, performance metrics need to be evaluated, as accuracy and error rate are not sufficient to make any conclusion as long as the data remains imbalanced. The second part of the paper involves implementing the most interesting methods and metrics on a fraud dataset.

Modeling

The conducted research is divided into two separate parts: traditional sampling methods and deep learning. The first approach is based on traditional classifiers, such as random forest and logistic regression; the second approach utilizes deep neural networks, which are currently the center of many experiments in fraud detection. The sampling approach consists of three stages: data reduction, re-sampling, and classification. This approach strives to preprocess the data such that the classification performance of traditional classifiers improves. The deep-learning approach overcomes the problem of preprocessing the data and consists of two types of neural networks. The first is a classic feed-forward network, and the second model is an autoencoder network.

Table of Contents

1. Introduction

Financial fraud is a growing issue with wide-reaching consequences in the financial industry and daily life. It is increasing with the development of modern technology and communication methods. Fraud can reduce confidence in industry, destabilize economies, and affect people's cost of living. [1] In the past, fraud detection was performed manually by human analysts. As it can be really complicated to find fraudulent patterns by hand, which are often characterized by a large number of samples and multi-dimensional data, fraud detection systems are necessary to find these patterns. Financial fraud is an overarching term used to describe several varieties of fraud. The major types are: credit card fraud, securities and commodities fraud, financial statement fraud, insurance fraud, mortgage fraud and money laundering. This paper does not focus on a specific kind of fraud, but the used data is synthetic money transaction data that is similar to credit card data.

Several problems arise when analyzing financial transactions to detect fraud. The main problem is obtaining a legitimate financial dataset that can be used for research. The confidential nature of this information means that it is difficult to obtain an open dataset. To overcome this problem, some researchers have created a financial payment simulator that simulates mobile money transactions in such a way that the simulated data becomes similar to the original financial data [2]. It is possible to analyze financial data just like real-world data. Further discussion this simulated data can be found in *Chapter 3: Data Description and Exploration*. The second problem, and the focus of this paper, is related to the distribution of fraud data. Since the number of legal transactions is much higher than that of fraudulent ones, the data is skewed towards non-fraudulent observations. In other words, the data is (highly) imbalanced. Imbalanced data is a common problem in real-world classification applications such as the diagnosis of rare diseases, machine component failure analysis and fraud detection. Data is regarded as imbalanced when the number of instances of a certain class is significantly smaller than those of other classes. This is usually caused by rare events/abnormal conditions or by limitations in data collection. It is difficult to learn from imbalanced data is difficult since most machine learning methods require balanced data, to predict the minority class properly. Without balanced data, the machine learning algorithms are biased towards the majority class as they try to optimize the overall accuracy [3]. The class imbalance problem depends on multiple factors: the degree of class imbalance, the complexity of the concept represented by the data, and the overall size of the training set. Using sampling methods such as re-sampling, over-sampling and under-sampling could make the classifier less sensitive to class imbalances [4].

The purpose of this paper is to discuss and evaluate more advanced methods for addressing the problem of imbalanced data in machine learning applications for fraud detection. In the next chapter, the paper concentrates on explaining the problem with class imbalance in machine learning applications. In Chapter 3, the data which was gathered for this research is explored and prepared for analysis. In Chapter 4, several methods to balance imbalanced data are discussed. After these methods are reviewed, they are implemented and then evaluated in Chapter 5. And finally, in Chapter 6 concluding remarks are made.

2. Literature Study

Classification with imbalanced data can be a most difficult task. Solutions to the imbalance problem can be classified into three groups: sampling techniques, cost-sensitive techniques and one-class learning [5]. Sampling can be used to create a balanced set by under-sampling, over-sampling, or a combination of both. More advanced techniques describe methods for generate new data for the minority classes from existing data. Cost-sensitive learning (CSL) does not use any sampling technique to optimize the data. When CSL misclassifies an instance of the minority class, it gives a higher penalty than when an instance of the majority class is misclassified. From this point of view, it is more important for the machine learning algorithm to classify the minority class correctly. One-class learning attempts to determine whether an instance belongs to a particular class or not. During training, only data from a single class is available. In the next subsection, several methods are reviewed from the relevant literature.

Most machine learning algorithms try to minimize the overall error rate, but as fraud data is extremely skewed, this focus on reducing accuracy/error rate is not the optimal measure. In cases where 99% of the instances belong to one class, an error rate of 1% is still not reliable since all the instances from the minority class can be predicted incorrectly. In *Section 2.2: Performance Metrics* are several measures discussed that tries to give a more useful value to evaluate the model.

2.1. Sampling Methods/Other Techniques

There are several possible techniques for handling imbalanced data. Changing class distribution is a common technique that is performed at the data level. This technique tries to balance the imbalanced data by under- sampling the majority class, over-sampling the minority class, applying a combination of both methods, or employing selective sampling. The simplest sampling methods are random over-sampling and random under-sampling. Random over-sampling (ROS) duplicates minority class instances until a balanced distribution exists. The main disadvantage of this is the possibility of overfitting the classifier. The classifier works well on training data, but does not perform adequately on new data. The random under-sampling (RUS) technique also has a major drawback. When under-sampling is carried out, a substantial amount of information from the majority class can potentially be discarded when sampling occurs randomly. This can result in a loss of classification performance since the decision boundary between the classes becomes unclear. More appropriate methods use selective sampling, hybrid techniques (where under- and over-sampling is combined), ensemble methods, or skew-insensitive classifiers.

Over-Sampling Methods

To overcome the problem of overfitting, a new oversampling technique called "Synthetic Minority Over-sampling Technique" (SMOTE) was introduced by Chawla [6]. This sample method generates artificial minority class instances from existing ones, instead of duplicating existing instances from the data. Synthetic Minority Over-sampling Technique works in the feature space rather than the data space. The new instances are created by combining features of the target instance and its nearest neighbors. To create the new artificial minority class instance, SMOTE randomly selects one existing minority class sample m.

Next, the algorithm should find its k-nearest neighbors and should select at random one of the k samples, called n. Subsequently, it is necessary to calculate the difference between samples m and n and then multiply this with a random number between 0 and 1. After the resulting value is added to the feature vector of m, m and n form a line segment in the feature space [7]. Several methods have been developed to improve the original SMOTE algorithm. SMOTE-NC is a generalization of SMOTE designed for handling mixed data with continuous and nominal features. Two alternative methods based on SMOTE are borderline-SMOTE, proposed by Han [8], and safe-level SMOTE. With borderline and safe-level SMOTE, only the minority samples near the borderline and safe area, respectively, are over-sampled. Borderline instances are located where minority and majority class instances overlap; safe-level instances are located in homogeneous areas with minority class instances.

Motivated by the success of synthetic approaches to deal with the class imbalance problem, Haibo He [9] has proposed a new algorithm, ADASYN. This algorithm uses a weighted distribution for the minority class instances and assigns the respective weight based on the difficulty in learning that particular instance. Minority instances that are harder to learn requires more generated synthetic data. The ADASYN algorithm starts by calculating the number of artificial data instances that should be generated for the minority class: $G = \beta(m_l - m_s)$ where m_s and m_l correspond, respectively, to the number of minority and majority class samples. Therefore, $m_s \leq m_l$ and $m_s + m_l = m$. The parameter $\beta \in [0,1]$ defines the desired balance level after the data generation process. For each instance of $\boldsymbol{x}_i \in$ minorityclass, find the k-nearest neighbors based on the Euclidean distance, and calculate ratio r_i, which is defined as $r_i = \frac{\Delta_i}{k}$ for $i = 1,2,\dots,m_s$. In this formula, Δ_i represents the number of instances in the k-nearest neighbors of $\boldsymbol{x}_i$ that belongs to the majority class. Then normalize r_i with Equation (1).

$$\widehat{r_i} = \frac{r_i}{\sum_{i=1}^{m_s} r_i} \quad \text{where } r_i \text{ is a density distribution} \quad \sum_{i=1}^{m_s} \widehat{r_i} = 1 \tag{1}$$

After the ratios are normalized, the number of artificial instances that need to be generated should be calculated for each minority instance $\boldsymbol{x}_i$ by the formula $g_i = \widehat{r_i} \cdot G$. In order to generate g_i artificial instances for each minority instance $\boldsymbol{x}_i$, the algorithm should randomly choose one minority instance $\boldsymbol{x}_{zi}$ from the k-nearest neighbors for instance $\boldsymbol{x}_i$. Subsequently, the artificial data can be generated by Equation (2), where λ is a random number between 0 and 1 [9].

$$\boldsymbol{s}_i = \boldsymbol{x}_i + \lambda \cdot (\boldsymbol{x}_{zi} - \boldsymbol{x}_i) \tag{2}$$

Although artificial data generation is a successful approach to the imbalanced data problem, Barua [10] has shown that many existing over-sampling methods generate, in some cases, the wrong artificial minority samples. Wrong minority instances make it more difficult for the classifier to learn properly. In the same paper, another method has been proposed by Barua, namely the "majority weighted minority oversampling technique" (MWMOTE). The majority weighted minority oversampling technique tries to improve the selection of important instances and the quality of the artificial instances. This method identifies the hard-to-learn minority class instances first and assigns a weight based on the Euclidean distance from the nearest majority class instance. Subsequently, new artificial instances are generated using a clustering approach, based on the weighted minority class.

Under-Sampling Methods

Since random under-sampling can have some disadvantages, several other techniques could prove to be more useful. Tomek links (T-links) remove unwanted majority class instances that are close to the minority class region until all minimally distanced, nearest neighbor pairs are from the same class. When two instances are a T-link, then one of them is a noisy instance or both instances are on the class boundaries. After those T-links are removed, the resulting dataset has a better separation between the minority and majority class. A T-link is defined as follows: let x and y be an instance from, respectively, the majority and minority class, and let $d(x, y)$ be their distance. The pair (x, y) is a T-link if there does not exist an instance such that $d(x, z) < d(x, y)$ or that $d(y, z) < d(x, y)$. An addition to T-links can be "condensed nearest neighbors" (CNN). A combination of both methods can be useful since T-links remove border line and noisy examples, while CNN removes redundant instances. In short, CNN tries to create a subset that is consistent with the original data, which means that the subset classifies the original dataset correctly. The idea behind CNN is to eliminate instances from the majority class that are distant from the borderline since those instances are easier to learn and less important for training the classifier. Harder-to-learn instances, the borderline samples, and the noisy samples are added to the CNN set as it is more likely to misclassify them. The drawback here is the sensitivity to noise.

A further two distance-based, under-sampling methods are "edited nearest neighbor" (ENN) and "neighborhood cleaning rule" (NCL). Moreover, Wilson [11] has proposed an edited k-NN rule to improve the 1-NN rule. The first algorithm (ENN) removes at least two of the closest instances from the other class. Only instances that are misclassified by their three-nearest neighbors are removed so as to overcome the problem of losing relevant information from the minority class. The second algorithm (NCL) uses the ENN algorithm to compute the three nearest neighbors for a certain instance. If a instance is from the majority class and is misclassified by its closest three neighbors, then that particular majority class instance is removed. When the instance is from the minority class and is misclassified by its three closest neighbors, the majority class instances around the minority class instance are removed.

Hybrid Methods

As an extension to the over- and under-sampling methods, hybrid sampling uses a combination of both sampling techniques to balance an imbalanced dataset. The major drawback of under-sampling is the information loss that occurs when instances are removed. When the data is highly imbalanced, the possible information loss is enormous. Unfortunately, over-sampling is not perfect since over-sampling with artificial or re-sampled instances can result in overfitting the classifier. To overcome the disadvantages and combine the best of both worlds, the hybrid approach makes it possible to over-sample the minority class without overfitting the data but also to under-sample the majority class without removing too many instances from the data. Common hybrid sampling methods are SMOTE+Tomek and SMOTE+ENN [7], but other hybrid strategies can also be applied by combining different techniques.

Ensemble Methods

Ensemble methods combine multiple-base learners to improve the classification performance over single classifier algorithms. Three ensemble methods are frequently discussed in literature on imbalanced data. These three methods are SMOTEBoost, EasyEnsemble and BalanceCascade; the last two ensembles were developed by Liu [12] in 2009. SMOTEBoost combines SMOTE with the AdaBoost M.2 algorithm. The main goal in this method is to sample the data before each round of boosting. The base learners focus more on difficult-to-classify instances, and the weights for the minority class increase during every round. Instead of oversampling the minority class with SMOTE, EasyEnsemble uses an under-sampling technique. EasyEnsemble randomly generates T multiple subsamples from the majority class N. The size of each subsample is equal to the size of the minority class P. The union of samples N_i and P is then used to train an AdaBoost ensemble. This final ensemble is a bagged ensemble that combines all the base learners from the boosted AdaBoost ensembles [13]. BalanceCascade uses another approach. It tries to delete well-considered (rather than random) majority class instances. Instead of operating in a parallel fashion like EasyEnsemble, BalanceCascade works sequentially in T rounds. In each round, i $(i = 1, \ldots, T)$ is a subsample N_i created from the majority class, and it is equal to the size of the minority class sample P. Subsequently, an ensemble H_i is trained from the union of N_i and P. Correct classified instances in N_i are removed from majority class set N. After T rounds, the final ensemble is formed by combining all base learners in all AdaBoost ensembles.

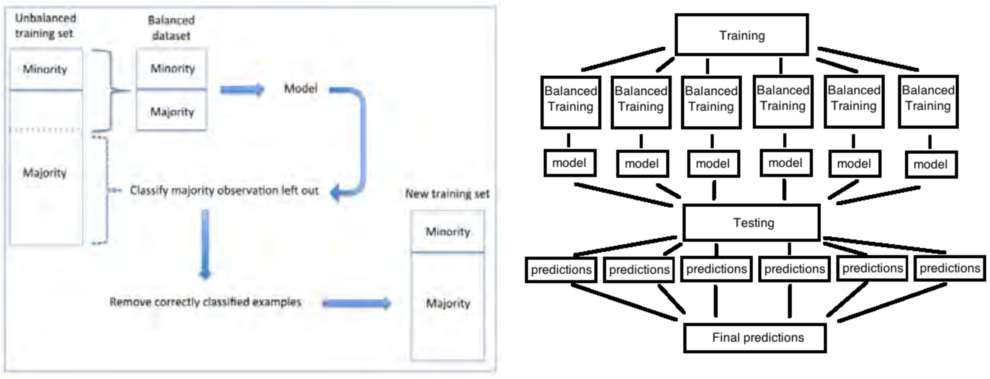

Figure 1: (left) BalanceCascade, a supervised method that keeps removing majority class examples until none are misclassified; (right) EasyEnsemble works parallel in a unsupervised manner [14].

2.2. Performance Metrics

To evaluate the performance of a classifier, evaluation criteria are necessary. Typically, a classifier is evaluated by a confusion matrix (see Table 1), which shows the results of correctly and incorrectly recognized examples of each class. From this point, predictive accuracy can be defined as all correctly classified observations divided by all (correctly and incorrectly) classified observations.

However, accuracy is not an appropriate measure when data is imbalanced and/or the costs of different errors vary markedly [15].

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Table 1: Confusion matrix for a two-class problem

In the imbalanced domain, is it more useful to measure the classification performance of the positive and negative classes independently. This can be achieved by the following metrics:

- TP rate: percentage of positive instances correctly classified. $TP_{rate} = \frac{TP}{TP + FN}$
- FP rate: percentage of negative instances misclassified. $FP_{rate} = \frac{FP}{FP + TN}$
- TN rate: percentage of negative instances correctly classified. $TN_{rate} = \frac{TN}{FP + TN}$
- FN rate: percentage of positive instances misclassified. $FN_{rate} = \frac{FN}{TP + FN}$

"Receiver operating characteristics" (ROC) curves are useful for visualizing the performance of a classifier. The ROC curve is created by plotting the sensitivity $(= TP_{rate})$ against the FP_{rate} $(= 1 - specificity)$ at different threshold settings. This plotting visualizes the trade-off between benefits and costs. The "area under the curve" (AUC, or, more correctly, AUROC), corresponds to the probability that a classifier will give a higher rank to a randomly chosen positive example, rather than a randomly chosen negative example. Models become optimal when they reach a score of 1.0. Models with higher AUC values are preferred over those with lower AUC values. As the ROC values have not been perfected as proposed by Kaymak [16], there are also other measures. The most used measures are precision-recall and its derived versions: F-measure and G-measure.

Precision-Recall

When the confusion matrix is analyzed, there are two more measures that can be derived. The first measure is called "precision"; it measures the fraction of all positive predictions that are correct. To calculate this, the number of correctly predicted positive values is divided by all positive predictions. The second measure is "recall"; this is the same as the true positive rate. Recall is a measure of the proportion of all real positive observations that are correct.

