

**Technische Hochschule Deggendorf**  
**Fakultät Angewandte Informatik**

Studiengang Master Artificial Intelligence and Data Science

**BETRUGSERKENNUNG MIT POSITIVEN UND  
UNMARKIERTEN DATENSÄTZEN DURCH  
INTERAKTIVE LERNTECHNIKEN**

**FRAUD DETECTION WITH POSITIVE AND  
UNLABELED DATASET USING INTERACTIVE  
LEARNING TECHNIQUE**

Masterarbeit zur Erlangung des akademischen Grades:

*Master of Science (M.Sc.)*

an der Technischen Hochschule Deggendorf

Vorgelegt von:

Mobin Al Hassan

Matrikelnummer: 22105632

Am: 15. July 2024

Prüfungsleitung:

Prof. Dr. Helena Liebelt

Ergänzende Prüfende:

Dr. Rui Li

Ergänzende Prüfende:

Dr. Yifeng Lu



## Erklärung

Name des Studierenden: Mobin Al Hassan

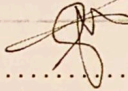
Name des Betreuenden: Prof. Dr. Helena Liebelt

Thema der Abschlussarbeit:

Betrugserkennung mit positiven und unmarkierten Datensätzen durch interaktive Lern-  
techniken .....

1. Ich erkläre hiermit, dass ich die Abschlussarbeit gemäß § 35 Abs. 7 RaPO (Rahmenprüfungsordnung für die Fachhochschulen in Bayern, BayRS 2210-4-1-4-1-WFK) selbständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen oder Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

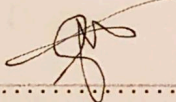
Deggendorf, 04.08.2024  
Datum

  
Unterschrift des Studierenden

2. Ich bin damit einverstanden, dass die von mir angefertigte Abschlussarbeit über die Bibliothek der Hochschule einer breiteren Öffentlichkeit zugänglich gemacht wird:

- ☐ Nein  
☒ Ja, nach Abschluss des Prüfungsverfahrens  
☐ Ja, nach Ablauf einer Sperrfrist von ... Jahren.

Deggendorf, 04.08.2024  
Datum

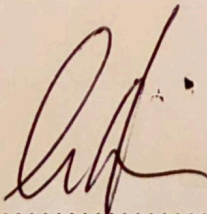
  
Unterschrift des Studierenden

Bei Einverständnis des Verfassenden vom Betreuenden auszufüllen:

Eine Aufnahme eines Exemplars der Abschlussarbeit in den Bestand der Bibliothek und die Ausleihe des Exemplars wird:

- ☒ Befürwortet  
☐ Nicht befürwortet

Deggendorf, .....  
Datum

  
Unterschrift des Betreuenden



# Abstract

In the field of Machine Learning, we often practice with datasets that are clean and well-balanced, meaning that each class has almost the same number of examples. However, data is usually messy and not labeled in the real world. This makes it hard to get large amounts of high-quality data, even though having more data is crucial for better performance.

Positive and Unlabeled Learning (PU learning) is crucial in machine learning for cases with few positive examples and many unlabeled ones, such as detecting fraud in financial transactions where labeling unlabeled data is costly or impractical. For instance, if a user clicks on one of six ads on a webpage, the clicked ad is a positive example, while the others remain unlabeled, not necessarily negative. We propose using a tree-based classifier for tabular datasets with a reinforcement algorithm. Unlike existing algorithms that rely on co-training or interactive learning with neural networks and require class priors or estimators, our approach leverages the strengths of tree-based algorithms for better performance on tabular data.

This thesis focuses on fraud detection using Positive and Unlabeled (PU) datasets with an interactive learning approach. In many practical situations, especially in fraud detection, we only have a few positive examples and a large subset of unlabeled data. This research explores how we can effectively use such data to train machine-learning models. By using interactive learning techniques, we aim to enhance the precision and effectiveness of fraud detection systems in practical scenarios where data quality and labeling present significant challenges.

