# Machine Learning Meets Tax Fraud: Insights from Slovakia[1]

*Eduard BAUMÖHL* – Roderik ANTOL** – Tomáš VÝROST* – Tomáš BAČO****

**Abstract**

*One of the most intriguing topics in the field of corporate finance is the detection of tax fraud. We consider a unique dataset of outcomes from Slovak tax authority audits, obtaining valuable insights into verified instances of tax manipulation and avoiding the misclassification problem that is common in this stream of literature. We apply artificial neural networks, random forests, XGBoost, and support vector machines to verify the extent to which we can classify tax manipulators on the basis of publicly available financial statement indicators. Our results show that the XGBoost model demonstrated the highest effectiveness, achieving an F1 score of 0.75 in the full sample, slightly lower scores within the industry groups, and excellent results in sector A – Agriculture, with an F1 score of 0.85. Our results indicate that the use of nowadays commonly known machine learning methods along with standard financial variables can provide a useful tool for tax fraud detection and, as such, can contribute to higher efficiency of tax audits.*

**Keywords:** *tax frauds, detection models, machine learning, earnings management*

**JEL Classification:** C63, G30, G38, K22, K42, M41

**DOI:** https://doi.org/10.31577/ekoncas.2025.05-06.01

**Article History:** *Received*: February 2025 *Accepted*: August 2025

* Eduard BAUMÖHL, corresponding author – Tomáš VÝROST, University of Economics, Faculty of Commerce, Dolnozemská cesta 1, 852 35 Bratislava, Slovakia; Institute of Economic Research, Slovak Academy of Sciences, Šancová 56, 811 05 Bratislava, Slovakia; Masaryk University, Faculty of Economics and Administration, Department of Finance, Lipová 41a, 602 00 Brno, Czech Republic; e-mail: eduard.baumohl@euba.sk, ORCID: 0000-0002-5444-7348; tomas.vyrost@savba.sk, ORCID: 0000-0002-8384-5724

** Roderik ANTOL, Comenius University Bratislava, Faculty of Mathematics, Physics and Informatics, Mlynská dolina F1, 842 48 Bratislava, Slovakia; e-mail: roderik.antol@gmail.com

*** Tomáš BAČO, Technical University of Košice, Faculty of Economics, B. Němcovej 32, 040 01 Košice, Slovakia; e-mail: tomas.baco@tuke.sk

[1] This work was supported by the Slovak Research and Development Agency (grant No. APVV-22-0126). We are thankful to the Anti-Fraud and Risk Analysis Section of the Financial Directorate of the Slovak Republic for their cooperation. The authors have no competing interests to declare relevant to this article's content. This work is based on a bachelor's thesis by Roderik Antol under the supervision of Eduard Baumöhl. All the detailed results and codes are available from the corresponding author upon request.

## Introduction

Income taxes are a vital source of revenue for countries worldwide, playing a crucial role in supporting public services and infrastructure. Ensuring the integrity of this income stream is paramount for economic stability and maintaining public trust. However, tax manipulation poses a significant challenge, as it undermines the state's ability to support its citizens effectively.

As one would expect, companies can try to declare lower revenues or higher costs to reduce their state tax liabilities (Harris et al., 1993). There are two main strategies for this practice; one is that legal techniques are applied to reduce tax liabilities and maximize after-tax income by exploiting tax loopholes or, generally, conducting strategic planning and structuring of financial activities to exploit tax incentives, deductions, or credits. These legal techniques are usually referred to as earnings management or simply tax avoidance (Beneish, 2001; Ball and Shivakumar, 2008; Huseynov and Klamm, 2012). The other type of strategy is nonlegal, i.e., manipulating financial information or engaging in illegal actions to evade taxes, which we refer to as tax evasion or tax fraud; as (Slemrod, 2007) puts it, „Tax evasion is widespread, always has been, and probably always will be." Even though both types of strategy for nonpayment of taxes have similar negative effects on the national budget, they are perceived differently by the legislation and individuals (Kirchler et al., 2003).

Because such earnings management and tax fraud „explicitly involves potential wrongdoing, mischief, conflict, cloak and dagger, and a sense of mystery" (Lo, 2008), it has been studied extensively. This is perhaps the most provocative topic in accounting and finance. Early models for detecting possible financial statement manipulation were mainly built on the principle of „red flags" (Moyes et al., 2006; Kenyon and Tilton, 2012), which indicate various financial reporting anomalies. In the past, one of the most commonly used detection methods was logistic regression (Persons, 1995; Dechow et al., 2011).

Later, artificial neural networks (ANNs), Naïve Bayes (NB) methods, and decision trees (DTs) were used to detect financial manipulations (see, e.g., Feroz et al., 2000; Lin et al., 2003; Ngai et al., 2011; Ravisankar et al., 2011). They were joined by the support vector machine (SVM) (Perols, 2011; Albashrawi, 2016) and K-nearest neighbour (KNN) and random forest (RF) methods. RF achieves reasonably good detection results among these techniques (Whiting et al., 2012; An and Suh, 2020; Wyrobek, 2020). However, a general consensus is still lacking.[2] For example, Dutta et al. (2017) reported that the ANN outperforms other data mining

[2] For a list of indicative recent studies on the identification of falsified financial statements, see, e.g., Tragouda et al. (2024).

algorithms – such as the DT, NB, SVM, and Bayesian belief network (BBN) classifiers – in terms of its empirical setup according to the accuracy and area under the ROC curve.

To address this lack of consensus, we explored the application of machine learning (ML) and artificial intelligence (AI) algorithms – specifically, ANNs, RFs, XGBoost, and SVMs – to detect and predict income tax fraud in Slovakia.

A common problem with studies in this field is that manipulative firms are poorly categorized during the development of models because they are not explicitly revealed (Dechow et al., 2011). In constructing detection models, researchers usually work under the somewhat naive assumption that companies that the authorities have not identified as manipulators are categorized as nonmanipulators, significantly distorting the results of these models. However, if the model is later applied to the entire sample of companies in a given country, the results will likely be affected by either type I or type II errors.

This paper considers a unique dataset provided by a state authority (the Financial Directorate of the Slovak Republic). The dataset includes outcomes of tax audits, offering unique insights into verified instances of tax manipulation. Hence, our study is not affected by the abovementioned problem of misclassification.

Our analysis was conducted across three different datasets: the full sample, a Winsorized sample, and a Winsorized sample segmented by industry. The XGBoost model demonstrated the highest effectiveness, achieving an F1 score of 0.75 in the full sample. The industry-specific results varied, with XGBoost outperforming the other methods in sector A (F1 score of 0.85), whereas the ANN and SVM yielded significant improvements in the uncategorized cases with an F1 score of 0.91.

## 1. Data and Methodology

The initial dataset represents the entire Register of Financial Statements[3] of companies operating within Slovakia, covering financial activities from January 2014 to January 2024. All Slovak firms must provide their balance sheets and income statements, making this one of the most extensive data collections available for this type of analysis in Slovakia. It spans a wide array of economic sectors represented by SK NACE codes and consists of approximately 770,000 companies, and each entry in the dataset includes more than 574 pieces of firm-specific information. However, our initial data do not include information on whether a company has had a finding of manipulation.

[3] <www.registeruz.sk>.

The Financial Directorate of the Slovak Republic also provided us with information on whether a company committed income tax fraud. This dataset includes slightly over 15,000 tax audits over the given period. Of course, not all findings can be labeled as tax fraud. Even small, unintentional deviations from accounting rules are classified as audit findings. Moreover, some sample selection bias is present. Additionally, since our research is based on sensitive data from the Slovak Financial Directorate, we are bound by a non-disclosure agreement that prevents us from sharing operational details, including the criteria used to select audited companies or any specific information about them.

We calculated 43 variables from the financial statements on the basis of data availability (see Appendix A). We addressed data imbalance issues by considering only firms in which a tax audit had been conducted. Our final dataset thus comprises 2,589 firms with findings (manipulators) and 1,952 without findings (nonmanipulators). We emphasize that our dataset is unbiased in terms of misclassification of nonmanipulators, as tax audits were conducted on all firms in our dataset.

We then employed the ANN, RF, XGBoost (XGB), and SVM to classify manipulating and nonmanipulating firms. More details on the setup are given below.

### 1.1. ANN Description

A neuron in an ANN mimics the basic structure of a biological neuron, receiving input signals or values from other neurons or external sources, such as data. In an ANN, each input value $x_i$ is usually associated with a weight $w_i$, which indicates the strength or importance of the input. The inputs are processed within the neuron via a weighted sum, including a bias term $b$. The information is then passed into an activation function $\phi$ to determine the neuron's output. The bias term is crucial, as it allows the activation function to shift to the left or right, effectively adjusting the threshold level at which the neuron is activated. The formula for this process is as follows:

$$y = \phi\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{1}$$

Common activation functions include the sigmoid, tanh, and rectified linear unit (ReLU) functions. The choice of the activation function is crucial, as it affects the neuron's ability to capture nonlinear relationships. The ReLU function is currently the most successful and widely used activation function (Ramachandran et al., 2017). In our models, we mainly apply the ReLU activation function in our input and hidden layers because of its ability to introduce nonlinearity, which is essential for learning complex patterns in data. The ReLU is defined by the function

$\mathrm{ReLU}(x) = \max(0, x)$, which is a simple piecewise linear function that allows the network to combine linear transformations in complex ways.

Statistically, the ReLU outperforms sigmoidal activation functions in classification problems (Schmidt-Hieber, 2020). However, for our output layer, we use the sigmoid function, which compresses the outputs to between 0 and 1; this makes the output useful for binary classification, as it can be interpreted as the probability of the input belonging to one class or the other. The mathematical formula for the sigmoid function is $\sigma(x) = \frac{1}{1+e^{-x}}$.