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{\text{correct positive predicted instances}}{\text{all positive predicted instances}} \tag{3}$$

$$\text{Recall} = TP_{rate} = \frac{TP}{TP + FN} = \frac{\text{correct positive predicted instances}}{\text{all actual positive instances}} \tag{4}$$

The precision measurements indicate how precise or useful the algorithm is. Recall is concerned with the completeness of an algorithm. Thus, it follows that precision and recall have no linear relationship between one another. If a given machine learning algorithm performs well at recall, it is not self-evident that the same will apply to the performance of precision. To overcome this problem, another measure can be ascertained by means of recall and precision, namely the F-measure (or the F1-score). This can be found in Equation (5). The general equation is represented by the F_β-measure (see Equation (6)), where β represents the relative importance of precision and recall.

$$\text{F-measure} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{5}$$

$$\text{F}_\beta\text{-measure} = (1+\beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \tag{6}$$

The F-measure is the harmonic mean of precision and recall. This measure can indicate how precise and how robust (in terms of recall) a certain model is. It is necessary to have a trade-off between precision and robustness. When precision and recall are equal to each other, the F-measure is maximal. The harmonic mean decreases quickly when only one of the metrics is optimized. Since the F-measure is available per class - say for example, fraud and non-fraud class - the F-score for the fraud class would be more important than the non-fraud class. This is because it is more important to classify the fraud cases correctly than the non-fraudulent ones. The geometric mean of precision and recall (known as Fowlkes–Mallows index) determines the similarity between two clustering's. A higher value indicates a greater similarity.

$$\text{G-mean} = \text{FMI} = \sqrt{\text{precision} \cdot \text{recall}} = \frac{TP}{\sqrt{(TP+FP)\cdot(TP+FN)}} \tag{7}$$

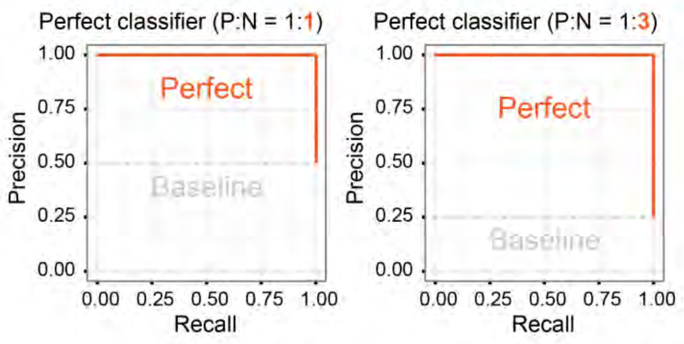

Figure 2: An ideal classifier shows a perfect precision recall curve and is a combination of two straight lines. From the top left corner to the top right corner, and from the top right corner to the end point in $\left(1, \frac{P}{P+N}\right)$ where P and N are respectively the number of positive and negative samples. [18]

A visual representation of precision and recall is possible through a precision-recall plot (like the previous ROC plot). In contrast to the ROC curve, the "precision-recall curve" (PRC) does not use the number of true-negatives, in the same way in which the ROC does. Both the PRC plot and ROC plot are model-wide evaluations. The PRC plot shows precision values (on the y-axis) against sensitivity/recall values (on the x-axis) for a certain threshold. The baseline of the curve is determined by the ratio of positive and negative values, in the form $y = \frac{P}{P+N}$. When the data is balanced, the baseline is at 0.5, but this changes as the data becomes imbalanced. This mean that the value of AUC-PRC also changes when the positive/negative instance ratio changes. This is explained in Figure 2. Takaya Saito has concluded that the PRC is more informative than the ROC when working with imbalanced data [19].

H-Measure

The evaluation of how a classifier performs, is mostly achieved in terms of misclassification costs. In the majority of occasions, this results in a weighted combination of both types of misclassification (false positives and false negatives). In the case of fraud, a false positive means that a case is unjustly classified as fraudulent; a false negative means that a fraud case has been missed by the classifier. Thus, in all likelihood, the cost of missing a fraud case can be higher than that of a falsely classified case of fraud. Accordingly, it is essential to decide which classification error is weighted more. The H-measure is an alternative to the AUC and was proposed by D.J. Hand [20]. It overcomes several critiques of the AUC – especially in those cases where AUC treats false positives as equally important as false negatives. The H-measure is designed to control the misclassification costs by a prior. This ensures that the end user does not need to set any weights for the costs, and captures the end user's uncertainty about the exact values. [21]

$$\text{H-measure} = 1 - \frac{\int Q(T(c); b, c)u(c)dc}{\pi_0 \int_0^{\pi_1} cu(c)dc + \pi_1 \int_{\pi_1}^{1} (1-c)u(c)dc} \tag{8}$$

In this equation, c_0 and c_1 are the misclassification costs of, respectively, a positive and negative instance. The prior probabilities are π_0 and π_1, corresponding to the positive and negative instances. From this point, c is defined as $c = \frac{c_0}{c_0+c_1}$ [22].

$$T(c) = \underset{t}{\operatorname{argmin}}\{c\pi_0(1 - TPR(t)) + (1-c)\pi_1 FPR(t)\} \tag{9}$$

$$Q(t, c) = \{c\pi_0(1 - TPR(t)) + (1-c)\pi_1 FPR(t)\}(c_0 + c_1) \tag{10}$$

$$u(c) = \frac{c(1-c)}{\int_0^1 c(1-c)dc} \tag{11}$$

The H-measure is an improvement upon the AUC since it takes the priors into account. These priors are important when working with imbalanced datasets, such as fraud data. Some critiques of the H-measure have been based on the assumptions that are made by this means of measurement. To clarify, some assumptions regarding the loss functions may not be valid for all application domains. When these assumptions are replaced, the measure will lose its objectivity – different assumptions by different researchers will lead to different H-measure values – which is required from all performance measures [16]. To overcome this problem, Hand [23] has proposed a universal standard distribution for the H-measure: the $Beta(1 + \pi_1, 1 + \pi_0)$. In this case, π_i is the proportion of instances from class i ($i = 0, 1$).

Cohen's Kappa

Kappa is a measure of the agreement in scores of two nominal variables. The kappa statistic is frequently used to test inter-rater reliability [24]. Inter-rater reliability is the measurement of the extent to which raters assign the same score to the same variable. It was proposed by Jacob Cohen in 1960 as an alternative to the percentage agreement because the percentage agreement does not take change into account. Accordingly, kappa compares the observed accuracy with an expected accuracy (the random chance). Because it takes random chance into account, this method is more robust than using accuracy as a metric. Kappa can be defined as follows:

$$\text{Kappa} = \frac{\text{observed accuracy} - \text{expected accuracy}}{1 - \text{expected accuracy}} \tag{12}$$

The "observed accuracy" is the number of instances that were classified correctly, throughout all instances. The "expected accuracy" is defined as the accuracy that a random classifier would expect to achieve, based on the confusion matrix.

$$\text{Observed accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

$$\text{Exp. accuracy} = \frac{\text{Actual False} \cdot \text{Pred. False} + \text{Actual True} \cdot \text{Pred. True}}{(\text{total number of instances})^2} \tag{14}$$

$$\text{Exp. accuracy} = \frac{(TN + FP) \cdot (TN + FN) + (FN + TP) \cdot (FP + TP)}{(TP + TF + FP + FN)^2} \tag{15}$$

Kappa is a normalized statistic, and it can be used even as the number of observations grows. Its value never exceeds 1. There is no fixed interpretation for the kappa value, but Cohen [24] has suggested the kappa results should be interpreted as follows:

Kappa	**Level of Agreement**	**Percentage of Data Reliable**
0 - 0.20	None	0 – 4%
0.21 – 0.39	Minimal	4-15%
0.40 – 0.59	Weak	15-35%
0.60 – 0.79	Moderate	35-63%
0.80 – 0.90	Strong	64-81%
> 0.90	Almost perfect	82-100%

Table 2: One possible interpretation of kappa.

An extension to the Kappa measure and an alternative to the AUC value, is the "area under kappa" (AUK). This measure has been proposed by Kaymak [16]. Similar to the H-measure but dissimilar to the AUC, the AUK depends on the priors. It does not, however, assume any knowledge about the underlying distribution as the H-measure does. This renders the kappa curve and AUK value more intuitive.

$$AUK = \int_0^1 k(f)\, df \tag{16}$$

$$k = \frac{2p(2-p)(TPR-FPR)}{p+FPR(1-2p)+p(1-2p)(TPR-FPR)} \tag{17}$$

The AUK is a performance metric that can be used in the same way as AUC. The AUK measures the area under the kappa curve. The kappa curve consists of kappa values plotted against the false positive rates; similar to the ROC plot, where the true positive rates are plotted against the false positive rates. One advantage of AUK over AUC is the accountability of class imbalance. Kaymak [16] has proposed that the kappa method prefers to classify the minority class correctly rather than the majority class. This makes the AUK a more preferable measure when one works with imbalanced data. The kappa curve function is defined in Equation (17), where $p = TP + FN$ and $n = FP + TN$. As the curve has often a unique maximum, it is a useful tool for selecting a suitable threshold for a model. The threshold that maximizes kappa is typically unique [16].

3. Data Description and Exploration

Traditional fraud detection was performed manually by human analysts or by means of information systems that were based on predefined rules. The main drawback with those information systems is that rules have to be defined in advance, before the fraud is committed. Since all rules have to be defined manually, the system is unable to recognize other fraudulent patterns, which ensures that criminals can find ways to avoid detection. As I stated earlier in this paper, the lack of publicly available fraud datasets makes it challenging to undertake research on fraud detection techniques so as to improve the recognition of fraudulent patterns. Lopez-Rojas, Elmir and Axelsson [2] have overcome the problem of not having suitable financial data by creating a financial transaction simulator. This simulator, called PaySim, generates synthetic data that is most similar to real transaction data. Based on real data (provided by Ericsson) did Lopez-Rojas prove that the simulated data is similar.

The data used in this paper is from the Synthetic Financial Datasets for Fraud Detection, a dataset which is openly distributed by Kaggle [25], and which consists of data generated through the PaySim simulator. There are 6,362,620 transactions simulated in a period of 30 days. Modern techniques such as machine learning, can classify the fraudulent transactions and can predict fraud in the future. As we have ascertained from the literature review, machine learning algorithms are sensitive to imbalances between classes in data. The PaySim data consists of 8213 fraudulent transactions out of 6,362,620 in total; this makes the data highly imbalanced. Only 0.0013% of the instances are from the minority class; this, therefore, makes the dataset suitable for testing different types of balancing methods.

The data covers five types of transactions: cash-in, cash-out, debit, payment and transfer. Customers can cash-in and cash-out money from a given account through a local agent who serves as an ATM. The balance of the account increases when paying the agent (cash-in) and decreases when the customer withdraws cash from the agent (cash-out). Debit is the process wherein the money is transferred from the mobile money service to a bank account; this decreases the account balance. A payment transaction is the process of paying for goods or services to an agent; this decreases the balance of the account and increases the balance of the receiver. Transfer is similar to a payment transaction but is the process of sending money to another user of the mobile money service [26]. In Table 3, the number of times a transaction occurs is shown.

Transaction type	Fraudulent	Frequency
Cash-in	No	1399284
Cash-out	No	2233384
	Yes	4116
Debit	No	41432
Payment	No	2151495
Transfer	No	528812
	Yes	4097

Table 3: The number of times each transaction type occurs in the synthetic dataset, divided into fraud and non-fraud cases.

From Table 3, it follows that fraud only arises from two types of transactions: cash-out and transfer. When the other types are ignored, a fraudulent transaction occurs in approximately 1 out of 336 transactions, which means that approximately 0,3% of transactions are fraudulent.

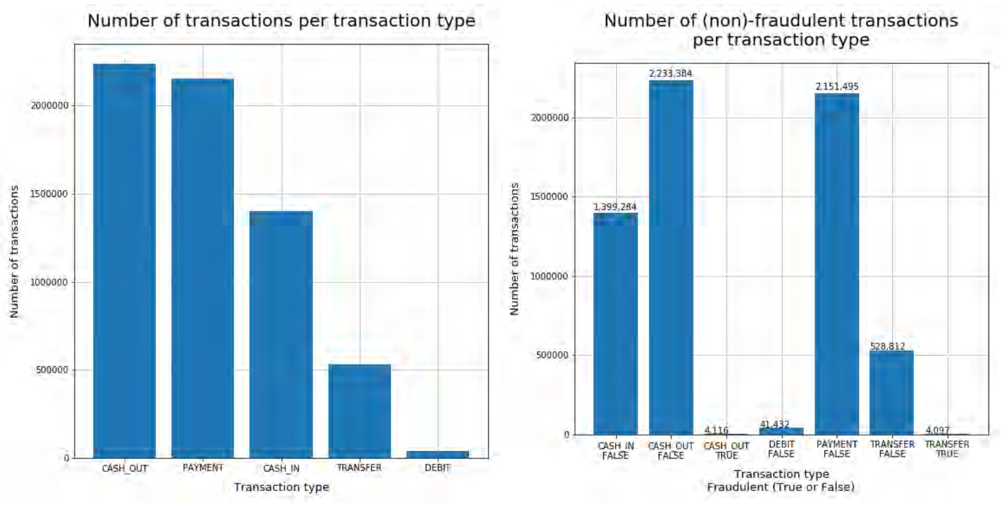

Figure 3: (left) The number of transactions for each type of transaction. (right) The number of times each transaction type occurs in the synthetic dataset, divided into fraud and non-fraud cases. These are the visualized results from Table 3.

Column name	Description of column data
Step	This is the time interval in which the transaction occurs. The time interval maps with a unit of real-world time, where 1 step is equal to 1 hour. There are a total of 744 steps which maps to 30 days in real time.
Type	This is the type of the transaction. There are five types of transactions: CASH_IN, CASH_OUT, DEBIT, PAYMENT or TRANSFER.
Amount	This is the transaction amount in local currency.
nameOrig	This is the account name of the person who started the transaction.
oldBalanceOrig	This is the initial balance before the transaction starts.
newBalanceOrig	This is the new account balance after the transaction is completed.
nameDest	This is the account name of the receiver of the transaction.
oldBalanceDest	This is the initial balance of the receiver before the transaction starts.
newBalanceDest	This is the new account balance of the receiver after the transaction is finished.
isFraud	The instance is marked as fraudulent or not. Fraudulent = 1, non-fraudulent = 0.
isFlaggedFraud	Huge transactions are flagged as fraud when the amount of money that is transferred between mobile money accounts is larger than 200.000.

*Table 4: Data description of the 11 columns of data from the "*Synthetic Financial Datasets for Fraud Detection*" dataset, which is publicly available from Kaggle.com.*

The dataset contains 11 columns of data, which are described in Table 4. Not all the information in this dataset seems relevant. After data analysis was performed, it appeared that "isFlaggedFraud" only occurs in 16 instances, while there are over 400.000 instances from the "TRANSFER" type where the amount of money transferred is higher than 200.000. Therefore, the isFlaggedFraud variable does not seem useful since it does not flag the data as the description declares. Also there do not appear to be any relations when one tries to find a correlation between isFlaggedFraud and the other variables. Thus, it is unclear why the simulator flagged 16 instances as fraudulent. Accordingly, the feature is not used in ensuing models.

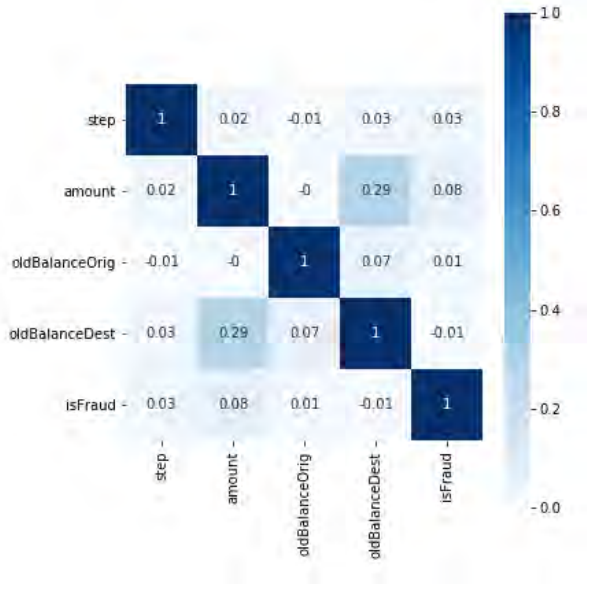

Figure 4: Correlation heatmap

The correlation heatmap in Figure 4 shows no correlation between features. There is a slight correlation between “amount” and “oldBalanceDest”, but with a value of 0.29, this is not significant. Further investigation into the time data, which is expressed as time steps provides more information. Since every time step is equal to one hour of real time, the time information can be used to analyze transactions based on the day of the month, the day of the week, and the hour of the day. There are a total of 744 time steps, which corresponds to 1 month of 31 days. In Figure 5, one can observe the transaction distribution, which is split by the hour of the day. The distribution follows an expected pattern, where most transactions occur during daytime. Since, the number of fraud transactions is the same during the day as the night; this results in an increase in the fraud/non-fraud ratio during the night.

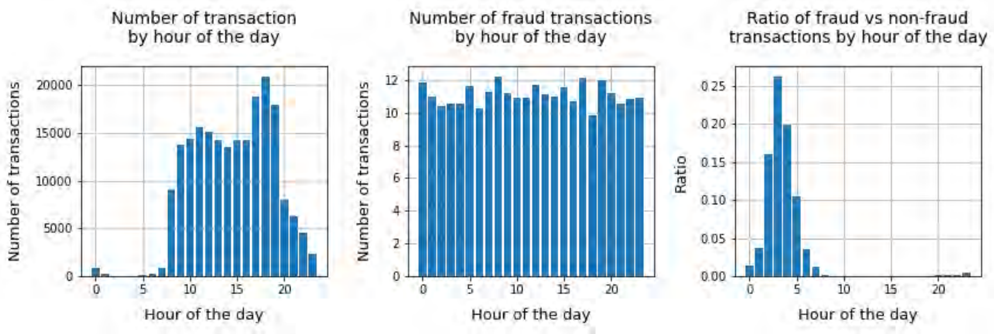

Figure 5: (left) The average number of transactions per hour. (middle) The average number of fraudulent transactions. (right) The ratio between fraudulent and non-fraudulent transactions.

An analysis of the transactions by day of the month reveals that there are far more transactions during the 1st, 2nd, and between the 6th and 17th day of the month than on other days. Nevertheless, the number of fraudulent transactions is still constant over all days, as one can observe in Figure 6.

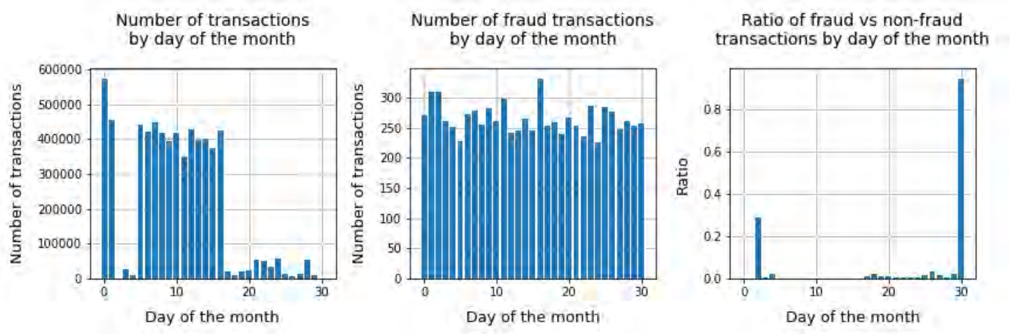

Figure 6: (left) The total amount of transactions. (middle) The total number of fraudulent transactions. (right) The ratio between fraudulent and non-fraudulent transactions.

