ATENEO DE MANILA UNIVERSITY

Investigation of Stacked Ensemble Machine Learning for Tax Fraud Detection

A CAPSTONE PROJECT SUBMITTED TO
THE GRADUATE FACULTY OF
THE SCHOOL OF SCIENCE AND ENGINEERING
IN CANDIDACY FOR THE DEGREE OF
MASTER IN APPLIED MATHEMATICS MAJOR IN
MATHEMATICAL FINANCE

DEPARTMENT OF MATHEMATICS

BY

Albert Matthew Alejo
Kenrick Raymond So

QUEZON CITY, PHILIPPINES
APRIL 2025

The CAPSTONE PROJECT entitled:

Investigation of Stacked Ensemble Machine Learning for Tax Fraud Detection

submitted by Albert Matthew Alejo, and Kenrick Raymond So has been examined and is recommended for **Oral Defense**.

ELVIRA P. DE LARA-TUPRIO, Ph.D.
Adviser

JERIC C. BRIONES, Ph.D.
Adviser

EMMANUEL A. CABRAL, Ph.D
Adviser

ROMINA ANN YAP, Ph.D.
Chair

RAPHAEL A. GUERRERO, Ph.D.
Dean
School of Science and Engineering

The Faculty of the Department of Mathematics, School of Science and Engineering, Ateneo de Manila University ACCEPTS THE CAPSTONE PROJECT entitled:

Investigation of Stacked Ensemble Machine Learning for Tax Fraud Detection

submitted by Albert Matthew Alejo, and Kenrick Raymond So, in partial fulfillment of the requirements for the degree of Master in Applied Mathematics Major in Mathematical Finance.

ELVIRA P. DE LARA-TUPRIO, Ph.D.
Adviser

JERIC C. BRIONES, Ph.D.
Adviser

EMMANUEL A. CABRAL, Ph.D
Adviser

ROMINA ANN YAP, Ph.D.
Chair

RAPHAEL A. GUERRERO, Ph.D.
Dean
School of Science and Engineering

Grade : A (Excellent)

Date : May 15, 2025

ABSTRACT

This thesis examined how machine learning can be applied to improve tax fraud detection in the Philippines, with the goal of helping the Bureau of Internal Revenue (BIR) more effectively identify fraudulent activities. Traditional methods, which relied on manually reviewing financial records, were often slow and inefficient. To address this, the study combined financial data from the BIR with both supervised and unsupervised machine learning techniques to automate the detection process. The methodology followed a two-stage approach: first, unsupervised models such as Isolation Forest, Support Vector Machines, and K-means clustering were used to assign preliminary fraud labels to transactions. In the second stage, supervised learning algorithms, including logistic regression, gradient boosting, and artificial neural networks, were used to refine these labels and improve classification accuracy. Additionally, association rule mining was employed to uncover hidden relationships between transaction attributes, providing further insights into suspicious patterns. The results showed that the ensemble approach, particularly the XGBoost model, achieved a 95% out-of-sample accuracy, demonstrating its promise for automating tax fraud detection. Overall, the findings suggest that machine learning can significantly enhance the efficiency and scalability of fraud detection systems.

ACKNOWLEDGEMENTS

We would like to thank our advisors for their guidance and support throughout this research. Their feedback and encouragement were important in shaping our study. We especially appreciate Dr. Jeric Briones for his helpful suggestions and feedback, which improved our methodological framework. His expertise in machine learning helped strengthen the foundation of our work. We are also grateful to Mr. Luis Esquivias for his hospitality during our visits to the Bureau of Internal Revenue. His willingness to share his knowledge and his engaging approach to teaching us about tax filings made our learning experience both informative and enjoyable.

AUTHORSHIP CONTRIBUTION STATEMENT

AMA: Conceptualization, Data Curation, Investigation, Methodology, Software, Investigation, Formal analysis, Validation, Visualization, Writing - original draft, Writing - review & editing.

KRS: Conceptualization, Data Curation, Investigation, Methodology, Software, Investigation, Formal analysis, Validation, Visualization, Writing - original draft, Writing - review & editing.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

Tax fraud remains a significant challenge in the Philippines, resulting in lost government revenue and undermining fair business practices. Traditional detection methods, which rely on manual evaluation, are often slow and ineffective. Although machine learning offers potential for identifying fraudulent activities, the Bureau of Internal Revenue (BIR) has faced difficulties in adopting automated detection due to data quality issues.

This project aims to improve tax fraud detection by integrating financial records from the BIR. By analyzing these datasets, we seek to identify inconsistencies and automate the currently manual process of detecting potential fraud. Using both supervised and unsupervised machine learning techniques, the project focuses on developing a robust model capable of accurately identifying fraudulent entities. This model is intended to help the BIR strengthen tax compliance and enhance enforcement efficiency.

1.1. Tax Fraud in Philippines

The Bureau of Internal Revenue (BIR)[1] is the Philippine government agency responsible for assessing and collecting all national internal revenue taxes, fees, and charges. Operating under the Department of Finance, the BIR enforces tax laws to ensure compliance and manage the country's revenues.

BIR plays a key role in combating tax evasion by overseeing tax collection and enforcement, and ensuring that corporations and individuals accurately report their taxable income. Detecting fraudulent activities, however, often requires auditing financial statements and other corporate disclosures. These records can reveal inconsistencies such as misstated revenues, unreported subsidiaries, or undisclosed ownership, which may indicate fraudulent tax practices. By analyzing

[1] `https://www.bir.gov.ph`

financial data alongside tax records, BIR can more effectively identify discrepancies and detect companies involved in tax evasion schemes. This data-driven approach helps uncover hidden financial manipulation, improve risk assessment, and strengthen enforcement efforts.

1.2. Machine Learning Models

Machine learning (ML) models are algorithms that learn from data to make predictions or uncover patterns without being explicitly programmed. ML models are commonly categorized as supervised, unsupervised, or reinforcement learning (Murphy, 2012). This paper focuses on supervised and unsupervised approaches.

Supervised learning uses labeled data, where each input is paired with a known output. The model learns to generalize from these examples, aiming to predict outcomes for new, unseen data by minimizing the difference between its predictions and actual results. In contrast, unsupervised learning works with unlabeled data, seeking to discover hidden structures, groupings, or anomalies without predefined outputs. We employ both supervised and unsupervised machine learning algorithms to predict tax evasion, as they offer complementary strengths. Supervised models are effective when historical cases of tax evasion are available, enabling the identification of features linked to fraudulent activity. Unsupervised models are valuable when labeled data is limited, as they can reveal unusual patterns or outliers in taxpayer behavior that may signal potential fraud. Combining these methods enhances the accuracy and efficiency of tax evasion detection.

1.3. Relevant Literature

Many studies have investigated how machine learning can help detect tax fraud, but few have used a combined, or ”ensemble,” approach that brings together several different models for this specific purpose. Most existing research has focused on using single machine learning techniques or combining just two types. supervised (which learns from labeled examples) and unsupervised (which finds patterns in unlabeled data). However, a comprehensive framework that blends multiple machine learning methods to boost accuracy and reliability has not yet been fully explored.

Recent progress in machine learning has made it easier to spot suspicious patterns in large and complex financial datasets. For example, Murorunkwere et al. (2023) compared several supervised machine learning models, such as artificial neural networks (which mimic how the brain processes information), logistic regression (a method for predicting outcomes), decision trees, random forests, and XGBoost, and found that artificial neural networks were the most reliable for predicting tax fraud. Similarly, Shujaaddeen et al. (2024) developed a hybrid model that combines both supervised and unsupervised learning, and tested it using data from the Yemeni Tax Authority. Their approach was more efficient than previous methods, reducing the time and cost needed to detect tax fraud. Other researchers have focused on specific techniques. For instance, Baghdasaryan et al. (2022) used gradient boosting (a method that builds a series of simple models to improve predictions) to develop a fraud detection model, identifying key warning signs such as past fraud, audits, and certain economic activities. Because labeled data, where past fraud cases are clearly identified, can be scarce, de Roux et al. (2018) proposed an unsupervised approach that looks for unusual patterns in tax returns. This method helped reduce the number of audits needed while still catching previously undetected fraud. In another study, Andrade et al. (2021) trained several models, including K-Nearest Neighbors (which classifies data based on similarity), Random Forest (which combines many decision trees), Support Vector Machine (which finds the best boundary between classes), and Neural Networks, using financial and tax data. Random Forest performed best, correctly identifying fraudulent entities with an average accuracy of 92.98%. Ariyibi et al. (2024) showed that artificial intelligence techniques, including deep learning (which uses complex neural networks) and natural language processing (which analyzes text), can improve fraud detection rates by up to 85% compared to traditional rule-based methods. However, these advances also bring challenges, such as ensuring data quality, protecting privacy, and keeping models up to date. Careful oversight is needed to maintain fairness and transparency in tax administration. Finally, Matos et al. (2020) explored ways to select the most important features, or indicators, for detecting fraud. Using data from Brazil, they analyzed relationships between different warning signs, built a network to visualize these connections, and used a centrality measure to find the most relevant features for identifying potential fraudsters.

1.4. Objectives of the Study

This study aims to enhance tax fraud detection by applying a range of machine learning techniques to classify transactions and their counterparties as either fraudulent or non-fraudulent, using ghost receipt data. In addition, it seeks to uncover connections between entities that may signal a higher risk of fraudulent activity. The results from these two analyses will be combined in a traffic-light model, providing the BIR with a prioritized list of entities for further investigation or audit.

The first objective is to develop models that can accurately classify entities based on their likelihood of engaging in fraudulent behavior. This involves training predictive models on historical ghost receipt data, using features such as transaction patterns, counterparty relationships, and financial indicators. By leveraging both supervised and unsupervised machine learning techniques, the study aims to improve the accuracy and reliability of fraud detection. Multiple algorithms, including decision trees, neural networks, and gradient boosting, will be combined in an ensemble approach to maximize classification performance. The second objective is to identify entities that are closely linked to fraudulent transactions, which may indicate their involvement in tax fraud. To achieve this, the study will use association analysis, including network analysis and clustering methods, to reveal hidden relationships and high-risk groups within the data. Together, these approaches are designed to provide actionable insights for the BIR, supporting more effective risk assessment and targeted enforcement.