The learning process within an ANN is facilitated by backpropagation, a method in which the errors between the predicted outputs and actual outputs are calculated and used to update the network's weights and biases. We apply the binary cross-entropy cost function, calculated as follows:

$$C_{CE}\left(W, B, S^T, E^r\right) = -\sum_j \left[ E_j^r \ln a_j^L + \left(1 - E_j^r\right) \ln\left(1 - a_j^L\right) \right] \tag{2}$$

where $W$ represents our neural network's weights, $B$ represents our neural network's biases, $S^T$ represents the input of a single training sample, $E^r$ represents the desired output of that training sample and $a^L$ represents the activation value of the $j^{th}$ neuron in the output layer (Nielsen, 2015).

Gradient descent is then employed to minimize the loss function with respect to each weight in the network, guiding the way adjustments should be made to reduce error. These updates propagate backward from the output layer to the input layer, refining the model iteratively over multiple epochs until the performance stabilizes or reaches a satisfactory level.

T a b l e 1

**Parameter Configuration for the ANN Model**

| Parameter | Values |
|---|---|
| Number of layers | 1, 2, 3, 4 |
| Number of neurons | 16, 32, 64, 128 |
| Optimizer | 'adam', 'rmsprop' |
| Learning rate | 0.001, 0.01, 0.1 |
| Batch size | 10, 20 |
| Epochs | 100 |

*Source*: Own calculations.

The optimization was carried out via RandomizedSearchCV, and a broad and efficient exploration of the hyperparameter space was achieved over 100 iterations with 5-fold cross-validation. The focus was on improving the model's performance

on the basis of the F1 score metric. Finally, the output layer consisted of a single neuron with a sigmoid activation function tailored for binary classification tasks in fraud detection.

### 1.2. Random Forest Description

Consider a dataset $S$ with $N$ samples and $p$ features, where $F$ represents the full set of features. RFs first construct $B$ bootstrapped datasets $S_1$, $S_2$, …, $S_B$. Each dataset $S_i$ is generated by randomly sampling $N$ instances from $S$ with replacement, which we can describe mathematically as:

$$S_i = \left\{\left(x_{i1}, y_{i1}\right), \left(x_{i2}, y_{i2}\right), \ldots, \left(x_{iN}, y_{iN}\right)\right\} \text{ for } i = 1 \text{ to } B \tag{3}$$

where $\left(x_{ij}, y_{ij}\right)$ are randomly sampled with replacement from $S$.

For each bootstrapped dataset $S_i$, a decision tree $T_i$ is grown. At each node of the tree $T_i$, rather than considering all $p$ features, a subset of $m$ features is randomly selected without replacement from $F$. Once the subset is selected, the best split $F_m$, where $F_m \subset F$ and $\left|F_m\right| = m$, is determined on the basis of the splitting criterion $J$ (such as entropy for classification or mean squared error for regression; see, e.g., Biau, 2012; Biau and Scornet, 2016; Parmar et al., 2019). Mathematically, this is represented as:

$$S = \arg\min_{s \in F_m} J\left(s\right) \tag{4}$$

This process is repeated for each tree node $T_i$ in the forest, and each tree is built independently. The randomness of feature selection ensures that the trees are decorrelated, improving the ensemble's generalization ability and reducing overfitting.

T a b l e 2

**Parameter Configuration for the Random Forest Model**

| Parameter | Values |
|---|---|
| n_estimators | 100, 200, 300, 400 |
| max_features | 'sqrt', 'log2' |
| max_depth | 4, 6, 8, 10, 20, 30, 40 |

*Source*: Own calculations.

Here, we employ the RandomForestClassifier from the scikit-learn library.[4] The choice of this classifier is rooted in its proven ability to handle complex datasets with a mixture of feature types and its robustness against overfitting,

[4] <scikit-learn.org>.

particularly when dealing with high-dimensional data (Pedregosa et al., 2011). In contrast to the original work of Breiman (2001), the scikit-learn implementation combines classifiers by averaging their probabilistic predictions instead of letting each classifier vote for a single class.

The RF algorithm was configured to adapt to our dataset's characteristics dynamically. The flexibility of this model is indicated by its ability to handle many input features and complex data structures efficiently. Each tree in the RF was built by considering both depth and number of estimators to prevent overfitting while maximizing the predictive power. The ensemble approach of the RF, which combines multiple decision trees, inherently increases the model's accuracy and generalizability. GridSearchCV was employed to determine the optimal configuration of the RF model. This method searches through a predefined grid of parameters, which in our case was the number of trees in the forest, the number of features to consider when looking for the best split, and the maximum depth of each tree, as detailed in Table 2.

### 1.3. XGBoost Description

XGBoost, as described in (Chen and Guestrin, 2016), is an advanced implementation of gradient boosting that is widely used in the field of machine learning because of its efficiency, performance, and flexibility. It was designed to be highly scalable and faster than other implementations of gradient boosting (Friedman, 2001).

The core of XGBoost's methodology involves sequentially constructing an ensemble of decision trees, where each tree is built to correct the residuals (errors) left by its predecessors. The algorithm optimizes a regularized objective function that combines a differentiable convex loss function and a regularization term to prevent overfitting and improve model generalizability. Mathematically, the objective function during the training phase at each step $t$ can be described as:

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t\left(x_i\right)\right) + \Omega\left(f_t\right) \tag{5}$$

Here, $y_i$ represents the true value, $\hat{y}_i^{(t-1)}$ is the prediction from the existing ensemble of trees up to iteration $t-1$, $f_t\left(x_i\right)$ is the output of the newly added tree, and $l$ denotes the loss function that measures prediction accuracy, which is typically taken as the squared error for regression and logistic loss for classification tasks.

The regularization term, $\Omega\left(f_t\right)$, is crucial to the XGBoost algorithm. It is formulated as follows:

$$\Omega\left(f\right) = \gamma T + \frac{1}{2}\lambda \left\|w\right\|^2 \tag{6}$$

In this formula, $T$ is the number of leaves in tree $f_t$, $w$ represents the vector of the scores on the leaves, and $\gamma$ and $\lambda$ are parameters that control the complexity of the model. The term $\gamma T$ penalizes the number of leaves, and $\frac{1}{2}\lambda\|w\|^2$ penalizes the magnitude of the leaf weights, thus controlling tree growth and ensuring that the model does not overfit.

XGBoost improves the gradient boosting method by optimizing the loss function via second-order Taylor expansion. This approximation considers not only the gradients, which represent the first derivatives of the loss function with respect to the predictions but also the Hessians, the second derivatives, thereby capturing the curvature of the loss function. The objective function (5) at each iteration $t$ is expanded to a second-order Taylor series around the prediction from the previous iteration $\hat{y}_i^{(t-1)}$, which can be represented as:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n}\left[ g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \tag{7}$$

In this equation, $g_i$ and $h_i$ represent the gradient and Hessian, and $f_t(x_i)$ denotes the output of the tree at the $t$-th iteration. This nuanced approach enables XGBoost to assess and update the model's predictive power accurately and converge more rapidly toward the optimal solution.

When XGBoost constructs a tree, it uses the potential „score" or „gain" of a split to determine the tree structure. This score measures the benefit of making a split on the basis of how much it will improve the model's predictions considering the model's complexity.

The score is computed via the following formula:

$$\text{Score} = \frac{1}{2}\left[ \frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I} g_i\right)^2}{\sum_{i\in I} h_i + \lambda} \right] - \gamma \tag{8}$$

where $I_L$ and $I_R$ are the data instances on the split's left and right sides, respectively. Maximizing this score allows XGBoost to partition the data optimally, thereby efficiently reducing the loss and increasing the prediction accuracy while controlling complexity.

The update rule in XGBoost adjusts the model by adding the contribution of a new tree and scales this contribution by a learning rate $\eta$:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta \cdot f_t(x_i) \tag{9}$$

T a b l e 3

**Parameter Configuration for the XGBoost Model**

| Parameter | Values |
|---|---|
| n_estimators | 100, 200, 300 |
| learning_rate | 0.01, 0.1, 0.2 |
| max_depth | 4, 6, 8 |

*Source*: Own calculations.

The XGBoost algorithm was adapted to align with the specific characteristics of our dataset. Known for its ability to handle many input features and complex data structures efficiently, the model leverages gradient boosting to optimize speed and accuracy. The architecture of the XGBoost model includes parameters that are configured to enhance performance without sacrificing computational efficiency. GridSearchCV was utilized to fine-tune the settings of the XGBoost model. This optimization tool searches through a comprehensive parameter grid, including the number of estimators, learning rate, and maximum depth. The specifics of these parameters are described in detail in Table 3.

### 1.4. SVM Description

The SVM is a supervised machine learning algorithm primarily used for classification tasks (Zhou, 2021). It operates by identifying a hyperplane that optimally separates the training data points into different classes. The primary goal is to select a hyperplane with the maximum margin, defined as the distance between the hyperplane and the nearest data points from each class, which are known as support vectors. These support vectors are crucial because they represent the margin and are the closest points to the decision boundary.

The equation defining this separating hyperplane is:

$$w^T x + b = 0 \tag{10}$$

where $w$ is the weight vector that determines the direction of the hyperplane and $b$ is the bias, which controls the distance of the hyperplane from the origin.

The optimization problem for identifying the most effective hyperplane can be formulated as minimizing:

$$\frac{1}{2}\|w\|^2 \tag{11}$$

subject to the following constraints:

$$y_i\left(w^T x_i + b\right) \geq 1 \tag{12}$$

for each training sample $i$, where $y_i$ are the labels.

