Clash of titans on imbalanced data: TabNet vs XGBoost

1st Róbert Kanász
Faculty of Electrical Engineering and Informatics
Technical University of Košice
Košice, Slovakia
robert.kanasz@student.tuke.sk

2nd Peter Drotár
Faculty of Electrical Engineering and Informatics
Technical University of Košice
Košice, Slovakia
peter.drotar@tuke.sk

3rd Peter Gnip
Faculty of Electrical Engineering and Informatics
Technical University of Košice
Košice, Slovakia
peter.gnip@tuke.sk

4th Martin Zoričák
Faculty of Economic
Technical University of Košice
Košice, Slovakia
martin.zoricak@tuke.sk

***Abstract*—In machine learning, particularly with tabular data, ensemble methods and neural networks stand as the preeminent approaches for predictive modeling. Among these, XGBoost and TabNet have demonstrated remarkable efficacy and interpretability. However, one critical challenge in these methodologies is their performance on imbalanced datasets, a common yet intricate issue in many real-world applications. This research paper proposes novel modifications to TabNet, tailored to enhance its performance on imbalanced tabular datasets. Our methodology introduces multiple loss functions for the TabNet architecture. These modifications improve the models' sensitivity to minority classes and enhance overall predictive accuracy on imbalanced data. We conducted a comprehensive performance comparison using various synthetic and real-world datasets characterized by significant class imbalances. TabNet, combined with IBLoss, achieved a GM score on real-world data up to 92%. On synthetic data, the highest GM score was up to 82% using TabNet in combination with BVSLoss. These results demonstrate that TabNet is robust to imbalanced datasets and can learn well even on imbalanced data. The performance is further boosted by incorporating a loss function built for imbalanced data, such as BVSLoss or IBLoss. On the other hand, XGBoost fails to converge if not adapted to imbalanced data with sampling or cost-sensitive learning, resulting in less accurate prediction performance.**

***Index Terms*—imbalanced learning, TabNet, XGBoost, oversampling, bankruptcy**

I. Introduction

Machine learning has seen significant advancements in recent years, revolutionizing domains such as natural language processing, computer vision, and healthcare. As machine learning models grow in complexity and capability, they face a critical challenge when confronted with imbalanced datasets [1]. Imbalanced datasets are characterized by a significant disparity in the distribution of classes, where, in the case of binary classification tasks, one class (minority class) noticeably outnumbers another (majority class). This inherent class imbalance poses a significant obstacle to the practical application of machine learning techniques, as it can lead to biased and suboptimal model performance [2].

Imbalanced learning is a crucial issue that is present in many real-world scenarios such as medical diagnoses [3], bankruptcy prediction [4] or fraud detection [5]. Positive cases (e.g., diseases, frauds, or rare events) are often significantly lower than negative cases, creating a natural class imbalance. Therefore, addressing imbalanced data is essential for successfully deploying machine learning in these critical domains.

Secondly, the imbalanced nature of data can severely affect model training, evaluation, and generalization [6]. Traditional machine learning algorithms are biased towards the majority class when applied directly to imbalanced datasets as they assume relatively balanced class distribution. Consequently, they may achieve high accuracy in predicting the majority class but perform poorly on the minority class. This bias can have serious consequences, mainly when misclassification of minority instances is costly or has critical implications. Additionally, traditional evaluation metrics such as the accuracy score can be misleading in imbalanced settings, as a model that predicts all instances as the majority class can still achieve high accuracy but provide little practical value.

TabNet [7] and XGBoost [8] have emerged as significant breakthroughs for their unparalleled efficacy in handling tabular data, a standard format in many real-world applications. TabNet, with its unique deep learning approach, leverages attention mechanisms to selectively consider features at each decision step, offering interpretability akin to decision trees while harnessing the power of neural networks. On the other hand, XGBoost, a scalable and efficient implementation of gradient boosting, has gained immense popularity due to its performance and speed, making it a go-to algorithm for many predictive tasks. They not only provide state-of-the-art accuracy but also introduce a level of interpretability that is critical in fields ranging from finance to healthcare.