1.5. Scope and Limitations

This study uses a variety of machine learning models to detect fraudulent entities and conducts a separate association analysis to identify related risks. Although these tasks are conceptually connected, they are carried out independently, which may result in inconsistencies between the classification outcomes and the patterns revealed by association analysis. While this separation streamlines the methodology, it may also miss important interactions between fraudulent behavior and its broader network of relationships.

The dataset is composed of entities previously flagged as fraudulent under Philippine tax law. Consequently, the model's performance is closely tied to this specific legal context, and results may differ if applied elsewhere. Although the dataset includes confirmed cases of fraud, entities labeled as non-fraudulent are assumed to be compliant, which could introduce bias, some fraudulent entities may be incorrectly classified as non-fraudulent, affecting the model's reliability. Additional challenges include data imbalance and potential selection bias, which may limit the generalizability of the findings. Future research should consider using unsupervised or semi-supervised approaches to further validate and refine the models, and incorporate data from other jurisdictions to improve their robustness and applicability.

1.6. Method Overview

To identify potential cases of tax fraud, the BIR typically conducts lengthy and resource-intensive audits. Given limited capacity, efficiently managing these resources is a significant challenge. This capstone project seeks to enhance current systems by introducing a traffic light model, which integrates the outputs of two independent analyses: a fraud detection model and an association analysis model. The traffic light model provides the BIR with a prioritized list of entities for further investigation, helping to focus audit efforts where they are most needed.

Detecting fraud in financial transactions is difficult due to the rarity and constantly changing nature of fraudulent activities. To address this, the project employs a two-stage machine learning approach that leverages both unsupervised and supervised techniques. In the first stage, unsupervised models, such as Isolation Forest, Support Vector Machines, and K-means clustering, are used for pseudo-labeling, assigning preliminary labels to transactions that lack explicit fraud indicators. In the second stage, these pseudo-labels are used as ground truth to train supervised models, including logistic regression, gradient boosting, and artificial neural networks, which classify transactions as fraudulent or non-fraudulent. Because many real-world datasets lack explicit fraud labels, preprocessing steps include feature engineering to extract relevant indicators such as transaction amount, frequency, counterparty relationships, and temporal patterns. The dataset is then split into training and test sets to ensure reliable evaluation of model performance.

Beyond classification, this thesis incorporates association rule mining to uncover hidden patterns and relationships in transaction data. While classification models predict the likelihood of fraud, association rule mining identifies co-occurring behaviors and anomalies such as frequent transactions between certain entities or unusual transaction patterns that may signal fraudulent activity. This approach adapts to evolving fraud tactics without requiring frequent retraining and offers greater transparency, though it requires careful rule selection to avoid irrelevant or misleading patterns. By integrating association analysis with classification, the system becomes more robust and adaptive, providing the BIR with actionable insights for risk assessment and targeted enforcement.

1.6.1. Pseudolabelling

Once the data is prepared, the pseudolabelling phase addresses the lack of reliable labels by applying unsupervised learning models. This phase uses three approaches: Isolation Forest, Support Vector Machines in a one-class setting, and k-means clustering. In this context, the one-class SVM models typical transaction behavior and identifies anomalies by drawing a boundary around the majority of normal data points. Similarly, Isolation Forest and k-means clustering detect outliers and group transactions based on their similarity. These unsupervised techniques generate pseudolabels by assigning each transaction a tentative status of “fraudulent” or “non-fraudulent,” depending on how far it deviates from established patterns. Each model is set to divide the data into two groups, aiming to separate legitimate from potentially fraudulent transactions. To determine which group corresponds to legitimate or fraudulent activity, an initial analysis is performed using data with known labels. The group containing more known legitimate transactions is labeled as legitimate, while the group with more known fraudulent transactions is labeled as fraudulent. If both groups are predominantly legitimate or fraudulent, further analysis is conducted to clarify the classification. Although these pseudolabels are not perfect, they provide a valuable foundation for training more accurate supervised models in the next stage.

1.6.2. Classification

After provisional labels are assigned through pseudolabeling, the process advances to supervised learning. The pseudolabeled dataset is treated as if it were fully annotated, allowing the use of several supervised algorithms to improve classification accuracy. Logistic regression is used as a baseline model for binary classification, providing interpretable results and insights into the influence of individual features. More advanced models, such as XGBoost, are included to capture complex, non-linear relationships in the data. Artificial Neural Networks are also employed for their ability to learn intricate patterns that may indicate subtle forms of fraud. Random forests, which aggregate the predictions of multiple decision trees, are used to further enhance predictive performance. To maximize accuracy and robustness, an ensemble learning approach is adopted. This involves combining the predictions of logistic regression, XGBoost, ANNs, and random forests using stacking and weighted averaging methods. By integrating the strengths of each model, the ensemble approach reduces the risk of overfitting and ensures that the final fraud detection system benefits from both interpretable and highly expressive models.

The two-step approach, pseudolabeling followed by classification, offers flexibility and depth. Generating pseudolabels first addresses the common challenge of missing or unreliable labels in fraud datasets and allows for the discovery of hidden patterns that might be missed with only manually annotated labels. In the supervised learning phase, using a variety of models ensures that different algorithmic strengths are leveraged to improve detection accuracy. However, this method has limitations. The reliance on unsupervised techniques to generate pseudolabels introduces some uncertainty, as these provisional labels may include misclassifications that could affect the performance of the supervised models. Additionally, the two-stage approach assumes that the patterns identified during pseudolabeling are directly applicable to the classification task, and any discrepancies between these stages could lead to inconsistencies in the final fraud predictions. In summary, k-means clustering, one-class SVM, and isolation forests are used for pseudolabel generation, while supervised models include logistic regression, XGBoost, ANNs, and random forests. The ensemble strategy aims to maximize fraud detection accuracy and maintain stability across different data distributions by combining the strengths of diverse models.

1.6.3. Association Rule Mining

In addition to classification, this capstone used association rule mining to uncover hidden relationships and patterns within the data. While classification predicts specific outcomes (e.g., fraudulent vs. non-fraudulent transactions), association rule mining complements this by identifying co-occurrence patterns and dependencies among variables. This approach is especially valuable for exploratory data analysis, offering actionable insights into the structure and behavior of the dataset. The process begins by detecting frequent combinations of transaction attributes, which are then evaluated for statistical significance. A key challenge in fraud detection is distinguishing meaningful patterns from coincidental correlations, given the diversity and complexity of financial transactions. To address this, association rule mining employs statistical measures to assess the strength and relevance of detected patterns, helping to reveal behavioral anomalies such as sudden spikes in transaction volume or repeated interactions with unverified merchants.

A major advantage of association rule mining is its adaptability to evolving fraud tactics without frequent model retraining. As fraudsters change their methods, rule-based approaches enable the continuous discovery of new relationships between transactional features, supporting ongoing investigative and preventive efforts. Additionally, the transparency of association rules offers greater interpretability compared to black-box machine learning models. However, this flexibility can result in an overwhelming number of rules, many of which may be redundant or irrelevant. Effective rule selection, validation, and domain expertise are essential to ensure that only the most meaningful associations inform fraud detection. Despite its strengths, association rule mining is most effective when integrated with other analytical techniques. While it can highlight suspicious co-occurrences, it does not directly classify transactions as fraudulent or non-fraudulent, this requires further validation through supervised learning or anomaly detection. Moreover, it assumes that historical transaction patterns are indicative of future fraud, which may not always hold true. By combining association rule mining with classification-based methods, a more robust and adaptive fraud detection framework can be established, leveraging both structured patterns and predictive modeling to enhance detection accuracy.

1.7. Organization of the Manuscript

This thesis is structured as follows: Chapter 2 introduces the preliminary concepts and theoretical frameworks that underpin the study. Chapter 3 reviews the reference papers and outlines their relevance to the thesis. Chapter 4 details the experimental implementation, while Chapter 5 presents and discusses the results. Finally, Chapter 6 provides the conclusion and summarizes the key findings.

CHAPTER 2
PRELIMINARIES

In this capstone project, we employ XGBoost, Artificial Neural Networks (ANN), and Logistic Regression for supervised learning tasks. XGBoost's ensemble approach enhances predictive accuracy, ANN captures complex nonlinear patterns, and Logistic Regression offers interpretability for classification. For unsupervised learning, we utilize k-means clustering to group similar cases, Support Vector Machines (SVM) to detect outliers in high-dimensional spaces, and isolation forests to identify anomalies and underlying structures in taxpayer behavior. Additionally, Principal Component Analysis (PCA) is applied for dimensionality reduction, with its outputs serving as inputs to the aforementioned models, as described in the methodology section. This study also incorporates elements of Benford's analysis, a widely used technique in financial fraud detection. To further improve predictive performance, we integrate ensemble learning techniques to combine the outputs of individual models. The following sections provide an overview of the theoretical foundations underlying these approaches.

2.1. Benford Analysis

Benford analysis examines the distribution of leading digits in numerical datasets (Benford, 1938). According to Benford's Law, in many naturally occurring datasets, smaller digits appear as the first digit more frequently than larger ones. Specifically, the digit 1 occurs as the leading digit about 30% of the time, with the frequency decreasing for higher digits. The probability $P(d)$ that a number has leading digit d (where $d \in \{1, 2, \ldots, 9\}$) is given by the logarithmic formula:

$$P(d) = \log_{10}\left(1 + \frac{1}{d}\right)$$

This distribution typically emerges in datasets that span several orders of magnitude and are not artificially constrained. Benford's Law has practical applications in anomaly and fraud detection, as deviations from the expected digit distribution can indicate data manipulation or irregularities (Nigrini, 2012). However, it is important to note that not all datasets conform to Benford's Law. Datasets with inherent limits, such as those involving assigned numbers, fixed ranges, or human-generated data, may not exhibit the expected logarithmic pattern.

2.2. Backpropagation

Most machine learning models uncover complex, often nonlinear relationships in data by minimizing a loss function, which quantifies the difference between predicted and actual values. This minimization is typically achieved using gradient descent, an iterative optimization algorithm that updates model parameters in the direction that reduces the loss. For complex models, especially neural networks, finding a closed-form solution to this optimization problem is infeasible due to the high dimensionality and nonlinearity of the loss surface. The backpropagation algorithm addresses this challenge by efficiently computing gradients of the loss function with respect to each model parameter using the chain rule of calculus (Rumelhart et al., 1986). Backpropagation consists of two main steps: the forward pass and the backward pass (Bengio, 2016). In the forward pass, input data is propagated through the network to compute intermediate activations and the final output. In the backward pass, the algorithm calculates the gradients of the loss with respect to each parameter by propagating errors backward through the network. These gradients are then used to update the parameters via gradient descent. Despite its effectiveness, gradient-based optimization can encounter issues such as getting stuck in local minima, slow convergence, or instability due to inappropriate learning rates. Careful tuning and advanced optimization techniques are often required to address these challenges and ensure effective training.