However, the training samples may not be linearly separable within the original feature space in many cases. To address this, the SVM can be extended to a kernel SVM, which operates in a higher-dimensional feature space without explicitly mapping the data points. This extension uses the following kernel function:

$$\kappa\left(x_i, x_j\right) = \phi(x_i)^T \phi\left(x_j\right) \tag{13}$$

which computes the inner product of data points in the transformed feature space. Commonly used kernel functions include the Gaussian radial basis function (RBF), polynomial, and sigmoid kernels.

The dual form of the kernel SVM problem is introduced as a maximization problem, shifting from the minimization used in the primal form. This shift is necessary because maximizing the Lagrangian in the dual form is equivalent to minimizing the primal objective function under the condition that the constraints are incorporated through Lagrange multipliers. These multipliers ( $\alpha_i$ ) play a crucial role by allowing the constraints of the primal problem to be embedded in the optimization of the dual problem. The dual problem involves maximizing:

$$\sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \kappa\left(x_i, x_j\right) \tag{14}$$

subject to the conditions:

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \tag{15}$$

and

$$\alpha_i \geq 0 \tag{16}$$

for all $i$ . The solution to this problem yields the decision function:

$$f\left(x\right) = \sum_{i=1}^{m} \alpha_i y_i \kappa\left(x_i, x\right) + b \tag{17}$$

which classifies new samples. This approach enables the SVM to find a separating hyperplane in the transformed feature space, effectively solving nonlinear classification problems by applying linear methods in this higher-dimensional setting. This ability makes the kernel SVM a powerful tool for addressing complex classification tasks.

The SVM was specifically tailored to exploit its kernel functions, enabling the transformation of data into a higher dimension where a hyperplane can separate

classes. This ability is vital for handling the intricate feature interactions involved in fraud detection tasks. To optimize the SVM parameters, GridSearchCV was employed to methodically explore various settings to identify the most effective configurations for our specific needs. Key parameters such as the regularization strength, kernel type, and kernel coefficient play critical roles in the model performance and are thoroughly described in Table 4.

T a b l e 4

**Parameter Configuration for the SVM Model**

| Parameter | Values |
|---|---|
| C | 1, 10, 100 |
| kernel | 'linear', 'rbf' |
| gamma | 0.001, 0.0001 |
| probability | True |

*Source*: Own calculations.

### 1.5. Model Evaluation Metrics

We employ several metrics to assess the models, primarily focusing on the F1 score and supplementary evaluations via the confusion matrix and ROC curve. These metrics are critical for understanding how well the models perform, especially in tasks such as fraud detection, where the misclassification costs are high.

*The F1 score* is a critical metric in the evaluation of binary classification systems; it combines precision (the accuracy of positive predictions) and recall (the ability to find all the relevant cases within a dataset) into a single measure. It is calculated as the harmonic mean of precision and recall. The F1 score is particularly valuable in situations where there is class imbalance, such as in fraud detection, where there are typically many fewer fraudulent transactions than nonfraudulent ones. A high F1 score indicates that the model achieves a robust balance between precision and recall, making it ideal for scenarios in which both false positives (incorrectly flagged transactions) and false negatives (missed fraudulent transactions) have significant consequences.

*The confusion matrix* is a straightforward visualization tool for assessing a classifier's performance. It indicates the numbers of correct and incorrect predictions, broken down by class. This matrix is especially useful for determining how many instances are correctly identified as fraud (true positives), incorrectly identified as fraud (false positives), missed fraud instances (false negatives), and correctly identified nonfraud instances (true negatives). The insights from the confusion matrix feed directly into the computation of precision and recall and, subsequently, the F1 score, providing a clear picture of model performance across different classes.

*A homogeneity test* for confusion matrices can be used to perform a pairwise comparison between confusion matrices and assess the precision of classification. When evaluating the performance of the classification, two standard measures include the overall accuracy and the Kappa statistic (Cohen, 1960). However, the kappa statistic has been designed as a coefficient of agreement, not accuracy, and should not be considered for our case (see, Nishii and Tanaka, 2002). Instead, we perform the homogeneity test for confusion matrices designed by García-Balboa et al. (2018), which uses a statistic based on the discrete Hellinger distance, with the null distribution approximated by bootstrapping. The null hypothesis of the test corresponds to homogeneity of two confusion matrices, i.e., a similar classification accuracy.

*The receiver operating characteristic (ROC) curve* is another essential tool for evaluating the predictive performance of our models. The ROC curve plots the true positive rate (sensitivity) against the false positive rate (1 – specificity) at various threshold settings. The area under the ROC curve (AUC) provides a single measure of overall model performance across all classification thresholds. A model with perfect discrimination (no overlap between positive and negative distributions) has an AUC of 1.0, whereas a model with no better accuracy than random guessing has an AUC of 0.5. The ROC curve and AUC are particularly useful for comparing models and selecting optimal models on the basis of their ability to discriminate between classes under various threshold settings.

## 2. Results

This section evaluates the performance of machine learning models used to detect fraudulent activities across various sample modifications. Initially, the models were applied to the full dataset to establish a baseline, allowing an assessment of the impacts of outlier management and sector-specific variations. The analysis was then extended to a Winsorized sample to mitigate the influence of extreme values that could skew the model's accuracy and interpretability. Finally, the models were tailored to specific industries within the Winsorized framework to identify industry-specific patterns and anomalies that might affect fraud detection.

T a b l e 5

**Full-Sample Model Performance**

| Model | Training F1-Score | Test F1-Score |
|---|---|---|
| Neural Network | 0.7058 | 0.7252 |
| Random Forest | 0.7340 | 0.7332 |
| XGBoost | 0.7305 | 0.7528 |
| Support Vector Machine | 0.6181 | 0.7363 |

*Source*: Own calculations.

T a b l e 6

**Full-Sample Homogeneity Test for Confusion Matrices (p-values)**

| Model | Random Forest | XGBoost | Support Vector Machine |
|---|---|---|---|
| Neural Network | 0.978 | 0.001 | 0.000 |
| Random Forest | | 0.001 | 0.000 |
| XGBoost | | | 0.026 |

*Source*: Own calculations.

### 2.1. Full Sample

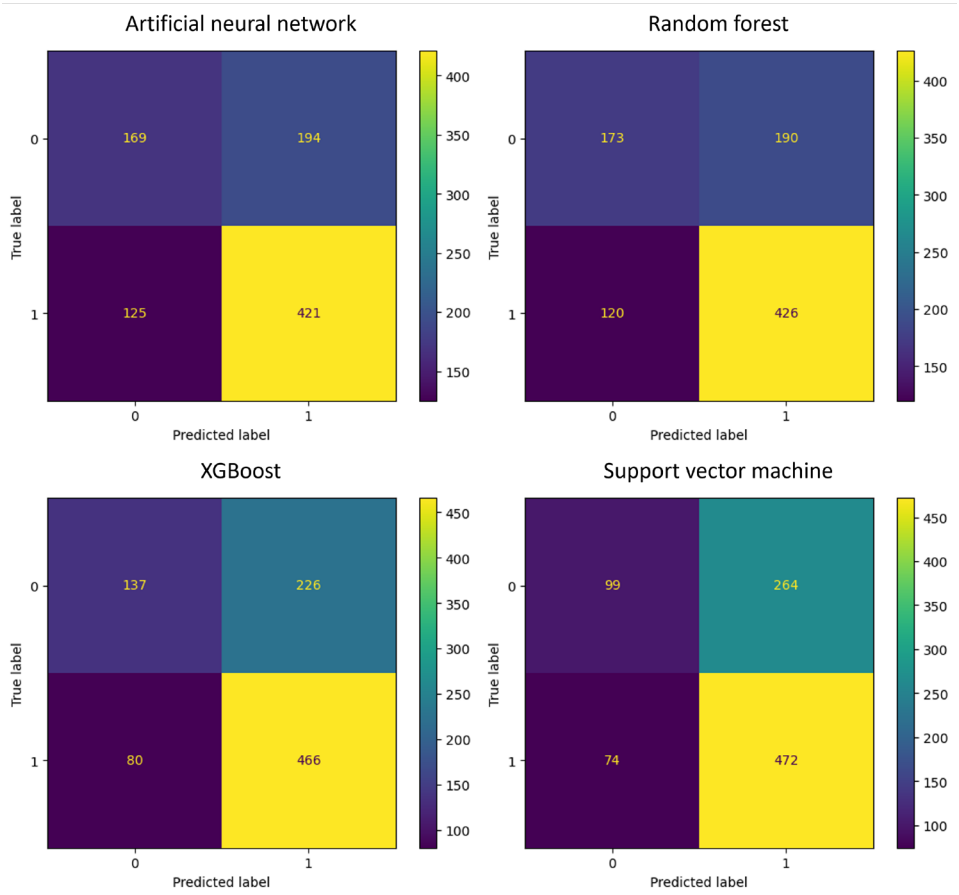
First, we aim to determine which models demonstrate the most reliable and robust predictive capabilities to obtain best practices for deploying these technologies in fraud detection scenarios.

In the training phase, as shown in Table 5, the ensemble tree models have the highest F1-scores, approximately 0.73, followed by the ANN, with 0.7058, and SVM, with a significantly lower score of 0.6181. During the test phase, the results indicate good generalizability for the RF and XGBoost methods, with F1 scores of 0.7332 and 0.7528, respectively. Notably, the ANN and SVM show improved performance in the test phase, with the SVM achieving a remarkable F1 score of 0.7363.

According to the confusion matrices in Figure 1, while all the models effectively identify fraudulent cases, they tend to misclassify nonfraudulent cases as fraudulent. This trend is particularly pronounced for the SVM, which, despite achieving a high F1 score, also records significant numbers of true and false positives. The RF algorithm adopts a more conservative approach, yielding the highest number of true negatives; this suggests that it uses a cautious classification strategy. In contrast, XGBoost maintains a good balance, which contributes to its high F1 score and underscores its effectiveness, despite the high rate of false positives. Following the results in Table 6, we may conclude that while the results of the ANN are very similar to RF, in all other cases, the homogeneity of the confusion matrices may be rejected. Thus, the classification models generally significantly differ in their accuracy.