An even more useful feature is the fraud transaction by day of the week. In Figure 7, it can be observed that the number of transactions varies by day. Most transactions occur during the first two days of the week. The transactions decrease significantly until the fifth day and are more or less stable over the last two days of the week. The number of fraudulent transactions is higher on the first three days than on the last four days of the week.

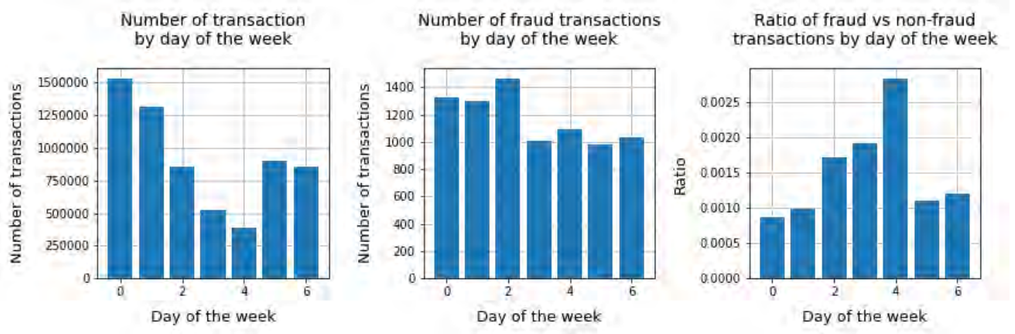

Figure 7: (left) The total amount of transactions. (middle) The total amount of fraudulent transactions. (right) The ratio between fraudulent and non-fraudulent transactions.

Based on the previous analysis, the number of transactions per hour and the number of transactions by day of the week could be an interesting feature to include for the machine learning and/or sampling algorithms. An analysis of the number of transactions uncovers certain differences between non-fraud and fraud cases.

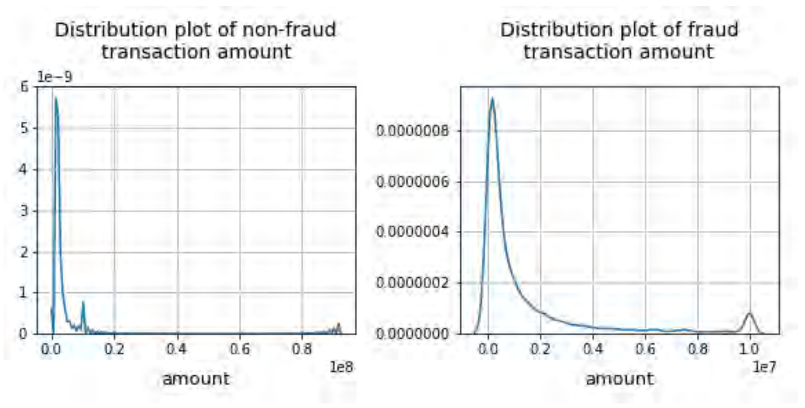

Figure 8: The distribution plot of transaction amount split by fraud and non-fraud cases.

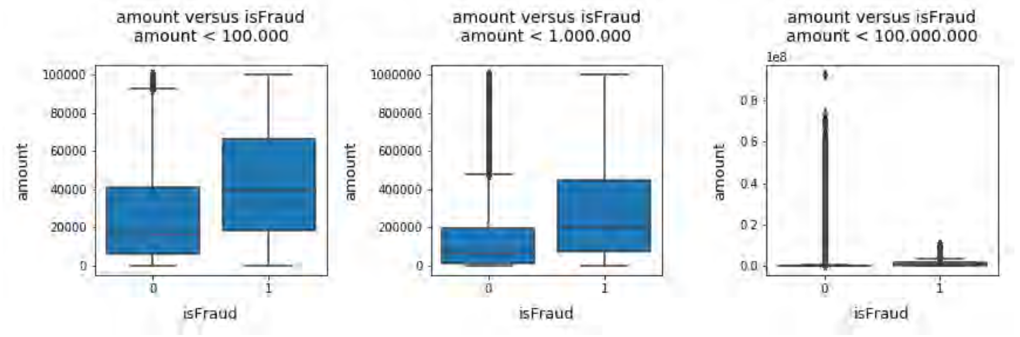

Figure 9: The boxplot of amount split on fraud and non-fraud cases. There are three boxplots, for amounts smaller than 100.000 (left), smaller than 1.000.000 (middle) and for unrestricted amounts, which is in practice smaller than 100.000.000 (right).

As one can observe from Figure 8 and Figure 9, there is a substantial difference in the size of transactions. The amount of money that is involved in fraud is higher than in legal, non-fraud transactions. Although, the most expensive transactions are not flagged as fraud, more than 99,5% of non-fraud transactions involve a smaller value of money than the upper 17% of fraud cases.

4. Methods

The research conducted was divided in three parts – namely data reduction, re-sampling, and classification. The classification part is divided in two different approaches. The first approach is based on traditional classifiers such as random forest and logistic regression; the second approach uses deep neural networks, which are currently the center of many experiments in fraud detection [27]. The schematic representation of the research methodology can be observed in Figure 10.

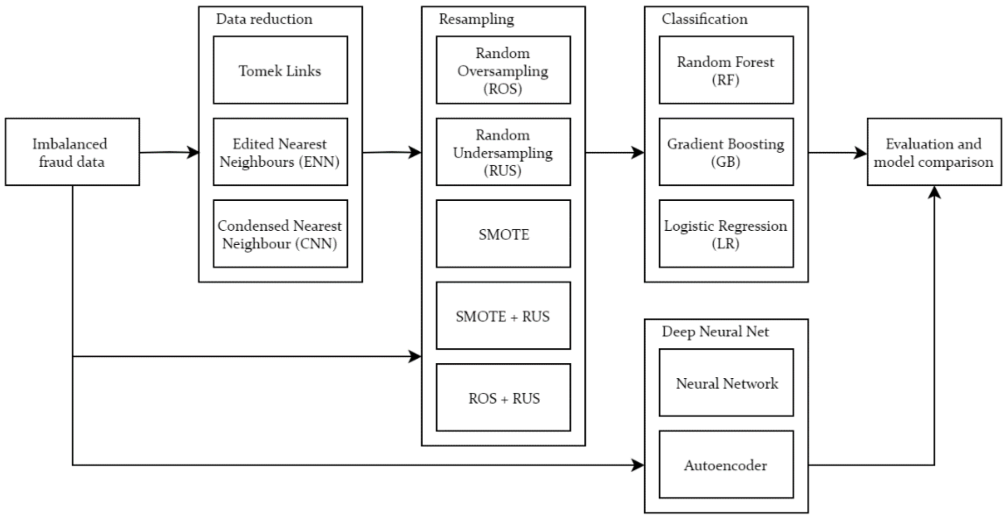

Figure 10: Research methodology

Before any research on the dataset was conducted, preprocessing was performed so that all models could use the same data source as an input. The transaction type was then converted into Boolean (0/1) dummy variables for each type. These were: “cash_in”, “cash_out”, “debit”, “payment”, and “transfer”. Afterwards, two copies were made from the original dataset with dummy variables. In one of the copies, all the values were standardized, by means of Equation (18), and the other copy was normalized, by means of Equation (19). After this preprocessing, the data was split into a training set and test set. Eighty percent of the data was used for training purposes, while the other twenty percent was used to test the performance of the trained classifier.

$$X_{\text{standardized}} = \frac{X - \bar{X}}{s_X} \tag{18}$$

$$X_{\text{normalized}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{19}$$

In all experiments, the following features were used: “isFraud” as target variable and “step”, “cash_in”, “cash_out”, “debit”, “payment”, “transfer”, “amount”, “oldBalanceOrig”, “newBalanceOrig”, “oldBalanceDest”, and “newBalanceDest” were used as features.

4.1. Data Reduction

Since the majority class has many more instances than the minority class, it makes sense to remove noisy data. Only majority class instances should be removed so as to overcome the problem of the minority class becoming even smaller. Three methods will be compared as data reduction algorithm: edited nearest neighbors, T-links, and condensed nearest neighbors. As I described earlier, T-links remove unwanted majority class instances that are close to the minority class region until all minimally distanced, nearest-neighbor pairs are from the same class. When two instances are a T-link, then one of them is a noisy instance or both instances are on the class boundaries. The T-link method removes noisy instances and those instances close to the borderline. Another approach is CNN; this method tries to remove redundant instances from the data. This is achieved by eliminating instances that are distant from the borderline. Thus, CNN tries to create a subset which is as close as possible to the original data, in terms of data representation. The third method, ENN, is another distance-based method. Firstly, a reference set that is equal to the original data is created. Subsequently, every instance in the original data is classified by the k-NN rule and added to the edited set. All instances that are not correctly classified by the k-NN rule are removed from the reference set. This results in smoother boundaries since noisy and borderline instances are removed.

4.2. Sampling and Classification

The next step in the process is resampling and, finally, classification. There are three sampling approaches used and two combinations of the three approaches. In my experiments, the conducted sampling techniques were random over-sampling (*ROS*), random under-sampling (*RUS*), *SMOTE*, *ROS + RUS,* and *SMOTE + RUS*. The last two approaches use a combination of over- and under-sampling. An explanation for these methods can be found in Chapter 2.1. For each method, there are several ratios of under- and/or oversampling used. In Chapter 5, more details are supplied concerning these ratios. When the samples are generated, the reduced data sets are used as inputs for three different classifiers; random forest, gradient boosting and logistic regression.

Random Forest

"Random forest" is a supervised classification algorithm that works by means of an ensemble approach. The preparatory work that led to the random decision forests method was undertaken by Tin Kam Ho [28]. In 2001, an extension of the algorithm was developed by Breiman [29]; this extension is today known as the "random forest". The basic idea behind the classifier involves the use of several weak learners, the small decision trees, that are based on a subset of the feature set. Many of these decision trees can be built in parallel and combined into a strong learner [30].

The random forest algorithm works as follows: [31]

1. Create N subsets of the data, at random, by sampling with replacement. Each subset should be around 2/3 of the of the total set size.
2. Learn a tree and do for each node in the tree:
    a. Select at random m feature variables, where m is much smaller than the total number of features available.
    b. Take a (binary) split based on the best feature according to a certain objective function.
    c. At the next node, repeat the above with another m variables.

Gradient Boosting

Another machine learning algorithm used is the gradient boosting machine (GBM). This method belongs to the family of boosting algorithms that use weak learners and convert them into strong ones, just as the random forest method does. The difference is that random forest is a bagging algorithm (not a boosting algorithm) that works in parallel. Boosting algorithms function sequentially and try to add new models that perform well for those instances where previous models failed [32]. Although this appears to be a more effective option, the performance does, in general, depend on the data and the specific case of usage. Both bagging and boosting decrease the variance of an estimate since they combine estimates from different models. This ensures that a given model becomes more stable [33]. In cases where the model easily overfits, bagging outperforms boosting methods. Boosting does not help to avoid overfitting and can even increase the overfit, while bagging tries to prevent overfitting.

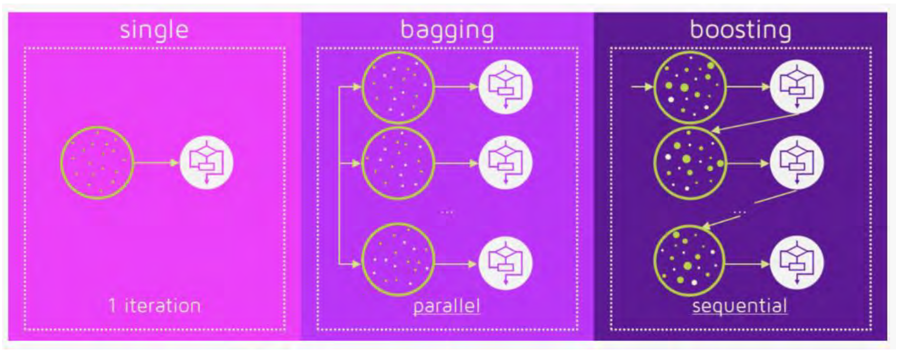

Figure 11: Bagging (independent models) and boosting (sequential models).

In general, GBM is a gradient decent combined with boosting. Gradient boosting creates a prediction model from an ensemble of weak learners (typically decision trees). These models are sequentially built and introduce at each successive stage a weak learner so as compensate for the shortcomings of existing weak learners. The gradient decent method is used to minimize a differentiable loss function.

Logistic Regression

Logistic regression is a regression model where the response variable is binary; this means that it can have two states: 0 or 1. Although the response variable is binary, the explanatory variables can be either discrete or continuous. Logistic regression models the probability distribution $p(y|\boldsymbol{x})$, where y is the response variable and $\boldsymbol{x}$ the explanatory vector, containing all the features. For LR, the hypothesis is defined in the following form [34]:

$$h(x) = \begin{Bmatrix} 1 : & p(y = 1|\boldsymbol{x}) > t \\ 0 : & \text{otherwise} \end{Bmatrix} \tag{20}$$

In this case, t represents a certain threshold. Usually, this threshold is chosen as 0.5. The probability distribution function $p(y|\boldsymbol{x})$ is defined as:

$$p(y = 1|\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\beta}^T x}} = \sigma(\boldsymbol{\beta}^T \boldsymbol{x}) \tag{21}$$

In this equation, $\boldsymbol{\beta}$ is a vector containing all the weights, and σ is the sigmoid function, which is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{22}$$

Logistic regression uses odds ratios to calculate the probability. This is defined as the ratio of the odds of an event happening to its not happening. The odds ratio is used rather than directly calculating the probability distribution in order to ensure that the probability is always between 0 and 1. The link function that is used to do this is called the logit function. If we define $\pi = p(x = 1|\boldsymbol{x})$, then the following equation applies:

$$\begin{aligned} \sigma(x) &= \frac{1}{1 + e^{-x}}\ \pi = \alpha + \boldsymbol{\beta}^T \boldsymbol{x} \\ \text{logit}(\pi) &= \log\left(\frac{\pi}{1 - \pi}\right) = \alpha + \boldsymbol{\beta}^T \boldsymbol{x} \end{aligned} \tag{23}$$

Since the inverse of the logit function is the sigmoid function, we can describe π in the following terms:

$$\begin{aligned} \pi &= \text{logit}^{-1}(\alpha + \boldsymbol{\beta}^T \boldsymbol{x}) \\ &= \sigma(\alpha + \ \boldsymbol{\beta}^T \boldsymbol{x}) \\ &= \sigma(\boldsymbol{\beta}^T \boldsymbol{x}) \end{aligned} \tag{24}$$

4.3. Deep Neural Net

An alternative to the sampling strategies discussed in Paragraph 2.1 and the preceding algorithms discussed in this chapter, is the use of (deep) neural networks. To overcome the problem of over- and/or under-sampling, two methods can be discussed; feed-forward networks and autoencoder networks. Both networks are built with the deep learning library Tensorflow, with the high-level neural networks Keras API on top. Tensorflow is an open-source machine learning library originally developed by researchers from the Google Brain division. Keras works on a higher level and is designed to enable fast experimentation. It is user friendly, modular, and easy to extend. It is possible to build deep neural nets with only a few lines of code.

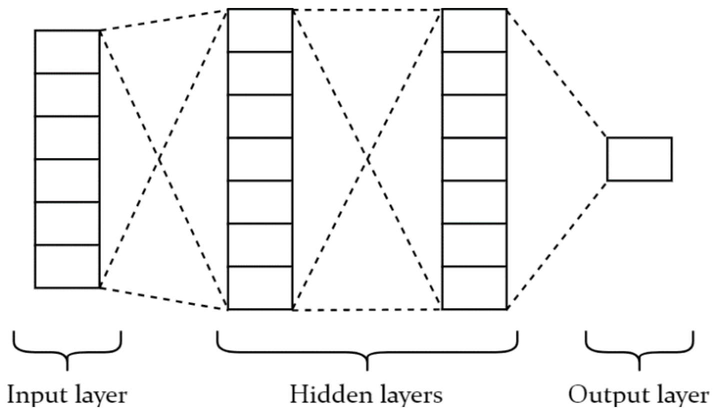

Figure 12: A simple representation of layers in a feed forward neural network.

The first neural network that this project uses is the so-called “feed forward neural network”. This network was the first and simplest type of artificial neural network to be devised and was, therefore, probably the most commonest type in practical applications [35, 36]. The network consists of an input layer, output layer, and in-between (multiple) hidden layers. Each layer consists of a number of nodes that are connected with adjacent layers, all these connections have weights associated with them. A graphical representation can be observed in Figure 12. In a feed forward network, the information moves from the input layer to the output layer via the hidden layers in the middle, and there are no loops in the network. The hidden layers compute a series of transformations that change the similarities between cases and ensure that each layer has a non-linear relation to adjacent layers. Through the employment of backpropagation, the weights in each layer are optimized.

Autoencoder Networks

An autoencoder is a neural network that is used to learn a representation for a set of data. It is an unsupervised machine learning algorithm that takes a sample from the fraud dataset as an input and tries to reconstruct it. The goal is to copy the input to its output by using a reconstruction process [37]. The autoencoder consists of an encoder and decoder part. The encoder is a network of layers that takes a sample as an input and produces a much smaller representation (encoding). The decoder part tries to reconstruct the input by using only the encoding as input. Back-propagation is used to optimize the autoencoder network in such a way that the right amount of information is encoded to ensure that the decoder network is still able to reproduce the input. In the case of fraud data, fraud samples are supposed to have a different distribution from non-fraud samples. So, it should be possible to recognize fraud transactions through the amount of reconstruction errors. This is expected to function as long as the network is trained only on the non-fraud samples. In that case, fraud samples (the assumption here is that they have a different distribution) are recognized by the reconstruction error.