**Keywords:** PU Learning; weakly supervised learning; semi-supervised learning; classification; and Imbalanced data



# Contents

<b>Abstract</b>	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background and Motivation . . . . .	2
1.2 Problem Statement . . . . .	4
1.3 Research Objectives . . . . .	5
1.4 Structure of this Thesis . . . . .	6
<b>2 Literature Review</b>	<b>9</b>
2.1 Related Work . . . . .	9
2.2 Imbalanced Datasets and Their Challenges . . . . .	12
2.2.1 Nature of Imbalanced Datasets . . . . .	12
2.2.2 Challenges of Imbalanced Datasets . . . . .	12
2.2.3 Fraud Detection . . . . .	13
2.2.4 Conclusion . . . . .	14
2.3 Positive and Unlabeled Learning . . . . .	15
2.3.1 Challenges in PU Learning . . . . .	15
2.3.2 Strategies for PU Learning . . . . .	15
2.3.3 Conclusion . . . . .	16
2.4 Co-training and Semi-Supervised Learning . . . . .	17
2.4.1 Co-training . . . . .	17
2.4.2 Semi-Supervised Learning . . . . .	19
2.5 Bayesian Approaches to PU Learning . . . . .	19
2.5.1 Bayesian Classifiers in PU Learning . . . . .	20
2.6 Policy Gradient Methods in PU Learning . . . . .	21
2.6.1 Core Concepts . . . . .	21
2.6.2 Methodology . . . . .	21
2.6.3 Advantages . . . . .	21
2.6.4 Applications . . . . .	22
2.7 Differentiating Techniques for Interactive Learning . . . . .	22
2.7.1 Active Learning . . . . .	22
2.7.2 Reinforcement Learning . . . . .	23
2.7.3 Semi-Supervised Learning . . . . .	23
2.7.4 Human-in-the-Loop . . . . .	23
2.8 Limitations of SVM for PU Learning . . . . .	23
2.8.1 Challenges . . . . .	23
2.8.2 Alternatives to SVM in PU Learning . . . . .	24

2.8.3	Conclusion . . . . .	24
<b>3</b>	<b>Methodology</b>	<b>25</b>
3.1	Data Collection and Preprocessing . . . . .	25
3.1.1	Datasets . . . . .	25
3.1.2	MNIST Dataset Preprocessing . . . . .	25
3.1.3	Construction of Imbalanced Dataset from Balanced Dataset . . . . .	25
3.1.4	Label Distribution Before and After Preprocessing . . . . .	27
3.1.5	Preprocessing steps . . . . .	28
3.1.6	Credit Card Transaction Dataset Preprocessing . . . . .	29
3.2	Reinforcement Learning in PU Learning . . . . .	29
3.2.1	Policy Network . . . . .	29
3.2.2	Reward Function . . . . .	29
3.2.3	Calculation of the Threshold . . . . .	30
3.3	Tree-Based Algorithm in PU Learning . . . . .	32
3.3.1	LightGBM . . . . .	32
3.4	Proposed Interactive Learning Framework . . . . .	32
3.4.1	Pre-Training Phase . . . . .	32
3.4.2	Interactive Training Process . . . . .	33
3.5	Model Selection and Justification . . . . .	33
3.5.1	Tree-Based Algorithm . . . . .	33
3.5.2	Dense Neural Network . . . . .	33
3.5.3	Why Tree-Based Algorithm over Neural Network for classification . . . . .	35
3.6	Evaluation Metrics . . . . .	35
3.6.1	Precision and Recall . . . . .	35
3.6.2	F1-Score . . . . .	36
3.6.3	Area Under the ROC Curve (AUC-ROC) . . . . .	36
3.6.4	Area Under the Precision-Recall Curve (AUC-PR) . . . . .	36
3.6.5	Implementation . . . . .	36
3.7	Conclusion . . . . .	36
<b>4</b>	<b>Experimental Setup</b>	<b>37</b>
4.1	Hardware and Software Environment . . . . .	37
4.1.1	Hardware . . . . .	37
4.1.2	Software . . . . .	37
4.2	Model Configuration . . . . .	39
4.2.1	Tree-Based Algorithm . . . . .	39
4.2.2	Reinforcement Learning Algorithm . . . . .	40
4.3	Conclusion . . . . .	40
<b>5</b>	<b>Results</b>	<b>41</b>
5.0.1	Experimental Setting . . . . .	41
5.0.2	Performance Analysis on MNIST Image Dataset . . . . .	42
5.0.3	Performance Analysis on Credit Card Tabular Dataset . . . . .	42