The ROC curves, displayed in Figure 2, reveal that the ensemble models, RF and XGBoost, both achieve an AUC of 0.70, reflecting their competence in discriminating fraudulent and nonfraudulent cases. However, there is still potential for improvement. The lower AUC scores of the ANN and SVM, indicating lower classification accuracy, suggest that while the ANN tends to misclassify cases in both classes, the SVM is biased toward classifying cases as fraudulent.

F i g u r e 1
**Full Sample Confusion Matrices**

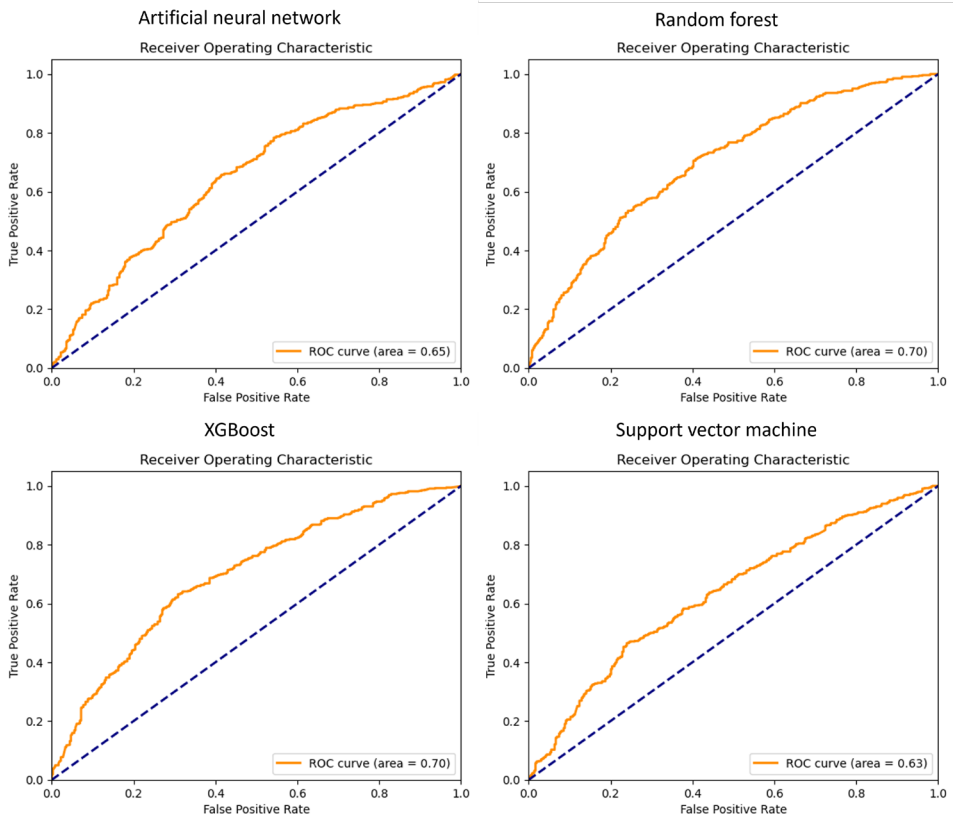



*Source*: Own calculations.

In conclusion, the models demonstrated moderate results, which could be attributed to the complexity of the dataset, the mixture of industries represented, and the presence of significant outliers. We also experimented with the logarithmic transformation of the data.

Although this transformation was expected to significantly benefit linear models such as artificial neural networks and support vector machines, it had limited impact and, in some cases, even worsened the results. Notably, it surprisingly improved the performance of the ensemble models (detailed results are available upon request).

F i g u r e 2
**Full Sample ROC Curves**




*Source*: Own calculations.

## 2.2. Winsorized Sample

This section discusses the performance of the four machine learning models applied to the Winsorized sample to identify manipulators on the basis of financial indicators. We keep the values that do not exceed the 5th and 95th percentiles, which is also referred to as 10% Winsorization or Winsorizing the top and bottom 5% (Sullivan et al., 2021). The models' effectiveness was evaluated via the F1 score. The training and test results are summarized in Table 7.

During the training phase, the RF model achieved the highest F1 score of 0.7423, closely followed by the XGBoost model, with a score of 0.7364. The ANN and SVM models had comparable performances, approximately 0.67. In the testing phase, there were slight variations in the scores; RF and XGBoost improved the F1 scores to 0.7428 and 0.7423, respectively, demonstrating the robust performance of the ensemble trees in both the training and testing phases. Notably, the

SVM showed significant improvement, achieving an F1 score of 0.7427, closely matching the scores of the tree-based models. However, the ANN had an F1 score of 0.6698, indicating some challenges in model generalization.

T a b l e 7

**Model Performance on the Winsorized Sample**

| Model | Training F1-Score | Test F1-Score |
|---|---|---|
| Neural Network | 0.6747 | 0.6698 |
| Random Forest | 0.7423 | 0.7428 |
| XGBoost | 0.7364 | 0.7423 |
| Support Vector Machine | 0.6702 | 0.7427 |

*Source*: Own calculations.

From the configurations presented in Table 12 (Appendix), the RF utilized a larger number of trees but less depth than XGBoost, which opted for fewer trees. This difference in configurations suggests varying approaches to managing the dataset's complexity. Given this complexity, the ANN employed a surprisingly simple architecture, with just one hidden layer and 16 neurons. The SVM utilized the RBF kernel function and a high regularization parameter (C), indicating its preference for a more flexible decision boundary over a simple linear approach.

T a b l e 8

**Winsorized Sample Homogeneity Test for Confusion Matrices (p-values)**

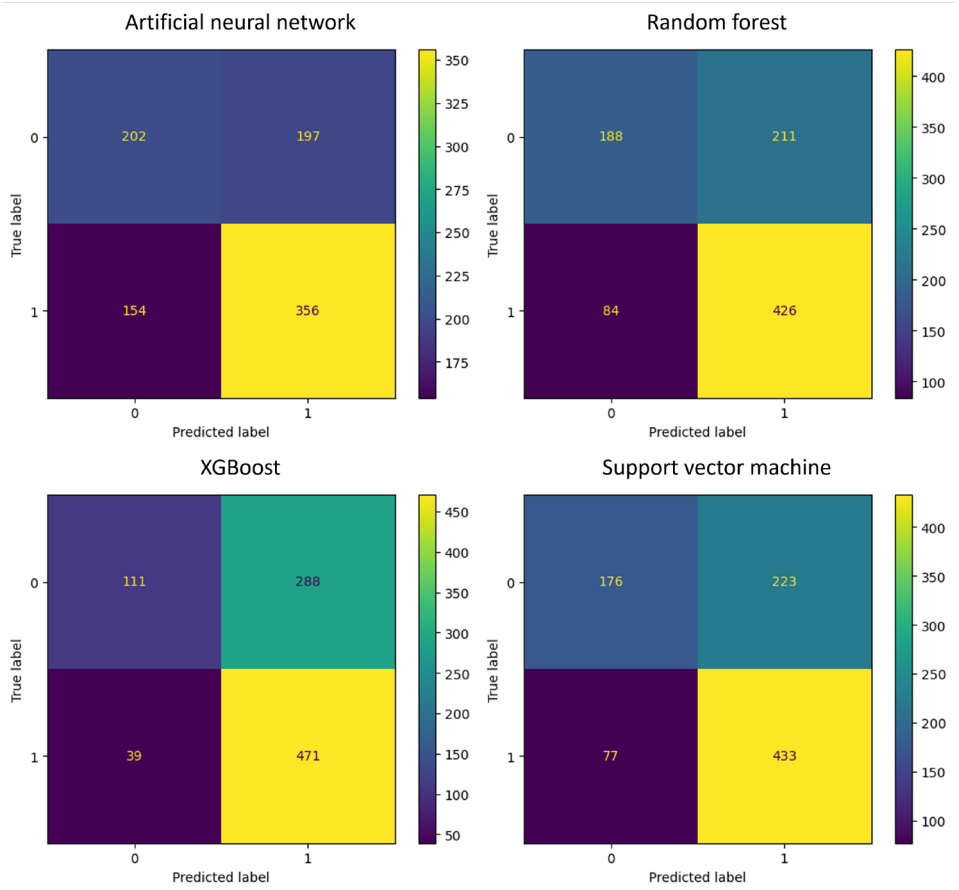
| Model | Random Forest | XGBoost | Support Vector Machine |
|---|---|---|---|
| Neural Network | 0.000 | 0.000 | 0.000 |
| Random Forest | | 0.001 | 0.780 |
| XGBoost | | | 0.000 |

*Source*: Own calculations.

Further analysis of the confusion matrices in Figure 3 reveals that despite the RF model's superior F1 score, XGBoost and the SVM surpassed it in terms of the number of true positives, achieving values of 471 and 433, respectively, in contrast to the RF model's value of 426. Although the ANN obtained the lowest number of true positives (356), it had the highest number of true negatives (202), surpassing RF's 188.

When formally comparing the homogeneity of the confusion matrices in Table 8, we can see that with the exception of RF and SVM, which have similar performance, there are significant differences in model accuracy. This outcome highlights a significant challenge across the models – they largely fail to accurately classify nonfraudulent subjects and often mislabel them as manipulators, including the top-performing RF.

