Cyber Security Data Science: Machine Learning Methods and their Performance on Imbalanced Datasets

Mateo Lopez-Ledezma* and Gissel Velarde†

Abstract

Cybersecurity has become essential worldwide and at all levels, concerning individuals, institutions, and governments. A basic principle in cybersecurity is to be always alert. Therefore, automation is imperative in processes where the volume of daily operations is large. Several cybersecurity applications can be addressed as binary classification problems, including anomaly detection, fraud detection, intrusion detection, spam detection, or malware detection. In many cases, the positive class samples, those that represent a problem, occur at a much lower frequency than negative samples, and this poses a challenge for machine learning algorithms since learning patterns out of under-represented samples is hard. This is known in machine learning as imbalance learning. In this study, we systematically evaluate various machine learning methods using two representative financial datasets containing numerical and categorical features. The Credit Card dataset contains 283 726 samples, 31 features, and 0.2 percent of the transactions are fraudulent (imbalance ratio of 598.84:1). The PaySim dataset contains 6 362 620 samples, 11 features and 0.13 percent of the transactions are fraudulent (imbalance ratio of 773.70:1). We present three experiments. In the first experiment, we evaluate single classifiers including Random Forests, Light Gradient Boosting Machine, eXtreme Gradient Boosting, Logistic Regression, Decision Tree, and Gradient Boosting Decision Tree. In the second experiment, we test different sampling techniques including over-sampling, under-sampling, Synthetic Minority Over-sampling Technique, and Self-Paced Ensembling. In the last experiment, we evaluate Self-Paced Ensembling and its number of base classifiers. We found that imbalance learning techniques had positive and negative effects, as reported in related studies. Thus, these techniques should be applied with caution. Besides, we found different best performers for each dataset. Therefore, we recommend testing single classifiers and imbalance learning techniques for each new dataset and application involving imbalanced datasets as is the case in several cyber security applications. We provide the code with all experiments as open-source.[1]

Keywords: Machine Learning, Cyber Security, Classification, Imbalance Learning, Data Science.

*Computational Systems Engineering, Universidad Privada Boliviana, Bolivia. E-mail: lopezmateo97@yahoo.com

†Extended Artificial Intelligence, IU International University of Applied Sciences, Erfurt, 99084, Germany E-mail: gissel.velarde@iu.org

[1]Available at `https://github.com/MateoLopez00/Imbalanced-Learning-Empirical-Evaluation`.

1 Introduction

Cyber security is gaining prominence worldwide as cybercrime increases. The cost of cybercrime to the global economy is estimated at 400 billion USD (Sarker et al., 2020). Cyber security data science aims at discovering patterns from data to detect anomalous, malicious, or fraudulent events that can harm a system. Applications in cyber security include anomaly detection, fraud detection, intrusion detection, spam detection, malware detection, and phishing detection (Sarker et al., 2020; Shaukat et al., 2020). Cyber security systems should detect possible attacks at any time, even if the number of operations increases suddenly. Availability and scalability pose a challenge to systems that depend on human monitoring, therefore, automation in cyber security is wanted. Artificial Intelligence (AI), and more specifically, machine learning allows intelligent automation based on the available data to detect cybercrime activities automatically.

Cyber security is now a trending topic just like data science and machine learning (Sarker et al., 2020). In some countries, the term data science is more popular than machine learning and vice versa. Some authors consider that data science would encompass machine learning (Sarker et al., 2020). Sculley et al. (2015) argue that the code devoted to purely machine learning algorithms in real-world systems represents a small percentage of the required infrastructure consisting of several tasks including configurations, data collection, feature extraction, data verification, analysis tools, process and machine resource management tools, serving infrastructure and monitoring. Still, the distinction between what is considered data science and machine learning is often not clear, and these terms may be used as synonyms in some fields or as complementary terms in other domains (Alpaydin, 2014). What is certain is that cyber security is a must for individuals, institutions, and governments, as cyber-attacks increase. Therefore, extracting patterns from data to automatically detect attacks can only be done computationally to stay alert 24 hours a day, 7 days a week. However, in addition to availability and scalability, a fundamental component is the effectiveness of a system at detecting attacks. Therefore, detection quality depends on the performance of the machine learning methods in place.

In this paper, we present machine learning methods and their performance on two representative imbalanced datasets, as many cyber security applications will face the problem of learning from imbalanced distributions. Several cyber security machine learning applications can be addressed as binary classification problems, where the positive class samples represent a problem to a system. In many cases, the positive samples are under-represented in the datasets, and therefore, it is hard to learn statistics from the minority class (Lemaître, Nogueira, & Aridas, 2017; Kim & Hwang, 2022; Hajek, Abedin, & Sivarajah, 2022; Li et al., 2023; Velarde et al., 2024; Yang et al., 2021).