5.1	Policy Gradient with Neural Network Classifier on MNIST Dataset: Detailed Analysis . . . . .	43
5.1.1	Overview of the Learning Curves . . . . .	43
5.1.2	Initial High Performance and Subsequent Dip . . . . .	44
5.1.3	Influence of High Weight Decay . . . . .	44
5.1.4	Recovery and Stabilization . . . . .	44
5.1.5	Insights and Discussion . . . . .	45
5.2	Policy Gradient with LightGBM Classifier on MNIST Dataset: Detailed Analysis	45
5.2.1	Overview of the Learning Curves . . . . .	46
5.2.2	Steady Improvement and Model Learning . . . . .	46
5.2.3	Detailed Insights . . . . .	47
5.2.4	Discussion . . . . .	47
5.3	Policy Gradient with Neural Network Classifier on Credit Card Transaction Dataset: Detailed Analysis . . . . .	48
5.3.1	Overview of the Learning Curves . . . . .	48
5.3.2	Fluctuating Performance and Learning Instability . . . . .	49
5.3.3	Detailed Insights . . . . .	49
5.3.4	Discussion . . . . .	50
5.4	Policy Gradient with LightGBM Classifier on Credit Card Transaction Dataset: Detailed Analysis . . . . .	51
5.4.1	Overview of the Learning Curves . . . . .	51
5.4.2	Steady Improvement and Model Learning . . . . .	52
5.4.3	Detailed Insights . . . . .	52
5.4.4	Discussion . . . . .	53
<b>6</b>	<b>Conclusion</b>	<b>55</b>
6.0.1	Policy Network with Neural Network Classifier on MNIST Dataset . .	55
6.0.2	Policy Network with Neural Network Classifier on Credit Card Transaction Dataset . . . . .	55
6.0.3	Policy Network with LightGBM Classifier on MNIST Dataset . . . . .	55
6.0.4	Policy Network with LightGBM Classifier on Credit Card Transaction Dataset . . . . .	56
6.0.5	Overall Insights and Future Directions . . . . .	56
6.0.6	Overall Insights and Future Directions . . . . .	56
6.1	Future Work . . . . .	57



# List of Figures

2.1	Pictorial representation of CO-training . . . . .	18
2.2	Policy gradient interactive learning process. Inspired by the work in [1] . . . .	22
3.1	Labels in the original MNIST dataset [2]. . . . .	26
3.2	Sample of MNIST images labeled as 0 and 1 after preprocessing. . . . .	26
3.3	Label Distribution in MNIST Dataset Before Preprocessing . . . . .	27
3.4	Label Distribution in MNIST Dataset After Preprocessing . . . . .	28
3.5	Proposed framework for PU learning . . . . .	34
5.1	Models performance results . . . . .	42
5.2	Learning curves for Policy Gradient with NN Classifier on MNIST . . . . .	43
5.3	Learning curves for Policy Gradient with LightGBM on MNIST Dataset . . . .	46
5.4	Learning curves for Policy Gradient with NN on Credit Card Dataset . . . . .	49
5.5	Learning curves for Policy Gradient with LightGBM on Credit Card Dataset .	51



# 1 Introduction

Traditional machine learning methods heavily labeled data depended, which can be both costly and difficult to obtain. The necessity for labeled data presents significant challenges, particularly when high-quality labeled examples are limited. Numerous real-world applications struggle to acquire a substantial amount of completely labeled data due to the high costs and extensive time required for labeling efforts.

To overcome the obstacle of having insufficient labeled data, Blum and Mitchell [3] introduced the co-training algorithm, which employs two separate learning models, each trained on distinct views of the same dataset. These views leverage different features or complementary learning tasks, and each model iteratively improves its training by incorporating the most confident predictions from the other model on the unlabeled data. This process enhances the accuracy of both models, effectively utilizing the abundance of unlabeled data, which is especially valuable when labeled data is scarce.

Binary classification aims to develop a model that distinguishes between positive and negative examples. Traditionally, this requires training data that includes fully labeled examples of both classes, a problem setup extensively studied in machine learning. However, in many practical scenarios, the training data comprises only positive and unlabeled examples. This approach, known as (PU learning) Positive and Unlabeled, seeks to construct a classifier using a dataset with positive examples and a large set of unlabeled data.