Several machine-learning approaches have emerged in recent years to combat imbalanced scenarios. The most popular ones are sampling techniques [9], ensemble learning [10], outlier detection methods [11], cost-sensitive learning [12], algorithmic modifications [13], and a mixture of the approaches mentioned above, well-known as hybrid methods [14].

Sampling involves either oversampling the minority class or undersampling the majority class to balance the dataset. Cost-sensitive learning adjusts the learning process to place a higher penalty on misclassifying the minority class. Algorithmic modifications involve adapting existing algorithms to be more sensitive to the imbalance, such as using different thresholds for classification. Ensemble methods, like bagging and boosting, can be effective, especially when combined with oversampling or cost-sensitive techniques [15]. These approaches aim to improve model performance on minority classes without compromising overall accuracy.

In this paper, we propose several new variations of TabNet architecture and compare them with the performance of imbalanced XGBoost. The experimental results on multiple synthetic and real-world datasets indicate that imbalanced TabNet can match and even outperform the predictions of imbalanced XGBoost.

II. Imbalanced TabNet

This section briefly describes the imbalanced TabNet utilized in the experimental study.

A. TabNet

TabNet [7] is a novel tabular learning method that combines the strengths of both deep neural networks and decision trees. Its encoder structure comprises multiple steps (N_{steps}). In this sequence, each step i receives information processed from the preceding step $(i-1)$. This information is used to select relevant features, and the step then produces a processed feature representation. These representations from each step are collectively aggregated and contribute to the final decision-making process. The model takes in a dataset with a specified batch size B and features of specific dimensions D. Importantly, this input does not require global feature normalization before being fed into the model. Once the data is inputted, it goes through a batch normalization layer, which is forwarded to a feature transformer for further processing.

The feature transformer in TabNet is composed of a series of n gated linear unit (GLU) blocks. Each GLU block incorporates three essential layers: a fully connected layer, a batch normalization layer, and a GLU layer. In a scenario where four GLU blocks are utilized, the configuration includes two shared blocks and two operated independently. This design aids in achieving robust and efficient learning in terms of parameter usage. Additionally, the architecture features a skip connection between every two consecutive blocks, enhancing the flow of information and gradients during the learning process. After each block, a normalization step is applied to ensure stability and control variance, explicitly using the $\sqrt{0.5}$. After batch normalization, the processed features are directed through the feature transformer. The transformed output is subsequently channeled to the attentive transformer corresponding to the i-th step via a split layer. This mechanism allows for effective and nuanced processing of features, which is crucial for the model's performance in handling tabular data.

The attentive transformer in TabNet comprises four layers: a fully connected layer, batch normalization layer, prior scales, and sparsemax. It starts by receiving input from a split layer and then sequentially processes this data through the fully connected and batch normalization layers. Following these, the input is directed to the prior scales layer, aggregating the magnitudes of features used in previous decision steps. This aggregation informs the model about the extent of each feature's usage before the current step, playing a key role in dynamic feature selection. Finally, the data passes through the sparsemax layer, which refines the focus on the most relevant features for the model's predictions. The sparsemax layer can be defined by the following equation

$$P[i] = \prod_{j=1}^{i} (\gamma - M[j]), \qquad (1)$$

where γ is the relaxation parameter. The attentive transformer in TabNet calculates the mask layer for the current step using the results from the prior step. Its main role is to create a learnable mask ($M[i] \in \mathbb{R}^{B*D}$), which selectively identifies the most important features for that step. This selective process ensures the model's efficiency, as it concentrates the learning on relevant features without wasting capacity on irrelevant ones. Masking is a multiplicative process, applying these learned masks to the features processed in the previous step ($a[i-1]$), thereby guiding the model to focus on key features in each decision step. The mask can be expressed with the following equation

$$M[i] = \text{sparsemax}\, P[i-1] * h_i(a[i-1]), \qquad (2)$$

where $h(\bullet)$ is the trainable function that represents the fully connected and batch normalization layers, $P[i-1]$ is the prior scales item, and the sparsemax layer is used for coefficient normalization, resulting in sparse feature selection. The mask enables interpretability and improves feature selection from the attentive transformer. $M_{bj}[i]$ represents $j-th$ feature of $b-th$ samlpe. If $M_{bj}[i] = 0$, there is no contribution from the feature at that step. Aggregating these masks at each step creates a coefficient that weights the importance of each step in a final decision.