2.3. Principal Component Analysis

Principal Component Analysis (PCA) is a statistical technique used to reduce the dimensionality of data by identifying key directions (principal components) that capture the most variance (Bishop, 2006). Given a data matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, where each of the n rows is an observation and each of the p columns is a variable, PCA finds new orthogonal axes that maximize the variance of the projected data (Jolliffe & Cadima, 2016). For simplicity, we assume the data is centered (mean zero for each variable); if not, centering can be performed as a preprocessing step. PCA seeks an orthonormal basis $\{\mathbf{v}_1, \ldots, \mathbf{v}_p\}$ for $\mathbb{R}^p$ such that the first vector $\mathbf{v}_1$ captures the largest variance, the second vector $\mathbf{v}_2$ captures the next largest variance orthogonal to $\mathbf{v}_1$, and so on. Formally, the kth principal component is found by solving:

$$\mathbf{v}_k = \underset{\|\mathbf{v}\|=1,\, \mathbf{v} \perp \{\mathbf{v}_1, \ldots, \mathbf{v}_{k-1}\}}{\arg\max} \mathbf{v}^\top \mathbf{\Sigma} \mathbf{v}, \quad \text{where } \mathbf{\Sigma} = \frac{1}{n-1} \mathbf{X}^\top \mathbf{X}.$$

Here, $\|\cdot\|$ denotes the Euclidean norm, and orthogonality is enforced with respect to previously found components.

By the spectral decomposition theorem, the optimal directions $\mathbf{v}_k$ are the eigenvectors of the covariance matrix $\mathbf{\Sigma}$, with corresponding eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$. The variance explained by the kth principal component is λ_k, and the total variance in the data is:

$$\operatorname{tr}(\mathbf{\Sigma}) = \sum_{k=1}^{p} \lambda_k.$$

2.4. Apriori Association Rule Mining

The Apriori algorithm is a method for mining association rules and discovering frequent patterns in large transactional datasets (Agrawal & Srikant, 1994). Its primary goal is to identify itemsets that commonly occur together in transactions and to generate association rules that reveal hidden relationships among these items.

Consider a dataset of transactions $\mathcal{D} = \{T_1, T_2, \ldots, T_m\}$, where each transaction T_j is a subset of a set of items $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$. An itemset $X \subseteq \mathcal{I}$ is a group of items that may appear together in one or more transactions. The frequency of an itemset X is measured by its support, defined as the proportion of transactions containing X:

$$\text{support}(X) = \frac{|\{T_j \in \mathcal{D} : X \subseteq T_j\}|}{|\mathcal{D}|}$$

where $|\cdot|$ denotes set cardinality.

Apriori operates iteratively, first identifying all itemsets that meet a minimum support threshold σ, ensuring only frequent itemsets are considered. The algorithm's efficiency is rooted in the *Apriori property*: if an itemset X is frequent, then all of its subsets must also be frequent. This property enables effective pruning of the search space, as any itemset with an infrequent subset can be immediately discarded. After identifying frequent itemsets, Apriori generates association rules of the form $X \Rightarrow Y$, where X and Y are disjoint itemsets and $X \cup Y \subseteq \mathcal{I}$. The strength of a rule is evaluated using two key metrics: confidence and lift. Confidence measures the conditional probability that Y appears in a transaction given X is present:

$$\text{confidence}(X \Rightarrow Y) = \frac{\text{support}(X \cup Y)}{\text{support}(X)}$$

Lift assesses how much more likely Y is to occur with X than would be expected by chance:

$$\text{lift}(X \Rightarrow Y) = \frac{\text{confidence}(X \Rightarrow Y)}{\text{support}(Y)}$$

Rules that meet minimum thresholds for confidence (γ) and lift (λ) are retained as strong rules. These rules provide valuable insights into item co-occurrence and can be applied in areas such as market basket analysis and fraud detection. The Apriori algorithm's pruning mechanism, based on the Apriori property, makes it highly efficient for large-scale association analysis.

2.5. Unsupervised Models

Unsupervised machine learning refers to techniques that analyze unlabeled data to uncover hidden patterns, structures, or relationships without predefined outputs. Unlike supervised learning, which relies on labeled data for prediction tasks, unsupervised learning explores the intrinsic organization of data using methods such as clustering, dimensionality reduction, and anomaly detection. This approach is especially valuable when labeled data is scarce, making it ideal for discovering insights, segmenting datasets, or identifying unusual behaviors. In contrast, supervised learning is better suited for tasks with clear input-output mappings, such as classification and regression.

Mathematically, unsupervised learning can be formulated as an optimization problem. Given a dataset $\mathbf{X} = \{x_1, x_2, \ldots, x_n\} \subseteq \mathbb{R}^d$ with no labels, the goal is to find a function $f : \mathbb{R}^d \to \mathcal{S}$ that captures underlying structure in the data. The nature of $\mathcal{S}$ depends on the specific task: for dimensionality reduction, $\mathcal{S} = \mathbb{R}^c$ with $c < d$; for clustering, $\mathcal{S}$ is a set of cluster assignments.

The learning process typically involves minimizing a loss function tailored to the task:

$$\min_{f \in \mathcal{F}} \mathcal{L}(f(\mathbf{X}))$$

where $\mathcal{F}$ is the set of allowable models or functions, and $\mathcal{L}$ is a predefined loss function. For example, in clustering, $\mathcal{L}$ might represent the within-cluster variance. This section provides a general overview of unsupervised learning. The following subsections will discuss the specific models applied in this capstone project.

2.5.1. Support Vector Machines

Support Vector Machines (SVM) are powerful supervised learning algorithms, particularly effective in high-dimensional spaces and in cases where the number of features exceeds the number of samples (Cristianini & Shawe-Taylor, 2000; Cortes & Vapnik, 1995). The fundamental idea behind SVM is to find an optimal hyperplane that separates data points of different classes while maximizing the margin, the distance between the hyperplane and the nearest data points, known as support vectors.

For binary classification, consider a dataset of m training examples, where each example is a pair (x_i, y_i) with $x_i \in \mathbb{R}^n$ representing an n-dimensional feature vector and $y_i \in \{-1, 1\}$ the class label. SVM seeks a decision boundary of the form:

$$f(x) = w^T x + b$$

where w is the weight vector and b is the bias term. The objective is to maximize the margin while ensuring correct classification, formulated as:

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

subject to:

$$y_i(w^T x_i + b) \geq 1, \quad \forall i = 1, \ldots, m.$$

In practice, data is often not perfectly separable. To address this, SVM introduces slack variables ξ_i to allow some misclassifications, resulting in the soft-margin SVM:

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \xi_i$$

subject to:

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i.$$

Here, C is a regularization parameter that balances maximizing the margin and minimizing classification errors. A larger C emphasizes correct classification, while a smaller C allows more flexibility for misclassification, potentially improving generalization.

When data is not linearly separable, SVM employs the kernel trick, mapping data into a higher-dimensional space where a linear separator may exist. Instead of explicitly computing this mapping, SVM uses a kernel function $K(x_i, x_j)$ to measure similarity in the transformed space. Common kernels include:

$$\begin{aligned}
K(x_i, x_j) &= x_i^T x_j && \text{(Linear Kernel)} \\
K(x_i, x_j) &= (x_i^T x_j + c)^d && \text{(Polynomial Kernel)} \\
K(x_i, x_j) &= \exp\left(-\gamma \|x_i - x_j\|^2\right) && \text{(Radial Basis Function Kernel)}
\end{aligned}$$

The choice of kernel function significantly affects model performance and should be guided by the data's structure.

2.5.2. k-means Clustering

K-means clustering is a widely used unsupervised learning algorithm for partitioning a dataset into K distinct, non-overlapping clusters (Jain, 2010). Given n data points $\{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^d$, the objective is to find K cluster centroids $\{\mu_1, \mu_2, \ldots, \mu_K\}$ that minimize the total within-cluster variance. Formally, the optimization problem is:

$$\min_{\mu_1,\ldots,\mu_K} \sum_{i=1}^{n} \|x_i - \mu_{c_i}\|^2, \tag{2.1}$$

where $c_i \in \{1, \ldots, K\}$ denotes the cluster assignment for data point x_i.

The k-means algorithm proceeds iteratively as follows:

1. **Initialization:** Select K initial centroids, either randomly or using a heuristic such as k-means++.

2. **Assignment step:** Assign each data point to the nearest centroid:

$$c_i = \arg\min_k \|x_i - \mu_k\|^2. \tag{2.2}$$

3. **Update step:** Recompute each centroid as the mean of all points assigned to that cluster:

$$\mu_k = \frac{1}{|\mathcal{C}_k|} \sum_{i:c_i=k} x_i, \tag{2.3}$$

   where $\mathcal{C}_k$ is the set of points assigned to cluster k.

4. **Convergence:** Repeat steps 2 and 3 until cluster assignments stabilize or centroid shifts fall below a predefined threshold.

K-means is efficient and scalable, but its performance can depend on the initial centroid selection and the value of K.

2.5.3. Isolation Forest

Isolation Forest is an unsupervised anomaly detection algorithm that identifies anomalies by isolating them, rather than profiling normal data points (Liu et al., 2008). The core principle is that anomalies are few and different, making them easier to isolate through recursive partitioning. The algorithm constructs an ensemble of randomly generated binary trees, known as isolation trees, to partition the data.

The construction of each isolation tree proceeds as follows:

1. Randomly select a feature from the dataset.
2. Randomly choose a split value within the range of the selected feature.
3. Recursively partition the data based on the split until a stopping criterion is met (e.g., a single data point in a node or a maximum tree depth).