We summarise the contributions of this study as follows:

- We systematically evaluate the performance of six machine learning algorithms in terms of recognition and speed in two representative cyber security datasets for fraud detection.
- We test the effects of imbalance learning techniques including sampling and ensembling.
- We show that imbalance classification depends strongly on the characteristics of each dataset, such that different classifiers should be evaluated for each dataset to select the best approach.

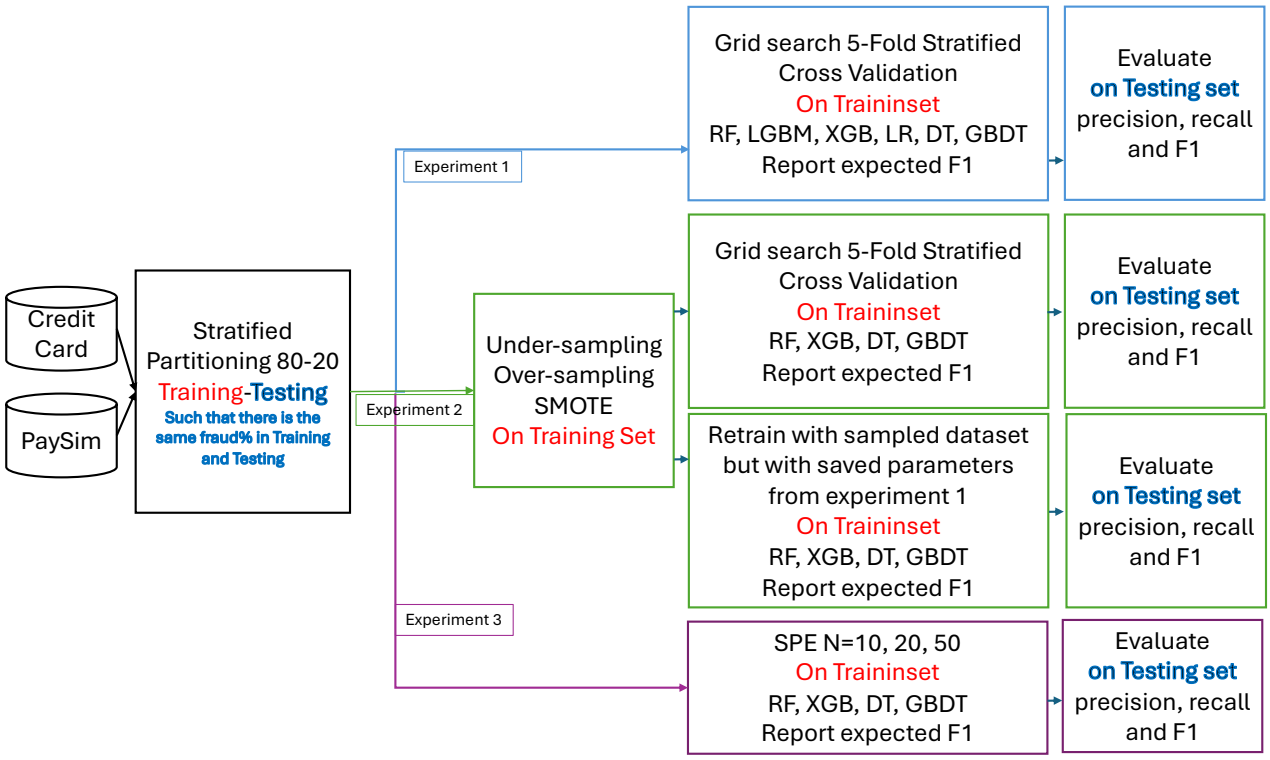

Figure 1: A schematic representation of the method.

.

2 The Method

The proposed method can be seen in Figure 1. Each dataset follows a stratified partition of 80-20 percent for training-testing sets, respectively. In the first experiment, we evaluate the performance of single classifiers: Random Forests (RF), Light Gradient Boosting Machine (LGBM), eXtreme Gradient Boosting (XGB), Logistic Regression (LR), Decision Tree (DT), and Gradient Boosting Decision Tree (GBT), see section 2.2 for details. Each classifier goes under grid search on 5-Fold Stratified cross-validation on the training set, optimizing F1 score. In the second experiment, we evaluate the effect of sampling the training set to balance it, such that it has the same number of positive and negative samples, see section 2.4. First, we use the parameters found in Experiment 1 on the resampled training set. Then, we retrain the classifiers with the resampled training set in 5-fold Stratified cross-validation. In the third experiment, we evaluate Self-Paced Ensembling (SPE) with N=10, 20, and 50 base classifiers, see section 2.4.