B. Vector-scaling Loss and Binary Vector-scaling Loss

The vector-scaling loss (VSLoss) [16] and binary vector-scaling loss (BVSLoss) are both loss functions designed for classification tasks involving imbalanced datasets. However, there are some key differences between these loss functions.

The VSLoss is a more general loss function that can be used for binary and multi-class classification. It is based on the idea of scaling the logits of the classifier before applying the cross-entropy loss. This scaling is done in a way that is designed to give more weight to the minority classes. The

VSLoss involves two stages of scaling. In the first stage, the logits are scaled by a vector of multiplicative logit parameters. These parameters are optimized during training to achieve a specific goal, such as minimizing the loss or maximizing the accuracy of the classifier for the minority class. In the second stage, the scaled logits are further scaled by a vector of additive logit parameters. These parameters are also optimized during training to improve the classifier's performance, particularly in the terminal phase of training. The VSLoss loss is expressed as follows

$$L_{VS}(y, \mathbf{f}_\omega(\mathbf{x})) = -\omega_y \cdot \log\left(\frac{e^{\Delta_y \mathbf{f}_y(\mathbf{x})+\iota_y}}{\sum_{c\in[C]} e^{\Delta_c \mathbf{f}_c(\mathbf{x})+\iota_c}}\right). \quad (3)$$

The BVSLoss is a special VSLoss case specifically designed for binary classification. It is based on the same idea of scaling the logits but uses a different scaling function that is more effective for binary classification. The BVSLoss is also more efficient than the VSLoss as it requires no additional parameters. The following equation defines the BVSLoss

$$L_{BVS}(y, \mathbf{f}_\omega(\mathbf{x})) = \omega_y \cdot \log\left(1 + e^{\iota_y} \cdot e^{-\Delta_y y f_\omega(\mathbf{x})}\right). \quad (4)$$

Here, in formulas (3) and (4) $\mathbf{f_w} : \mathbb{R}^d \to \mathbb{R}^C$, $\mathbf{f_w}(\mathbf{x}) = [\mathbf{f_1}, ..., \mathbf{f_C}]$ is the vector of logits.

C. Influence-balanced Loss

Influence-balanced loss (IBLoss) [17] is a newly developed loss function for classification tasks that involve imbalanced datasets. It aims to address the class imbalance issue by adapting the loss function to the importance of each sample rather than treating them equally. This is achieved by considering the influence of each sample on the decision boundary during the training process. The influence of the sample can be defined by the influence-balanced weighting factor expressed by the following formula

$$IBLoss(x; \omega) = \parallel \nabla_\omega L(y, f(x, \omega)) \parallel_1, \quad (5)$$

also called the magnitude of the gradient descent. Re-weighting samples by the magnitude of the gradient vector can successfully down-weight samples from dominant classes. To define the IBLoss, let $h = [h_1, ..., h_L]^T$ be an input to the last fully connected layer of the model. Next, define the output of the fully connected layer as $f(x, \omega) = [f_1, ..., f_K]^T$. The gradient of the loss function can be computed as follows

$$\frac{\partial}{\partial \omega_{kl}} L(y, f(x, \omega)) = (f_k - y_k)h_l. \quad (6)$$

The weight matrix of the last fully connected layer is denoted by $\omega = [\omega_1, ..., \omega_K]^T \in R^{K\times L}$. Using (6), the $IBLoss(x; \omega)$ is given by the following equation

$$IBLoss(x; \omega) = \parallel f(x, \omega) - y \parallel_1 \cdot \parallel h \parallel_1 . \quad (7)$$