Each data point is assigned an *isolation depth*, defined as the path length from the root to the terminating leaf node. Since anomalies are more susceptible to isolation, they tend to have shorter average path lengths across the ensemble of trees. The anomaly score for a data point x is computed as:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}, \tag{2.4}$$

where $E(h(x))$ is the average path length of x over all trees, and $c(n)$ is the average path length of unsuccessful searches in a binary search tree of size n, given by:

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n},$$

with $H(i)$ denoting the i-th harmonic number. Points with shorter average path lengths (i.e., higher anomaly scores) are more likely to be anomalies. Isolation Forest is efficient and effective for high-dimensional datasets, as it does not rely on distance or density measures and scales well with large data.

2.6. Supervised Models

Consider a set of covariates $X = \{x_1, \ldots, x_N\} \in \mathcal{X}$ and a response variable $Y \in \mathcal{Y}$. The objective of supervised learning is to learn a function $F^*(x)$ that approximates the true underlying relationship $F(X) : \mathcal{X} \to \mathcal{Y}$, based on a training dataset $\{(X_i, Y_i)\}_{i=1}^N$ where both covariates and responses are observed.

The optimal function F^* is chosen to minimize the expected value of a loss function $L(Y, F(X))$ with respect to the joint distribution of (X, Y). Formally,

$$F^* = \arg\min_F \mathbb{E}_{X,Y} \left[L(Y, F(X)) \right], \tag{2.5}$$

where $\mathbb{E}_{X,Y}$ denotes the expectation over the joint distribution of X and Y. This can also be expressed as:

$$F^* = \arg\min_F \mathbb{E}_X \left[\mathbb{E}_{Y|X} \left[L(Y, F(X)) \mid X \right] \right]. \tag{2.6}$$

A commonly used loss function is the mean squared error (MSE), defined as $L(Y, F(X)) = (Y - F(X))^2$, especially in regression tasks. The choice of loss function depends on the specific problem and learning objective.

2.6.1. Boosting and Weak Learning Algorithms

Boosting is a powerful ensemble technique that addresses challenging machine learning problems by combining multiple weak learners to form a highly accurate predictor (Schapire, 1990). The core idea is to iteratively apply a weak learning algorithm to modified versions of the training data, extracting a sequence of simple models (weak hypotheses) that, when aggregated, yield a strong overall model.

Given a training set

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

where each $x_i \in \mathcal{X}$ is an input instance and $y_i \in \mathcal{Y}$ is its corresponding label, boosting aims to approximate the true underlying function $F(x)$ by an ensemble of the form

$$F_M(x) = \sum_{m=1}^{M} \beta_m h_m(x),$$

where $h_m(x)$ are base (weak) learners, $\beta_m \in \mathbb{R}$ are their associated weights, and M is the total number of boosting rounds.

The boosting process typically starts with an initial model $F_0(x)$, often a constant. At each iteration m, a new weak learner $h_m(x)$ and its coefficient β_m are chosen to minimize the empirical risk with respect to a specified loss function L:

$$(\beta_m, h_m) = \arg\min_{\beta,h} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \beta h(x_i)\right), \quad m = 1, \ldots, M. \tag{2.7}$$

The ensemble model is then updated as:

$$F_m(x) = F_{m-1}(x) + \beta_m h_m(x). \tag{2.8}$$

Through this iterative process, boosting focuses on instances that are harder to predict, gradually improving the model's accuracy by correcting the errors of previous learners.

2.6.2. Gradient Boosting Machines

While selecting the exact optimal pair (β_m, h_m) at each boosting step is theoretically appealing, it is often computationally infeasible in practice. To address this, gradient boosting is employed when the loss function L is differentiable.

At each iteration m, instead of fully optimizing for the best base learner, gradient boosting fits $h_m(x)$ to the negative gradients (also known as pseudo-residuals) of the loss function with respect to the current model prediction $F_{m-1}(x)$. Specifically, the negative gradients are computed as:

$$\tilde{y}_{i,m} = -\left.\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right|_{F(x)=F_{m-1}(x)}, \qquad i = 1, 2, \ldots, n.$$

The next base learner h_m is then chosen to approximately solve:

$$h_m = \arg\min_{h\in\mathcal{H}} \sum_{i=1}^{n} L\left(\tilde{y}_{i,m}, h(x_i)\right), \tag{2.9}$$

where $\mathcal{H}$ denotes the set of possible base learners (e.g., all decision stumps, shallow trees, or linear functions).

After fitting h_m, the optimal step size β_m is determined by:

$$\beta_m = \arg\min_{\beta\in\mathbb{R}} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \beta h_m(x_i)\right). \tag{2.10}$$

Finally, the model is updated as:

$$F_m(x) = F_{m-1}(x) + \beta_m h_m(x).$$

This iterative procedure allows gradient boosting to efficiently minimize the loss function, leveraging the flexibility of differentiable losses and the power of weak learners.

2.6.3. Artificial Neural Network

Artificial Neural Networks (ANNs) are computational models inspired by the structure and function of the human central nervous system (Bishop, 2006). ANNs are designed to detect complex, non-linear relationships in data by mimicking the way biological neurons process and transmit information. The fundamental building block of an ANN is the artificial neuron, which receives inputs, processes them through an activation function, and produces an output. Each neuron computes a weighted sum of its inputs, adds a bias term, and applies a non-linear activation function f:

$$\text{Output} = f\left(\sum_{i=1}^{n} x_i w_i + b\right), \tag{2.11}$$

where x_i are the input features, w_i are the corresponding weights, b is the bias, and f is the activation function (e.g., sigmoid, ReLU, or tanh).

Neurons are organized into layers: an input layer, one or more hidden layers, and an output layer. The connections between neurons, known as synapses, are characterized by their weights, which determine the strength and direction of the signal transmitted. A perceptron is the simplest form of an ANN, consisting of a single layer of weighted inputs and an activation function, and is primarily used for binary classification tasks. The activation function transforms the input signal into an output, which is then passed to subsequent layers in deeper networks. Training an ANN involves optimizing the weights and biases to minimize prediction errors, typically using algorithms such as gradient descent. The error is quantified by a loss function, which measures the discrepancy between the predicted and actual outputs. For binary classification, the most common loss function is the binary cross-entropy (log-loss):

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(p(y_i)) + (1-y_i)\log(1-p(y_i))\right], \tag{2.12}$$

where y_i is the true class label, $p(y_i)$ is the predicted probability for class 1, and N is the number of data points. Through iterative optimization, ANNs adjust their parameters to learn complex mappings from inputs to outputs, making them highly effective for a wide range of tasks, including classification, regression, and pattern recognition.

2.6.4. Logistic Regression

Logistic regression is a fundamental statistical method used for binary classification, where the goal is to predict the probability that a given input belongs to one of two possible classes (typically labeled as 0 and 1) (Bishop, 2006). Unlike linear regression, which predicts continuous outcomes, logistic regression predicts probabilities constrained to the interval $[0, 1]$.

The model assumes that the log-odds (logit) of the probability of the positive class is a linear function of the input features:

$$\text{logit}(P(Y=1 \mid X)) = \log\left(\frac{P(Y=1 \mid X)}{1-P(Y=1 \mid X)}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n \tag{2.13}$$

To map the log-odds to a probability, the logistic (sigmoid) function is applied:

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{2.14}$$

where $z = \beta_0 + \sum_{i=1}^{n} \beta_i x_i$. Thus, the predicted probability that $Y = 1$ given X is:

$$P(Y=1 \mid X) = \frac{1}{1+e^{-(\beta_0+\sum_{i=1}^{n}\beta_i x_i)}} \tag{2.15}$$

Parameter Estimation: The parameters $\beta_0, \beta_1, \ldots, \beta_n$ are estimated using Maximum Likelihood Estimation (MLE). The likelihood function for N independent observations is:

$$L(\beta) = \prod_{i=1}^{N} P(y_i \mid x_i) = \prod_{i=1}^{N} [\sigma(z_i)]^{y_i} [1 - \sigma(z_i)]^{1-y_i} \tag{2.16}$$

where $z_i = \beta_0 + \sum_{j=1}^{n} \beta_j x_{ij}$. The log-likelihood, which is easier to maximize, is:

$$\ell(\beta) = \sum_{i=1}^{N} [y_i \log \sigma(z_i) + (1 - y_i) \log(1 - \sigma(z_i))] \tag{2.17}$$

Optimization algorithms such as gradient descent or Newton-Raphson are used to find the parameter values that maximize the log-likelihood. After fitting the model, predictions are made by thresholding the predicted probability. For example, if $P(Y = 1 \mid X) > 0.5$, the instance is classified as class 1; otherwise, it is classified as class 0. The coefficients β_j represent the change in the log-odds of the outcome for a one-unit increase in the corresponding feature x_j, holding all other features constant.

2.6.5. Random Forest

A decision tree is a hierarchical model that recursively splits data based on feature values to make predictions (Breiman, 2001). While decision trees are intuitive and interpretable, a single tree is prone to overfitting, performing well on training data but generalizing poorly to unseen data. Random Forest addresses this limitation by constructing an ensemble of decision trees and aggregating their predictions, thereby improving both accuracy and robustness.

Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{m}$ with m training samples, where $x_i \in \mathbb{R}^n$ is an n-dimensional feature vector and y_i is the target variable, Random Forest builds an ensemble of T trees. Each tree is trained on a bootstrap sample (random sampling with replacement) of the original data. At each split within a tree, only a randomly selected subset of features is considered, which introduces additional diversity among the trees. For classification, the final prediction is made by majority voting:

$$\hat{y} = \arg\max_{c} \sum_{t=1}^{T} \mathbb{I}(h_t(x) = c), \tag{2.18}$$

where $h_t(x)$ is the prediction of the t-th tree and $\mathbb{I}$ is the indicator function. For regression, the predictions are averaged:

$$\hat{y} = \frac{1}{T} \sum_{t=1}^{T} h_t(x). \tag{2.19}$$

The Random Forest algorithm proceeds as follows:

1. Draw T bootstrap samples from the training data.
2. For each sample, grow a decision tree:
    - At each node, select a random subset of features.
    - Determine the best split among the selected features.
    - Recursively grow the tree until a stopping condition is met.
3. Aggregate the predictions from all trees:
    - Use majority voting for classification.
    - Use averaging for regression.

Common stopping conditions include limiting the maximum tree depth and specifying a minimum number of samples required to split a node. Limiting tree depth helps control overfitting and computational cost, while setting a minimum sample size prevents splits on small, potentially noisy subsets of data.