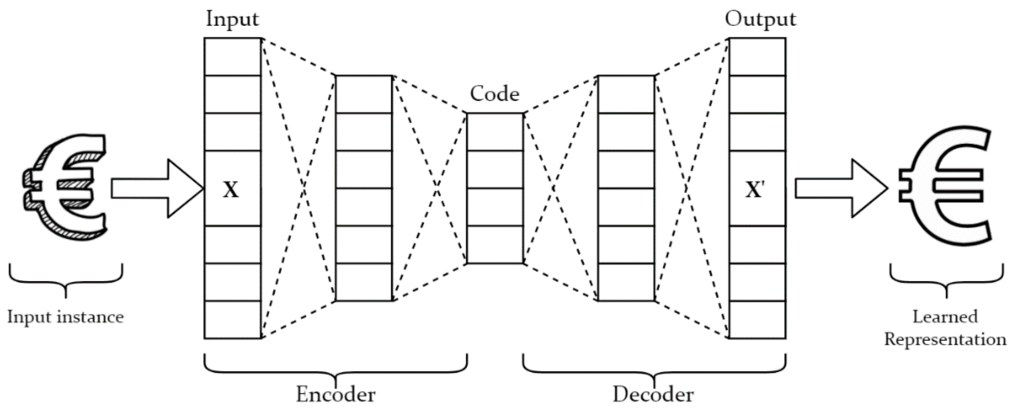

Figure 13: Schematic representation of an autoencoder where a general representation is learned by raw input instances.

5. Results

The previous chapter, Methods, outlined the steps this paper takes to test the models in combination with the imbalanced data (as described in Chapter 3). This chapter describes the results of the conducted research. Each model is tested with normalized (NRM) and standardized (STD) data in combination with various types of reduction techniques (no reduction, T-Link, ENN, and CNN). After the data is resampled, three classifiers are trained with the resampled data and evaluated on a test set, which was unseen by the classifier previously. There are two test sets: one where the data is normalized, and one where the data is standardized. The classifier that is trained with the normalized data should be tested with the normalized test set. Besides the rescaling technique, no differences exist between the test sets. A baseline has been defined based on the classification scores when raw (unprocessed) data is used in order to ensure that the results can be compared and placed in perspective. Without this baseline, is it difficult to identify whether the models increase the classification performance. The baseline values can be found in Table 5. Performance is measured in terms of area under the ROC curve (ROC-AUC) and normalized F1 scores. The ROC-AUC values give the probability that a fraudulent transaction has a higher mean squared error (MSE) than a non-fraud one.

		NRM			**STD**		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
No reduction	**ROC-AUC**	0,53982	0,89443	0,66839	0,71174	0,89285	0,72072
	F1	0,68485	0,90451	0,75095	0,77623	0,90322	0,77845
T-Link	**ROC-AUC**	0,53919	0,89474	0,84317	0,71047	0,89064	0,66039
	F1	0,68455	0,90477	0,86269	0,77547	0,90142	0,73052
ENN	**ROC-AUC**	0,54172	0,89916	0,65740	0,71205	0,89474	0,81278
	F1	0,68574	0,90840	0,74301	0,77642	0,90476	0,83045
CNN	**ROC-AUC**	0,61651	0,97032	0,95336	0,81513	0,98373	0,98630
	F1	0,72276	0,97112	0,95400	0,84381	0,98394	0,98644

Table 5: Baseline values

5.1. Random Oversampling

First, the evaluation is performed on the random oversampling model. The minority (fraud) instances, which comprise 6.631 instances by default, are oversampled into 11 steps (0-10), where the number of oversampled instances are, respectively, 9.750, 19.500, 39.000, 78.000, 156.250, 312.500, 625.000, 1.250.000, 2.500.000, 5.000.000, and 5.083.465. The 10th step does not provide an exact number of minority instances. This is because the number of majority instances differ in the case of normalized and standardized data. The corresponding ratio of non-fraud versus fraud transactions varies from 521:1 to 1:1. Table 6 presents the results for random oversampling with no reduction. In appendix tables Table 13, Table 14 and Table 15, the results are presented for random oversampling with, respectively, T-link, ENN and CNN as reduction techniques.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (9.750)	**ROC-AUC**	0,55373	0,89695	0,74555	0,7291	0,89537	0,6971
	F1	0,69143	0,90658	0,79695	0,78684	0,90528	0,76747
1 (19.500)	**ROC-AUC**	0,5749	0,90137	0,74585	0,76446	0,89979	0,74679
	F1	0,7017	0,91022	0,79734	0,80934	0,90892	0,79794
2 (39.000)	**ROC-AUC**	0,58944	0,90863	0,90507	0,7843	0,90548	0,91333
	F1	0,70893	0,91628	0,91328	0,82253	0,91364	0,92023
3 (78.000)	**ROC-AUC**	0,63776	0,90769	0,9132	0,81696	0,89821	0,92777
	F1	0,73407	0,91548	0,9201	0,84518	0,90761	0,93261
4 (156.250)	**ROC-AUC**	0,59729	**0,90927**	0,92279	0,84054	0,90643	0,94116
	F1	0,71288	**0,91681**	0,92827	0,86229	0,91443	0,94439
5 (312,500)	**ROC-AUC**	0,65886	0,90643	0,93253	0,85423	0,90453	0,95438
	F1	0,74535	0,91443	0,93671	0,87248	0,91284	0,95631
6 (625.000)	**ROC-AUC**	0,738	0,90579	0,94364	0,86858	0,90295	0,97043
	F1	0,79156	0,9139	0,94647	0,88334	0,91153	0,9712
7 (1.250.000)	**ROC-AUC**	0,76506	0,90643	0,95362	0,90497	0,90295	0,98629
	F1	0,80442	0,91443	0,95547	0,91227	0,91153	0,98639
8 (2.500.000)	**ROC-AUC**	0,84203	0,90232	0,96692	0,93364	0,90674	**0,9915**
	F1	**0,8548**	0,91101	0,96767	0,93618	0,91469	**0,99148**
9 (5.000.000)	**ROC-AUC**	0,8384	0,90863	0,97124	0,94726	**0,90706**	0,99045
	F1	0,83328	0,91628	0,97159	0,94741	**0,91496**	0,99038
10 (1:1)	**ROC-AUC**	**0,85049**	0,90295	**0,97164**	**0,94747**	0,90453	0,99056
	F1	0,84616	0,91153	**0,972**	**0,94756**	0,91285	0,99048

Table 6: Results for random oversampling with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. Best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
No reduction	**Ratio**	1,00	32,53	1,00	1,00	1,02	2,03
	ROC-AUC	0,85049	0,90927	0,97164	0,94747	0,90706	0,99150
	F1	0,84616	0,91681	0,97200	0,94756	0,91496	0,99148
T-Link	**Ratio**	2,03	32,53	1,00	1,00	32,53	1,02
	ROC-AUC	**0,87828**	0,91148	0,97209	**0,95143**	0,90864	0,99068
	F1	**0,88722**	0,91867	0,97242	**0,95131**	0,91628	0,99060
ENN	**Ratio**	1,00	4,06	1,00	1,02	8,13	2,03
	ROC-AUC	0,87104	0,91306	0,97180	0,95092	0,91021	0,99106
	F1	0,86889	0,92001	0,97214	0,95087	0,91760	0,99104
CNN	**Ratio**	0,84	0,42	1,00	0,42	0,84	1,00
	ROC-AUC	0,84558	**0,98188**	**0,99446**	0,95130	**0,99146**	**0,99331**
	F1	0,85761	**0,98216**	**0,99446**	0,95105	**0,99150**	**0,99328**

Table 7: Best results summarized from tables in Appendix A: Results random oversampling. Highest performance for each model is indicated in bold. Sampling ratio (number of non-fraud transaction versus one fraudulent transaction) is provided for each model in combination with the area under the curve and normalized F1 score.

When we look more closely at the results from Table 6, it can be seen that the ROC-AUC and F1 score increase when the data becomes more balanced. A definitive conclusion cannot be drawn from the four tables with oversampling results. However, it seems that the random forest classifier achieves slightly better results with normalized data, while the other two classifiers obtain better results with standardized data. The best results are combined in Table 7: Best results summarized from tables in Appendix A: Results random oversampling. Highest performance for each model is indicated in bold. Sampling ratio (number of non-fraud transaction versus one fraudulent transaction) is provided for each model in combination with the area under the curve and normalized F1 score.. This table indicates that the best results are achieved with CNN as a reduction technique. Looking deeper into the results, it seems that oversampling always performs better than the baseline where no sampling technique is applied.

5.2. Random Under-sampling

Another strategy is random under-sampling. This is performed on the majority class while the minority instances are frozen. All the experiment results can be found in appendix tables Table 16, Table 17 and Table 18. The best results are summarized in Table 8. As the CNN reduction technique reduces the majority instances to approximately 16.000 samples, random under-sampling is only applied to the other models. In practice, it is not useful to reduce the majority class further. Evaluating the results seems to reveal that the usage of a reduction technique results in lower performance, or just a slight increase. From the ratios in Table 8, we can conclude that the classifiers work best when the majority and minority classes re-balanced. A ratio between 1 and 1.5 seems to work best in this case. The best result is obtained with no data-reduction technique in combination with the random forest classifier with normalized data.

		NRM			**STD**		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
	Ratio	1,47	1,47	1,47	1,00	1,47	1,47
No reduction	**ROC-AUC**	**0,83339**	**0,99385**	**0,99138**	0,92616	**0,99369**	0,99069
	F1	**0,83844**	**0,99383**	**0,99133**	0,92686	**0,99367**	0,99064
	Ratio	1,47	1,00	2,94	1,00	1,00	1,47
T-Link	**ROC-AUC**	0,83174	0,99223	0,99047	**0,92636**	0,99273	0,99089
	F1	0,83698	0,99219	0,99047	**0,92698**	0,99269	0,99083
	Ratio	1,47	1,00	1,47	1,00	1,00	1,47
ENN	**ROC-AUC**	0,83106	0,99227	0,99026	0,92597	0,99252	**0,99144**
	F1	0,83590	0,99223	0,99020	0,92675	0,99248	**0,99138**

Table 8: Best results summarized from tables in Appendix B: Results Random Under-sampling. Highest performance for each model is indicated in bold. Sampling ratio (number of non-fraud transaction versus one fraudulent transaction) is provided for each model in combination with the area under the curve and normalized F1 score.

5.3. SMOTE

Next to random over- and under-sampling, the SMOTE technique is applied to the data. In this case, the majority class is frozen and the minority class is oversampled by generating artificial fraud transactions. All the experiment results can be found in appendix tables Table 19, Table 20, Table 21, and Table 22. The best results are summarized in Table 9. After evaluating these results, there seems to be a similar result as in the case of random oversampling. CNN performs the best in almost every case, except for the gradient boosting algorithm in combination with normalized data, where ENN outperforms the others. Also, in this case, the performance of the classifiers is increased when the data becomes more balanced.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
No reduction	**Ratio**	2,03	2,03	2,03	1,02	2,03	1,02
	ROC-AUC	0,98470	0,94646	0,87245	0,99005	0,98858	0,94717
	F1	0,98492	0,94889	0,88331	0,99014	0,98860	0,94768
T-Link	**Ratio**	8,13	2,03	1,00	4,07	2,03	1,02
	ROC-AUC	0,98293	0,94703	0,90274	0,99012	0,98826	0,9484
	F1	0,98320	0,94941	0,90572	0,99021	0,98828	0,94883
ENN	**Ratio**	2,03	2,03	1,00	1,02	2,03	1,00
	ROC-AUC	0,98722	0,94744	**0,94060**	0,99160	0,98858	0,94749
	F1	0,98737	0,94979	**0,90673**	0,99166	0,98859	0,94795
CNN	**Ratio**	0,42	0,84	0,84	0,48	1,00	0,48
	ROC-AUC	**0,99307**	**0,99354**	0,85015	**0,99452**	**0,99247**	**0,94976**
	F1	**0,99308**	**0,99355**	0,86304	**0,99451**	**0,99245**	**0,94974**

Table 9: Best results summarized from tables in Appendix C: Results SMOTE. Highest performance for each model is indicated in bold. Sampling ratio (number of non-fraud transaction versus one fraudulent transaction) is provided for each model in combination with the area under the curve and normalized F1 score.

5.4. SMOTE + RUS

Instead of using only oversampling or under-sampling, these are now combined. The SMOTE + RUS model combines oversampling of the minority class with random under-sampling of the majority class. First, the minority instances are oversampled into five steps of 9.750, 19.500, 39.000, 78.000, and 156.250 transactions, respectively. Then, the majority class is under-sampled into six steps. The number of transactions in each step is, respectively, 5.000.000, 2.500.000, 1.250.000, 625.000, 312.500, and 156.250. This strategy results in a completely balanced system in the latest step, where the minority and majority classes possess an equal number of instances. All the experiment results can be found in Appendix D: Results SMOTE + RUS. The best results are summarized in Table 10. In line with the results from previous experiments, the addition of a reduction technique does not add significant value. Only in the case of the random forest classifier does it perform slightly better. Although, SMOTE+RUS for RF+ENN with standardized data performs best out of all experiments, with a ROC-AUC value of 0,99627. Further observation of the ratios reveals that best results are always obtained with a ratio between one and two.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
No reduction	Ratio	2,00	1,00	2,00	1,00	1,00	2,00
	ROC-AUC	**0,86493**	0,99259	**0,98825**	**0,94639**	0,99345	**0,98915**
	F1	**0,87571**	0,99262	**0,98828**	**0,9469**	0,99347	**0,98917**
T-Link	Ratio	1,00	2,00	1,00	1,00	1,00	2,00
	ROC-AUC	0,85493	**0,99427**	0,98744	0,94586	0,99388	0,98867
	F1	0,85198	**0,99429**	0,98741	0,94638	0,99389	0,98868
ENN	Ratio	1,00	1,00	2,00	1,00	1,00	2,00
	ROC-AUC	0,85372	0,99368	0,98768	0,9457	**0,99627**	0,9886
	F1	0,85067	0,99369	0,98771	0,94625	**0,99627**	0,98862

Table 10: Best results summarized from tables in Appendix D: Results SMOTE + RUS. Highest performance for each model is indicated in bold. Sampling ratio (number of non-fraud transaction versus one fraudulent transaction) is provided for each model in combination with the area under the curve and normalized F1 score.

5.5. ROS + RUS

The ROS + RUS model combines random oversampling of the minority class with random under-sampling of the majority class. The experiment set-up is the same as within the case of SMOTE+RUS. First, the minority class is oversampled in five steps, and afterwards, the majority class is under-sampled in six steps until the dataset is fully balanced. Based on the results in Table 11, it is not possible to make any definitive conclusions regarding whether it is better to use oversampling, under-sampling, or a combination of both. Something of note is that the use of a reduction technique can result in a performance boost (RF-classifier) or in a performance decrease (LC-classifier)

		NRM			STD		
		LC	RF	GB	LC	RF	GB
No reduction	Ratio	8,01	1,00	2,00	1,00	1,00	2,00
	ROC-AUC	**0,97714**	0,84620	0,99139	**0,97338**	0,94187	0,99114
	F1	**0,97762**	0,84187	0,99138	**0,97404**	0,94223	0,99113
T-Link	Ratio	1,00	2,00	2,00	1,00	1,00	1,00
	ROC-AUC	0,84700	**0,97899**	0,99188	0,94733	**0,97779**	0,99093
	F1	0,85183	**0,97939**	0,99185	0,94742	**0,97825**	0,99091
ENN	Ratio	2,00	1,00	2,00	1,00	2,00	2,00
	ROC-AUC	0,83932	0,97644	**0,99222**	0,93993	0,97711	**0,99161**
	F1	0,85220	0,97694	**0,99220**	0,94038	0,97759	**0,99160**

Table 11: Best results summarized from tables in Appendix E: Results ROS + RUS. Highest performance for each model is indicated in bold. Sampling ratio (number of non-fraud transaction versus one fraudulent transaction) is provided for each model in combination with the area under the curve and normalized F1 score.

5.6. Deep Learning

As an alternative to the use of preprocessing techniques in combination with a classifier, two types of neural networks are applied. The first method is a traditional feed-forward network, while the second approach is the autoencoder network.

Feed-forward Network

The first deep learning network is a feed-forward network with two hidden layers containing, respectively, 64 and 32 units. In Network 1, the implemented network is summarized. After each dense layer, a dropout layer is implemented to prevent overfitting of the model. The first and second dropout layers possess respectively, dropout values of 0.2 and 0.1. After tuning the number of hidden layers, layer size, dropout values, and batch size, this model performed best with an ROC-AUC score of 0.54551. This is far below the results where sampling techniques are used, and sometimes even lower than the sampling baseline.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_4 (InputLayer)         (None, 11)                0         
_________________________________________________________________
dense_10 (Dense)             (None, 64)                768       
_________________________________________________________________
dropout_7 (Dropout)          (None, 64)                0         
_________________________________________________________________
dense_11 (Dense)             (None, 32)                2080      
_________________________________________________________________
dropout_8 (Dropout)          (None, 32)                0         
_________________________________________________________________
dense_12 (Dense)             (None, 1)                 33        
=================================================================
Total params: 2,881
Trainable params: 2,881
Non-trainable params: 0
```

Network 1: Summary of implemented feed-forward network.

Autoencoder Network

The second network is the autoencoder network. In Network 2, the implemented autoencoder network is summarized. To optimize the parameters in the autoencoder network, a small grid search is implemented to find the best parameters for the model. Due to the enormous computation time required by the deep learning models, only a small set of parameters is tested. In this case, the batch size and learning rate are optimized via the grid search. For the batch size, the following values are tested: 32, 64, 128, 256, 512, and 1024. For the learning rates, the tested parameters are as follows: 0.0005, 0.001, 0.005, 0.01, 0.02, and 0.05. The best ROC-AUC scores on the test set are achieved with a batch size of 32 and a learning rate equal to 0.005. The achieved ROC-AUC score is 0.71982. Although this is still lower than the results from the sampling models, it already performs better than the feed-forward network.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_2 (InputLayer)         (None, 11)                0         
_________________________________________________________________
dense_4 (Dense)              (None, 8)                 96        
_________________________________________________________________
dense_5 (Dense)              (None, 4)                 36        
_________________________________________________________________
dense_6 (Dense)              (None, 8)                 40        
_________________________________________________________________
dense_7 (Dense)              (None, 11)                99        
=================================================================
Total params: 271
Trainable params: 271
Non-trainable params: 0
```

Network 2: Summary of autoencoder network.