Then, the equation (7) can be used as a re-weighting factor to downscale the influence of the majority class samples. The IBLoss function is expressed as follows

$$L_{IBLoss}(y, f(x, \omega)) = \frac{L(y, f(x, \omega))}{\parallel f(x, \omega) - y \parallel_1 \cdot \parallel h \parallel_1}. \quad (8)$$

D. Label-Distribution-Aware Margin Loss

The label-distribution-aware margin loss (LDAMLoss) [18] is a sophisticated approach for handling class imbalance in classification tasks. The foundation of LDAMLoss is cross-entropy loss, commonly used in classification tasks. The cross-entropy loss measures the difference between the predicted probability distribution (outputted by the model) and the true distribution (actual class labels). The standard cross-entropy loss can be expressed as follows

$$L_{CE} = -\log\left(\frac{e^{f_{y_i}}}{\sum_{j=1}^{C} e^{f_j}}\right), \quad (9)$$

where f_j is the logit (i.e., the raw output of the neural network) for class j, y_i is the true class for the $i-$th sample. The LDAMLoss introduces a margin term to the cross-entropy loss. A margin in the context of classification can be thought of as a buffer zone that separates classes. In simple terms, it's the "extra effort" the model needs to put into classifying a sample correctly. LDAMLoss introduces this concept by adding a margin term to the logits of the true class before computing the cross-entropy loss. The margin is not constant; it varies inversely with the square root of class frequency, which can be expressed by the following formula

$$\Delta_c = \frac{C}{\sqrt{n_c}}, \quad (10)$$

where C is a hyperparameter that controls the overall scale of the margins, and n_c is the number of samples in class c. This design inherently gives more "breathing space" to minority classes. Then, the LDAMLoss formula is as follows

$$L_{LDAMLoss} = -\log\left(\frac{e^{f_{y_i}-\Delta_{y_i}}}{e^{f_{y_i}-\Delta_{y_i}} + \sum_{j\neq y_i} e^{f_j}}\right). \quad (11)$$

Here, Δ_{y_i} is the margin for the true class y_i of the sample. These equations are the basis of the LDAMLoss, capturing the essence of how it adjusts the standard cross-entropy loss to account for class imbalance by introducing a class-dependent margin.

E. Mean Divergence Regularization

Mean divergence regularization (MDR) [19] is a machine learning technique used to enhance a model's generalization capability by ensuring that its outputs or some internal representations do not deviate excessively from their average values across the training dataset. This approach can be particularly useful in complex models like deep neural networks.

Let be $\hat{y} = F(x) \in [0, 1]$ the output of the model. If the distribution of the network's output $p(\hat{y}|x)$, the prior probability of each class is not identical. Let's consider standard cross-entropy loss and dataset $D = \{D_+, D_-\}$. The mean of the cross-entropy for the dataset D can be expressed with the following formula

$$\begin{aligned} L_{CE} &= -\frac{1}{|D|}\sum_{i=1}^{|D|}\left[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)\right] \\ &\approx -\int m \log \hat{y} + n \log(1-\hat{y})\, dx. \end{aligned} \tag{12}$$

Here, $m = p(x = D_+)$ and $n = p(x = D_-)$. If $\hat{y} = \frac{m}{m+n}$, the equation (12) takes on a minimum. In case $m = n$, $\hat{y}$ converges to $1/2$. The distribution has a skewed mean when m is greater than n or n is greater than m. In unbalanced datasets, negative samples are greater than positive; thus, the mean of distribution $p(\hat{y}|x)$ becomes close to 0. A regularization term is added to the loss function to mitigate the model's bias to the negative class samples. The cross-entropy loss function with regularization term can be defined as follows

$$L_{CE+MDR} = L_{CE} + \frac{\lambda}{2}\left|\frac{1}{2} - E_x\left[F(x)\right]\right|^2. \tag{13}$$