Both datasets were analyzed and cleaned before classifier initialization. We performed exploratory data analysis to understand the datasets' structure, number of features, types of features, and imbalance ratio. The Credit Card dataset presented in total 1 081 duplicates but no missing (null) values. To address duplicated values, the first occurrences of each duplicated row have been kept and subsequent occurrences have been removed for each dataset. PaySim dataset did not present missing (null) values or duplicated rows.

2.1 The datasets

We use two imbalanced datasets: **Credit Card** introduced in (Dal Pozzolo, Boracchi, Caelen, Alippi, & Bontempi, 2017), and **PaySim** introduced in (Lopez-Rojas, Elmir, & Axelsson, 2016), see Table 1. Credit Card consists of 283 726 credit card transactions of a European e-commerce, where daily fraud transactions vary over time, but roughly account for 0.2 percent of the transactions (Dal Pozzolo et al., 2017). PaySim consists of 6 362 620 simulated transactions from a real dataset of African mobile payments. Fraud transactions account for 0.13 percent of all transactions (Lopez-Rojas et al., 2016).

Table 1: Characteristics of Credit Card and PaySim Datasets.

Dataset	#Attributes	#Samples	Feature Format	Imbalance Ratio	Fraud %
Credit Card (Dal Pozzolo et al., 2017)	31	283 726	Numerical	598.84:1	0.2
PaySim (Lopez-Rojas et al., 2016)	11	6 362 620	Numerical & categorical	773.70:1	0.13

2.2 Classifiers

We evaluated the following classifiers:

- **Random Forests (RF)** are combinations of trees that depend on an independently sampled vector with the same distribution of the trees in a forest (Breiman, 2001).
- **Gradient Boosting Decision Tree (GBDT)** uses the principle of boosting to create a series of predictive models, each correcting the errors of its predecessors (Friedman, 2001).
- **Light Gradient Boosting Machine (LGBM)** aims at speeding up the training process of Gradient Boosting Decision Tree algorithms, delivering a scalable implementation by reducing the number of features (Ke et al., 2017).
- **eXtreme Gradient Boosting (XGB)** also belongs to the Gradient Boosting Decision Tree family, where successive models are added to correct the residuals or errors of previous models. This iterative approach minimizes the overall predictive error through a series of decision trees that progressively improve the model's accuracy (Chen & Guestrin, 2016).
- **Logistic Regression (LR)** is a statistical method for binary classification that estimates the probabilities of the outcomes based on a weight vector (Yu, Huang, & Lin, 2011).
- **Decision Tree (DT)** implements the divide-and-conquer strategy on a hierarchical data structure, and can be represented as a set of simple rules easy to interpret and visualize (Alpaydin, 2014; Quinlan, 1986).

2.3 Evaluation criteria

In binary imbalance classification, the minority class has a substantially less number of samples than the majority class. Identifying the minority class is hard but essential (Velarde et al., 2024). Therefore, the minority class represents the positive class, and the majority class, the negative class.

Table 2: Confusion matrix for binary classification.

		Predicted	
		Negative	Positive
Actual	Negative	TN	FP
	Positive	FN	TP

Table 2 shows the confusion matrix for binary classification. The confusion matrix allows us to compute different evaluation metrics. We do not consider accuracy as an evaluation metric, since it does not reflect well the model performance and may be deceiving for imbalanced scenarios (Saito & Rehmsmeier, 2015; Liu et al., 2020; Velarde et al., 2024). The evaluation metrics considered in this study include Recall = $\frac{TP}{TP+FN}$, Precision=$\frac{TP}{TP+FP}$, and F1 score=$2 \cdot \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$.

2.4 Imbalance Learning techniques

We use four different imbalance learning techniques, including sampling and ensembling. Sampling techniques are used to change the training set original distribution.

- **Over-Sampling**: Aims at increasing the number of samples in the minority class by randomly resampling the minority class samples. This is a data augmentation technique. The size of the final training set after oversampling is two times the size of the original majority class, since we upsample the minority class until it is equal to the majority class. For example, for PaySim, the training set had 5 083 526 majority samples, and 6 570 minority samples, we upsample the minority class so it also has 5 083 526 samples. The size of the final training set is two times the size of the original majority class.

- **Synthetic Minority Over-sampling Technique (SMOTE)**: Aims at balancing the class distribution by creating synthetic minority class examples and can under-sampling the majority class (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). In our case, the minority class was increased to reach the size of the majority class.

- **Under-Sampling**: Randomly removes samples from the majority class. The size of the final training set after under-sampling is two times the size of the original minority class, since we under-sample the majority class until it is equal to the minority class. For example for PaySim, the training set had 5 083 526 majority samples, and 6570 minority samples, we under-sample the majority class so it also has 6 570 samples. The size of the final training set is two times the size of the original minority class.