F i g u r e 3
**Winsorized Sample Confusion Matrices**

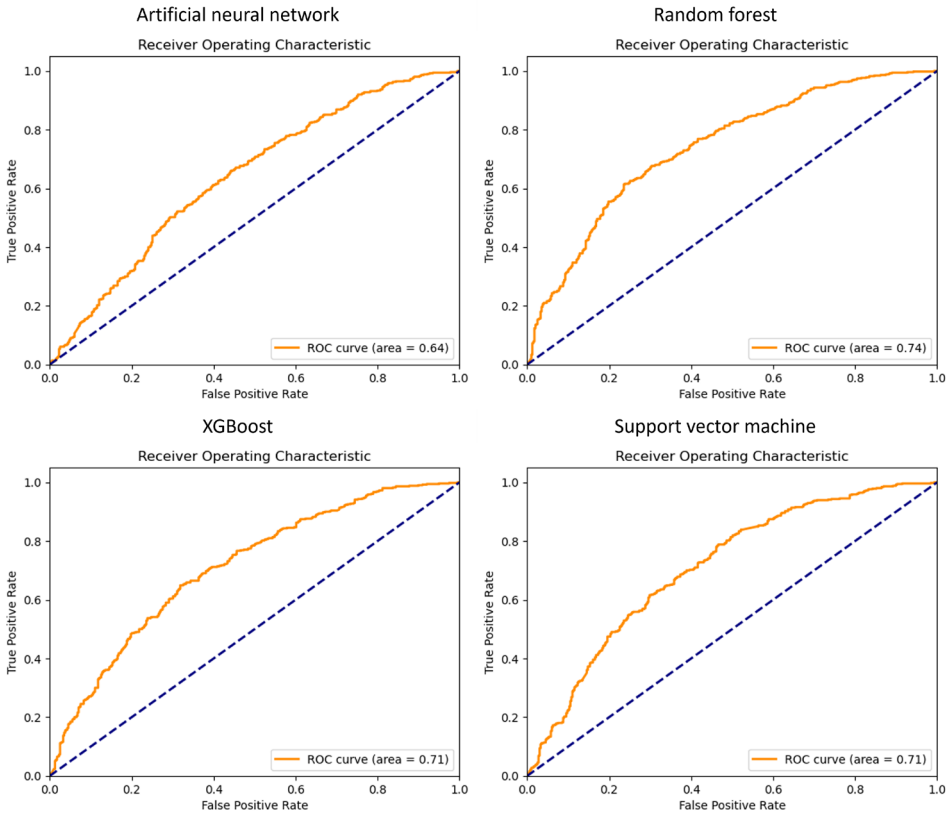



*Source*: Own calculations.

Examining the ROC curves in Figure 4, the ANN exhibited the lowest AUC at 0.64, suggesting that its performance was only marginally better than that of random guessing. XGBoost and the SVM achieved an AUC of 0.71, and RF yielded a slightly greater AUC of 0.74. These results indicate a reasonably good ability among the models to identify manipulators, although they are far from perfect. This analysis indicates that while the RF model leads in overall accuracy, challenges remain in distinguishing between fraudulent and nonfraudulent subjects across all models. The varying performances and parameter selections highlight the complexity of financial fraud detection even after the number of outliers in the data is decreased via Winsorization. Furthermore, it underscores the need for further refinement and possibly the integration of additional data sources or features

to improve detection accuracy. The results also indicate the importance of choosing appropriate model architectures and parameters to balance sensitivity and specificity in real-world applications.

F i g u r e 4
**Winsorized Sample ROC Curves**




*Source*: Own calculations.

## 2.3. Breakdown by Industry

In the previous sections, we employed machine learning models on a Winsorized dataset with a rich variety of financial indicators to detect fraud. This section expands on the previous analysis by segmenting the dataset according to industry sector to deepen our understanding of the nuances of fraud detection. By analyzing each sector individually, we aim to identify industry-specific patterns and variations in model performance that may be obscured in a broader analysis. The detailed F1 score results are given in Table 9.

The sectors under review include agriculture, forestry, and fishing (Sector A); mining and manufacturing (Sectors B – E); construction (Sector F); and services (Sectors G – S). Companies not fitting these categories are labeled Uncategorized. Each sector presents unique challenges and characteristics that could impact the effectiveness of our predictive models.[5]

Sector A yields notable results, with the XGBoost model achieving the highest test F1 score of 0.8485. This superior performance can be attributed to XGBoost's ability to handle sparse data and its robust implementation of gradient boosting algorithms, which increases its ability to distinguish between fraudulent and non-fraudulent activities. Additionally, implementing GridSearchCV optimizes the model parameters, maximizing efficiency while balancing the risks of overfitting and underfitting. The RF and SVM models also perform well in Sector A, with test F1 scores of 0.7742 and 0.7333, respectively, although they do not reach the effectiveness of XGBoost. Conversely, the ANN achieved a score of only 0.6207, potentially due to its limitations in capturing the complexity of the data. Importantly, the RF algorithm was superior during the training phase, with an F1 score of 0.8176, compared with XGBoost and the SVM, which scored 0.75. This discrepancy highlights the importance of model performance evaluation across different stages to ensure robust fraud detection.

T a b l e 9

**Model Performance in Different Sectors**

| Sector | Model | Training F1-Score | Test F1-Score |
|---|---|---|---|
| A | Neural Network | 0.7854 | 0.6207 |
| A | Random Forest | 0.8176 | 0.7742 |
| A | XGBoost | 0.7539 | 0.8485 |
| A | SVM | 0.7546 | 0.7333 |
| B – E | Neural Network | 0.7522 | 0.6789 |
| B – E | Random Forest | 0.7899 | 0.7119 |
| B – E | XGBoost | 0.7741 | 0.7107 |
| B – E | SVM | 0.7734 | 0.7050 |
| F | Neural Network | 0.7243 | 0.6250 |
| F | Random Forest | 0.7749 | 0.6912 |
| F | XGBoost | 0.7494 | 0.7092 |
| F | SVM | 0.7474 | 0.6622 |
| G – S | Neural Network | 0.6671 | 0.6453 |
| G – S | Random Forest | 0.7380 | 0.7271 |
| G – S | XGBoost | 0.7331 | 0.7353 |
| G – S | SVM | 0.7394 | 0.7300 |
| Uncategorized | Neural Network | 0.6574 | 0.9091 |
| Uncategorized | Random Forest | 0.5686 | 0.6667 |
| Uncategorized | XGBoost | 0.6724 | 0.7273 |
| Uncategorized | SVM | 0.5732 | 0.9091 |

*Source*: Own calculations.

[5] To preserve space, we do not provide confusion matrices or ROC curve figures, as in the previous subsections. However, all these detailed results are available upon request.

The confusion matrices indicate similar abilities across all the models in classifying nonmanipulators in Sector A, with the main differences seen in identifying manipulators. XGBoost identifies nearly all the manipulators, missing only one. The RF and SVM methods yield almost identical matrices, differing on just one subject. Despite its lower performance, the ANN correctly identifies many true negatives; it struggles with false positives, which impacts its overall effectiveness. The ROC curves further confirm the superiority of the XGBoost model on Sector A. The ANN has a lower AUC (0.57), suggesting weaker performance and a tendency toward randomness. The SVM and RF have commendable AUCs (0.69 and 0.79, respectively), confirming their effectiveness in classification tasks, although this performance is not as high as that of XGBoost (0.84). In conclusion, the RF and XGBoost models demonstrate superior performance in Sector A for fraud detection. However, XGBoost slightly edges them out due to its higher test F1 score and potentially better handling of model tuning and complexity, which highlights the advantage of ensemble trees in this context.

Sectors B – E demonstrate robust results, with the RF model achieving the highest training and test F1 scores of 0.7899 and 0.7119, respectively. XGBoost and the SVM are comparable in terms of training performance, with F1 scores of approximately 0.77; however, XGBoost slightly outperforms the SVM in testing, with a small improvement of 0.0057. The neural network, however, lagged with a training score of 0.7522 and a test score of only 0.6789, indicating that it was the least effective model in this evaluation. By examining the confusion matrices, it is found that all the models can identify fraudulent cases. However, the SVM model incorrectly labels all instances as fraudulent, which results in many false positives despite ensuring that there are no false negatives. If implemented, this approach would drastically impact its utility and cost-effectiveness; this shows why the F1 score is crucial, as it penalizes the lack of precision in such classifications. The ROC curves highlight the models' ability to discriminate between classes under various thresholds. The RF modelled with an AUC of 0.70, with XGBoost close behind (0.66). Interestingly, the ANN has a better AUC than the SVM (0.61 vs 0.59), likely because the SVM tends to classify all instances as manipulative, which affects its overall discriminative performance. In summary, whereas the SVM model's strategy of classifying all observations as fraudulent proves less effective, ensemble tree models such as RF and XGBoost demonstrate superior performance. This trend mirrors the findings for Sector A, reinforcing the dominance of ensemble methods in effectively addressing fraud detection across various sectors.

The complexity of Sector F is indicated by its model performance scores, which are generally lower than those for previous sectors. The XGBoost model

emerges as the top performer, achieving the highest test F1 score of 0.7092, although this score is slightly lower than for other sectors. The RF model closely follows, with a robust test F1 score of 0.6912. In contrast, the ANN and SVM models yield lower test F1 scores of 0.6250 and 0.6622, respectively, highlighting the difficulties these models have with the intricate data patterns in Sector F. The confusion matrices show that while all models are relatively effective in identifying fraudulent cases, their ability to correctly classify nonfraudulent cases varies, reflecting the sector's complexity. The ROC curves for the RF and XGBoost models exhibited an AUC of 0.67, indicating decent discriminative power. However, there is notable potential for improvement, particularly in terms of reducing the false positive rate to increase the overall predictive accuracy. The SVM and ANN models have lower AUC values (0.59 and 0.54, respectively), indicating less effectiveness in this sector than the ensemble methods have. Overall, the results for Sector F, construction, underscore the industry's complexity and the efficacy of ensemble methods under varying conditions. Regarding XGBoost, although it leads in terms of the F1 score and RF, its results indicate that significant improvements could be achieved through further parameter tuning or the exploration of alternative modeling techniques.