In the case of training the model with mini-batches of size M, the function has the following form:

$$L_{CE+MDR} = \frac{1}{|M|}\sum_{i=1}^{|M|} L_{CE(i)} + \frac{\lambda}{2}\left|\frac{1}{2} - \frac{1}{|M|}\sum_{i=1}^{|M|} F(x_i)\right|^2. \tag{14}$$

In (13) and (14), the λ is the hyperparameter. The main goal of the regularization term is to set a penalty as the difference between the optimal mean $1/2$ and the mean of the network's output and prevent the model from being highly skewed.

III. XGBoost

XGBoost is a decision-tree-based ensemble machine learning algorithm that uses a gradient-boosting framework. Prediction problems involving unstructured data (like images or text) are not as effective as deep learning, but structured/tabular data is often considered one of the best. The core of XGBoost is the gradient-boosting algorithm. Gradient boosting involves building models sequentially, where each new model incrementally reduces the errors of the previous models. Models are added until no further improvements can be made. The gradient boosting approach minimizes a loss function, which measures the difference between the predicted and actual values. In XGBoost, decision trees are used as base estimators. Each decision tree is added to correct the errors made by the previous trees. Unlike random forests, which build each tree independently, gradient boosting sequentially combines them.

The complexity of the XGBoost algorithm can be described in terms of its objective function and the factors contributing to the computational complexity of training the model. The objective function in XGBoost for training at each step includes both a loss function and a regularization term. If we have a dataset with n samples and m features, and f represents the model (ensemble of trees), the objective function at iteration t is given as

$$Obj^t = \sum_{i=1}^{n} l(y_i, \hat{y}^{y-1} + f_t(x_i)) + \Omega(f_t), \tag{15}$$

where l is a differentiable convex loss function that measures the difference between the predicted value $\hat{y}^{y-1} + f_t(x_i)$ and the actual value and y_i. The function $\Omega(f_t)$ represents the regularization term, which typically includes the complexity of the tree f_t being added at step t. The regularization term $\Omega(f_t)$ is usually defined as

$$\Omega(f_t) = \alpha T + \frac{1}{2}\beta\sum_{j=1}^{T}\omega_j^2, \tag{16}$$

where T represents the number of leaves in the tree, ω_j is the score on the $j-th$ leaf, α is the parameter penalizing the number of leaves, and β is the L2 regularization term on the leaf weights.

IV. Experiments

This section briefly describes the data and methodology used throughout the experimental study.

A. Datasets

This study focuses on two types of data, the real-world and the synthetic, to assess the performance of the proposed imbalanced TabNet approaches. The overview of the utilized datasets is presented in Table I. The synthetic datasets (*Synth_1*, *Synth_2*, and *Synth_3*) were artificially generated using Scikit-learn [21] library generators.

TABLE I
Detailed characteristic of utilized datasets

Dataset	Samples	Features	Imbalance ratio
Synth_1	1000	50	3:1
Synth_2	500	100	12:1
Synth_3	500	200	33:1
Bankruptcy_13	1230	60	50:1
Bankruptcy_14	1448	60	49:1
Bankruptcy_16	2174	60	156:1
Car_Claims	1800	25	7:1

The real-world data were represented by bankruptcy prediction datasets (*Bankruptcy_13*, *Bankruptcy_14* and *Bankruptcy_16*) [22] and fraudulent claim on cars physical damage dataset (*Car_Claims*) [23]. Bankruptcy prediction datasets are constituted by financial ratios of thousands of small and medium-sized enterprises from construction business areas operating in the Slovak Republic. These datasets represent the timespan of 3 different evaluation years, namely 2013, 2014, and 2016. In this context, the evaluation year is defined as the year a company is evaluated as either a going concern (non-bankrupt) or financially distressed (bankrupt). The data gathered from publicly accessible annual reports characterized each company through financial ratios based on information from annual reports spanning three years before the evaluation year.