- **Self-Paced Ensemble (SPE)** is an ensemble learning technique designed to handle imbalanced datasets by iteratively re-weighting the training data to focus on "hard" examples, aiming to improve recognition of the minority class (Liu et al., 2020).

3 Results

The results of the three experiments are presented in this section. Both datasets, Credit Card and PaySim, are used independently in each experiment. See Appendix A for more information.

3.1 Experiment 1. Individual classifiers.

Experiment 1 aims to evaluate individual classifiers without any sampling techniques. Figure 2 presents the experimental results of the six selected machine learning classifiers. XBG and RF are the best-performing classifiers, and their F1 score is similar in both datasets. However, XGB is much faster to train than RF, see Table 3. GBDT and DT perform on a similar level on the Credit Card dataset, but DT outperforms all classifiers on PaySim. GDBT in contrast does a poor job on PaySim. LGBM is the weakest of all in both datasets. For the following experiments, we select four classifiers: XGB, RF, DT, and GBDT.

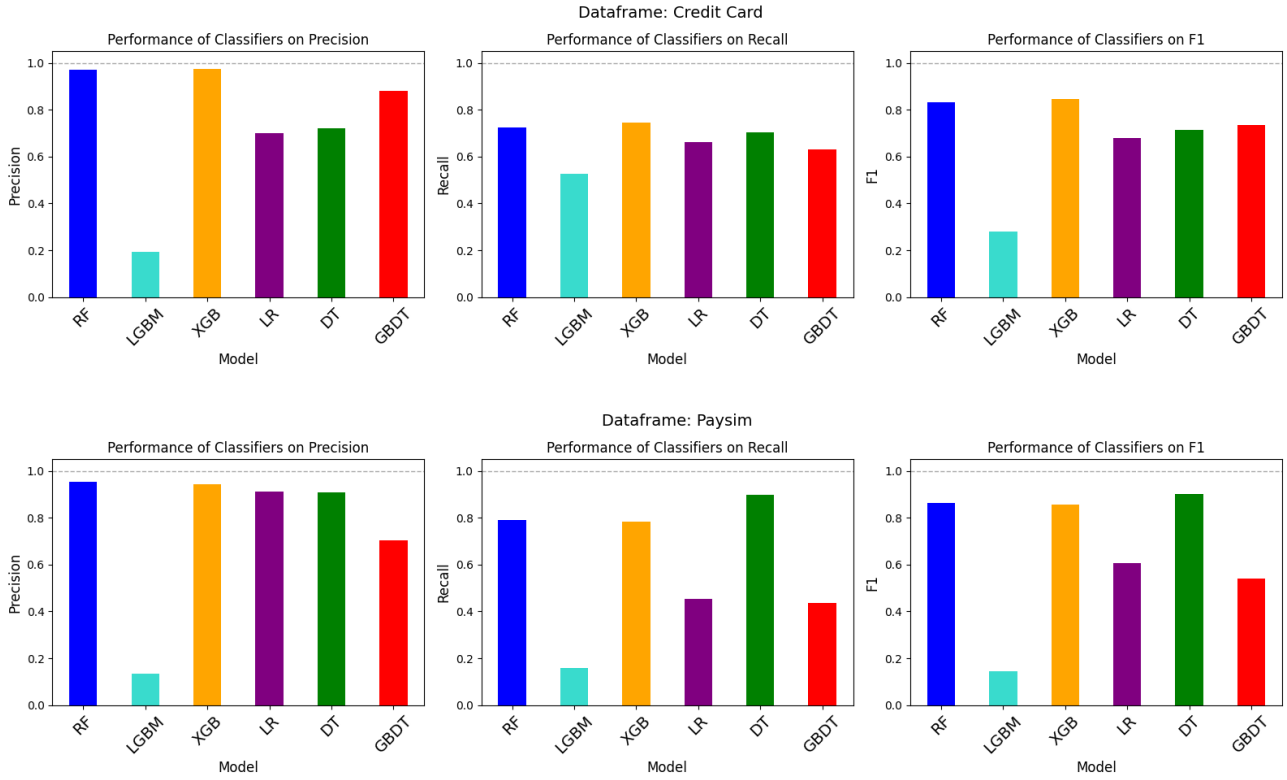

Figure 2: Experiment 1. Performance of individual classifiers evaluated over Precision, Recall, and F1. The upper row of figures corresponds to the results on Credit Card. The lower row corresponds to the results on PaySim.

3.2 Experiment 2. Imbalance Learning Methods

Experiment 2 aims at evaluating the effect of sampling techniques, see Figure 3. Compare the blue, orange, green, and red bars marked as (New) against the purple, brown, pink, and grey, marked as (Old) for No Sampling, Over-Sampling, SMOTE, and Under-Sampling, respectively. Sampling techniques have different effects. Over-Sampling has either no effect or helps improve recognition.