Although both deep learning models are outperformed by the sampling models, a number of possibilities remain for them to perform better. Implementing dropout layers and some sort of regularizers could prevent the model from overfitting. In addition, several parameters can be tuned further. Many companies have also based their fraud detection algorithms on autoencoder or deep learning models. So, there should be a great deal potential, although this cannot be demonstrated by this results.

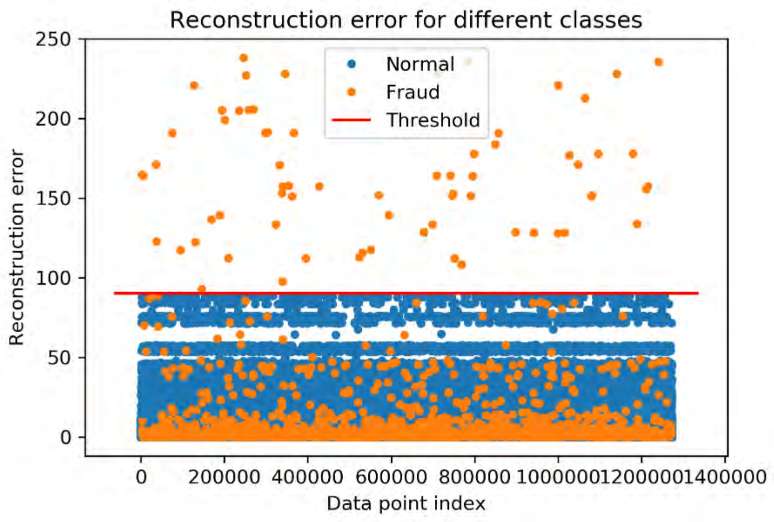

Figure 14: Reconstruction error (mean squared error) for each data point in the test set. Blue points are legal transactions, while orange points are fraud ones.

Diving deeper into the autoencoder results reveals an interesting detail. In Figure 14, the reconstruction error of the autoencoder model is illustrated for each data point in the test set. As the model is trained on the legal transactions, the assumption is that the error would be low for the legal transactions and higher for the fraud transactions. As visible in the figure, however, this is not the case. Although a part of the fraud transactions do indeed possess a higher error than the threshold value, most of them are below the threshold line. Lowering the threshold value makes no sense, as the legal transactions are then classified as fraud.

6. Discussion and Conclusion

During this research, the primary goal was to review various methods that could improve the classification performance in the case of imbalanced data. Now, we can conclude that methods exist that can improve the classification performance, but also that neither of the discussed methods work under all conditions. The first conclusion that could be made following the experiments is that resampling always improves the classification performance. Although this sounds promising, it is not clear that a particular method works under all conditions. Instead, it depends on the imbalance of the data, scaling and reduction method, and the used classifier. So, a clear decision cannot be made regarding which methods should be used. In practice, several methods should be evaluated to achieve the best performance. After evaluating three distance-based reduction techniques — T-Link, ENN, and CNN — it seems that only CNN adds significant value in performance by itself. The results achieved without reduction technique are almost always close to or better than the results obtained with some reduction. Although CNN adds some value by itself, when combined with an oversampling method, it does not perform better in any case. Regarding the rescaling method, standardizing performs better in 74,5% of the cases. An interesting observation was that the difference in classification performance is lower than 0,01 in all cases where normalization performed better. In 57,8% of the cases where standardization works better, the difference in performance is greater than 0,01. So, in this case, it can be concluded that standardization works better than normalization.

An alternative to the use of preprocessing techniques in combination with a classifier is the use of deep learning. Although these techniques are promising, it was not possible to achieve better results compared to using the classic sampling approach. Unless results were not better than the baseline values obtained with the logistic classification, random forest, and gradient boosting classifiers, there remains a number of opportunities to achieve competitive results. As deep learning requires a great deal of computation power, it was not possible to train more complex models. So, when more time and/or more computation power is available, it is likely that the models will obtain better performance. Furthermore, more advanced strategies like combining multiple methods into ensembles and tuning several parameters to achieve better results were only performed on a simple level. Although the idea was to make sampling methods obsolete with neural networks, it would also be possible to combine both techniques. Generating more data with a method like SMOTE (or other data-augmentation techniques) could improve the model further.

With all this research conducted, is it possible to determine that strategies exist that handle the imbalanced data problem. One works better than the other, but at least we can say that sampling is always worthwhile to use. However, although sampling works, it is not self-evident which sampling strategy works best. This makes it virtually impossible to conclude in advance that one technique will always work better than another.

Bibliography

[1] J. W. M. Bhattacharya, "Intelligent financial fraud detection: A comprehensive review," *Computers & Security,* vol. 57, pp. 47-66, 08-09-2015 2015.

[2] E. A. L.-R. A. E. S. Axelsson, "PAYSIM: A FINANCIAL MOBILE MONEY SIMULATOR FOR FRAUD DETECTION," presented at the Conference: 28th European Modeling and Simulation Symposium 2016 (EMSS 2016), Larnaca, Cyprus, 2016.

[3] P. Phoungphol, "A Classification Framework for Imbalanced Data," Doctor of Philosophy (PhD), Computer Science, Georgia State University, 2013.

[4] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.,* vol. 6, no. 5, pp. 429-449, 2002.

[5] H. H. E. A. Garcia, "Learning from Imbalanced Data," *IEEE Transactions on Knowledge and Data Engineering* vol. 21, no. 9, pp. 1263 - 1284, 26-09-2009 2009.

[6] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *J. Artif. Int. Res.,* vol. 16, no. 1, pp. 321-357, 2002.

[7] T. R. Hoens and N. V. Chawla, "Imbalanced Datasets: From Sampling to Classifiers," in *Imbalanced Learning*: John Wiley & Sons, Inc., 2013, pp. 43-59.

[8] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning," in *Advances in Intelligent Computing: International Conference on Intelligent Computing, ICIC 2005, Hefei, China, August 23-26, 2005, Proceedings, Part I*, D.-S. Huang, X.-P. Zhang, and G.-B. Huang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 878-887.

[9] H. Haibo, B. Yang, E. A. Garcia, and L. Shutao, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2008, pp. 1322-1328.

[10] S. Barua, M. M. Islam, X. Yao, and K. Murase, "MWMOTE--Majority Weighted Minority Oversampling Technique for Imbalanced Data Set Learning," *IEEE Transactions on Knowledge and Data Engineering,* vol. 26, no. 2, pp. 405-425, 2014.

[11] D. L. Wilson, *Asymptotic Properties of Nearest Neighbor Rules Using Edited Data*. 1972, pp. 408-421.

[12] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *Trans. Sys. Man Cyber. Part B,* vol. 39, no. 2, pp. 539-550, 2009.

[13] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Chapman \& Hall/CRC, 2012, p. 236.

[14] A. D. Pozzolo, O. Caelen, S. Waterschoot, and G. Bontempi, "Racing for Unbalanced Methods Selection," presented at the Proceedings of the 14th International Conference on Intelligent Data Engineering and Automated Learning --- IDEAL 2013 - Volume 8206, Hefei, China, 2013.

[15] N. V. Chawla, "Data Mining for Imbalanced Datasets: An Overview," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Boston, MA: Springer US, 2005, pp. 853-867.

[16] U. Kaymak, A. Ben-David, and R. Potharst, "The AUK: A simple alternative to the AUC," *Engineering Applications of Artificial Intelligence,* vol. 25, no. 5, pp. 1082-1089, 2012/08/01/ 2012.
[17] (09-07-2017). *What does AUC stand for and what is it?* Available: https://stats.stackexchange.com/questions/132777/what-does-auc-stand-for-and-what-is-it
[18] T. Saito. (2015). *Introduction to the precision-recall plot*. Available: https://classeval.wordpress.com/introduction/introduction-to-the-precision-recall-plot/
[19] 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.
[20] D. J. Hand, "Measuring classifier performance: a coherent alternative to the area under the ROC curve," *Machine Learning,* journal article vol. 77, no. 1, pp. 103-123, October 01 2009.
[21] D. J. H. C. Anagnostopoulos, N.M. Adams, "Measuring classification performance: the hmeasure package," 10 september 2012. Accessed on: 28-08-2017
[22] D. Berrar, "On the Noise Resilience of Ranking Measures," in *Neural Information Processing: 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16–21, 2016, Proceedings, Part II,* A. Hirose, S. Ozawa, K. Doya, K. Ikeda, M. Lee, and D. Liu, Eds. Cham: Springer International Publishing, 2016, pp. 47-55.
[23] D. J. Hand and C. Anagnostopoulos, "A better Beta for the H measure of classification performance," *Pattern Recognition Letters,* vol. 40, pp. 41-46, 2014/04/15/ 2014.
[24] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia Medica,* vol. 22, no. 3, pp. 276-282, 2012.
[25] "Synthetic Financial Datasets For Fraud Detection," 2 ed, 2017.
[26] E. A. Lopez-Rojas, "Applying Simulation to the Problem of Detecting Financial Fraud," PhD Thesis, Department of Computer Science and Engineering, Blekinge Tekniska Högskola, Karlskrona, 2016.
[27] A. Roy, J. Sun, R. Mahoney, L. Alonzi, S. Adams, and P. Beling, "Deep learning detecting fraud in credit card transactions," in *2018 Systems and Information Engineering Design Symposium (SIEDS)*, 2018, pp. 129-134.
[28] T. K. Ho, "Random decision forests," presented at the Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1, 1995.
[29] L. Breiman, "Random Forests," *Mach. Learn.,* vol. 45, no. 1, pp. 5-32, 2001.
[30] M. N. Bernstein, "Random Forest," 21-08-2015 2015.
[31] D. Benyamin, "A Gentle Introduction to Random Forests, Ensembles, and Performance Metrics in a Commercial System," ed, 2012.
[32] P. Grover, "Gradient Boosting from scratch," 2017.
[33] Quantdare. (2016). *What is the difference between Bagging and Boosting?* Available: https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/
[34] M. Bernstein, "The logistic regression model," 2016.
[35] U. Karn, "A Quick Introduction to Neural Networks," 2016.

[36] J. Le, "The 8 Neural Network Architectures Machine Learning Researchers Need to Learn," 2018.

[37] J. C. J. Sanchez, "Methodology for Fraud Detection in credit card transactions with small manual labelling effort (Keras / MLP / Autoencoder)," 2018.

Appendix A: Results Random Oversampling

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (9.750)	**ROC-AUC**	0,55373	0,89695	0,74555	0,7291	0,89537	0,6971
	F1	0,69143	0,90658	0,79695	0,78684	0,90528	0,76747
1 (19.500)	**ROC-AUC**	0,5749	0,90137	0,74585	0,76446	0,89979	0,74679
	F1	0,7017	0,91022	0,79734	0,80934	0,90892	0,79794
2 (39.000)	**ROC-AUC**	0,58944	0,90863	0,90507	0,7843	0,90548	0,91333
	F1	0,70893	0,91628	0,91328	0,82253	0,91364	0,92023
3 (78.000)	**ROC-AUC**	0,63776	0,90769	0,9132	0,81696	0,89821	0,92777
	F1	0,73407	0,91548	0,9201	0,84518	0,90761	0,93261
4 (156.250)	**ROC-AUC**	0,59729	**0,90927**	0,92279	0,84054	0,90643	0,94116
	F1	0,71288	**0,91681**	0,92827	0,86229	0,91443	0,94439
5 (312,500)	**ROC-AUC**	0,65886	0,90643	0,93253	0,85423	0,90453	0,95438
	F1	0,74535	0,91443	0,93671	0,87248	0,91284	0,95631
6 (625.000)	**ROC-AUC**	0,738	0,90579	0,94364	0,86858	0,90295	0,97043
	F1	0,79156	0,9139	0,94647	0,88334	0,91153	0,9712
7 (1.250.000)	**ROC-AUC**	0,76506	0,90643	0,95362	0,90497	0,90295	0,98629
	F1	0,80442	0,91443	0,95547	0,91227	0,91153	0,98639
8 (2.500.000)	**ROC-AUC**	0,84203	0,90232	0,96692	0,93364	0,90674	**0,9915**
	F1	**0,8548**	0,91101	0,96767	0,93618	0,91469	**0,99148**
9 (5.000.000)	**ROC-AUC**	0,8384	0,90863	0,97124	0,94726	**0,90706**	0,99045
	F1	0,83328	0,91628	0,97159	0,94741	**0,91496**	0,99038
10 (1:1)	**ROC-AUC**	**0,85049**	0,90295	**0,97164**	**0,94747**	0,90453	0,99056
	F1	0,84616	0,91153	**0,972**	**0,94756**	0,91285	0,99048

Table 12: Experiment results for random oversampling with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. Best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (9.750)	**ROC-AUC**	0,70408	0,97088	0,97895	0,84427	0,9819	0,9932
	F1	0,77129	0,97165	0,97934	0,86491	0,98215	0,99321
1 (19.500)	**ROC-AUC**	**0,84558**	0,97459	0,9936	0,90574	0,99146	0,9929
	F1	**0,85761**	0,97516	0,99359	0,91274	0,9915	0,99288
2 (39.000)	**ROC-AUC**	0,78304	0,98188	0,99111	**0,9513**	0,98635	0,99132
	F1	0,72292	0,98216	0,99106	**0,95105**	0,98647	0,99126
10 (1:1)	**ROC-AUC**	0,81873	0,97754	**0,99446**	0,90082	0,98886	**0,99331**
	F1	0,84149	0,97798	**0,99446**	0,90873	0,98893	**0,99328**

Table 13: Experiment results for random oversampling with CNN as a reduction technique. By default, there are 16.475 and 18.958 legal transactions (for the normalized and standardized data, respectively) versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (9.750)	**ROC-AUC**	0,55278	0,90422	0,74829	0,73226	0,90074	0,65835
	F1	0,69098	0,91259	0,79886	0,78879	0,9097	0,74366
1 (19.500)	**ROC-AUC**	0,57237	0,90516	0,74868	0,75972	0,90516	0,74741
	F1	0,70046	0,91337	0,79914	0,80625	0,91337	0,79833
2 (39.000)	**ROC-AUC**	0,59385	0,90706	0,90601	0,78618	0,90674	0,91363
	F1	0,71116	0,91496	0,91406	0,82379	0,91469	0,92049
3 (78.000)	**ROC-AUC**	0,59985	0,91148	0,91605	0,79556	0,90705	0,92761
	F1	0,7142	0,91867	0,92252	0,83019	0,91495	0,93245
4 (156.250)	**ROC-AUC**	0,59602	0,91085	0,92436	0,83275	0,90516	0,94116
	F1	0,71224	0,91814	0,92963	0,85655	0,91337	0,94439
5 (312,500)	**ROC-AUC**	0,6968	0,90989	0,93509	0,8566	0,90547	0,95808
	F1	0,76708	0,91734	0,93896	0,87425	0,91363	0,9597
6 (625.000)	**ROC-AUC**	0,69384	0,908	0,94386	0,88915	**0,91021**	0,9688
	F1	0,7645	0,91574	0,9467	0,89954	**0,9176**	0,96966
7 (1.250.000)	**ROC-AUC**	0,7635	**0,91306**	0,95486	0,91191	0,9039	0,98753
	F1	0,80328	**0,92001**	0,9566	0,91796	0,91232	0,98761
8 (2.500.000)	**ROC-AUC**	0,85573	0,90548	0,96671	0,93229	0,90294	**0,99106**
	F1	0,86675	0,91364	0,96745	0,93492	0,91153	**0,99104**
9 (5.000.000)	**ROC-AUC**	0,85503	0,91148	0,97161	**0,95092**	0,90548	0,99052
	F1	0,85153	0,91868	0,97196	**0,95087**	0,91364	0,99044
10 (1:1)	**ROC-AUC**	**0,87104**	0,908	**0,9718**	0,94708	0,90642	0,98988
	F1	**0,86889**	0,91575	**0,97214**	0,94716	0,91442	0,9898

Table 14: Experiment results for random oversampling with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data, respectively) versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (9.750)	**ROC-AUC**	0,55436	0,89822	0,74422	0,72784	0,89758	0,6971
	F1	0,69173	0,90762	0,79629	0,78605	0,90709	0,76747
1 (19.500)	**ROC-AUC**	0,57269	0,90548	0,7449	0,75435	0,90232	0,74615
	F1	0,70062	0,91364	0,79674	0,80277	0,91101	0,79753
2 (39.000)	**ROC-AUC**	0,55562	0,90421	0,90632	0,78272	0,90611	0,91459
	F1	0,69234	0,91258	0,91433	0,82146	0,91416	0,9213
3 (78.000)	**ROC-AUC**	0,6447	0,9039	0,91604	0,816	0,90263	0,9265
	F1	0,73782	0,91232	0,92251	0,8445	0,91126	0,93151
4 (156.250)	**ROC-AUC**	0,59571	**0,91148**	0,92404	0,84148	**0,90864**	0,94089
	F1	0,71208	**0,91867**	0,92935	0,86298	**0,91628**	0,94416
5 (312,500)	**ROC-AUC**	0,65886	0,90611	0,93402	0,85537	0,90643	0,95599
	F1	0,74535	0,91416	0,938	0,87333	0,91443	0,95779
6 (625.000)	**ROC-AUC**	0,74505	0,91085	0,94321	0,88863	0,90326	0,97012
	F1	0,79606	0,91814	0,94611	0,89914	0,91179	0,9709
7 (1.250.000)	**ROC-AUC**	0,74962	0,90958	0,95452	0,91135	0,90231	0,98588
	F1	0,79315	0,91708	0,95629	0,91749	0,911	0,986
8 (2.500.000)	**ROC-AUC**	**0,87828**	0,90263	0,96629	0,93495	0,90516	0,99041
	F1	**0,88722**	0,91127	0,96705	0,93725	0,91337	0,99039
9 (5.000.000)	**ROC-AUC**	0,85801	0,90864	0,97089	0,94548	0,90547	**0,99068**
	F1	0,85473	0,91628	0,97127	0,94567	0,91363	**0,9906**
10 (1:1)	**ROC-AUC**	0,83291	0,908	**0,97209**	**0,95143**	0,90579	0,99045
	F1	0,82663	0,91575	**0,97242**	**0,95131**	0,9139	0,99037

Table 15: Experiment results for random oversampling with T-Link as a reduction technique. By default, there are 5.082.736 and 5.082.814 legal transactions (for the normalized and standardized data, respectively) versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. The best results are indicated in bold.