The *Car_Claims* dataset contains a wide range of information about car insurance claims, including details about the claimant, the vehicle involved, the damage's extent, and the claim's outcome. We randomly subsampled the original dataset with 17998 samples to decrease the number of samples while keeping the imbalanced ratio.

B. Methodology

Before applying the selected classifiers, it was necessary to clean the real-world data. This involved two main steps: replacing missing values with the attribute's average and standardizing each feature to have a zero mean and unit variance. Additionally, we applied one-hot-encoding to categorical data for the *Car_Claims* dataset. We incorporated a genetic algorithm for fine-tuning. This approach allowed for an optimized selection of parameters, enhancing the overall performance of the machine-learning models. For each set of hyperparameters, we implemented a 5-fold stratified cross-validation process.

Choosing an appropriate evaluation metric is crucial when dealing with imbalanced data. We used the geometric mean (GM) score [20] to assess the selected classification models' performance. GM is defined as the square root of the product of the sensitivity and specificity. Mathematically, it can be defined by the following formula

$$GM = \sqrt{sensitivity * specificity}. \tag{17}$$

Sensitivity represents a true positive rate, whilst specificity represents a true negative rate.

C. Results

Our experiments aimed to explore how TabNet performs with different loss functions on the provided datasets, specifically focusing on identifying the most effective combination for handling imbalanced classification tasks.

The overview of the GM scores for the experiments performed on synthetic datasets is depicted in Table II. The most notable outcomes were observed using Weighted-XGBoost, Tabnet+BVSLoss, and Tabnet+BVSLoss+MDR approaches, with the best GM score of over 82%. Specifically for the *Synth_1* dataset, the Weighted-XGBoost model demonstrated superior performance. The best results were obtained for the other two synthetic datasets using a combination of Tabnet and BVSLoss functions. Here, the best GM score yielded from 68.2% to 82.13%. In those cases, only XGBoost-based models perform poorly, especially for *Synth_2* and *Synth_3* datasets, where the number of features was high compared to the number of samples. This caused some of the features to be irrelevant or noisy, providing little to no predictive power. XGBoost might focus on these features, leading to a decrease in overall model performance.

The results of experiments performed on real-world data are depicted in Table III. The best results in terms of GM score were obtained using Weighted-XGBoost, TabNet+LDAMLoss, SMOTE+TabNet, and SMOTE+XGBoost models. In the case of the *Bankruptcy_13* dataset, the Weighted XGBoost performed the best with a GM score greater than 83%. A slightly lower performance was achieved using TabNet-based models combined with the SMOTE technique or IBLoss function. The other eight TabNet-based models achieved similar results, while the pure XGBoost and XGBoost combined with SMOTE achieved significantly decreased prediction performances. On dataset *Bankruptcy_14*, the best performance achieved a combination of TabNet with LDAMLoss function and MDR. From TabNet-based models, only the pure TabNet model achieved performance lower than 80%. From XGBoost models, only weighted XGBoost achieved performance higher than 80%. A similar situation occurred in the case of *Bankruptcy_16*, where TabNet-based models outperformed most of the XGBoost-based models. The best-performing model was TabNet, which was combined with the SMOTE technique. On *Car_Claims* dataset, the best performance was achieved by the XGBoost model in combination with the SMOTE technique. The weighted XGBoost model achieved slightly lower performance, whilst the pure XGBoost achieved the lowest performance from all classifiers.

V. Conclusions

The challenge of uneven tabular data distribution significantly impacts the current landscape of machine learning. In our study, we conducted experiments using the newly proposed TabNet architectures. This was compared with the performance of the XGBoost classifier, a state-of-the-art approach to imbalanced learning. We utilized several different loss functions and comparatively analyzed classification outcomes. The results of these experiments led to a series of insightful conclusions. First, the vanilla TabNet model, without using the oversampling technique and modified loss function, was better than the vanilla XGBoost. Second, on synthetic datasets with a higher imbalanced ratio, imbalanced TabNet achieved a higher GM score than imbalanced XGBoost. Even though TabNet is a deep neural network with a vast internal capacity, it can be generalized well even on datasets with smaller samples. Third, the loss functions designed for imbalanced problems helped to increase the predictive performance of TabNet compared to TabNet with a cross-entropy loss function.