2.7. Stacked Ensemble Learning

Stacked Ensemble Learning, or stacking, is a meta-learning technique that combines the predictions of multiple base models to enhance predictive performance (Wolpert, 1992). Unlike traditional ensemble methods such as bagging and boosting, which aggregate predictions through averaging or weighted voting, stacking introduces a higher-level model, known as a meta-learner, to learn how to best combine the outputs of diverse base models.

Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{m}$, where $x_i \in \mathbb{R}^n$ is an n-dimensional feature vector and y_i is the target variable, stacking proceeds as follows:

1. Train multiple base models $h_1(x), h_2(x), \ldots, h_K(x)$, each potentially using different algorithms or hyperparameters.

2. For each instance, collect the predictions from all base models to form a new feature vector.

3. Train a meta-learner $g(\cdot)$ on these new features to make the final prediction:

$$\hat{y} = g(h_1(x), h_2(x), \ldots, h_K(x)). \tag{2.20}$$

The meta-learner can be any machine learning algorithm, such as linear regression, logistic regression, or another ensemble method. To prevent overfitting and ensure unbiased meta-features, stacking typically employs cross-validation when generating base model predictions. The standard procedure is as follows:

1. Split the training data into K folds for cross-validation.

2. For each base model, train on $K - 1$ folds and generate predictions on the holdout fold. Repeat for all folds to obtain out-of-fold predictions for the entire dataset.

3. Stack the out-of-fold predictions from all base models to create a new training set for the meta-learner.

4. Train the meta-learner on this new dataset using the original target labels.

5. For new data, obtain predictions from all base models and feed them into the trained meta-learner to produce the final output.

Stacking leverages the complementary strengths of different models, often resulting in improved generalization and predictive accuracy compared to any single model.

2.8. Performance Metrics

To evaluate the performance of the supervised models, we focus on three key metrics: Positive Predictive Value (PPV), False Discovery Rate (FDR), and the F1 Score. Each metric provides a distinct perspective on the model's ability to distinguish between legitimate and fraudulent companies. A **true positive (TP)** is defined as a legitimate company correctly classified as legitimate, while a **true negative (TN)** is a fraudulent company correctly identified as fraudulent. These correct predictions indicate the model's effectiveness in recognizing legitimate and fraudulent behavior.

Positive Predictive Value (PPV), also known as precision, measures the proportion of positive predictions that are actually correct. In this context, it answers the question: among all companies predicted as legitimate, how many are truly legitimate? It is defined as:

$$\text{PPV} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

False Discovery Rate (FDR) quantifies the proportion of positive predictions that are incorrect, i.e., the fraction of companies incorrectly labeled as legitimate. FDR is the complement of precision and is given by:

$$\text{FDR} = \frac{\text{False Positives}}{\text{True Positives} + \text{False Positives}}$$

F1 Score provides a balance between precision and recall, making it especially useful when both false positives and false negatives are important. The F1 Score is the harmonic mean of precision and recall, and is calculated as:

$$\text{F1 Score} = \frac{2 \times \text{True Positives}}{2 \times \text{True Positives} + \text{False Positives} + \text{False Negatives}}$$

CHAPTER 3
METHODOLOGY

3.1. Raw Data

Figure 3.1 provides an overview of the datasets used in this study, which are based on administrative tax data collected by the Bureau of Internal Revenue (BIR) in the Philippines. The key components of the dataset are described below:

3.1.1. Summary List of Sales (SLS)

The SLS is a transaction-level report submitted quarterly by VAT-registered businesses, detailing all sales of goods and services to other VAT-registered purchasers. The specific columns extracted from this dataset are listed in Table A.1.

3.1.2. Summary List of Purchases (SLP)

The SLP is a transaction-level report submitted quarterly by VAT-registered businesses, detailing all purchases of goods and services from VAT-registered suppliers. The relevant columns extracted from this dataset are outlined in Table A.2.

3.1.3. VAT Returns

VAT returns are aggregated reports, typically filed monthly or quarterly, by VAT-registered businesses. These reports summarize VAT collected from sales (output VAT) and VAT paid on purchases (input VAT). The net difference determines the VAT payable to or refundable from the BIR. The columns extracted from this dataset are shown in Table A.3.

3.1.4. Industry Classification

This dataset provides a standardized code or category identifying the type of economic activity a taxpayer is engaged in. The classification follows the BIR's

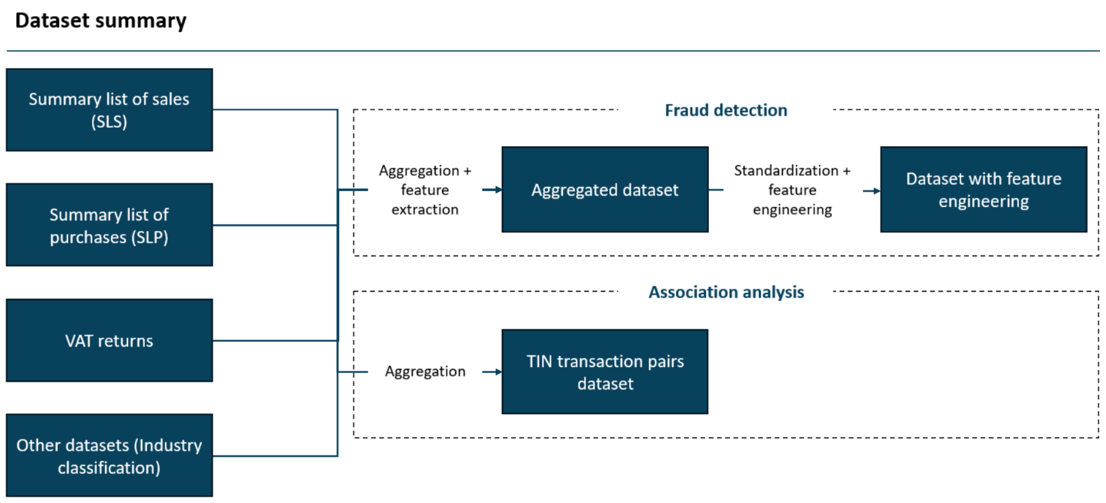

Dataset summary
Summary list of sales (SLS)
Summary list of purchases (SLP)
VAT returns
Other datasets (Industry classification)
Fraud detection
Aggregation + feature extraction
Aggregated dataset
Standardization + feature engineering
Dataset with feature engineering
Association analysis
Aggregation
TIN transaction pairs dataset

Figure 3.1: Summary of datasets used in fraud detection and association analysis

adopted scheme, which is generally aligned with the Philippine Standard Industry Classification (PSIC), and is used for segmentation and compliance monitoring.

For data preprocessing and feature extraction, we utilized `NumPy` and `Pandas`, which provided efficient tools for handling and manipulating large datasets. These libraries facilitated data cleaning, transformation, and organization. For feature engineering and model development, we used `Scikit-learn`, `TensorFlow`, `DMLC XGBoost`, and `MLxtend`. `Scikit-learn` and `XGBoost` were primarily used for building machine learning models, while `TensorFlow` enabled the implementation of deep learning models. `MLxtend` was employed to enhance model performance through techniques such as stacking and ensemble learning.

3.2. Aggregated Dataset for Fraud Classification

The four raw datasets (SLS, SLP, VAT Returns, and Industry Classification) were aggregated by `owner_tin`, `tax_year`, and `qtr`. This aggregation provides the total taxable sales and purchases per taxpayer per year and quarter, which are then merged with the VAT returns dataset. The resulting dataset contains aggregated values for taxable sales, purchases, and VAT payable for each `owner_tin`, `tax_year`, and `qtr`.

3.3. Aggregated Dataset for Association Mining

For association mining, the transaction-level datasets (SLS and SLP) were aggregated by `owner_tin` and the ratio of `pur_tin/sel_tin`. Here, `pur_tin` and `sel_tin` are redefined as `counterparty_tin`. This aggregation results in a dataset of transaction TIN pairs, which serves as the basis for association rule mining.

3.4. Ratio Calculation

Given the aggregated dataset for fraud classification, the following ratios were computed:

3.4.1. Sales-to-Purchases and Purchases-to-Sales

The Sales-to-Purchases and Purchases-to-Sales ratios are calculated using the aggregate sales and aggregate purchases for a given owner TIN, tax year, and quarter:

$$\text{Sales-to-Purchases}_{i,y,q} = \frac{\text{Sales}_{i,y,q}}{\text{Purchases}_{i,y,q}} \tag{3.1}$$

$$\text{Purchases-to-Sales}_{i,y,q} = \frac{\text{Purchases}_{i,y,q}}{\text{Sales}_{i,y,q}} \tag{3.2}$$

where i denotes the owner TIN, y the tax year, and q the quarter of filing.

3.4.2. Sales-to-VAT and VAT-to-Sales

The Sales-to-VAT and VAT-to-Sales ratios are calculated using aggregate sales and VAT for each owner TIN, tax year, and quarter:

$$\text{Sales-to-VAT}_{i,y,q} = \frac{\text{Sales}_{i,y,q}}{\text{VAT}_{i,y,q}} \tag{3.3}$$

$$\text{VAT-to-Sales}_{i,y,q} = \frac{\text{VAT}_{i,y,q}}{\text{Sales}_{i,y,q}} \tag{3.4}$$

3.4.3. Purchases-to-VAT and VAT-to-Purchases

The Purchases-to-VAT and VAT-to-Purchases ratios are calculated using aggregate purchases and VAT for each owner TIN, tax year, and quarter:

$$\text{Purchases-to-VAT}_{i,y,q} = \frac{\text{Purchases}_{i,y,q}}{\text{VAT}_{i,y,q}} \tag{3.5}$$

$$\text{VAT-to-Purchases}_{i,y,q} = \frac{\text{VAT}_{i,y,q}}{\text{Purchases}_{i,y,q}} \tag{3.6}$$

In all cases, i refers to the owner TIN, y to the tax year, and q to the quarter of filing.

3.5. Growth Rate Calculation

To analyze growth trends, the following growth rates were computed from the aggregated dataset:

3.5.1. Year-on-Year Sales, Purchases, and VAT Growth

The year-on-year growth rate measures the relative change in aggregate sales, purchases, or VAT for a given quarter compared to the same quarter in the previous year:

$$\text{Year-on-Year Growth}_{x,i,y,q} = \frac{x_{i,y,q}}{x_{i,y-1,q}} \tag{3.7}$$

where x denotes the feature of interest (e.g., Sales, Purchases, VAT), i the owner TIN, y the tax year, and q the quarter.