The concept of PU learning started to develop in the early 2000s and has subsequently attracted considerable interest because of its relevance to numerous fields. For instance, in personalized advertising, visited pages and clicked ads are treated as positive examples, while all other ads are considered unlabeled rather than negative. Similarly, in medical records, the absence of a diagnosis does not necessarily indicate or directly mean the absence of a disease; it may simply mean the disease was not diagnosed [4].

Learning from Positive and Unlabeled (PU) data is essential because explicit negative examples are often not naturally available in many domains. For example, specialized databases in molecular biology might define a set of positive examples, such as specific genes or proteins, without including explicit negative examples [5]. Instead, all other examples remain unlabeled, and the objective is to identify additional relevant examples from this pool.

This thesis digs into applications of PU learning to fraud detection, an area where fully labeled datasets are frequently unavailable. By employing interactive learning techniques, this research aims to develop methods that effectively utilize positive and unlabeled data to improve the precision of fraud detection systems. Although neural networks are widely popular and frequently applied across various cases, this research focuses on using tree-based algorithms in conjunction with neural networks within the interactive learning process. This is particularly relevant for tabular datasets, where tree-based algorithms have demonstrated superior performance [6] [7]. Interactive learning involves iterative processes where the model's predictions are continuously refined through feedback to the reinforcement algorithm, making it especially well-suited for situations with finite labeled data.

To sum up, PU learning is a major breakthrough in machine learning that makes it possible to build strong models even with partly labeled data. The major goal of this thesis is to make a contribution to this field by investigating the efficacy of fusing neural networks and tree-based algorithms, and by using PU learning and interactive learning approaches to fraud detection. This strategy highlights the usefulness of these techniques in practical settings, possibly enhancing the precision and resilience of models in a range of domains where labeled data is hard to come by.

## 1.1 Background and Motivation

Over the past few decades, machine learning has made major strides that have revolutionized different sectors via the creation of intelligent systems and predictive models. However, the availability of high-quality labeled data, which is sometimes hard to come by and costly to acquire, is a major need for these models' success. This problem is most noticeable in applications that work with unbalanced datasets, when the dominant class is much outnumbering the minority class of interest, which might include fraud instances or uncommon illnesses. Because typical machine learning models are biased toward the majority class, the inherent imbalance in such datasets presents a significant difficulty, resulting in worse performance on the minority class.

In the context of fraud detection, for instance, the difficulty in identifying fraudulent activities is exacerbated by the limited availability of labeled fraud cases compared to the overwhelming number of non-fraud/unknown transactions. This imbalance can result in models that fail to accurately detect fraud, thereby compromising the effectiveness of fraud detection systems. Similar challenges are observed in medical diagnosis, where rare diseases may go undiagnosed due to the unavailability or lack of labeled examples, and in personalized advertising, where the lack of negative feedback complicates the task of predicting user

preferences.

The idea of (PU) Positive and Unlabeled learning has come to light as a potential remedy for these problems. A specific type of semi-supervised learning known as "positive and unlabeled learning" concentrates on training classifiers with datasets made up of positive and unlabeled examples, rather than labeled negative examples. This strategy is especially applicable in situations when it is difficult or impossible to assign labels to unfavorable occurrences. Similar to how a lack of a diagnosis in a medical record does not always indicate the presence of a condition, unclicked adverts in customized advertising should not always be interpreted as a negative choice.

The co-training algorithm introduced by Blum and Mitchell in 1998 [3] laid the groundwork for leveraging unlabeled data by training two separate models on distant views of the identical dataset. Each model iteratively incorporates the most confident predictions from the other model, thereby enhancing overall accuracy. This innovative approach underscores the potential of utilizing unlabeled data to enhance model performance, particularly when labeled data is insufficient.

Building on the principles of co-training, PU learning techniques have evolved to address the specific challenges of working with positive and unlabeled data. One notable method is the cost-sensitive approach, which adjusts the learning process to account for the absence of negative labels, as detailed in the work by Elkan and Noto [8]. This method has demonstrated significant improvements in various applications, including text classification and bioinformatics, where labeled negative examples are not readily available.