Acknowledgment

This work was supported by the Slovak Research and Development Agency under contact no. APVV-21-0318 and by the Ministry of Education, Science, Research and Sport of the Slovak Republic under contact no. VEGA 1/0174/24.

References

[1] Oksuz, K., Cam, B. C., Kalkan, S., & Akbas, E. (2020). Imbalance problems in object detection: A review. IEEE transactions on pattern analysis and machine intelligence, 43(10), 3388-3415.

[2] Thabtah, F., Hammoud, S., Kamalov, F., & Gonsalves, A. (2020). Data imbalance in classification: Experimental evaluation. Information Sciences, 513, 429-441.

[3] Liu, N., Li, X., Qi, E., Xu, M., Li, L., & Gao, B. (2020). A novel ensemble learning paradigm for medical diagnosis with imbalanced data. IEEE Access, 8, 171263-171280.

TABLE II
OVERVIEW OF THE BEST GM SCORES (%) ACHIEVED ON THE SYNTHETIC DATASETS (± FOR STANDARD DEVIATION)

Method	Synth_1	Synth_2	Synth_3	AVG_GM
XGBoost	54.73±3	0	0	18.24±32
SMOTE+XGBoost	60.65±3	20.66±25	0	27.10±31
Weighted XGBoost	**66.04±2**	59.61±14	32.00±18	52.55±18
TabNet	62.25±3	61.91±5	75.63±12	66.6±8
SMOTE+TabNet	64.26±3	66.42±15	63.53±32	64.74±2
TabNet+BVSLoss	62.08±3	65.46±6	**82.13±10**	**69.89±11**
TabNet+BVSLoss+MDR	61.03±5	**68.20±13**	75.27±14	68.16±7
TabNet+VSLoss	62.11±5	61.68±9	70.12±16	64.63±5
TabNet+VSLoss+MDR	60.71±2	66.50±14	68.71±14	65.3±4
TabNet+IBLoss	63.47±3	57.56±10	74.46±18	65.16±9
TabNet+IBLoss+MDR	60.40±3	61.66±19	77.61±10	66.41±10
TabNet+LDAMLoss	62.28±2	65.96±14	69.01±11	65.75±3
TabNet+LDAMLoss+MDR	63.44±2	60.10±9	73.07±17	65.54±7

TABLE III
OVERVIEW OF THE BEST GM SCORES (%) ACHIEVED ON THE REAL-WORLD DATASETS (± FOR STANDARD DEVIATION)

Method	Bankruptcy_13	Bankruptcy_14	Bankruptcy_16	Car_Claims	AVG_GM
XGBoost	37.06±32	42.72±22	27.88±35	28.97±6	34.16±7
SMOTE+XGBoost	68.49±22	79.71±12	69.80±11	**60.49±2**	69.62±8
Weighted XBoost	**83.82±1**	83.84±5	94.35±2	58.12±2	**80.03±15**
TabNet	78.94±7	79.55±3	91.77±6	58.67±3	77.23±14
SMOTE+TabNet	81.01±6	83.36±2	**97.17±1**	58.60±2	**80.03±16**
TabNet+BVSLoss	75.87±7	81.10±7	90.59±7	58.58±2	76.54±13
TabNet+BVSLoss+MDR	76.23±5	80.25±10	89.88±7	59.06±3	76.35±13
TabNet+VSLoss	80.05±7	81.45±10	89.20±7	59.71±2	77.60±13
TabNet+VSLoss+MDR	78.42±6	81.01±7	89.26±3	59.15±3	76.68±12
TabNet+IBLoss	80.29±7	81.09±4	92.99±5	59.21±2	78.39±14
TabNet+IBLoss+MDR	78.08±9	82.88±8	91.39±8	57.22±1	72.96±12
TabNet+LDAMLoss	79.88±5	**84.51±8**	92.52±9	59.54±4	79.11±14
TabNet+LDAMLoss+MDR	77.25±8	81.60±4	90.40±7	59.81±3	77.26±13