SMOTE most of the time worsens F1, except for GBDT, such that it improves recognition only when optimization is done on the resampled dataset. More generally, sampling techniques improve recall at the expense of lowering precision. However, RF, XGB, and DT prove robust to resampling the training set. In contrast, GBDT performs better when optimising over the resampled dataset, such that the orange, green, and red bars marked as (New) achieve higher F1 score than the brown, pink, and grey, marked as (Old).

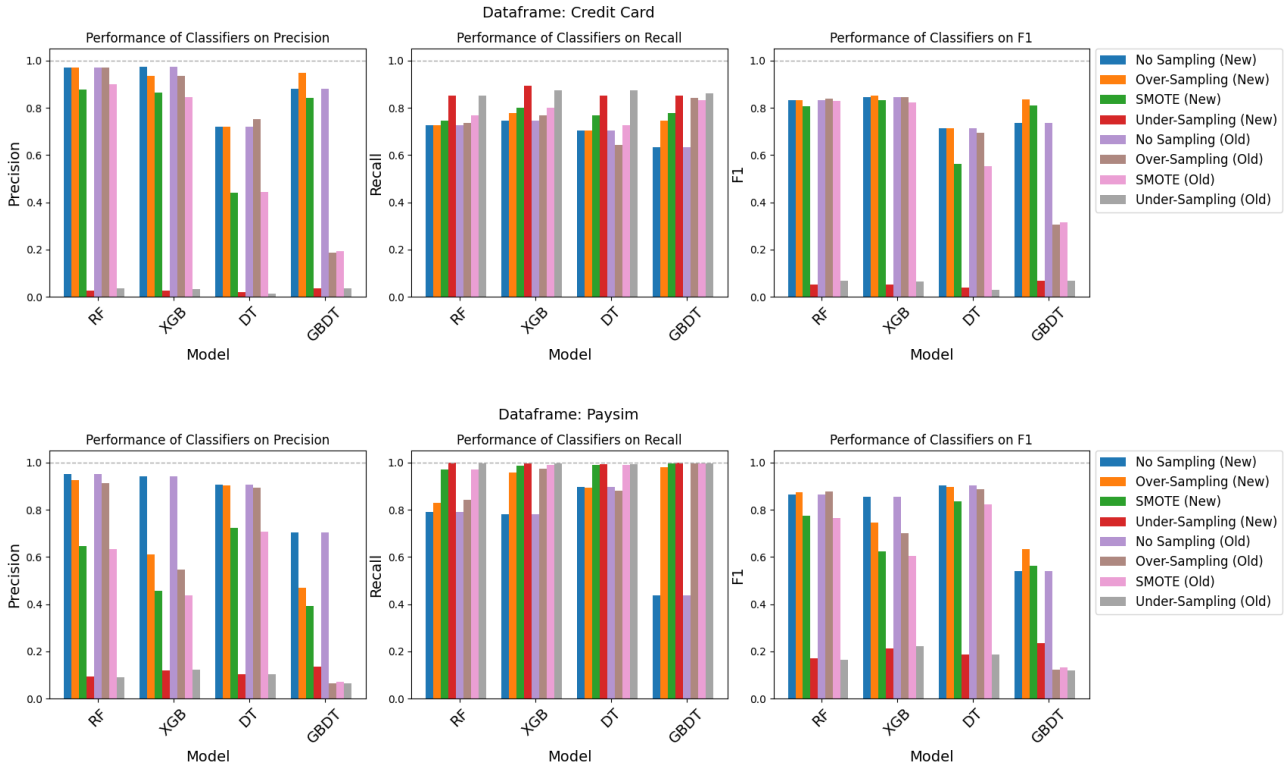

Figure 3: Experiment 2. Effect of sampling techniques: No Sampling, Over-Sampling, Under-Sampling, and SMOTE. Evaluation over Precision, Recall, and F1. Blue, orange, green, and red, marked as (New), correspond to optimising classifiers using the resampled training set. Purple, brown, pink, and grey, marked as (Old), correspond to using the parameters found in Experiment 1 and re-training with the resampled dataset. The upper row of figures corresponds to the results on Credit Card. The lower row corresponds to the results on PaySim.

3.3 Experiment 3. SPE and the number of base classifiers

Experiment 3 aims at evaluating SPE (ensembling), see Figure 4. As in Experiment 2, we evaluate XGB, RF, DT, and GBDT as initial classifiers for SPE with N=10, 20, and 50 classifiers. Interestingly, the more base classifiers, the higher the precision but the lower the recall. On Credit Card dataset, the optimal number of base classifiers is N=20. On PaySim, there are different effects for each classifier. XGB performs best with no more than N=10 base classifiers, while RF, DT, and GBDT do with N=20 base classifiers. Increasing complexity seems unnecessary.