However, the reliance on neural networks in PU learning, driven by their powerful feature extraction capabilities, has predominantly shaped the research landscape. Neural networks have proven effective in handling large, unstructured data like images and text, but their applicability to tabular data—common in many real-world applications—remains less explored in interactive learning approach. Tree-based algorithms, known for their robustness and superior performance on tabular datasets [6], present a compelling alternative. These algorithms, including Random Forests and Gradient Boosting, excel in handling categorical features and capturing complex interactions within the data.

My motivation to dig into the challenges of imbalanced datasets and explore PU learning techniques is rooted in my professional experiences and broader interest in advancing machine learning methodologies. Working with a leading insurance company, I encountered the significant challenge of detecting fraudulent activities within highly imbalanced datasets. This experience highlighted the critical need for advanced techniques that can effectively

utilize available data to improve model performance. Fraud detection is just one of many applications where imbalanced datasets are prevalent, and the solutions developed can have far-reaching implications for other fields facing similar challenges.

Interactive learning techniques, which involve iteratively refining model predictions through continuous feedback, offer a promising avenue for enhancing the performance of PU learning models. By integrating tree-based algorithms with neural networks within an interactive learning framework, I aim to evaluate the effectiveness of this hybrid approach. The iterative nature of interactive learning is highly suitable for scenarios with limited labeled data, as it allows for continuous improvement of the model based on new information.

The broader goal of my thesis research is to advance machine learning methodologies that do not rely exclusively on fully supervised data. By exploring and validating novel approaches like PU learning and interactive learning, my goal is to contribute to the development of robust models capable of operating effectively in environments with sparse or incomplete labeled data. This research extends beyond improving fraud detection systems to encompass a wide range of applications affected by similar data challenges, such as medical diagnosis and personalized advertisement.

In conclusion, addressing the issue of imbalanced datasets and the scarcity of labeled data is required for the advancement of the machine learning across various fields. My research aims to develop innovative solutions that leverage positive and unlabeled data, enhancing the accuracy and robustness of models in practical applications. By integrating tree-based algorithms with neural networks within an interactive learning framework, this work holds the potential to significantly improve not only fraud detection systems but also other applications impacted by similar data challenges, thereby contributing to the wider field of machine learning.

## 1.2 Problem Statement

The domain of Positive-Unlabeled (PU) learning has primarily focused on image datasets such as MNIST and CIFAR-10, which offer well-defined benchmarks and facilitate comparative studies. However, this focus leaves a gap in understanding the applicability and effectiveness of PU learning techniques on tabular datasets, which are more general in various real-world strategies such as fraud detection and personalized advertisement. These domains often involve complex, high-dimensional tabular data, posing unique challenges for PU learning.

Furthermore, the majority of prior research [1] [9] [10] in PU learning has heavily relied on



neural networks due to their powerful feature extraction capabilities and success in handling large, unstructured data like images and text. However, neural networks may not always be the best fit for tabular data, where tree-based algorithms have consistently outperformed due to their inherent ability to handle categorical features [6] [7] and capture non-linear interactions. The question then arises whether tree-based algorithms, either alone or in combination with neural networks, can be effectively utilized in PU learning, especially in the case of tabular data.

Additionally, there is a significant need to explore interactive learning frameworks that iteratively refine model predictions through continuous feedback. This technique is quite beneficial in scenarios with limited labeled data, such as fraud detection. The effectiveness of combining tree-based algorithms with neural networks within an interactive learning framework remains underexplored, and there is a lack of systematic studies addressing this integration.

Thus, the problem statement for this thesis can be summarized as follows:

- Prior research has primarily utilized image datasets (e.g., MNIST, CIFAR-10) for PU learning, leaving a gap in understanding the applicability of PU learning techniques on tabular datasets.
- Neural networks are widely used for PU learning, but their effectiveness compared to tree-based algorithms for tabular data is not well-established.
- There is a need to investigate the potential of integrating tree-based algorithms with neural networks within an interactive learning framework to enhance model performance on tabular datasets.
- Developing strategies for effectively identifying positive samples from unlabeled data in the context of PU learning.

## 1.3 Research Objectives

The primary objective of this thesis is to advance the field of PU learning by addressing the identified gaps and challenges, specifically focusing on tabular datasets and the integration of tree-based algorithms with neural networks. The research objectives are outlined as follows:

- **Objective 1: Evaluate the Effectiveness of Tree-Based Algorithms in PU Learning**
  - Investigate the performance of tree-based algorithms, such as Random Forests and Gradient Boosting, in the context of PU learning on tabular datasets.