3.5.2. Quarter-on-Quarter Sales, Purchases, and VAT Growth

The quarter-on-quarter growth rate measures the relative change in aggregate sales, purchases, or VAT for a given quarter compared to the previous quarter:

$$\text{Quarter-on-Quarter Growth}_{x,i,y,q} = \begin{cases} \dfrac{x_{i,y,1}}{x_{i,y-1,4}} & \text{if } q = 1 \\ \dfrac{x_{i,y,q}}{x_{i,y,q-1}} & \text{if } q > 1 \end{cases} \tag{3.8}$$

where x denotes the feature of interest, i the owner TIN, y the tax year, and q the quarter of filing.

3.6. Standardization using a Modified Z-Score Formula

Standardization is the process of transforming feature values so that they follow a standard normal distribution (mean 0 and standard deviation 1). This transformation was applied to features with continuous or floating-point data types. The standard score, denoted by Z, is computed as:

$$Z_i = \frac{x_i - \bar{x}}{s} \tag{3.9}$$

where x_i is the observed value, $\bar{x}$ is the sample mean, and s is the sample standard deviation, defined as:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \qquad s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{3.10}$$

However, due to the presence of extreme outliers in the data, a robust standardization approach was also used. The modified Z-score, denoted by mZ, is calculated as:

$$mZ_i = \frac{x_i - \tilde{x}}{\text{MAD}} \tag{3.11}$$

where $\tilde{x}$ is the median of the feature and MAD is the Median Absolute Deviation:

$$\tilde{x} = \text{med}(x_i), \qquad \text{MAD} = \text{med}(|x_i - \tilde{x}|) \tag{3.12}$$

Standardization was performed in two ways:

1. **Company-specific:** Standardization based on the historical values of the same company, enabling detection of intra-firm anomalies.

2. **Industry-specific:** Standardization based on the historical values of all companies within the same industry, enabling detection of deviations from industry norms.

3.6.1. Company-Specific Standardization

For each company, the modified Z-score for feature x in year y and quarter q is:

$$mZ_{x,i,y,q} = \frac{x_{i,y,q} - \tilde{x}_i}{\text{MAD}_{x,i}} \tag{3.13}$$

where i is the owner TIN, $\tilde{x}_i$ and $\text{MAD}_{x,i}$ are the median and median absolute deviation of feature x for company i.

3.6.2. Industry-Specific Standardization

For each industry, the modified Z-score for feature x in year y and quarter q is:

$$mZ_{x,j,y,q} = \frac{x_{j,y,q} - \tilde{x}_j}{\text{MAD}_{x,j}} \tag{3.14}$$

where j is the industry code, and $\tilde{x}_j$ and $\text{MAD}_{x,j}$ are the median and median absolute deviation of feature x for industry j.

3.6.3. Features Subjected to Standardization

The following features were standardized using the approaches above:

- **Aggregates:** Total Sales, Purchases, VAT
- **Ratios:** Sales-to-Purchases, Purchases-to-Sales, Sales-to-VAT, VAT-to-Sales, Purchases-to-VAT, VAT-to-Purchases
- **Growth Rates:** Year-on-year and quarter-on-quarter growth rates for Sales, Purchases, and VAT

3.7. Final Extracted Features

The following features were extracted from the BIR dataset for statistical analysis and modeling:

- **Aggregates:** Standardized Total Sales, Purchases, and VAT (company-specific)

- **Ratios:** Standardized ratios (company- and industry-specific)
- **Growth Rates:** Standardized year-on-year and quarter-on-quarter growth rates (company- and industry-specific)
- **Benford Analysis:** Benford conformity measures and scores for Sales, Purchases, and VAT
- **Boolean Indicators:** Presence of VAT penalties, amended VAT returns
- **Exogenous Features:** Industry code, industry classification

3.8. Dimensionality Reduction using PCA

To improve computational efficiency and reduce feature redundancy, Principal Component Analysis (PCA) was applied. PCA transforms the standardized features into orthogonal principal components, which are linear combinations of the original features and are ordered by the variance they explain. By retaining only the top components, the dimensionality of the dataset is reduced, collinearity is minimized, and noise is suppressed, resulting in a more compact and informative representation for downstream modeling.

3.9. Model Pipeline

An overview of the model pipeline is shown in Figure 3.2. Due to the limited availability of labeled data, a pseudolabeling approach was adopted. Unsupervised models (Isolation Forest, SVM, k-means Clustering) were first applied to the labeled subset to identify patterns associated with legitimate and fraudulent transactions. The outputs were cross-referenced with known labels to interpret the groupings. These unsupervised models were then used to generate pseudolabels for the unlabeled data, which were subsequently used as ground truth for training supervised models (Logistic Regression, Random Forest, XGBoost, and ANNs). A stacked ensemble learning approach was also implemented to combine the strengths of individual models. Hyperparameters were tuned iteratively, and model performance was evaluated on a separate test set. Note that the use of

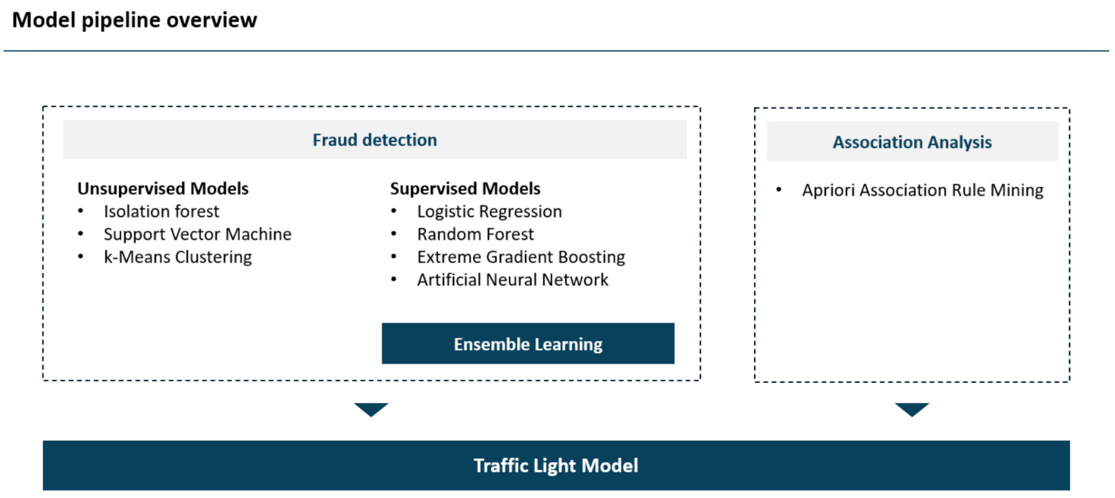


Figure 3.2: Overview of methodology

pseudolabels may introduce compounded errors, potentially affecting classification accuracy.

In parallel, Apriori Association Rule Mining was conducted to detect frequently occurring patterns and relationships among entities involved in suspicious transactions. The algorithm identifies item combinations exceeding a minimum support threshold, eliminates infrequent patterns, and generates interpretable "if-then" rules that satisfy confidence and lift criteria. This process uncovers hidden networks of related parties, such as individuals or companies linked to ghost receipts or suspicious intercompany activities.

Finally, outputs from both the supervised classification models and association rule mining were integrated into a risk assessment system. Each company was classified using a traffic light model with three levels: low, medium, and high risk of tax fraud. Specifically, a company is flagged as high risk if identified as fraudulent by both the classification model and association rules; medium risk if flagged by only one method; and low risk if neither method indicates fraudulent behavior. This integrated approach provides a robust framework for identifying and prioritizing potentially fraudulent activity within the dataset.

CHAPTER 4
RESULTS AND DISCUSSION

This section presents and interprets the results from the unsupervised learning models, supervised learning models, and association rule mining. The goal is to extract actionable insights and highlight the strengths and limitations of each approach.

4.0.1. Unsupervised Learning

Tables 4.1, 4.3, and 4.5 summarize the confusion matrices for Isolation Forest, SVM, and k-means, respectively. The corresponding outputs on the full dataset (labeled and unlabeled) are shown in Tables 4.2, 4.4, and 4.6.

The confusion matrices for the Isolation Forest, SVM, and k-means models are presented in Tables 4.1, 4.3, and 4.5, respectively, with their corresponding outputs on both labeled and unlabeled data shown in Tables 4.2, 4.4, and 4.6. Among these models, the Isolation Forest demonstrated the most effective separation between fraudulent and legitimate entities. Specifically, Cluster B identified by the Isolation Forest contained exclusively fraudulent cases (100%), while Cluster A comprised a majority of legitimate entities (60%), though with some overlap. This clear delineation suggests that the Isolation Forest is particularly adept at isolating high-risk, potentially fraudulent entities, making it a valuable tool for audit prioritization.

In contrast, both the SVM and k-means models failed to achieve meaningful separation. The SVM produced clusters with similar class distributions, each containing approximately 60% fraudulent entities, which undermines its utility for distinguishing between classes. The k-means model, while producing a cluster (A) composed entirely of fraudulent entities, still resulted in significant overlap in Cluster B, which contained 60% fraudulent cases. This limited discriminative power is likely influenced by class imbalance in the dataset, where fraudulent cases

Isolation Forest	**Clusters**	
Type	A	B
Legitimate	227 (60%)	0 (0%)
Fraudulent	155 (40%)	259 (100%)

Table 4.1: Confusion matrix for Isolation forest on data with known labels

Isolation Forest	**Clusters**	
Type	A	B
Total	2,679	2,680
Percentage	50%	50%

Table 4.2: Isolation forest output on labeled and unlabeled data

outnumber confirmed legitimate ones.

When the Isolation Forest was applied to the full unlabeled dataset, it produced an almost even split between Clusters A and B. While this result may not reflect the true underlying distribution of legitimate and fraudulent entities, the model's ability to isolate a pure fraudulent cluster remains valuable for generating pseudolabels. Importantly, these pseudolabels are used only as an intermediate step for training supervised models, so any imperfections in clustering do not directly propagate to final predictions. Given its relative strength in distinguishing between classes, the Isolation Forest was selected as the basis for pseudolabel generation, supporting a more robust downstream supervised learning process and enabling more targeted audit strategies.