On Credit Card, XGB with Over-Sampling ($F1 = 0.851$, Experiment 2) performs similarly as in ensembling with SPE, N=20 XGB base classifiers ($F1 = 0.854$, Experiment 3). On PaySim, the best F1 score was obtained by DT No-Sampling ($F1 = 0.902$, Experiment 1) just like in ensembling with SPE, N=50 DT base classifiers ($F1 = 0.902$, Experiment 3). These results indicate that it is important

to evaluate different classifiers on each problem and dataset. These results align with the No Free Lunch Theorem (Wolpert & Macready, 1997).

Table 3 presents the execution times. Most of the time, XGB is the fastest, followed by DT, RF, and GBDT, which is the slowest algorithm. Given that under sampling removes samples, it makes execution times the fastest, followed by no sampling, over sampling, and SMOTE. The execution time of SPE increases with the number of base classifiers as expected.

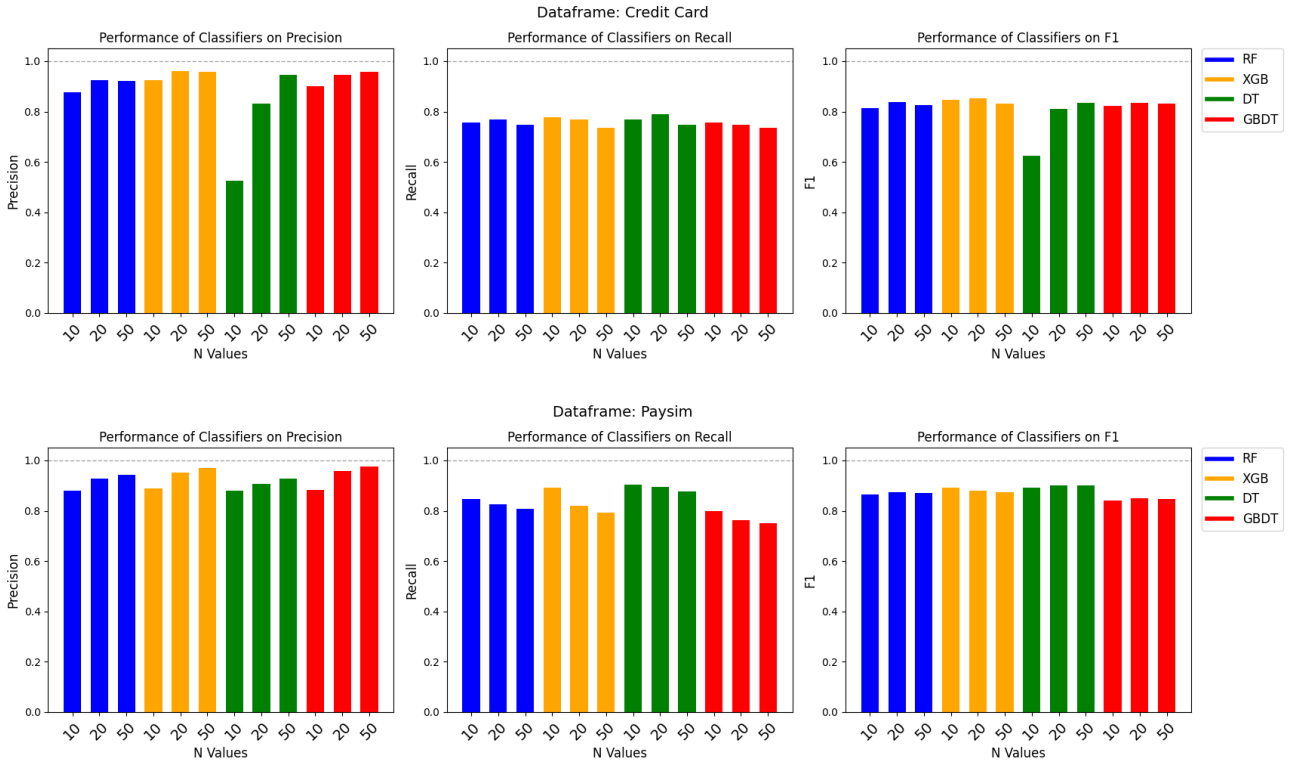

Figure 4: Experiment 3. SPE and the number of base classifiers (N = 10, 20, 50) evaluated over Precision, Recall, and F1. The upper row of figures corresponds to the results on Credit Card. The lower row corresponds to the results on PaySim.

Table 3: Execution time (minutes:seconds) for a single run including no-sampling, sampling or ensembling when model training. Fastest times per row in bold. Execution on Google Colab.