[4] Kanász, R., Gnip, P., Zoričák, M., & Drotár, P. (2023). Bankruptcy prediction using ensemble of autoencoders optimized by genetic algorithm. PeerJ Computer Science, 9, e1257.

[5] Singh, A., Singh, A., Aggarwal, A., & Chauhan, A. (2022, November). Design and Implementation of Different Machine Learning Algorithms for Credit Card Fraud Detection. In 2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME) (pp. 1-6). IEEE.

[6] Sadreddin, A., & Sadaoui, S. (2022, December). Training and Testing Cascades for Imbalanced Data Classification. In 2022 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 261-268). IEEE.

[7] Arik, S. Ö., & Pfister, T. (2021, May). Tabnet: Attentive interpretable tabular learning. In Proceedings of the AAAI conference on artificial intelligence (Vol. 35, No. 8, pp. 6679-6687).

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

[9] Ashtiani, M. N., & Raahemi, B. (2023, July). An Efficient Resampling Technique for Financial Statements Fraud Detection: A Comparative Study. In 2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME) (pp. 1-7). IEEE.

[10] Dong, X., Yu, Z., Cao, W., Shi, Y., & Ma, Q. (2020). A survey on ensemble learning. Frontiers of Computer Science, 14, 241-258.

[11] Wang, H., Bah, M. J., & Hammad, M. (2019). Progress in outlier detection techniques: A survey. Ieee Access, 7, 107964-108000.

[12] Chen, Y. (2021, May). Research on Cost-sensitive Classification Methods for Imbalanced Data. In 2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA) (pp. 224-228). IEEE.

[13] Ahmed, Z., & Das, S. (2023). A Comparative Analysis on Recent Methods for Addressing Imbalance Classification. SN Computer Science, 5(1), 30.

[14] Yotsawat, W., Wattuya, P., & Srivihok, A. (2021). A novel method for credit scoring based on cost-sensitive neural network ensemble. IEEE Access, 9, 78521-78537.

[15] Phankokkruad, M. (2020, August). Cost-sensitive extreme gradient boosting for imbalanced classification of breast cancer diagnosis. In 2020 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE) (pp. 46-51). IEEE.

[16] Kini, G. R., Paraskevas, O., Oymak, S., & Thrampoulidis, C. (2021). Label-imbalanced and group-sensitive classification under overparameterization. Advances in Neural Information Processing Systems, 34, 18970-18983.

[17] Park, S., Lim, J., Jeon, Y., & Choi, J. Y. (2021). Influence-balanced loss for imbalanced visual classification. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 735-744).

[18] Cao, K., Wei, C., Gaidon, A., Arechiga, N., & Ma, T. (2019). Learning imbalanced datasets with label-distribution-aware margin loss. Advances in neural information processing systems, 32.

[19] Kim, B., Ko, Y., & Seo, J. (2022). Novel regularization method for the class imbalance problem. Expert Systems with Applications, 188, 115974.

[20] Akosa, J. (2017, April). Predictive accuracy: A misleading performance measure for highly imbalanced data. In Proceedings of the SAS global forum (Vol. 12, pp. 1-4). Cary, NC, USA: SAS Institute Inc..

[21] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.

[22] Zoričák, M., Gnip, P., Drotár, P., & Gazda, V. (2020). Bankruptcy prediction for small-and medium-sized companies using severely imbalanced datasets. Economic Modelling, 84, 165-176.

[23] Xu, B., Wang, Y., Liao, X., & Wang, K. (2023). Efficient fraud detection using deep boosting decision trees. Decision Support Systems, 175, 114037.