4.0.2. Supervised Learning

Tables 4.10 to 4.17 present the confusion matrices for Logistic Regression, Random Forest, XGBoost, and Artificial Neural Networks on both training and testing splits. While training set results provide context for model behavior and potential overfitting, model selection is ultimately based on performance on the testing set, as this best reflects generalization to unseen data.

Among the evaluated models, XGBoost achieved the highest test accuracy at 95%, closely followed by the ANN and Random Forest models. Logistic Regression performed notably worse, with an accuracy of 87%. Both XGBoost and Random Forest attained perfect accuracy on the training set, which may indicate over-

SVM	**Clusters**	
Type	A	B
Legitimate	86 (39%)	141 (34%)
Fraudulent	136 (61%)	278 (66%)

Table 4.3: Confusion matrix for SVM on data with known labels

SVM	**Clusters**	
Type	A	B
Total	4,900	459
Percentage	91%	9%

Table 4.4: SVM output on labeled and unlabeled data

fitting, a common trait of tree-based models that are highly expressive and can capture complex patterns, sometimes at the expense of generalization. Addressing this overfitting in future work could involve expanding the dataset or applying regularization techniques such as early stopping or dropout. Interestingly, both the ANN and Logistic Regression models exhibited higher accuracy on the test set than on the training set, which is counterintuitive since models typically perform better on data they have already seen. This anomaly may be due to the relatively small sample size or the lack of a dedicated validation set for hyperparameter tuning, suggesting that further optimization and a more robust validation strategy could unlock additional performance.

Table 4.7 summarizes the key performance metrics: Positive Predictive Value (PPV), False Discovery Rate (FDR), and F1 score for each supervised model on both the training and testing sets. These metrics provide a nuanced view of each model's strengths and limitations beyond simple accuracy. Both Random Forest and XGBoost achieved perfect precision (PPV) and F1 scores on the training set, which is a strong indicator of potential overfitting, as these models may be capturing noise specific to the training data. However, on the test set, XGBoost slightly outperformed Random Forest, achieving a higher PPV (87.96% vs. 85.59%) and F1 score (93.60% vs. 92.23%), while also maintaining the lowest FDR (12.04%). The ANN model also demonstrated strong generalization, with test set metrics nearly matching those of XGBoost (PPV of 87.85%, FDR of 12.15%, and F1 score of 93.07%). This suggests that both XGBoost and ANN are robust to overfitting

k-means	**Clusters**	
Type	A	B
Legitimate	0 (0%)	227 (37%)
Fraudulent	30 (100%)	384 (63%)

Table 4.5: Confusion matrix for k-means on data with known labels

k-means	**Clusters**	
Type	A	B
Total	31	5,328
Percentage	1%	99%

Table 4.6: k-means output on labeled and unlabeled data

and are able to generalize well to unseen data. Logistic Regression, while showing improvement from training to testing, lagged behind the other models with a lower PPV (74.79%) and F1 score (83.18%) on the test set. This reinforces its relative weakness for this task, likely due to its limited capacity to capture complex, nonlinear relationships in the data.

Lastly, we explored the use of ensemble learning to combine the strengths of individual models in hopes of improving overall performance by mitigating their respective weaknesses. However, as shown in Tables 4.15 and 4.9, the ensemble model and the XGBoost model yielded identical performance. Given this result, we selected XGBoost as the final model due to its simplicity and parsimony, offering equivalent accuracy without the added complexity of an ensemble approach. Overall, these results support the selection of XGBoost as the final model. It consistently delivered the best balance between training and test performance across all evaluated metrics, offering high precision, low false discovery, and strong overall classification ability. Although ensemble learning was explored to potentially combine the strengths of individual models, it did not yield any improvement over XGBoost alone. Therefore, XGBoost was chosen for its simplicity, parsimony, and superior performance without the added complexity of an ensemble approach.

4.0.3. Association Rule Mining

The Apriori algorithm identified a total of 77 association rules involving 76 unique TINs and various features. The extracted rules exhibited lift values ranging

	Train			Test		
Score	PPV	FDR	F1	PPV	FDR	F1
Logistic Regression	62.98%	37.02%	52.61%	74.79%	25.21%	83.18%
Random Forest	100%	0%	100%	85.59%	14.41%	92.23%
XGBoost	100%	0%	100%	87.96%	12.04%	93.60%
ANN	97.67%	2.33%	93.43%	87.85%	12.15%	93.07%

Table 4.7: Accuracy metrics for the supervised models

Ensemble	**Predictions**	
Type	0	1
0	2,524	0
1	0	2,835

Table 4.8: Confusion matrix for stacked ensemble on training split

Ensemble	**Predictions**	
Type	0	1
0	95	0
1	0	167

Table 4.9: Confusion matrix for stacked ensemble on testing split

from 1.51 to 223.96, indicating varying degrees of association strength. Notably, 28 of these rules directly involve known fraudulent companies, and 28 distinct companies were found to be associated with these fraudulent entities.

A closer examination of the rules reveals several key insights:

- **Rule Coverage:** The 77 rules collectively cover a significant portion of the dataset, highlighting patterns of co-occurrence among TINs and features.
- **Fraudulent Associations:** Of the 77 rules, 28 specifically link TINs to known fraudulent companies, suggesting potential risk propagation through these associations.
- **Lift Values:** The lift values for all rules exceed 1, with the highest reaching 223.96. This indicates that the co-occurrence of items in these rules is much higher than would be expected by chance, signifying strong associations.

The interpretation of lift values is as follows:

- **Lift $>$ 1:** Items in the rule appear together more frequently than expected if they were independent, indicating a positive association.
- **Lift $=$ 1:** Items co-occur as often as expected by chance, implying no association.

Logistic	**Predictions**	
Type	0	1
0	1,140	1,384
1	670	2,165

Table 4.10: Confusion matrix for logistic regression on training split

Logistic	**Predictions**	
Type	0	1
0	89	6
1	30	150

Table 4.11: Confusion matrix for logistic regression on testing split

Random Forest	**Predictions**	
Type	0	1
0	2,524	0
1	0	2,835

Table 4.12: Confusion matrix for random forest on training split

Random Forest	**Predictions**	
Type	0	1
0	95	0
1	16	164

Table 4.13: Confusion matrix for random forest on testing split

- **Lift < 1:** Items appear together less frequently than expected, suggesting a negative association.

Since all discovered rules have lift values greater than 1, it can be concluded that the identified associations are statistically significant. In particular, TINs associated with known fraudulent companies are much more likely to be involved in fraudulent activity themselves. This highlights the effectiveness of association rule mining in uncovering hidden relationships and potential risk factors within the dataset.

4.0.4. Integrated Risk Assessment and Decision Matrix

By combining the outputs of the classification model and association rule mining, a comprehensive risk assessment system was developed (see Figure 4.1). This system classifies each company into low, medium, or high risk: high risk if flagged as fraudulent by both the classification model and association rules, medium risk if flagged by either method, and low risk if not flagged by either. High-risk entities should be prioritized for immediate audit and enforcement actions, medium-risk entities warrant monitoring and potential follow-up, and low-risk entities can be deprioritized, allowing resources to be focused where they are most needed.

Applying this framework to our testing set, the results are as follows:

- **High Risk:** 3 companies were flagged as fraudulent by both the supervised classification model and association rule mining. These entities represent

XGBoost	**Predictions**	
Type	0	1
0	2,524	0
1	0	2,835

Table 4.14: Confusion matrix for XGBoost on training split

XGBoost	**Predictions**	
Type	0	1
0	95	0
1	13	167

Table 4.15: Confusion matrix for XGBoost on testing split

ANN	**Predictions**	
Type	0	1
0	2,260	264
1	54	2,781

Table 4.16: Confusion matrix for ANNs on training split

ANN	**Predictions**	
Type	0	1
0	94	1
1	13	167

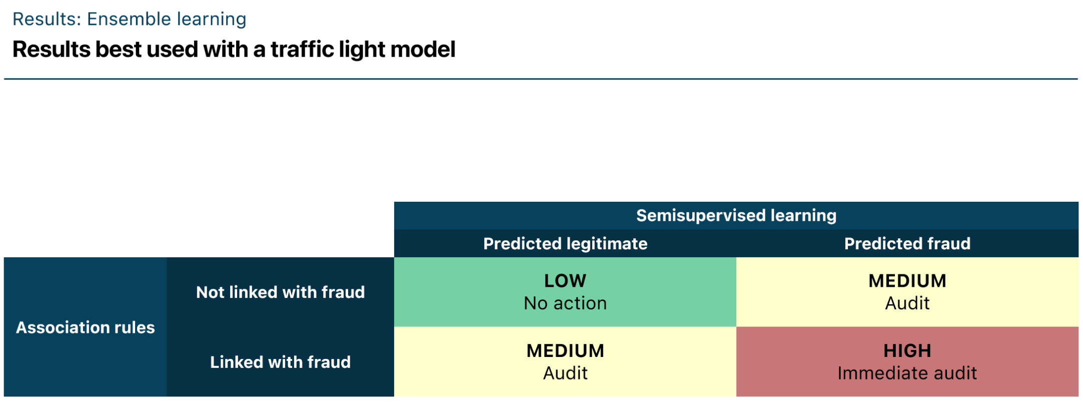
Table 4.17: Confusion matrix for ANNs on testing split

the highest priority for intervention.

- **Medium Risk:** 140 companies were predicted as fraudulent by the classification model but were not linked to any fraudulent association rules. Conversely, 173 companies were linked to fraud via association rules but were predicted as legitimate by the classification model. Both groups fall into the medium-risk category, indicating the need for further monitoring or investigation.

- **Low Risk:** The remaining companies in the testing set were not flagged by either method and are thus classified as low risk.

By focusing enforcement and monitoring efforts according to this traffic light model, resources can be allocated more efficiently and effectively.