Dataset	**Method**		**RF**	**XGB**	**DT**	**GBDT**
Credit Fraud	No Sampling		00:52	**00:03**	00:26	07:20
	Over-sampling		00:24	**00:08**	00:15	09:16
	SMOTE		00:59	**00:11**	01:02	15:16
	Under-sampling		**00:00**	**00:00**	**00:00**	**00:00**
	SPE	Base Classifiers = 10	00:03	00:05	**00:01**	00:16
	SPE	Base Classifiers = 20	00:08	00:13	**00:06**	00:28
	SPE	Base Classifiers = 50	00:21	00:35	**00:04**	01:12
PaySim	No Sampling		00:34	**00:18**	00:50	23:47
	Over-sampling		00:56	**00:53**	00:55	34:58
	SMOTE		01:25	**00:59**	01:42	45:23
	Under-sampling		**00:00**	**00:00**	**00:00**	00:01
	SPE	Base Classifiers = 10	00:26	00:41	**00:16**	01:52
	SPE	Base Classifiers = 20	00:53	01:26	**00:31**	03:48
	SPE	Base Classifiers = 50	02:04	03:33	**01:15**	09:35

4 Conclusion

This study extensively evaluated various machine learning strategies to address the challenge posed by imbalanced datasets, particularly in classification tasks where the minority class is crucial yet underrepresented. We presented three experiments executed on two highly imbalanced and representative datasets containing numerical and categorical features: Credit Card dataset (283 726 samples, imbalance ratio: 598.84:1), and PaySim dataset (6 362 620 samples, imbalance ratio: 773.70:1). XGB and RF proved to be robust classifiers and performed well with or without sampling on both datasets in terms of classification accuracy, with XGB being faster than RF. Sampling techniques had positive and negative effects, similarly as reported by previous studies (Kim & Hwang, 2022; Hajek et al., 2022; Velarde et al., 2024). Sampling helped improve recall at the expense of deteriorating precision. While in many cases over-sampling the minority class helped improve recognition, under-sampling had a strong deteriorating effect, presumably due to loss of information. Except for GBDT, SMOTE had deteriorating effects, possibly because the generation of synthetic minority samples might introduce noise. Ensembling with SPE improved recognition in some cases. Therefore, we recommend evaluating different setups as these might perform differently depending on the dataset characteristics. The findings presented in this study contribute to better understanding and addressing several cybersecurity applications formulated as binary classification problems with imbalanced data, including anomaly detection, fraud detection, intrusion detection, spam detection, or malware detection.

Author contributions

M.L. experimental design, code, and preparation of tables and figures. G.V. supervision, experimental design, and writing.

References

Alpaydin, E. (2014). *Introduction to machine learning*. MIT press.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321–357.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794).

Dal Pozzolo, A., Boracchi, G., Caelen, O., Alippi, C., & Bontempi, G. (2017). Credit card fraud detection: a realistic modeling and a novel learning strategy. *IEEE transactions on neural networks and learning systems*, *29*(8), 3784–3797.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Hajek, P., Abedin, M. Z., & Sivarajah, U. (2022). Fraud detection in mobile payment systems using an xgboost-based framework. *Information Systems Frontiers*, 1–19.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*.

Kim, M., & Hwang, K.-B. (2022). An empirical evaluation of sampling methods for the classification of imbalanced data. *PLoS ONE*, *17*(7). (https://doi.org/10.1371/journal.pone.0271260)

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, *18*(17), 1-5. Retrieved from `http://jmlr.org/papers/v18/16-365.html`

Li, Y., Jin, J., Ma, J., Zhu, F., Jin, B., Liang, J., & Chen, C. P. (2023). Imbalanced least squares regression with adaptive weight learning. *Information Sciences*, *648*, 119541.

Liu, Z., Cao, W., Gao, Z., Bian, J., Chen, H., Chang, Y., & Liu, T.-Y. (2020). Self-paced ensemble for highly imbalanced massive data classification. In *2020 ieee 36th international conference on data engineering (icde)* (pp. 841–852).

Lopez-Rojas, E., Elmir, A., & Axelsson, S. (2016). Paysim: A financial mobile money simulator for fraud detection. In *28th european modeling and simulation symposium, emss, larnaca* (pp. 249–255).

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, *1*, 81–106.

Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, *10*(3), e0118432.

Sarker, I. H., Kayes, A., Badsha, S., Alqahtani, H., Watters, P., & Ng, A. (2020). Cybersecurity data science: an overview from machine learning perspective. *Journal of Big data*, *7*, 1–29.

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., ... Dennison, D. (2015). Hidden technical debt in machine learning systems. *Advances in neural information processing systems*, *28*.

Shaukat, K., Luo, S., Varadharajan, V., Hameed, I. A., Chen, S., Liu, D., & Li, J. (2020). Performance comparison and current challenges of using machine learning techniques in cybersecurity. *Energies*, *13*(10), 2509.