Appendix B: Results Random Under-sampling

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,52023	0,896	0,62456	0,71237	0,89411	0,79421
	F1	0,67578	0,9058	0,72681	0,77661	0,90425	0,82932
1 (2.500.000)	**ROC-AUC**	0,54551	0,90832	0,81907	0,73415	0,90358	0,82538
	F1	0,68753	0,91601	0,84675	0,78997	0,91206	0,8513
2 (1.250.000)	**ROC-AUC**	0,55594	0,91872	0,88013	0,75434	0,91809	0,88743
	F1	0,69249	0,92482	0,89295	0,80277	0,92428	0,89881
3 (625.000)	**ROC-AUC**	0,56194	0,92688	0,92371	0,77198	0,93574	0,92276
	F1	0,69538	0,93184	0,92911	0,81427	0,9396	0,92829
4 (312.500)	**ROC-AUC**	0,57963	0,94666	0,934	0,79553	0,94728	0,93338
	F1	0,70404	0,94934	0,93806	0,83016	0,9499	0,93751
5 (156.250)	**ROC-AUC**	0,59744	0,9597	0,94952	0,81204	0,95936	0,9485
	F1	0,71277	0,96123	0,9519	0,84164	0,96091	0,95096
6 (78.000)	**ROC-AUC**	0,64713	0,97467	0,96383	0,84048	0,97174	0,96603
	F1	0,73838	0,97525	0,96502	0,86214	0,97246	0,96708
7 (39.000)	**ROC-AUC**	0,71241	0,97732	0,98234	0,86714	0,97977	0,98131
	F1	0,77297	0,97774	0,98257	0,88217	0,9801	0,98157
8 (19.500)	**ROC-AUC**	0,77371	0,99057	0,99088	0,8965	0,98961	0,9897
	F1	0,80416	0,99061	0,99088	0,90498	0,98966	0,98972
9 (9.750)	**ROC-AUC**	**0,83339**	**0,99385**	**0,99138**	0,91972	**0,99369**	**0,99069**
	F1	**0,83844**	**0,99383**	**0,99133**	0,92298	**0,99367**	**0,99064**
10 (1:1)	**ROC-AUC**	0,80312	0,99193	0,9903	**0,92616**	0,99159	0,99029
	F1	0,7909	0,9919	0,99022	**0,92686**	0,99156	0,99021

Table 16: Experiment results for random under-sampling with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. Only non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	**ROC-AUC**	0,53887	0,90233	0,64949	0,71521	0,89569	0,74315
	F1	0,6844	0,91102	0,73863	0,77833	0,90554	0,78185
1 (2.500.000)	**ROC-AUC**	0,54488	0,90769	0,61091	0,73131	0,90452	0,79791
	F1	0,68723	0,91548	0,71989	0,7882	0,91284	0,83184
2 (1.250.000)	**ROC-AUC**	0,55436	0,91556	0,90859	0,74771	0,92094	0,88901
	F1	0,69173	0,92212	0,91623	0,79852	0,92672	0,90009
3 (625.000)	**ROC-AUC**	0,56226	0,93919	0,92117	0,77513	0,9332	0,92212
	F1	0,69553	0,94266	0,92691	0,81636	0,93737	0,92773
4 (312.500)	**ROC-AUC**	0,57964	0,94948	0,93462	0,79742	0,95141	0,934
	F1	0,70404	0,95188	0,9386	0,83146	0,95364	0,93806
5 (156.250)	**ROC-AUC**	0,59683	0,9682	0,94828	0,82022	0,96125	0,95265
	F1	0,71246	0,96915	0,95077	0,84746	0,96266	0,95474
6 (78.000)	**ROC-AUC**	0,63038	0,98006	0,9658	0,84016	0,97602	0,96484
	F1	0,72932	0,98042	0,96687	0,86191	0,97654	0,96597
7 (39.000)	**ROC-AUC**	0,71327	0,98169	0,97802	0,86553	0,98315	0,98085
	F1	0,77358	0,98195	0,97839	0,88091	0,98338	0,98113
8 (19.500)	**ROC-AUC**	0,77989	0,98904	0,98933	0,89517	0,98996	0,989
	F1	0,80918	0,9891	0,98935	0,90387	0,99	0,98903
9 (9.750)	**ROC-AUC**	**0,83106**	0,98892	**0,99026**	0,91884	0,99251	**0,99144**
	F1	**0,8359**	0,98892	**0,9902**	0,92221	**0,99251**	**0,99138**
10 (1:1)	**ROC-AUC**	0,80078	**0,99227**	0,98896	**0,92597**	**0,99252**	0,98997
	F1	0,78817	**0,99223**	0,98886	**0,92675**	0,99248	0,98988

Table 17: Experiment results for random under-sampling with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data, respectively) versus 6.631 fraudulent ones. Only non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,51928	0,8938	0,65929	0,71047	0,89601	0,63263
	F1	0,67535	0,90399	0,74406	0,77547	0,9058	0,71867
1 (2.500.000)	**ROC-AUC**	0,54267	0,90484	0,76033	0,71488	0,90137	0,60902
	F1	0,68618	0,91311	0,80663	0,77813	0,91022	0,71891
2 (1.250.000)	**ROC-AUC**	0,55467	0,92061	0,90544	0,75371	0,91808	0,8909
	F1	0,69188	0,92644	0,9136	0,80236	0,92427	0,90162
3 (625.000)	**ROC-AUC**	0,55562	0,93255	0,92054	0,77544	0,93162	0,92307
	F1	0,69234	0,9368	0,92637	0,81657	0,93598	0,92855
4 (312.500)	**ROC-AUC**	0,56098	0,94724	0,93115	0,79615	0,95106	0,93146
	F1	0,69491	0,94986	0,93555	0,83058	0,95332	0,93583
5 (156.250)	**ROC-AUC**	0,60916	0,96695	0,94887	0,82396	0,96155	0,94885
	F1	0,71878	0,96798	0,95131	0,85015	0,96293	0,95129
6 (78.000)	**ROC-AUC**	0,64705	0,9748	0,96388	0,84418	0,97275	0,96502
	F1	0,73831	0,97538	0,96507	0,86489	0,97342	0,96612
7 (39.000)	**ROC-AUC**	0,71327	0,98022	0,98146	0,86826	0,97847	0,9818
	F1	0,77335	0,98054	0,98171	0,88302	0,97886	0,98205
8 (19.500)	**ROC-AUC**	0,7769	0,99012	**0,99047**	0,89675	0,99015	0,98824
	F1	0,80671	0,99016	**0,99047**	0,90518	0,99019	0,98827
9 (9.750)	**ROC-AUC**	**0,83174**	0,98904	0,99045	0,9206	0,99236	**0,99089**
	F1	**0,83698**	0,98906	0,99039	0,92375	0,99234	**0,99083**
10 (1:1)	**ROC-AUC**	0,81103	**0,99223**	0,98913	**0,92636**	**0,99273**	0,98941
	F1	0,80124	**0,99219**	0,98903	**0,92698**	**0,99269**	0,98931

Table 18: Experiment results for random under-sampling with T-Link as a reduction technique. By default, there are 5.082.736 and 5.082.814 legal transactions (for the normalized and standardized data, respectively) versus 6.631 fraudulent ones. Only non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

Appendix C: Results SMOTE

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (9.750)	**ROC-AUC**	0,9039	0,78499	0,54361	0,90453	0,6317	0,73164
	F1	0,91233	0,82301	0,68663	0,91285	0,7308	0,78841
1 (19.500)	**ROC-AUC**	0,92	0,86329	0,59386	0,91779	0,84593	0,76414
	F1	0,92592	0,8797	0,71116	0,92403	0,86647	0,80913
2 (39.000)	**ROC-AUC**	0,93514	0,90252	0,61724	0,9364	0,91774	0,77957
	F1	0,93909	0,91116	0,72318	0,9402	0,92398	0,81934
3 (78.000)	**ROC-AUC**	0,9572	0,91506	0,61092	0,95911	0,92524	0,80253
	F1	0,95895	0,92168	0,71989	0,96071	0,93041	0,83502
4 (156.250)	**ROC-AUC**	0,96442	0,92185	0,64876	0,96819	0,93862	0,8294
	F1	0,96563	0,92746	0,74002	0,96916	0,94213	0,85412
5 (312,500)	**ROC-AUC**	0,97573	0,93566	0,70038	0,9776	0,95276	0,86551
	F1	0,97629	0,93943	0,76924	0,97808	0,95483	0,88109
6 (625.000)	**ROC-AUC**	0,97787	0,93401	0,77324	0,98513	0,97064	0,89624
	F1	0,97833	0,93796	0,81454	0,98533	0,9714	0,90538
7 (1.250.000)	**ROC-AUC**	0,98226	0,9425	0,79155	0,98886	0,98357	0,91538
	F1	0,98255	0,94534	0,82405	0,98897	0,98375	0,92094
8 (2.500.000)	**ROC-AUC**	**0,9847**	**0,94646**	**0,87245**	0,9866	**0,98858**	0,93087
	F1	**0,98492**	**0,94889**	**0,88331**	0,98676	**0,9886**	0,93382
9 (5.000.000)	**ROC-AUC**	0,98055	0,94292	0,86898	**0,99005**	0,98749	**0,94717**
	F1	0,9809	0,94519	0,8677	**0,99014**	0,98745	**0,94768**
10 (1:1)	**ROC-AUC**	0,98274	0,93971	0,83744	0,98943	0,9868	0,94661
	F1	0,98301	0,94211	0,83211	0,98953	0,98676	**0,94717**

Table 19: Experiment results for SMOTE with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (9.750)	**ROC-AUC**	0,97446	0,97744	0,70335	0,99239	0,99206	0,84583
	F1	0,97504	0,9779	0,7707	0,99241	0,99208	0,86607
1 (19.500)	**ROC-AUC**	0,99101	**0,99354**	**0,85015**	0,99245	0,99224	0,90679
	F1	0,99106	**0,99355**	**0,86304**	0,99246	0,99222	0,91369
2 (39.000)	**ROC-AUC**	**0,99307**	0,992	0,78309	**0,99452**	0,99067	**0,94976**
	F1	**0,99308**	0,99196	0,72301	**0,99451**	0,99061	**0,94974**
10 (1:1)	**ROC-AUC**	0,98249	0,99345	0,82223	0,99264	**0,99247**	0,90542
	F1	0,98273	0,99346	0,84494	0,99265	**0,99245**	0,91257

Table 20: Experiment results for SMOTE with CNN as a reduction technique. By default, there are 16.475 and 18.958 legal transactions (for the normalized and standardized data, respectively) versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (9.750)	**ROC-AUC**	0,90137	0,78076	0,55531	0,90643	0,69971	0,72626
	F1	0,91022	0,82012	0,69219	0,91443	0,76905	0,78508
1 (19.500)	**ROC-AUC**	0,92221	0,87791	0,59481	0,92348	0,84151	0,76320
	F1	0,92782	0,89117	0,71164	0,92892	0,86316	0,80851
2 (39.000)	**ROC-AUC**	0,93514	0,90064	0,61249	0,94209	0,91489	0,79185
	F1	0,93908	0,9096	0,72071	0,94526	0,92155	0,82766
3 (78.000)	**ROC-AUC**	0,9509	0,91599	0,66491	0,95499	0,92649	0,8003
	F1	0,95318	0,92247	0,74898	0,95692	0,93150	0,83346
4 (156.250)	**ROC-AUC**	0,96946	0,92184	0,63772	0,97231	0,93792	0,83805
	F1	0,97035	0,92745	0,73403	0,97304	0,94150	0,86044
5 (312,500)	**ROC-AUC**	0,97256	0,93563	0,70606	0,97538	0,95275	0,86593
	F1	0,97328	0,93939	0,77261	0,97596	0,95481	0,88144
6 (625.000)	**ROC-AUC**	0,98070	0,93401	0,73644	0,98353	0,97224	0,89527
	F1	0,98106	0,93796	0,79059	0,98379	0,97291	0,90452
7 (1.250.000)	**ROC-AUC**	0,98600	0,94374	0,80507	0,98191	0,9829	0,91588
	F1	0,98618	0,94645	0,83435	0,98221	0,98309	0,92134
8 (2.500.000)	**ROC-AUC**	**0,98722**	**0,94744**	0,85460	0,99005	**0,98858**	0,93738
	F1	**0,98737**	**0,94979**	0,86649	0,99014	**0,98859**	0,9396
9 (5.000.000)	**ROC-AUC**	0,97878	0,94263	0,89995	**0,99160**	0,98732	0,94709
	F1	0,97921	0,94494	0,90294	**0,99166**	0,98728	0,94760
10 (1:1)	**ROC-AUC**	0,98244	0,94268	**0,90406**	0,99097	0,98795	**0,94749**
	F1	0,98273	0,94499	**0,90673**	0,99104	0,98791	**0,94795**

Table 21: Experiment results for SMOTE with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data, respectively) versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			**STD**		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (9.750)	**ROC-AUC**	0,90390	0,78626	0,55499	0,90233	0,63076	0,73195
	F1	0,91233	0,82387	0,69204	0,91101	0,73029	0,78860
1 (19.500)	**ROC-AUC**	0,91873	0,85818	0,57237	0,91842	0,84594	0,76509
	F1	0,92484	0,87574	0,70046	0,92457	0,86647	0,80975
2 (39.000)	**ROC-AUC**	0,93577	0,90128	0,62577	0,94241	0,91552	0,79628
	F1	0,93964	0,91013	0,72767	0,94554	0,92209	0,83071
3 (78.000)	**ROC-AUC**	0,94964	0,91505	0,59259	0,95753	0,92712	0,81475
	F1	0,95205	0,92167	0,71052	0,95925	0,93205	0,84361
4 (156.250)	**ROC-AUC**	0,97075	0,92120	0,64214	0,96947	0,93764	0,83027
	F1	0,97157	0,92690	0,73642	0,97036	0,94126	0,85473
5 (312,500)	**ROC-AUC**	0,97574	0,93535	0,70607	0,97603	0,95215	0,86581
	F1	0,97631	0,93914	0,77262	0,97658	0,95427	0,88132
6 (625.000)	**ROC-AUC**	**0,98293**	0,93339	0,77324	0,98451	0,97222	0,89534
	F1	**0,98320**	0,93741	0,81453	0,98474	0,97289	0,90460
7 (1.250.000)	**ROC-AUC**	0,98255	0,94348	0,80200	**0,99012**	0,98191	0,91327
	F1	0,98284	0,94622	0,83211	**0,99021**	0,98213	0,91919
8 (2.500.000)	**ROC-AUC**	0,98281	**0,94703**	0,83649	0,98690	**0,98826**	0,93651
	F1	0,98309	**0,94941**	0,85050	0,98706	**0,98828**	0,93881
9 (5.000.000)	**ROC-AUC**	0,98276	0,94291	0,87250	0,98941	0,98717	**0,94840**
	F1	0,98304	0,94519	0,87203	0,98951	0,98712	**0,94883**
10 (1:1)	**ROC-AUC**	0,98087	0,94043	**0,90274**	0,98938	0,98738	0,94799
	F1	0,98121	0,94288	**0,90572**	0,98948	0,98734	**0,94840**

Table 22: Experiment results for SMOTE with T-Link as a reduction technique. By default, there are 5.082.736 and 5.082.814 legal transactions (for the normalized and standardized data, respectively) versus 6.631 fraudulent ones. Only fraudulent transactions are oversampled. The number of oversampled transactions is provided between brackets. The best results are indicated in bold.