For Sectors G – S, the models generally maintain similar training and test F1 scores, indicative of good generalizability to new, unseen data. This suggests that the models have successfully learned patterns that are representative of the broader data distribution rather than merely fitting the training examples. With respect to the F1-scores, the RF, XGBoost, and SVM methods yield comparable results at approximately 0.73, with XGBoost slightly outperforming the other methods in the test phase and the SVM method performing marginally better in the training phase. As observed for previous sectors, the ANN continues to face difficulties, reflecting these sectors' complexity; it has a lower test F1 score of 0.6453. The confusion matrices revealed that while all the models are relatively effective at identifying fraudulent cases, they face greater challenges in accurately classifying nonfraudulent cases, as already seen for previous industries.

Interestingly, despite its lower F1 score, the ANN has more true negatives than the other models do, indicating its cautious approach in labeling cases as fraudulent. This conservatism, however, leads to a higher incidence of false positives. The ROC curves highlight that all the models exhibit moderate discriminative ability, with AUCs of approximately 0.7. The ANN scores are slightly lower, with an AUC of 0.63. While these values suggest that the models perform better than random guessing, they also indicate that the reliability in distinguishing between positive and negative classes could be improved. To summarize, Sectors G – S illustrate the nuanced challenges of applying machine learning models in the service sector.

Although they achieve moderate success, there remains significant room for improvement in model accuracy, particularly in reducing false positives and increasing the ability to differentiate between classes.

For the uncategorized subjects, there are intriguing results, as illustrated in Table 9. The disparity between the training and test F1 scores across all the models is notable. During training, the models exhibit a moderate ability to identify fraudulent activities. However, there is a notable improvement in the testing phase, particularly for the ANN and SVM models, which achieve an impressive F1 score of 0.9091. They outperform the ensemble tree models – RF and XGBoost – which have F1 scores of 0.6667 and 0.7273, respectively. However, the significantly smaller dataset size could explain these unexpected outcomes in the training and testing phases, and these results should therefore be viewed cautiously.

To increase the accuracy and efficiency of our models, particularly the ANN and SVM, which are sensitive to the data distribution, we applied a logarithmic transformation to the dataset. However, adjusting the scale of the dataset's features to a more consistent range led to only a marginal improvement for the SVM and unexpectedly benefited the ensemble models, but had only a limited impact. Hence, our results do not indicate the need for data rescaling as a practice to improve model performance in fraud detection scenarios.

## Concluding Remarks

Our findings underscore the complexity of deploying ML models in real-world environments and the necessity for ongoing customization and refinement. This study contributes to the continuing dialog about the realistic application of ML in tax fraud detection, offering insights into the potential and limitations of these technologies in the Slovak business context.

Our models across three datasets – the full sample, a Winsorized sample, and a Winsorized sample segmented by industry – yield mixed results in terms of fraud detection. The overall best-performing model was XGBoost, which achieved an F1 score of 0.75 in the full sample. Winsorization did not significantly improve model performance as anticipated, with most models showing results similar to those for the full sample; only the ANN lagged, with an F1 score of 0.67.

In industry-specific analyses, XGBoost performed excellently in Sector A, with an F1 score of 0.85, whereas the ANN and SVM both dramatically improved on the uncategorized cases, with an F1 score of 0.91. Despite these promising results for the ANN and SVM, the inconsistency in model performance across different conditions suggests limitations in their real-world applicability.

In contrast to the literature, as discussed in Section 1, our results indicated only moderate success in applying these ML models for fraud detection, highlighting potential gaps and challenges in adapting these models to complex datasets such as ours. Notably, the ensemble tree models (RF and XGBoost) performed most consistently across all datasets, confirming their robustness in certain contexts. However, the ANN displayed significant weaknesses, often delivering moderate results that were unexpected on the basis of prior studies. The difference in performance across models and datasets indicates that further customization and refinement are necessary to optimize these technologies for specific applications.

As for possible extension of our results, several suggestions for future research may be considered to further enhance the detection of tax fraud using machine learning techniques. From a methodological standpoint, recent advancements in deep learning make it possible to explore models such as TabNet or Neural Oblivious Decision Ensembles (NODE). These approaches have demonstrated strong predictive performance in other domains and may offer improvements over traditional tree-based models like XGBoost, which might particularly be useful in capturing complex, non-linear interactions among financial indicators.

In addition to using more advanced models, future research could benefit from incorporating semi-supervised and unsupervised learning techniques. Given the natural imbalance in the dataset-where confirmed tax fraud cases are relatively scarce, methods such as autoencoders or isolation forests could be employed for anomaly detection. These approaches allow models to learn the structure of normal behavior and identify deviations as potentially fraudulent, even in the absence of extensive labeled data. Similarly, semi-supervised learning strategies that leverage both labeled and unlabeled data could help address the limitations posed by limited verified fraud cases, improving generalization in practical applications.

Further extensions might involve the application of explainable AI (XAI) methods to ensure interpretability of complex models, a critical factor when deploying decision systems in the sensitive context of tax auditing. Techniques such as SHAP values or layer-wise relevance propagation can provide insight into the drivers of model predictions.

Finally, our results indicate that relying solely on financial variables may not be sufficient for effective income tax fraud detection. This suggests another crucial area for future research: identifying and integrating more complex indicators that can more effectively distinguish fraudulent manipulators. Future studies could explore the inclusion of non-financial variables, such as governance and behavioral or demographic indicators, to increase the predictive power and accuracy of fraud detection models.

## References

ALBASHRAWI, M. (2016): Detecting financial Fraud Using Data Mining Techniques: A Decade Review from 2004 to 2015. Journal of Data Science, *14*, No. 7, pp. 553 – 569. DOI: 10.6339/JDS.201607_14(3).0010.

AN, B. – SUH, Y. (2020): Identifying Financial Statement Fraud with Decision Rules Obtained from Modified Random Forest. Data Technologies and Applications, *54*, No. 2, pp. 235 – 255. DOI: 10.1108/DTA-11-2019-0208.

BALL, R. – SHIVAKUMAR, L. (2008): Earnings Quality at Initial Public Offerings. Journal of Accounting and Economics, *45*, No. 2 – 3, pp. 324 – 349. DOI: 10.1016/j.jacceco.2007.12.001.

BENEISH, M. D. (2001): Earnings Management: A Perspective. Managerial Finance, *27*, No. 12, pp. 3 – 17. DOI: 10.1108/03074350110767411.

BIAU, G. (2012): Analysis of a Random Forests Model. The Journal of Machine Learning Research, *13*, pp. 1063 – 1095.

BIAU, G. – SCORNET, E. (2016): A Random Forest Guided Tour. Test, *25*, No. 2, pp. 197 – 227. DOI: 10.1007/s11749-016-0481-7.

BREIMAN, L. (2001): Random Forests. Machine Learning, *45*, No. 1, pp. 5 – 32. DOI: 10.1023/A:1010950718922.

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.

COHEN, J. (1960): A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement, *20*, No. 1, pp. 37 – 46. DOI: 10.1177/001316446002000104.

DECHOW, P. M. – GE, W. – LARSON, C. R. – SLOAN, R. G. (2011): Predicting Material Accounting Misstatements. Contemporary Accounting Research, *28*, No. 1, pp. 17 – 82. DOI: 10.1111/j.1911-3846.2010.01041.x.

DUTTA, I. – DUTTA, S. – RAAHEMI, B. (2017): Detecting Financial Restatements Using Data Mining Techniques. Expert Systems with Applications, *90*, No. 4, pp. 374 – 393. DOI: 10.1016/j.eswa.2017.08.030.

FEROZ, E. H. – KWON, T. M. – PASTENA, V. S. – PARK, K. (2000): The Efficacy of Red Flags in Predicting the Sec's Targets: An Artificial Neural Networks Approach. Intelligent Systems in Accounting, Finance & Management, *9,* No. 3, pp. 145 – 157. DOI: 10.1002/1099-1174(200009)9:3<145::AID-ISAF185>3.0.CO;2-G.

FRIEDMAN, J. H. (2001): Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics, pp. 1189 – 1232. DOI: 10.1214/aos/1013203451.

GARCÍA-BALBOA, J. L. – ALBA-FERNÁNDEZ, M. V. – ARIZA-LÓPEZ, F. J. – RODRIGUEZ-AVI, J. (2018): Homogeneity Test for Confusion Matrices: A Method and an Example. In: IGARSS 2018 – 2018 IEEE International Geoscience and Remote Sensing Symposium. IEEE, pp. 1203 – 1205.

HARRIS, D. – MORCK, R. – SLEMROD, J. B. (1993): Income Shifting in Us Multinational Corporations. In: Studies in International Taxation. University of Chicago Press, pp. 277 – 308.

HUSEYNOV, F. – KLAMM, B.K. (2012): Tax Avoidance, Tax Management and Corporate Social Responsibility. Journal of Corporate Finance, *18*, No. 4, pp. 804 – 827. DOI: 10.1016/j.jcorpfin.2012.06.005.

KENYON, W. – TILTON, P. D. (2012): Potential Red Flags and Fraud Detection Techniques. A Guide to Forensic Accounting Investigation, pp. 231 – 269. DOI: 10.1002/9781119200048.ch13.

KIRCHLER, E. – MACIEJOVSKY, B. – SCHNEIDER, F. (2003): Everyday Representations of Tax Avoidance, Tax Evasion, and Tax Flight: Do Legal Differences Matter? Journal of Economic Psychology, *24*, No. 4, pp. 535 – 553. DOI: 10.1016/S0167-4870(02)00164-2.

LIN, J. W. – HWANG, M. I. – BECKER, J. D. (2003): A Fuzzy Neural Network for Assessing the Risk of Fraudulent Financial Reporting. Managerial Auditing Journal, *18*, No. 8, pp. 657 – 665. DOI: 10.1108/02686900310495151.