Results: Ensemble learning

Results best used with a traffic light model

		Semisupervised learning	
		Predicted legitimate	Predicted fraud
Association rules	Not linked with fraud	**LOW** No action	**MEDIUM** Audit
	Linked with fraud	**MEDIUM** Audit	**HIGH** Immediate audit

Figure 4.1: Decision matrix based on results of the classification model and the association rules model

CHAPTER 5
CONCLUSION

This study, conducted in collaboration with the BIR, developed a data-driven framework to enhance the identification of entities for potential audit. By integrating supervised and unsupervised machine learning techniques with association rule mining, we aimed to automate and strengthen the BIR's tax fraud detection processes. Among the models evaluated, XGBoost demonstrated the highest out-of-sample accuracy at 95%, making it the most suitable candidate for operational deployment due to its strong predictive performance and interpretability. The Isolation Forest model played a key role in generating pseudolabels for the unlabeled dataset, enabling the extension of supervised learning to a broader set of taxpayers and informing audit prioritization, despite some limitations in cluster purity. Association rule mining further complemented these models by uncovering interpretable patterns and relationships, supporting both automation and transparency in decision-making.

Several limitations were encountered. The scarcity of labeled data constrained the development of fully supervised models, making the reliability of predictions dependent on the quality of pseudolabels. Data quality issues, such as missing TINs that prevented the integration of certain external datasets, also limited the scope for feature enrichment. Additionally, some evidence of overfitting in tree-based models and suboptimal performance in simpler models like logistic regression underscored the need for more comprehensive data and advanced tuning strategies. Despite these challenges, the resulting traffic light risk assessment system offers a practical and actionable tool for the BIR. By combining classification model outputs with interpretable association rules, the system provides clear, tiered signals for audit prioritization, enabling more efficient allocation of enforcement resources. With ongoing improvements in data integration, labeling, and model refinement, this approach has strong potential to further support the BIR's efforts to enhance tax compliance and reduce fraud at scale.

5.1. Recommendations for Further Work

To enhance the performance of supervised models, the BIR should prioritize expanding and improving its labeled datasets. The accuracy and robustness of these models are highly dependent on both the quality and quantity of labeled data, particularly for legitimate entities. Investing in the systematic creation and verification of comprehensive labels will enable the development of models with stronger generalization capabilities. Given the current reliance on pseudolabels, future work should also focus on refining pseudolabel generation methods. This could involve exploring advanced unsupervised or semi-supervised techniques, such as self-training, confidence-based thresholding, or generative approaches, to reduce labeling errors and increase the reliability of the training data.

Improved data integration across agencies is also essential. The inability to merge certain datasets, such as those from the Securities and Exchange Commission, due to missing Tax Identification Numbers limited the scope of analysis. Establishing robust data-matching systems and formalizing data-sharing agreements between agencies will help ensure more comprehensive and unified datasets for future studies. Additionally, addressing data quality issues such as inconsistent formatting, missing values, and unstructured entries is critical. Strengthening data cleaning processes and standardizing formats across BIR systems will provide higher-quality inputs for modeling.

To mitigate overfitting, particularly in tree-based models, regularization techniques such as early stopping, pruning, and careful feature selection should be employed. Access to larger and more diverse datasets will further help models learn patterns that generalize well. Moreover, the absence of a dedicated validation set hindered effective model tuning. Implementing a process to reserve a portion of the data for validation, even when labeled data is scarce, will facilitate better hyperparameter optimization and reduce the risk of overfitting.

BIBLIOGRAPHY

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th international conference on very large data bases* (p. 487–499). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Andrade, J., Paulucio, L., Paixão, T., Berriel, R., Carneiro, T., Carneiro, R., … Oliveira-Santos, T. (2021). A machine learning-based system for financial fraud detection. In *Anais do xviii encontro nacional de inteligência artificial e computacional* (pp. 165–176). Porto Alegre, RS, Brasil: SBC. doi: 10.5753/eniac.2021.18250

Ariyibi, K. O., Bello, O. F., Ekundayo, T. F., Oladepo, O. I., Wada, I. U., & Makinde, E. O. (2024, November). Leveraging artificial intelligence for enhanced tax fraud detection in modern fiscal systems. *GSC Advanced Research and Reviews*, *21*(2), 129–137. Retrieved from `http://dx.doi.org/10.30574/gscarr.2024.21.2.0415`

Baghdasaryan, V., Davtyan, H., Sarikyan, A., & and, Z. N. (2022). Improving tax audit efficiency using machine learning: The role of taxpayer's network data in fraud detection. *Applied Artificial Intelligence*, *36*(1), 2012002. doi: 10.1080/08839514.2021.2012002

Benford, F. (1938). The law of anomalous numbers. *Proceedings of the American Philosophical Society*, *78*(4), 551–572. Retrieved 2025-05-08, from `http://www.jstor.org/stable/984802`

Bengio, Y. (2016). *Deep learning.* London, England: MIT Press.

Bishop, C. (2006). *Pattern recognition and machine learning* (1st ed.). New York, NY: Springer.

Breiman, L. (2001).
Machine Learning, *45*(1), 5–32. Retrieved from `http://dx.doi.org/10.1023/A:1010933404324` doi: 10.1023/a:1010933404324

Cortes, C., & Vapnik, V. (1995, September). Support-vector networks. *Machine Learning*, *20*(3), 273–297. Retrieved from `http://dx.doi.org/10.1007/BF00994018` doi: 10.1007/bf00994018

Cristianini, N., & Shawe-Taylor, J. (2000, March). Preface. In *An introduction to support vector machines and other kernel-based learning methods* (pp. ix–xii). Cambridge: Cambridge University Press.

de Roux, D., Perez, B., Moreno, A., Villamil, M. d. P., & Figueroa, C. (2018, July). Tax fraud detection for under-reporting declarations using an unsupervised machine learning approach. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery &; data mining* (p. 215–222). ACM. doi: 10.1145/3219819.3219878

Jain, A. K. (2010, June). Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, *31*(8), 651–666. Retrieved from `http://dx.doi.org/10.1016/j.patrec.2009.09.011` doi: 10.1016/j.patrec.2009.09.011

Jolliffe, I. T., & Cadima, J. (2016, April). Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *374*(2065), 20150202. Retrieved from `http://dx.doi.org/10.1098/rsta.2015.0202` doi: 10.1098/rsta.2015.0202

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008, December). Isolation forest. In *2008 eighth ieee international conference on data mining* (p. 413–422). IEEE. Retrieved from `http://dx.doi.org/10.1109/ICDM.2008.17` doi: 10.1109/icdm.2008.17

Matos, T., Macedo, J. A., Lettich, F., Monteiro, J. M., Renso, C., Perego, R., & Nardini, F. M. (2020). Leveraging feature selection to detect potential tax fraudsters. *Expert Systems with Applications*, *145*, 113128. doi: https://doi.org/10.1016/j.eswa.2019.113128

Mururunkwere, B. F., Haughton, D., Nzabanita, J., Kipkogei, F., & and, I. K. (2023). Predicting tax fraud using supervised machine learning approach.

African Journal of Science, Technology, Innovation and Development, *15*(6), 731–742. doi: 10.1080/20421338.2023.2187930

Murphy, K. P. (2012). *Machine learning.* London, England: MIT Press.

Nigrini, M. (2012). *Benford's law* (M. J. Nigrini, Ed.). Nashville, TN: John Wiley & Sons.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533-536. Retrieved from https://api.semanticscholar.org/CorpusID:205001834

Schapire, R. E. (1990, June). The strength of weak learnability. *Machine Learning*, *5*(2), 197–227. Retrieved from http://dx.doi.org/10.1007/BF00116037 doi: 10.1007/bf00116037

Shujaaddeen, A., Ba-Alwi, F. M., & Al-Gaphari, G. (2024, 1). A new machine learning model for detecting levels of tax evasion based on hybrid neural network. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(11s), 450–468. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/4467

Wolpert, D. H. (1992, January). Stacked generalization. *Neural Networks*, *5*(2), 241–259. Retrieved from http://dx.doi.org/10.1016/S0893-6080(05)80023-1 doi: 10.1016/s0893-6080(05)80023-1

APPENDIX A

COLUMNS EXTRACTED FROM DATASET

Column	Data Type	Description	Sparse
owner_tin	int	Taxpayer identification number (TIN) of the filing company	No
pur_tin	int	TIN of the counterparty, i.e. the company that purchased goods and services from the filing company	Yes
tax_year	int	Tax year of when the transaction was filed	No
qtr	int	Quarter of when the transaction was filed	No
sls_taxable_sales	float	Taxable sales amount per transaction	No

Table A.1: Columns extracted from the summary list of sales

Column	Data Type	Description	Sparse
owner_tin	int	Taxpayer identification number (TIN) of the filing company	No
pur_tin	int	TIN of the counterparty, i.e. the company that purchased goods and services from the filing company	Yes
tax_year	int	Tax year of when the transaction was filed	No
qtr	int	Quarter of when the transaction was filed	No
sls_taxable_sales	float	Taxable sales amount per transaction	No

Table A.2: Columns extracted from the summary list of purchases

Column	Data Type	Description	Sparse
owner_tin	int	Taxpayer identification number (TIN) of the filing company	No
tax_year	int	Tax year of when the transaction was filed	No
qtr	int	Quarter of when the transaction was filed	No
amended_yn	boolean	Indicator whether the VAT return filing was amended/revised by the filing company after initial submission	No
penalties	boolean	Indicator whether the filing company was penalized due to filing violations (noncompliance, late submission, etc)	No
net_payable	float	Net VAT payable by the filing company for the specific tax year and quarter	No

Table A.3: Columns extracted from the VAT returns data

APPENDIX B

ASSOCIATION RULES RESULTS

Table B.1: Association rules with the 10 highest lift values

antecedents	**consequents**	**antecedent support**	**consequent support**	**support**	**confidence**	**lift**
{390718}	{10240063}	0.00179	0.00301	0.00120	0.67347	223.96
{10240063}	{390718}	0.00301	0.00179	0.00120	0.40000	223.96
{8214619}	{298393}	0.00171	0.01042	0.00166	0.96809	92.86
{298393}	{8214619}	0.01042	0.00171	0.00166	0.15909	92.86
{298393}	{8182020}	0.01042	0.00326	0.00231	0.22203	68.06
{8182020}	{298393}	0.00326	0.01042	0.00231	0.70950	68.06
{298393}	{9872871}	0.01042	0.00264	0.00171	0.16434	62.19
{9872871}	{298393}	0.00264	0.01042	0.00171	0.64828	62.19
{8182079}	{298393}	0.00503	0.01042	0.00219	0.43478	41.71
{298393}	{8182079}	0.01042	0.00503	0.00219	0.20979	41.71