Velarde, G., Weichert, M., Deshmunkh, A., Deshmane, S., Sudhir, A., Sharma, K., & Joshi, V. (2024). Tree boosting methods for balanced and imbalanced classification and their robustness over time in risk assessment. *Intelligent Systems with Applications*, *22*, 200354. Retrieved from `https://www.sciencedirect.com/science/article/pii/S2667305324000309` doi: https://doi.org/10.1016/j.iswa.2024.200354

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, *1*(1), 67–82.

Yang, K., Yu, Z., Chen, C. P., Cao, W., Wong, H.-S., You, J., & Han, G. (2021). Progressive hybrid classifier ensemble for imbalanced data. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *52*(4), 2464–2478.

Yu, H.-F., Huang, F.-L., & Lin, C.-J. (2011). Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, *85*, 41–75.

A Appendix

Table 4: Experiment 1. Performance of individual classifiers. Values close to 1 are better. Tabular representation of the results presented in Figure 2. Best results in bold.

Dataset	Metric	RF	LGBM	XGB	LR	DT	GBDT
Credit Card	Precision	0.972	0.192	**0.973**	0.7	0.72	0.882
	Recall	0.726	0.526	**0.747**	0.663	0.705	0.632
	F1	0.831	0.281	**0.845**	0.681	0.713	0.736
Paysim	Precision	**0.953**	0.135	0.943	0.913	0.907	0.705
	Recall	0.792	0.158	0.782	0.453	**0.898**	0.436
	F1	0.865	0.855	0.605	0.865	**0.902**	0.539

Table 5: Experiment 2. Effect of sampling the training set, results reported on test set. Values close to 1 are better. Tabular representation of the results presented in Figure 3, Blue, orange, green, and red, marked as (New). Best results in bold.

Dataset	Model	Metric	No Sampling	Over-Sampling	SMOTE	Under-Sampling
Credit Card	RF	Precision	0.972	**0.972**	0.877	0.027
		Recall	0.726	0.726	0.747	**0.853**
		F1	**0.831**	**0.831**	0.807	0.052
	XGB	Precision	**0.973**	0.937	0.864	0.026
		Recall	0.747	0.779	0.8	**0.895**
		F1	0.845	**0.851**	0.831	0.051
	DT	Precision	**0.72**	**0.72**	0.442	0.02
		Recall	0.705	0.705	0.768	**0.853**
		F1	**0.713**	**0.713**	0.562	0.04
	GBDT	Precision	0.882	0.947	0.841	0.036
		Recall	0.632	0.747	0.779	**0.853**
		F1	0.736	**0.835**	0.809	0.069
Paysim	RF	Precision	**0.953**	0.927	0.646	0.093
		Recall	0.792	0.829	0.971	**0.999**
		F1	0.865	**0.875**	0.776	0.17
	XGB	Precision	**0.943**	0.612	0.458	0.118
		Recall	0.782	0.957	0.987	**0.998**
		F1	**0.855**	0.747	0.625	0.211
	DT	Precision	**0.907**	0.903	0.722	0.103
		Recall	0.898	0.893	0.991	**0.994**
		F1	**0.902**	0.898	0.836	0.186
	GBDT	Precision	**0.705**	0.468	0.391	0.134
		Recall	0.436	0.981	0.995	**0.999**
		F1	0.539	**0.634**	0.562	0.236

Table 6: Experiment 3. SPE with N=10, 20, and 50 base classifiers. Tabular representation of the results presented in Figure 4. Best results in bold.

Dataset	**Method**	**Base Classifiers**	**Metric**	**RF**	**XGB**	**DT**	**GBDT**
Credit Card	SPE	N = 10	Precision	0.878	**0.925**	0.525	0.9
			Recall	0.758	**0.779**	0.768	0.758
			F1	0.814	**0.846**	0.624	0.823
		N = 20	Precision	0.924	**0.961**	0.833	0.947
			Recall	0.768	0.768	**0.789**	0.747
			F1	0.839	**0.854**	0.811	0.835
		N = 50	Precision	0.922	**0.959**	0.947	0.959
			Recall	**0.747**	0.737	**0.747**	0.737
			F1	0.826	0.833	**0.835**	0.833
Payment Simulation	SPE	N = 10	Precision	0.881	**0.89**	0.881	0.883
			Recall	0.848	0.892	**0.904**	0.8
			F1	0.864	0.891	**0.893**	0.84
		N = 20	Precision	0.928	0.951	0.908	**0.959**
			Recall	0.825	0.821	**0.894**	0.764
			F1	0.873	0.881	**0.901**	0.851
		N = 50	Precision	0.943	0.969	0.928	**0.975**
			Recall	0.809	0.794	**0.877**	0.751
			F1	0.871	0.873	**0.902**	0.848