LO, K. (2008): Earnings Management and Earnings Quality. Journal of Accounting and Economics, *45*, No. 2, pp. 350 – 357. DOI: 10.1016/j.jacceco.2007.08.002.

MOYES, G. D. – LIN, P. – LANDRY JR, R. M. – VICDAN, H. (2006): Internal Auditors' Perceptions of the Effectiveness of Red Flags to Detect Fraudulent Financial Reporting. Journal of Accounting, Ethics & Public Policy, JAEPP, *6*, pp. 75 – 75.

NGAI, E. W. – HU, Y. – WONG, Y. H. – CHEN, Y. – SUN, X. (2011): The Application of Data Mining Techniques in Financial Fraud Detection: A Classification Framework and an Academic Review of Literature. Decision Support Systems, *50*, No. 3, pp. 559 – 569. DOI: 10.1016/j.dss.2010.08.006.

NIELSEN, M. A. (2015): Neural Networks and Deep Learning. Volume 25. San Francisco, CA, USA: Determination Press.

NISHII, R. – TANAKA, S. (2002): Accuracy and Inaccuracy Assessments in Land-Cover Classification. IEEE Transactions on Geoscience and Remote Sensing, *37*, pp. 491 – 498.

PARMAR, A. – KATARIYA, R. – PATEL, V. (2019): A Review on Random Forest: An Ensemble Classifier, In: International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018. Springer, pp. 758 – 763.

PEDREGOSA, F. et al. (2011): Scikit-Learn: Machine Learning in Python. Journal of Machine Learning Research, *12*, pp. 2825 – 2830.

PEROLS, J. (2011): Financial Statement Fraud Detection: An Analysis of Statistical and Machine Learning Algorithms. Auditing: A Journal of Practice & Theory, *30*, No. 2, pp. 19 – 50. DOI: 10.2308/ajpt-50009.

PERSONS, O. S. (1995): Using Financial Statement Data to Identify Factors Associated with Fraudulent Financial Reporting. Journal of Applied Business Research, *11*, No. 3, pp. 38 – 46. DOI: 10.19030/jabr.v11i3.5858.

RAMACHANDRAN, P. – ZOPH, B. – LE, Q. V. (2017): Searching for Activation Functions. arXiv preprint arXiv:1710.05941.

RAVISANKAR, P. – RAVI, V. – RAO, G. R. – BOSE, I. (2011): Detection of Financial Statement Fraud and Feature Selection Using Data Mining Techniques. Decision Support Systems, *50*, No. 2, pp. 491 – 500. DOI: 10.1016/j.dss.2010.11.006.

SCHMIDT-HIEBER, J. (2020): Nonparametric Regression Using Deep Neural Networks with ReLU Activation Function. The Annals of Statistics, *48*, No. 4, pp. 1875 – 1897. DOI: 10.1214/19-AOS1875.

SLEMROD, J. (2007): Cheating Ourselves: The Economics of Tax Evasion. Journal of Economic Perspectives, *21*, No. 1, pp. 25 – 48. DOI: 10.1257/jep.21.1.25.

SULLIVAN, J. H. – WARKENTIN, M. – WALLACE, L. (2021): So Many Ways for Assessing Outliers: What Really Works and Does It Matter? Journal of Business Research, *132*, pp. 530 – 543.

TRAGOUDA, M. – DOUMPOS, M. – ZOPOUNIDIS, C. (2024): Identification of Fraudulent Financial Statements Through a Multi-Label Classification Approach. Intelligent Systems in Accounting, Finance and Management, *31*, No. 2, e1564. DOI: 10.1002/isaf.1564.

WHITING, D. G. – HANSEN, J. V. – MCDONALD, J. B. – ALBRECHT, C. – ALBRECHT, W. S. (2012): Machine Learning Methods for Detecting Patterns of Management Fraud. Computational Intelligence, *28*, No. 4, pp. 505 – 527. DOI: 10.1111/j.1467-8640.2012.00425.x.

WYROBEK, J. (2020): Application of Machine Learning Models and Artificial Intelligence to Analyze Annual Financial Statements to Identify Companies with Unfair Corporate Culture. Procedia Computer Science, *176*, No. 4, pp. 3037 – 3046. DOI: 10.1016/j.procs.2020.09.335.

ZHOU, Z.-H. (2021): Machine Learning. Singapore: Springer. ISBN 978-981-15-1966-6; ISBN 978-981-15-1967-3 (eBook). DOI: 10.1007/978-981-15-1967-3.

## A p p e n d i x A: Data Description

T a b l e 10

**List of Utilized Financial Variables**

| Variable | Indicator | Formula |
|---|---|---|
| AR | Accounts Receivable | $\text{Accounts Receivable}_{i,t}$ |
| GM | Gross Margin | $\text{Net Sales}_{i,t} - \text{Direct Costs}_{i,t}$ |
| TA | Total Assets | $\text{Total Assets}_{i,t}$ |
| TD | Total Debt | $\text{Total Debt}_{i,t}$ |
| CD | Cash and Deposits | $\text{Cash and Deposits}_{i,t}$ |
| NS | Net Sales | $\text{Net Sales}_{i,t}$ |
| OCF | Operating Cash Flows | $\text{Operating Cash Flows}_{i,t}$ |
| LTA | Logarithm of Total Assets | $\log(\text{Total Assets}_{i,t})$ |
| LNS | Logarithm of Net Sales | $\log(\text{Net Sales}_{i,t})$ |
| STA | Sales to Total Assets | $\frac{\text{Net Sales}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| RADA | Ratio of Allowance for Doubtful Accounts | $\frac{\text{Allowance for Doubtful Accounts}_{i,t}}{\text{Net Sales}_{i,t}}$ |
| RART | Ratio of Accounts Receivable to Assets | $\frac{\text{Accounts Receivable}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| RARS | Ratio of Accounts Receivable to Sales | $\frac{\text{Accounts Receivable}_{i,t}}{\text{Net Sales}_{i,t}}$ |
| RIS | Ratio of Inventory to Sales | $\frac{\text{Inventories}_{i,t}}{\text{Net Sales}_{i,t}}$ |
| RITA | Ratio of Inventory to Total Assets | $\frac{\text{Inventories}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| COGSS | Cost of Goods Sold to Sales | $\frac{\text{Material Consumption}_{i,t} + \text{Cost of Goods Sold}_{i,t}}{\text{Net Sales}_{i,t}}$ |
| FATA | Fixed Assets to Total Assets | $\frac{\text{Fixed Assets}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| PPETA | Property, Plant and Equipment to Total Assets | $\frac{\text{PP\&E}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| TDTA | Total Debt to Total Assets | $\frac{\text{Total Debt}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| ROA | Return on Assets | $\frac{\text{EBT}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| ROATA | ROA to Total Assets | $\frac{\text{ROA}_{i,t-1}}{\text{Total Assets}_{i,t}}$ |
| ROS | Return on Sales | $\frac{\text{Net Profit}_{i,t}}{\text{Net Sales}_{i,t}}$ |
| ROE | Return on Equity | $\frac{\text{EBT}_{i,t}}{\text{Equity}_{i,t}}$ |

| Variable | Indicator | Formula |
|---|---|---|
| CR | Current Ratio | $\frac{\text{Current Assets}_{i,t}}{\text{CL}_{i,t}}$ |
| QR | Quick Ratio | $\frac{\text{Cash}_{i,t} + \text{Accounts Receivable}_{i,t} + \text{Other Quick Assets}_{i,t}}{\text{CL}_{i,t}}$ |
| L | Liquidity | $\frac{\text{Working Capital}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| FL | Financial Leverage | $\frac{\text{Total Debt}_{i,t}}{\text{Total Equity}_{i,t}}$ |
| ETR | Effective Tax Rate | $\frac{\text{Tax Expense}_{i,t}}{\text{EBT}_{i,t}}$ |
| CFOTL | Operating Cash Flows to Total Liabilities | $\frac{\text{Operating Cash Flows}_{i,t}}{\text{Total Liabilities}_{i,t}}$ |
| WCTC | Personal Costs to Costs of Economic Activity | $\frac{\text{Personal Costs}_{i,t}}{\text{Costs of Economic Activity}_{i,t}}$ |
| DSRI | Days Sales in Receivable Index | $\frac{\text{Receivables}_{i,t} / \text{Net Sales}_{i,t}}{\text{Receivables}_{i,t-1} / \text{Net Sales}_{i,t-1}}$ |
| GMI | Gross Margin Index | $\frac{\text{Net Sales}_{i,t-1} - \text{Cost of Goods Sold}_{i,t-1} / \text{Net Sales}_{i,t-1}}{\text{Net Sales}_{i,t} - \text{Cost of Goods Sold}_{i,t} / \text{Net Sales}_{i,t}}$ |
| AQI | Asset Quality Index | $\frac{1 - \left(\text{Current Assets}_{i,t} + \text{PP\&E}_{i,t}\right) / \text{Total Assets}_{i,t}}{1 - \left(\text{Current Assets}_{i,t-1} + \text{PP\&E}_{i,t-1}\right) / \text{Total Assets}_{i,t-1}}$ |
| SGI | Sales Growth Index | $\frac{\text{Net Sales}_{i,t}}{\text{Net Sales}_{i,t-1}}$ |
| DEPI | Depreciation Index | $\frac{\text{Depreciation}_{i,t-1} / \left(\text{Depreciation}_{i,t-1} + \text{PP\&E}_{i,t-1}\right)}{\text{Depreciation}_{i,t} / \left(\text{Depreciation}_{i,t} + \text{PP\&E}_{i,t}\right)}$ |
| SGAI | Sales General Administrative Expenses Index | $\frac{\text{SG\&A}_{i,t} / \text{Net Sales}_{i,t}}{\text{SG\&A}_{i,t-1} / \text{Net Sales}_{i,t-1}}$ |
| TATA | Total Accruals to Total Assets | $\frac{\Delta\text{CA}_{i,t} - \Delta\text{Cash}_{i,t} - \left(\Delta\text{CL}_{i,t} - \Delta\text{LTD}_{i,t} - \Delta\text{TAX}_{i,t}\right) - \text{DA}_{i,t}}{\text{Total Assets}_{i,t}}$ |
| GMCH | Gross Margin Change | $\frac{\text{Net Sales}_{i,t} - \text{Direct Costs}_{i,t}}{\text{Net Sales}_{i,t-1} - \text{Direct Costs}_{i,t-1}}$ |
| TACH | Change in Total Assets | $\frac{\text{Total Assets}_{i,t}}{\text{Total Assets}_{i,t-1}}$ |
| ECH | Change in EAT | $\frac{\text{EAT}_{i,t}}{\text{EAT}_{i,t-1}}$ |
| CEREA | Cost-Effectiveness Indicator of Economic Activity | $\frac{\text{Cost of Economic Activity}_{i,t}}{\text{Revenue from Economic Activity}_{i,t}}$ |
| RSEAT | Receivables from Shareholders to EAT | $\frac{\text{Receivables from Shareholders}_{i,t}}{\text{EAT}_{i,t}}$ |