Appendix D: Results SMOTE + RUS

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,90580	0,53413	0,81031	0,90390	0,73289	0,63170
	F1	0,91391	0,68219	0,84054	0,91232	0,78919	0,73080
1 (2.500.000)	ROC-AUC	0,91842	0,55689	0,75312	0,91526	0,75404	0,84842
	F1	0,92457	0,69295	0,80199	0,92187	0,80257	0,86833
2 (1.250.000)	ROC-AUC	0,93101	0,57016	0,91234	0,93133	0,77137	0,91680
	F1	0,93545	0,69938	0,91939	0,93574	0,81387	0,92318
3 (625.000)	ROC-AUC	0,94104	0,56573	0,92930	0,94482	0,79116	0,93028
	F1	0,94430	0,69722	0,93394	0,94769	0,82717	0,93480
4 (312.500)	ROC-AUC	0,96013	0,58560	0,94141	0,95697	0,80741	0,93856
	F1	0,96163	0,70699	0,94461	0,95872	0,83840	0,94206
4 (156.250)	ROC-AUC	**0,96616**	**0,65602**	**0,95496**	**0,96935**	**0,84066**	**0,95893**
	F1	**0,96722**	**0,74377**	**0,95684**	**0,97022**	**0,86232**	**0,96048**

Table 23: Results for SMOTE + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 9.750 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,92822	0,56668	0,84742	0,92317	0,76541	0,75904
	F1	0,93303	0,69768	0,86757	0,92865	0,80996	0,80579
1 (2.500.000)	ROC-AUC	0,92977	0,59227	0,90570	0,93640	0,78146	0,91568
	F1	0,93437	0,71036	0,91381	0,94020	0,82061	0,92216
2 (1.250.000)	ROC-AUC	0,94328	0,60555	0,92302	0,94770	0,80657	0,92926
	F1	0,94631	0,71712	0,92850	0,95029	0,83783	0,93390
3 (625.000)	ROC-AUC	0,95673	0,58970	0,93950	0,96335	0,82121	0,93956
	F1	0,95850	0,70904	0,94290	0,96462	0,84819	0,94296
4 (312.500)	ROC-AUC	0,96819	0,68104	0,95086	0,97258	0,84186	0,95272
	F1	0,96914	0,75792	0,95309	0,97328	0,86320	0,95478
4 (156.250)	ROC-AUC	**0,97851**	**0,70026**	**0,96692**	**0,97723**	**0,87268**	**0,97235**
	F1	**0,97892**	**0,76832**	**0,96788**	**0,97769**	**0,88656**	**0,97300**

Table 24: Results for SMOTE + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 19.500 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,93766	0,58312	0,89969	0,94335	0,78744	0,91141
	F1	0,94131	0,70577	0,90882	0,94638	0,82465	0,91861
1 (2.500.000)	ROC-AUC	0,95277	0,64376	0,91569	0,95592	0,82045	0,92838
	F1	0,95489	0,73732	0,92221	0,95777	0,84769	0,93314
2 (1.250.000)	ROC-AUC	0,96276	0,64497	0,93451	0,96116	0,83904	0,93672
	F1	0,96408	0,73796	0,93849	0,96259	0,86118	0,94044
3 (625.000)	ROC-AUC	0,97390	0,64675	0,95152	0,97169	0,83754	0,95276
	F1	0,97454	0,73863	0,95369	0,97245	0,86001	0,95482
4 (312.500)	ROC-AUC	0,98028	0,74227	0,96635	0,97898	0,88298	0,97285
	F1	0,98063	0,79431	0,96736	0,97938	0,89469	0,97349
4 (156.250)	ROC-AUC	**0,98600**	**0,76967**	**0,98260**	**0,98755**	**0,89544**	**0,98583**
	F1	**0,98616**	**0,80757**	**0,98280**	**0,98767**	**0,90448**	**0,98594**

Table 25: Results for SMOTE + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 39.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,95722	0,61218	0,91788	0,96289	0,82009	0,92807
	F1	0,95896	0,72055	0,92408	0,96421	0,84742	0,93287
1 (2.500.000)	ROC-AUC	0,96659	0,64403	0,93032	0,96405	0,82900	0,93957
	F1	0,96766	0,73744	0,93480	0,96528	0,85381	0,94298
2 (1.250.000)	ROC-AUC	0,97677	0,65310	0,94105	0,97841	0,85388	0,95338
	F1	0,97728	0,74213	0,94424	0,97885	0,87220	0,95539
3 (625.000)	ROC-AUC	0,98385	0,70619	0,96273	0,97875	0,88951	0,97217
	F1	0,98409	0,77184	0,96396	0,97917	0,89990	0,97284
4 (312.500)	ROC-AUC	0,99007	0,75182	0,98167	0,98694	0,90758	0,98255
	F1	0,99014	0,79437	0,98190	0,98709	0,91438	0,98275
4 (156.250)	ROC-AUC	**0,99169**	**0,85143**	**0,98825**	**0,99157**	**0,92924**	**0,98915**
	F1	**0,99174**	**0,86351**	**0,98828**	**0,99161**	**0,93240**	**0,98917**

Table 26: Results for SMOTE + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 78.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,96725	0,64308	0,92246	0,96913	0,83688	0,93861
	F1	0,96828	0,73693	0,92798	0,97005	0,85959	0,94211
1 (2.500.000)	**ROC-AUC**	0,97311	0,70255	0,93556	0,97407	0,86762	0,95243
	F1	0,97380	0,77051	0,93936	0,97471	0,88271	0,95453
2 (1.250.000)	**ROC-AUC**	0,98399	0,70657	0,95430	0,98366	0,89147	0,97096
	F1	0,98423	0,77209	0,95617	0,98390	0,90146	0,97170
3 (625.000)	**ROC-AUC**	0,99092	0,77097	0,97349	0,98932	0,91107	0,98244
	F1	0,99099	0,80856	0,97403	0,98941	0,91730	0,98264
4 (312.500)	**ROC-AUC**	0,99258	**0,86493**	0,98612	0,99221	0,93162	**0,98882**
	F1	**0,99262**	**0,87571**	0,98618	0,99225	0,93447	**0,98883**
4 (156.250)	**ROC-AUC**	**0,99259**	0,83423	**0,98694**	**0,99345**	**0,94639**	0,98803
	F1	**0,99262**	0,82879	**0,98692**	**0,99347**	**0,94690**	0,98798

Table 27: Results for SMOTE + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 156.250 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,90643	0,54614	0,76752	0,90137	0,72816	0,63139
	F1	0,91443	0,68782	0,81132	0,91023	0,78625	0,73063
1 (2.500.000)	ROC-AUC	0,91810	0,55689	0,75471	0,91368	0,75372	0,84810
	F1	0,92430	0,69295	0,80301	0,92053	0,80237	0,86809
2 (1.250.000)	ROC-AUC	0,93260	0,56984	0,91675	0,93324	0,77167	0,91584
	F1	0,93684	0,69922	0,92313	0,93740	0,81407	0,92236
3 (625.000)	ROC-AUC	0,94230	0,58533	0,92835	0,94010	0,78895	0,92901
	F1	0,94543	0,70687	0,93312	0,94347	0,82566	0,93369
4 (312.500)	ROC-AUC	0,96676	0,59160	0,94174	0,95635	0,82028	0,94177
	F1	0,96780	0,71000	0,94491	0,95815	0,84753	0,94493
4 (156.250)	ROC-AUC	**0,97217**	**0,65637**	**0,95933**	**0,96994**	**0,83783**	**0,95717**
	F1	**0,97289**	**0,74398**	**0,96086**	**0,97078**	**0,86023**	**0,95886**

Table 28: Results for SMOTE + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 9.750 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,92537	0,59418	0,88773	0,92095	0,76572	0,69905
	F1	0,93055	0,71132	0,89905	0,92673	0,81017	0,76864
1 (2.500.000)	ROC-AUC	0,93483	0,55847	0,90633	0,93830	0,77357	0,91519
	F1	0,93881	0,69371	0,91433	0,94187	0,81534	0,92180
2 (1.250.000)	ROC-AUC	0,94707	0,60048	0,92363	0,94834	0,80782	0,92678
	F1	0,94972	0,71452	0,92902	0,95086	0,83871	0,93175
3 (625.000)	ROC-AUC	0,95832	0,60647	0,94174	0,96020	0,83154	0,93635
	F1	0,95996	0,71759	0,94491	0,96170	0,85567	0,94010
4 (312.500)	ROC-AUC	0,97700	0,68138	0,95057	0,97569	0,84365	0,95341
	F1	0,97749	0,75812	0,95283	0,97623	0,86451	0,95539
4 (156.250)	ROC-AUC	**0,98382**	**0,72673**	**0,97178**	**0,98039**	**0,87609**	**0,97519**
	F1	**0,98405**	**0,78456**	**0,97247**	**0,98073**	**0,88925**	**0,97571**

Table 29: Results for SMOTE + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 19.500 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,94019	0,60713	0,90189	0,93893	0,79343	0,91551
	F1	0,94355	0,71793	0,91063	0,94244	0,82875	0,92208
1 (2.500.000)	**ROC-AUC**	0,95435	0,63933	0,91917	0,94992	0,82328	0,92963
	F1	0,95633	0,73491	0,92518	0,95230	0,84973	0,93423
2 (1.250.000)	**ROC-AUC**	0,96025	0,64371	0,93321	0,96466	0,83808	0,93860
	F1	0,96175	0,73727	0,93734	0,96585	0,86047	0,94211
3 (625.000)	**ROC-AUC**	0,97329	0,62751	0,94964	0,97707	0,84949	0,95244
	F1	0,97396	0,72816	0,95199	0,97756	0,86887	0,95453
4 (312.500)	**ROC-AUC**	0,98910	0,74263	0,96814	0,98489	0,88478	0,97471
	F1	0,98920	0,79455	0,96904	0,98509	0,89612	0,97526
4 (156.250)	**ROC-AUC**	**0,99047**	**0,77039**	**0,98252**	**0,98782**	**0,90138**	**0,98476**
	F1	**0,99054**	**0,80819**	**0,98272**	**0,98793**	**0,90935**	**0,98491**

Table 30: Results for SMOTE + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 39.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,95090	0,66808	0,91851	0,96037	0,82573	0,92711
	F1	0,95318	0,75077	0,92461	0,96188	0,85148	0,93204
1 (2.500.000)	**ROC-AUC**	0,96911	0,64119	0,92872	0,96596	0,84029	0,94144
	F1	0,97002	0,73590	0,93340	0,96706	0,86211	0,94464
2 (1.250.000)	**ROC-AUC**	0,97684	0,67217	0,94388	0,97653	0,85792	0,95241
	F1	0,97734	0,75285	0,94678	0,97705	0,87528	0,95450
3 (625.000)	**ROC-AUC**	0,98638	0,70589	0,96310	0,98540	0,88821	0,97348
	F1	0,98655	0,77166	0,96431	0,98559	0,89884	0,97409
4 (312.500)	**ROC-AUC**	0,99099	0,77273	0,98167	0,98841	0,90850	0,98442
	F1	0,99106	0,80991	0,98190	0,98851	0,91516	0,98457
4 (156.250)	**ROC-AUC**	**0,99298**	**0,84206**	**0,98768**	**0,99318**	**0,92865**	**0,98850**
	F1	**0,99300**	**0,85516**	**0,98771**	**0,99320**	**0,93189**	**0,98852**

Table 31: Results for SMOTE + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 78.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,97357	0,64687	0,92591	0,97230	0,84460	0,93796
	F1	0,97424	0,73899	0,93096	0,97303	0,86531	0,94154
1 (2.500.000)	**ROC-AUC**	0,97661	0,69717	0,93646	0,98037	0,87054	0,95337
	F1	0,97713	0,76731	0,94015	0,98073	0,88502	0,95538
2 (1.250.000)	**ROC-AUC**	0,98208	0,70182	0,95394	0,98583	0,87010	0,97160
	F1	0,98237	0,76925	0,95584	0,98602	0,88451	0,97231
3 (625.000)	**ROC-AUC**	0,98748	0,77300	0,97791	0,98833	0,91188	0,98321
	F1	0,98761	0,81008	0,97827	0,98845	0,91798	0,98339
4 (312.500)	**ROC-AUC**	0,99296	0,83902	0,98468	0,99103	0,93064	**0,98860**
	F1	0,99299	**0,85257**	0,98477	0,99108	0,93359	**0,98862**
4 (156.250)	**ROC-AUC**	**0,99368**	**0,85372**	**0,98668**	**0,99627**	**0,94570**	0,98768
	F1	**0,99369**	0,85067	**0,98665**	**0,99627**	**0,94625**	0,98763

Table 32: Results for SMOTE + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 156.250 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,90801	0,55657	0,71512	0,90106	0,73132	0,62968
	F1	0,91575	0,69279	0,77825	0,90996	0,78821	0,72766
1 (2.500.000)	**ROC-AUC**	0,91305	0,54772	0,75345	0,91684	0,75340	0,84620
	F1	0,92000	0,68857	0,80220	0,92322	0,80216	0,86666
2 (1.250.000)	**ROC-AUC**	0,93070	0,56574	0,91171	0,93513	0,77325	0,91616
	F1	0,93518	0,69722	0,91885	0,93907	0,81512	0,92263
3 (625.000)	**ROC-AUC**	0,94200	0,58564	0,93120	0,94167	0,79461	0,92903
	F1	0,94517	0,70703	0,93560	0,94487	0,82953	0,93372
4 (312.500)	**ROC-AUC**	0,95698	0,58970	0,94172	0,96299	0,82090	0,94177
	F1	0,95873	0,70904	0,94488	0,96429	0,84797	0,94493
4 (156.250)	**ROC-AUC**	**0,97248**	**0,65149**	**0,95650**	**0,97055**	**0,82786**	**0,95590**
	F1	**0,97318**	**0,74123**	**0,95825**	**0,97136**	**0,85293**	**0,95770**

Table 33: Results for SMOTE + RUS with T-Link as a reduction technique. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 9.750 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,92032	0,54962	0,63996	0,92158	0,76572	0,84593
	F1	0,92620	0,68947	0,73525	0,92728	0,81017	0,86647
1 (2.500.000)	**ROC-AUC**	0,93607	0,58786	0,90949	0,93293	0,78366	0,91805
	F1	0,93990	0,70814	0,91699	0,93714	0,82209	0,92425
2 (1.250.000)	**ROC-AUC**	0,95053	0,60269	0,92300	0,94801	0,80848	0,92681
	F1	0,95285	0,71565	0,92848	0,95057	0,83918	0,93178
3 (625.000)	**ROC-AUC**	0,95833	0,59443	0,94209	0,96462	0,83091	0,93796
	F1	0,95997	0,71142	0,94522	0,96581	0,85521	0,94153
4 (312.500)	**ROC-AUC**	0,97321	0,68197	0,95122	0,97130	0,83969	0,95278
	F1	0,97388	0,75845	0,95343	0,97207	0,86160	0,95484
4 (156.250)	**ROC-AUC**	**0,98254**	**0,72432**	**0,97061**	**0,98321**	**0,86895**	**0,97461**
	F1	**0,98281**	**0,78301**	**0,97136**	**0,98345**	**0,88363**	**0,97516**

Table 34: Results for SMOTE + RUS with T-Link as a reduction technique. By default, there are 5.082.736 and 5.082.814 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 19.500 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,94114	0,57711	0,90190	0,94241	0,78745	0,91489
	F1	0,94441	0,70279	0,91065	0,94554	0,82466	0,92155
1 (2.500.000)	ROC-AUC	0,95278	0,60806	0,90500	0,95180	0,80972	0,92870
	F1	0,95490	0,71841	0,91321	0,95401	0,84005	0,93342
2 (1.250.000)	ROC-AUC	0,96814	0,64402	0,93259	0,96654	0,82246	0,93894
	F1	0,96911	0,73744	0,93680	0,96760	0,84909	0,94241
3 (625.000)	ROC-AUC	0,97209	0,69756	0,95081	0,97454	0,85943	0,95309
	F1	0,97282	0,76758	0,95305	0,97514	0,87643	0,95513
4 (312.500)	ROC-AUC	0,98185	0,72374	0,96725	0,98496	0,88114	0,97504
	F1	0,98215	0,78256	0,96820	0,98516	0,89322	0,97557
4 (156.250)	ROC-AUC	**0,98930**	**0,76966**	**0,98255**	**0,99089**	**0,90673**	**0,98434**
	F1	**0,98939**	**0,80756**	**0,98275**	**0,99094**	**0,91374**	**0,98449**

Table 35: Results for SMOTE + RUS with T-Link as a reduction technique. By default, there are 5.082.736 and 5.082.814 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 39.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,94301	0,63618	0,91631	0,95468	0,81729	0,92870
	F1	0,94607	0,73322	0,92274	0,95663	0,84542	0,93342
1 (2.500.000)	ROC-AUC	0,96974	0,64812	0,92972	0,96531	0,83958	0,93796
	F1	0,97061	0,73967	0,93428	0,96646	0,86156	0,94154
2 (1.250.000)	ROC-AUC	0,97873	0,66043	0,94458	0,97683	0,84916	0,95239
	F1	0,97915	0,74622	0,94742	0,97733	0,86863	0,95448
3 (625.000)	ROC-AUC	0,98322	0,71655	0,96297	0,98446	0,88883	0,97312
	F1	0,98347	0,77818	0,96418	0,98467	0,89934	0,97374
4 (312.500)	ROC-AUC	0,98909	0,76674	0,98098	0,98841	0,90696	0,98369
	F1	0,98918	0,80547	0,98123	0,98852	0,91387	0,98385
4 (156.250)	ROC-AUC	**0,99427**	**0,85209**	**0,98710**	**0,99214**	**0,92847**	**0,98867**
	F1	**0,99429**	**0,86404**	**0,98714**	**0,99217**	**0,93170**	**0,98868**

Table 36: Results for SMOTE + RUS with T-Link as a reduction technique. By default, there are 5.082.736 and 5.082.814 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 78.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,96316	0,64592	0,92215	0,97042	0,83779	0,93765
	F1	0,96445	0,73847	0,92772	0,97126	0,86026	0,94127
1 (2.500.000)	**ROC-AUC**	0,97630	0,70853	0,93687	0,97691	0,86211	0,95338
	F1	0,97683	0,77406	0,94052	0,97742	0,87846	0,95539
2 (1.250.000)	**ROC-AUC**	0,98145	0,76271	0,94757	0,98333	0,86468	0,97191
	F1	0,98177	0,80755	0,95004	0,98359	0,88033	0,97259
3 (625.000)	**ROC-AUC**	0,98717	0,80160	0,97799	0,98772	0,90809	0,98350
	F1	0,98732	0,83162	0,97835	0,98785	0,91485	0,98368
4 (312.500)	**ROC-AUC**	0,99270	0,85237	0,98580	0,99289	0,93062	**0,98856**
	F1	0,99273	**0,86425**	0,98586	0,99292	0,93357	**0,98857**
4 (156.250)	**ROC-AUC**	**0,99325**	**0,85493**	**0,98744**	**0,99388**	**0,94586**	0,98777
	F1	**0,99327**	0,85198	**0,98741**	**0,99389**	**0,94638**	0,98772

Table 37: Results for SMOTE + RUS with T-Link as a reduction technique. By default, there are 5.082.736 and 5.082.814 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 156.250 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

Appendix E: Results ROS + RUS

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,89695	0,55468	0,78178	0,89758	0,72753	0,80994
	F1	0,90658	0,69189	0,82083	0,90710	0,78586	0,84026
1 (2.500.000)	ROC-AUC	0,91464	0,55531	0,87448	0,90863	0,74330	0,82318
	F1	0,92135	0,69219	0,88847	0,91628	0,79571	0,84971
2 (1.250.000)	ROC-AUC	0,91870	0,56826	0,91329	0,91776	0,77357	0,91206
	F1	0,92481	0,69845	0,92019	0,92400	0,81533	0,91915
3 (625.000)	ROC-AUC	0,94074	0,54077	0,92679	0,93569	0,79146	0,92683
	F1	0,94404	0,68529	0,93176	0,93956	0,82737	0,93180
4 (312.500)	ROC-AUC	0,95323	0,59381	0,94242	0,94979	0,81338	0,94114
	F1	0,95530	0,71112	0,94551	0,95216	0,84260	0,94437
4 (156.250)	ROC-AUC	**0,96466**	**0,65628**	**0,95943**	**0,96876**	**0,83754**	**0,95902**
	F1	**0,96583**	**0,74390**	**0,96096**	**0,96967**	**0,86002**	**0,96057**