*Notes:* CA – Current Assets; CL – Current Liabilities; EAT – Earnings after Tax; EBT – Earnings before Tax; LTD – Current Maturities of LTD; PP&E – Property, Plant, and Equipment; SG&A – Selling, General, and Administrative Expenses; TAX – Income Tax Payable; DA – Depreciation and Amortization.

*Source*: Own calculations.

T a b l e 11

**List of Utilized Financial Variables – Definitions from Slovak Financial Statements**

| Variable | Indicator | Slovak financial statements |
|---|---|---|
| AR | Accounts Receivable | (41 S2) + (53 S2) |
| GM | Gross Margin | (03+04+05 V1) – (11 V1+12 V1+13 V1+14 V1) |
| TA | Total Assets | (01 S2) |
| TD | Total Debt | (79 S4 – 80 S4) |
| CD | Cash and Deposits | (71 S2) |
| NS | Net Sales | (03+04+05 V1) |
| OCF | Operating Cash Flows | [27 V1–08 V1+21 V1+24 V1+25 V1–57 V1 – (53 S2–53 S3) – (41 S2–41S3) – (34 S2–34 S3) + (101 S4–101 S5)] |
| LTA | Logarithm of Total Assets | ln(01 S2) |
| LNS | Logarithm of Net Sales | ln(03+04+05 V1) |
| STA | Sales to Total Assets | (03+04+05 V1) / (01 S2) |
| RADA | Ratio of Allowance for Doubtful Accounts | (25 V1) / (03+04+05 V1) |
| RART | Ratio of Accounts Receivable to Assets | (41 S2+53 S2) / (01 S2) |
| RARS | Ratio of Accounts Receivable to Sales | (41 S2+53 S2) / (03+04+05 V1) |
| RIS | Ratio of Inventory to Sales | (34 S2) / (03+04+05 V1) |
| ITA | Inventories to Total Assets | (34 S2) / (01 S2) |
| COGSS | Cost of Goods Sold to Sales | (11+12 V1) / (03+04+05 V1) |
| FATA | Fixed Assets to Total Assets | (02 S2) / (01 S2) |
| PPETA | Property, Plant and Equipment to Total Assets | (11 S2) / (01 S2) |
| TDTA | Total Debt to Total Assets | (79 S4-80 S4) / (01 S2) |
| ROA | Return on Assets | (56 V1) / (01 S2) |
| ROATA | ROA to Total Assets | (56 V2/01 S3) / (01 S2) |
| RS | Return on Sales | (61 V1) / (03+04+05 V1) |
| ROE | Return on Equity | (56 V1) / (80 S4) |
| CR | Current Ratio | (33 S2) / (122 S4) |
| QR | Quick Ratio | (71 S2+53 S2+34 S2) / (122 S4) |
| L | Liquidity | (33 S2–122 S4–136 S4–139 S4–140 S4) / (01 S2) |
| FL | Financial Leverage | (79 S4–80 S4) / (80 S4) |
| ETR | Tax Expense to EBT | (57 V1) / (56 V1) |
| CFOTL | Operating Cash Flows to Total Liabilities | (27 V1–08 V1+21 V1+24 V1+25 V1–57 V1–53 S2+53 S3–41 S2+41 S3–34 S2+34 S3+101 S4–101 S5) / (101 S4) |
| WCTC | Personal Costs to Costs of Economic Activity | (15 V1) / (10 V1) |
| DSRI | Days Sales in Receivable Index | [(41 S2+53 S2) / (03+04+05 V1)] / [(41 S2+53 S3) / (03+04+05 V2)] |
| GMI | Gross Margin Index | [(03+04+05–11–12–14 V2) / (03+04+05 V2)] / [(03+04+05–11–12–14 V1) / (03+04+05 V1)] |
| AQI | Asset Quality Index | [(1 – (33 S2+11 S2) / (01 S2)] / [(1 – (33 S3+11 S3) / (01 S3)] |
| SGI | Sales Growth Index | (03+04+05 V1) / (03+04+05 V2) |
| DEPI | Depreciation Index | [(22 V2) / (22 V2+11 S3)] / [(22 V1) / (22 V1+11 S2)] |
| SGAI | Sales General Administrative Expenses Index | [(14 V1) / (03+04+05 V1)] / [(14 V2) / (03+04+05 V2)] |
| TATA | Total Accruals to Total Assets | [(33 S2-33 S3–71 S2+71 S3 –122 S4+122 S5+57 V1-57 V2–21 V1)] / [(01 S2)] |
| GMCH | Gross Margin Change | (03+04+05-11-12-13-14 V1) / (03+04+05-11-12-13-14 V2) |
| TACH | Change in Total Assets | (01 S2) / (01 S3) |
| ECH | Change in EAT | (61 V1) / (61 V2) |
| CEREA | Cost-Effectiveness Indicator of Economic Activity | (10 V1) / (02 V1) |
| RSEAT | Receivables from Shareholders to EAT | (61 S2+49 S2) / (61 V1) |

*Source*: Own calculations.

## A p p e n d i x B: Best-Performing Models

T a b l e 12

**Full Sample: Best Parameters for Each Model**

| Model | Best parameters |
|---|---|
| Neural Network | optimizer: Adam, learning rate: 0.1, neurons: 16, layers: 2, epochs: 100, batch size: 20 |
| Random Forest | max depth: 10, max features: sqrt, n estimators: 100 |
| XGBoost | learning rate: 0.01, max depth: 4, n estimators: 100 |
| SVM | C: 100, gamma: 0.001, kernel: rbf, probability: True |

*Source*: Own calculations.

T a b l e 13

**Winsorized Sample: Best Parameters for Each Model**

| Model | Best parameters |
|---|---|
| Neural Network | optimizer: Adam, learning rate: 0.01, neurons: 16, layers: 1, epochs: 100, batch size: 20 |
| Random Forest | max depth: 10, max features: 'sqrt', n estimators: 400 |
| XGBoost | learning rate: 0.01, max depth: 4, n estimators: 100 |
| SVM | C: 100, gamma: 0.001, kernel: 'rbf', probability: True |

*Source*: Own calculations.

T a b l e 14

**Winsorized Sample by Industry: Best Parameters for Each Model**

| Model | Sector | Best parameters |
|---|---|---|
| Neural Network | A | optimizer: Adam, learn. rate: 0.001, neurons: 64, layers: 3, epochs: 100, batch size: 20 |
| Random Forest | A | max depth: 10, max features: 'sqrt', n estimators: 300 |
| XGBoost | A | learning rate: 0.1, max depth: 4, n estimators: 100 |
| SVM | A | C: 10, gamma: 0.0001, kernel: 'rbf', probability: True |
| Neural Network | B – E | optimizer: Adam, learn. rate: 0.001, neurons: 128, layers: 4, epochs: 100, batch size: 10 |
| Random Forest | B – E | max depth: 20, max features: 'log2', n estimators: 300 |
| XGBoost | B – E | learning rate: 0.01, max depth: 4, n estimators: 200 |
| SVM | B – E | C: 1, gamma: 0.001, kernel: 'rbf', probability: True |
| Neural Network | F | optimizer: Adam, learn. rate: 0.001, neurons: 32, layers: 2, epochs: 100, batch size: 10 |
| Random Forest | F | max depth: 40, max features: 'sqrt', n estimators: 100 |
| XGBoost | F | learning rate: 0.01, max depth: 8, n estimators: 200 |
| SVM | F | C: 1, gamma: 0.001, kernel: 'rbf', probability: True |
| Neural Network | G – S | optimizer: Adam, learn. rate: 0.001, neurons: 128, layers: 3, epochs: 100, batch size: 10 |
| Random Forest | G – S | max depth: 4, max features: 'sqrt', n estimators: 300 |
| XGBoost | G – S | learning rate: 0.01, max depth: 4, n estimators: 100 |
| SVM | G – S | C: 100, gamma: 0.0001, kernel: 'rbf', probability: True |
| Neural Network | Uncategorized | optimizer: Adam, learn. rate: 0.001, neurons: 32, layers: 4, epochs: 100, batch size: 10 |
| Random Forest | Uncategorized | max depth: 4, max features: 'sqrt', n estimators: 100 |
| XGBoost | Uncategorized | learning rate: 0.1, max depth: 4, n estimators: 100 |
| SVM | Uncategorized | C: 10, gamma: 0.001, kernel: 'rbf', probability: True |

*Source*: Own calculations.