Table 38: Results for ROS + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 9.750 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,90516	0,58849	0,74332	0,90106	0,74962	0,74647
	F1	0,91338	0,70846	0,79573	0,90996	0,79973	0,79773
1 (2.500.000)	ROC-AUC	0,91683	0,59354	0,90883	0,91558	0,78082	0,91490
	F1	0,92321	0,71100	0,91643	0,92215	0,82018	0,92156
2 (1.250.000)	ROC-AUC	0,92532	0,60744	0,92238	0,93386	0,80310	0,92778
	F1	0,93050	0,71809	0,92795	0,93795	0,83541	0,93262
3 (625.000)	ROC-AUC	0,94546	0,59349	0,93956	0,94449	0,82621	0,94178
	F1	0,94826	0,71096	0,94297	0,94739	0,85179	0,94495
4 (312.500)	ROC-AUC	0,96456	0,66677	0,95286	0,95540	0,83625	0,95349
	F1	0,96574	0,74978	0,95492	0,95728	0,85906	0,95550
4 (156.250)	ROC-AUC	**0,96838**	**0,69888**	**0,97290**	**0,97088**	**0,87052**	**0,96938**
	F1	**0,96931**	**0,76746**	**0,97354**	**0,97167**	**0,88487**	**0,97020**

Table 39: Results for ROS + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 19.500 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,90453	0,61661	0,90536	0,91116	0,78177	0,91522
	F1	0,91285	0,72286	0,91352	0,91841	0,82081	0,92183
1 (2.500.000)	ROC-AUC	0,92758	0,63555	0,91664	0,92347	0,80655	0,92746
	F1	0,93247	0,73288	0,92302	0,92890	0,83782	0,93234
2 (1.250.000)	ROC-AUC	0,93288	0,59350	0,93671	0,92784	0,82652	0,94429
	F1	0,93709	0,71097	0,94043	0,93268	0,85201	0,94719
3 (625.000)	ROC-AUC	0,95268	0,68646	0,95215	0,94575	0,85330	0,95787
	F1	0,95480	0,76106	0,95427	0,94852	0,87177	0,95951
4 (312.500)	ROC-AUC	0,96325	0,70011	0,97066	0,96484	0,86008	0,97318
	F1	0,96453	0,76818	0,97141	0,96601	0,87679	0,97380
4 (156.250)	ROC-AUC	**0,97714**	**0,77463**	**0,98727**	**0,97208**	**0,90571**	**0,98803**
	F1	**0,97762**	**0,81101**	**0,98735**	**0,97280**	**0,91287**	**0,98810**

Table 40: Results for ROS + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 39.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,90390	0,59764	0,91319	0,90706	0,80623	0,92872
	F1	0,91232	0,71308	0,92010	0,91496	0,83759	0,93345
1 (2.500.000)	ROC-AUC	0,92693	0,58685	0,92402	0,92693	0,82715	0,94464
	F1	0,93190	0,70762	0,92933	0,93190	0,85248	0,94751
2 (1.250.000)	ROC-AUC	0,93731	0,64326	0,95216	0,93636	0,85666	0,95284
	F1	0,94100	0,73673	0,95428	0,94016	0,87432	0,95490
3 (625.000)	ROC-AUC	0,95206	0,74007	0,97066	0,94858	0,88427	0,97038
	F1	0,95423	0,79282	0,97141	0,95107	0,89567	0,97115
4 (312.500)	ROC-AUC	0,96037	0,76207	0,98720	0,95945	0,89792	0,98641
	F1	0,96185	0,80206	0,98729	0,96100	0,90646	0,98650
4 (156.250)	ROC-AUC	**0,97144**	**0,83661**	**0,99091**	**0,97336**	**0,93437**	**0,99058**
	F1	**0,97220**	**0,84959**	**0,99089**	**0,97401**	**0,93685**	**0,99057**

Table 41: Results for ROS + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 78.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,90453	0,59223	0,92561	0,90674	0,83244	0,94182
	F1	0,91285	0,71032	0,93071	0,91469	0,85632	0,94498
1 (2.500.000)	ROC-AUC	0,92094	0,65874	0,94078	0,92473	0,85507	0,95841
	F1	0,92672	0,74524	0,94401	0,93000	0,87310	0,96001
2 (1.250.000)	ROC-AUC	0,93983	0,73381	0,96408	0,93574	0,88535	0,97131
	F1	0,94323	0,78891	0,96522	0,93961	0,89651	0,97203
3 (625.000)	ROC-AUC	0,94477	0,76320	0,98309	0,94795	0,90898	0,98643
	F1	0,94764	0,80287	0,98327	0,95051	0,91555	0,98653
4 (312.500)	ROC-AUC	0,95786	0,83732	**0,99139**	0,96165	0,93350	**0,99114**
	F1	0,95953	**0,85026**	**0,99138**	0,96304	0,93608	**0,99113**
4 (156.250)	ROC-AUC	**0,97298**	**0,84620**	0,99065	**0,97338**	**0,94187**	0,99011
	F1	**0,97365**	0,84187	0,99057	**0,97404**	**0,94223**	0,99002

Table 42: Results for ROS + RUS with no reduction. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 156.250 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,89632	0,53002	0,74829	0,89695	0,73384	0,80235
	F1	0,90606	0,68028	0,79886	0,90658	0,78978	0,83494
1 (2.500.000)	ROC-AUC	0,91336	0,56289	0,59728	0,92064	0,74803	0,74763
	F1	0,92027	0,69584	0,71287	0,92647	0,79872	0,79844
2 (1.250.000)	ROC-AUC	0,92629	0,56984	0,91550	0,92756	0,77514	0,91425
	F1	0,93134	0,69922	0,92206	0,93244	0,81638	0,92100
3 (625.000)	ROC-AUC	0,93980	0,55119	0,92617	0,94327	0,79367	0,92619
	F1	0,94321	0,69022	0,93122	0,94630	0,82888	0,93124
4 (312.500)	ROC-AUC	0,95765	0,59160	0,94243	0,95577	0,81967	0,94273
	F1	0,95935	0,71001	0,94552	0,95762	0,84709	0,94579
4 (156.250)	ROC-AUC	**0,96844**	**0,65124**	**0,95658**	**0,96878**	**0,83815**	**0,95563**
	F1	**0,96937**	**0,74112**	**0,95832**	**0,96969**	**0,86046**	**0,95745**

Table 43: Results for ROS + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 9.750 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,90769	0,54456	0,75123	0,90453	0,75782	0,74836
	F1	0,91549	0,68708	0,80078	0,91285	0,80501	0,79893
1 (2.500.000)	ROC-AUC	0,91746	0,59069	0,90698	0,91525	0,78239	0,91269
	F1	0,92375	0,70956	0,91488	0,92187	0,82124	0,91969
2 (1.250.000)	ROC-AUC	0,93226	0,61028	0,92270	0,92564	0,77447	0,92588
	F1	0,93655	0,71956	0,92822	0,93077	0,81592	0,93097
3 (625.000)	ROC-AUC	0,94515	0,59696	0,94020	0,94388	0,82870	0,93960
	F1	0,94798	0,71271	0,94353	0,94684	0,85360	0,94300
4 (312.500)	ROC-AUC	0,96389	0,64233	0,95661	0,95763	0,84929	0,95441
	F1	0,96512	0,73623	0,95835	0,95932	0,86875	0,95634
4 (156.250)	ROC-AUC	**0,97281**	**0,69965**	**0,97506**	**0,97305**	**0,86595**	**0,97190**
	F1	**0,97350**	**0,76798**	**0,97559**	**0,97372**	**0,88132**	**0,97259**

Table 44: Results for ROS + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 19.500 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,91053	0,62261	0,90568	0,90832	0,78681	0,91333
	F1	0,91787	0,72600	0,91379	0,91602	0,82423	0,92023
1 (2.500.000)	**ROC-AUC**	0,92473	0,58059	0,91793	0,91778	0,80907	0,92840
	F1	0,92999	0,70451	0,92412	0,92402	0,83958	0,93316
2 (1.250.000)	**ROC-AUC**	0,93543	0,58781	0,93292	0,93638	0,83587	0,94213
	F1	0,93934	0,70810	0,93709	0,94017	0,85883	0,94526
3 (625.000)	**ROC-AUC**	0,94923	0,65019	0,95498	0,95080	0,83467	0,95691
	F1	0,95166	0,74050	0,95685	0,95309	0,85789	0,95863
4 (312.500)	**ROC-AUC**	0,96324	0,73723	0,97255	0,96291	0,88101	0,97034
	F1	0,96451	0,79113	0,97320	0,96420	0,89309	0,97111
4 (156.250)	**ROC-AUC**	**0,97610**	**0,79362**	**0,98625**	**0,97711**	**0,90435**	**0,98714**
	F1	**0,97662**	**0,82517**	**0,98634**	**0,97759**	**0,91175**	**0,98722**

Table 45: Results for ROS + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 39.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,90958	0,60902	0,91761	0,90706	0,81539	0,92839
	F1	0,91707	0,71891	0,92385	0,91496	0,84407	0,93316
1 (2.500.000)	**ROC-AUC**	0,92062	0,59570	0,92534	0,92315	0,83713	0,94305
	F1	0,92645	0,71207	0,93048	0,92863	0,85977	0,94608
2 (1.250.000)	**ROC-AUC**	0,93857	0,63972	0,94641	0,93605	0,85913	0,95438
	F1	0,94211	0,73479	0,94906	0,93989	0,87620	0,95631
3 (625.000)	**ROC-AUC**	0,95018	0,74012	0,96786	0,95112	0,88133	0,97321
	F1	0,95252	0,79287	0,96877	0,95338	0,89335	0,97383
4 (312.500)	**ROC-AUC**	0,96387	0,76563	0,98733	0,96511	0,90836	0,98615
	F1	0,96510	0,80468	0,98741	0,96626	0,91503	0,98625
4 (156.250)	**ROC-AUC**	**0,97106**	**0,83798**	**0,99222**	**0,96893**	**0,93478**	**0,99098**
	F1	**0,97183**	**0,85087**	**0,99220**	**0,96982**	**0,93724**	**0,99096**

Table 46: Results for ROS + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 78.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,90642	0,59538	0,92403	0,90895	0,83678	0,94242
	F1	0,91442	0,71191	0,92934	0,91655	0,85950	0,94551
1 (2.500.000)	**ROC-AUC**	0,92693	0,64291	0,93906	0,92251	0,86298	0,95651
	F1	0,93190	0,73652	0,94247	0,92808	0,87912	0,95826
2 (1.250.000)	**ROC-AUC**	0,94047	0,70237	0,96460	0,93826	0,87891	0,97519
	F1	0,94380	0,76955	0,96571	0,94184	0,89143	0,97571
3 (625.000)	**ROC-AUC**	0,95048	0,79766	0,98454	0,94766	0,91220	0,98546
	F1	0,95280	0,82807	0,98468	0,95024	0,91820	0,98558
4 (312.500)	**ROC-AUC**	0,95943	**0,83932**	**0,99067**	0,96387	0,93332	**0,99161**
	F1	0,96098	**0,85220**	**0,99065**	0,96510	0,93585	**0,99160**
4 (156.250)	**ROC-AUC**	**0,97644**	0,81673	0,99055	**0,97525**	**0,93993**	0,99006
	F1	**0,97694**	0,80724	0,99047	**0,97582**	**0,94038**	0,98998

Table 47: Results for ROS + RUS with ENN as a reduction technique. By default, there are 5.079.703 and 5.079.902 legal transactions (for the normalized and standardized data respectively) versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 156.250 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,89664	0,54741	0,74829	0,89538	0,73321	0,80075
	F1	0,90632	0,68842	0,79886	0,90528	0,78938	0,83381
1 (2.500.000)	**ROC-AUC**	0,90484	0,55562	0,75090	0,90894	0,75404	0,77773
	F1	0,91310	0,69234	0,80056	0,91654	0,80257	0,81813
2 (1.250.000)	**ROC-AUC**	0,92124	0,54330	0,91108	0,92345	0,77419	0,91394
	F1	0,92698	0,68648	0,91833	0,92889	0,81575	0,92075
3 (625.000)	**ROC-AUC**	0,93886	0,57300	0,92681	0,93602	0,78233	0,92556
	F1	0,94237	0,70077	0,93178	0,93985	0,82118	0,93069
4 (312.500)	**ROC-AUC**	0,95419	0,59380	0,93958	0,94946	0,81966	0,94089
	F1	0,95617	0,71110	0,94299	0,95186	0,84708	0,94415
4 (156.250)	**ROC-AUC**	**0,97065**	**0,63035**	**0,95562**	**0,97282**	**0,83904**	**0,95907**
	F1	**0,97146**	**0,72967**	**0,95744**	**0,97351**	**0,86110**	**0,96062**

Table 48: Results for ROS + RUS with T-Link as a reduction technique. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 9.750 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	**RF**	**GB**	**LC**	**RF**	**GB**
0 (5.000.000)	**ROC-AUC**	0,90864	0,57301	0,74869	0,90295	0,75972	0,74615
	F1	0,91629	0,70077	0,79915	0,91153	0,80625	0,79752
1 (2.500.000)	**ROC-AUC**	0,91873	0,56321	0,90857	0,91936	0,78113	0,91080
	F1	0,92483	0,69599	0,91621	0,92537	0,82039	0,91809
2 (1.250.000)	**ROC-AUC**	0,92690	0,60743	0,92019	0,92817	0,80308	0,92651
	F1	0,93187	0,71808	0,92606	0,93298	0,83539	0,93152
3 (625.000)	**ROC-AUC**	0,94292	0,59507	0,93928	0,94356	0,82870	0,93834
	F1	0,94598	0,71175	0,94271	0,94656	0,85359	0,94188
4 (312.500)	**ROC-AUC**	0,95538	0,65719	0,95498	0,95508	0,85023	0,95661
	F1	0,95726	0,74438	0,95686	0,95698	0,86946	0,95835
4 (156.250)	**ROC-AUC**	**0,97060**	**0,70751**	**0,97131**	**0,96965**	**0,86852**	**0,97443**
	F1	**0,97140**	**0,77265**	**0,97202**	**0,97051**	**0,88328**	**0,97500**

Table 49: Results for ROS + RUS with T-Link as a reduction technique. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 19.500 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,90642	0,61661	0,90633	0,90484	0,78806	0,91490
	F1	0,91443	0,72286	0,91433	0,91311	0,82507	0,92156
1 (2.500.000)	ROC-AUC	0,92030	0,63239	0,91955	0,91840	0,80217	0,92524
	F1	0,92618	0,73119	0,92551	0,92455	0,83476	0,93042
2 (1.250.000)	ROC-AUC	0,93573	0,59571	0,93450	0,93763	0,82870	0,94119
	F1	0,93960	0,71208	0,93848	0,94128	0,85360	0,94442
3 (625.000)	ROC-AUC	0,95082	0,64199	0,95087	0,94828	0,85632	0,95627
	F1	0,95311	0,73604	0,95310	0,95081	0,87405	0,95805
4 (312.500)	ROC-AUC	0,96420	0,70077	0,97068	0,96420	0,86182	0,97382
	F1	0,96541	0,76858	0,97144	0,96541	0,87811	0,97441
4 (156.250)	ROC-AUC	**0,97456**	**0,76221**	**0,98886**	**0,97495**	**0,90475**	**0,98670**
	F1	**0,97515**	**0,80209**	**0,98891**	**0,97552**	**0,91205**	**0,98679**

Table 50: Results for ROS + RUS with T-Link as a reduction technique. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 39.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	ROC-AUC	0,91148	0,61533	0,91604	0,90927	0,81002	0,92872
	F1	0,91867	0,72218	0,92251	0,91682	0,84025	0,93344
1 (2.500.000)	ROC-AUC	0,92378	0,58209	0,92505	0,91872	0,83685	0,94147
	F1	0,92917	0,70523	0,93023	0,92482	0,85956	0,94467
2 (1.250.000)	ROC-AUC	0,93731	0,70408	0,94963	0,93005	0,84711	0,95620
	F1	0,94099	0,77140	0,95197	0,93461	0,86711	0,95797
3 (625.000)	ROC-AUC	0,94795	0,72967	0,97253	0,94732	0,87984	0,97004
	F1	0,95050	0,78631	0,97318	0,94993	0,89218	0,97083
4 (312.500)	ROC-AUC	0,96136	0,76276	0,98806	0,96233	0,90472	0,98805
	F1	0,96277	0,80254	0,98813	0,96368	0,91202	0,98811
4 (156.250)	ROC-AUC	**0,97899**	**0,83606**	**0,99188**	**0,97051**	**0,93422**	**0,99093**
	F1	**0,97939**	**0,84901**	**0,99185**	**0,97132**	**0,93670**	**0,99091**

Table 51: Results for ROS + RUS with T-Link as a reduction technique. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 78.000 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.

		NRM			STD		
		LC	RF	GB	LC	RF	GB
0 (5.000.000)	**ROC-AUC**	0,90485	0,59634	0,92437	0,90358	0,83526	0,94304
	F1	0,91311	0,71240	0,92963	0,91206	0,85839	0,94607
1 (2.500.000)	**ROC-AUC**	0,92410	0,65967	0,93849	0,91936	0,86031	0,95757
	F1	0,92945	0,74575	0,94197	0,92537	0,87710	0,95924
2 (1.250.000)	**ROC-AUC**	0,93479	0,75616	0,96197	0,94016	0,88902	0,97056
	F1	0,93877	0,80321	0,96325	0,94352	0,89947	0,97131
3 (625.000)	**ROC-AUC**	0,94671	0,76766	0,98502	0,95016	0,91249	0,98702
	F1	0,94938	0,80608	0,98515	0,95251	0,91845	0,98711
4 (312.500)	**ROC-AUC**	0,96072	0,83906	**0,99114**	0,96514	0,93565	**0,99079**
	F1	0,96218	**0,85183**	**0,99113**	0,96628	0,93793	**0,99078**
4 (156.250)	**ROC-AUC**	**0,97686**	**0,84700**	0,99061	**0,97779**	**0,94733**	0,99056
	F1	**0,97735**	0,84272	0,99053	**0,97825**	**0,94742**	0,99049

Table 52: Results for ROS + RUS with T-Link as a reduction technique. By default, there are 5.083.465 legal transactions versus 6.631 fraudulent ones. The fraudulent transactions are oversampled to 156.250 transactions. Then, non-fraud transactions are under-sampled. The number of under-sampled transactions is provided between brackets. The best results are indicated in bold.