- Compare the results with traditional neural network approaches to determine the relative advantages and limitations.
- **Objective 2: Develop a Hybrid Approach Combining Tree-Based Algorithms and Neural Networks**
  - Design a hybrid model that integrates tree-based algorithms with neural networks to leverage the strengths of both methodologies.
  - Implement the hybrid model within an interactive learning framework to iteratively improve model accuracy through continuous feedback.
- **Objective 3: Formulate Strategies for Identifying Positive Samples from Unlabeled Data**
  - Develop and evaluate methods for effectively selecting positive samples from unlabeled data to enhance the training process.
  - Assess the impact of different selection strategies on the overall model performance and robustness.
- **Objective 4: Apply the Proposed Methods to Fraud Detection**
  - Implement the developed PU learning techniques in the context of fraud detection, utilizing real-world tabular datasets.
  - Evaluate the performance improvements and practical implications of the proposed methods in a real-world application.

## 1.4 Structure of this Thesis

This thesis is arranged into several chapters, each focusing on different aspects of PU learning and the proposed research objectives. The design of the thesis is as given below:

- **Chapter 1: Introduction**
  - Provides an overview of traditional machine learning challenges related to labeled data scarcity.
  - Introduces the concepts of co-training and PU learning.
  - Discusses the relevance of PU learning in various applications and sets the stage for the thesis.

- **Chapter 2: Literature Review**

- Reviews existing publications on PU learning, with a focus on both neural network and tree-based approaches.
- Examines previous work on interactive learning and its applicability to PU learning.
- Identifies gaps and areas for further research.

- **Chapter 3: Methodology**

- Describes the experimental setup, including the datasets utilized, evaluation metrics employed, and baseline models referenced.
- Details the proposed hybrid model combining tree-based algorithms and neural networks.
- Explains the interactive learning framework and strategies for selecting positive samples from unlabeled data.

- **Chapter 4: Experiments and Results**

- Presents the experimental results, comparing the performance of tree-based algorithms, neural networks, and the proposed hybrid model.
- Discusses the effectiveness of different positive sample selection strategies.
- Analyzes the impact of the hybrid approach on fraud detection accuracy.

- **Chapter 5: Case Study in Fraud Detection**

- Applies the proposed PU learning techniques to a real-world fraud detection dataset.
- Evaluates the practical implications and performance improvements in the context of fraud detection.

- **Chapter 6: Discussion**

- Recaps the key discoveries and contributions of the thesis.
- Discusses the limitations of the current study and potential areas for future research.

- **Chapter 7: Conclusion**

- Provides a concluding outline of the thesis.
- Highlights the significance of the research and its contributions to the field of PU learning and fraud detection.



## 2 Literature Review

The challenge of imbalanced datasets is a common topic in machine learning. Traditional machine learning models are heavily dependent on labeled data to function well. However, acquiring a large amount of fully labeled data is often problematic due to the high cost and the time necessary for labeling, and sometimes it's impossible, for instance, in the case of identifying other diseases for the person who was diagnosed with a particular disease. In this chapter, I will review the existing literature related to Positive and Unlabeled (PU) learning and its applications, particularly focusing on fraud detection. The discussion will cover foundational theories, significant methodologies, and key developments in PU learning, highlighting the relevance and impact of this research area.

### 2.1 Related Work

Positive and Unlabeled learning is a growing field within machine learning that focuses on constructing models using datasets composed primarily of positive and unlabeled examples. This approach addresses the challenge of imbalanced datasets where labeled negative examples are either scarce or entirely unavailable. Bekker and Davis (2018) [4] provide a comprehensive survey of PU learning, discussing its theoretical foundations, various methodologies, and wide-ranging applications. The concept of PU learning began to gain significant attention in the early 2000s, driven by the need to effectively manage scenarios where negative examples are not naturally produced or easily accessible.

One of the foundational methodologies in leveraging unlabeled data is the Co-Training method introduced in 1998 [3]. This framework involves training two classifiers on different views of the same dataset, where each view exploits distinct features or complementary learning tasks. The models iteratively train on the most confident inference output by the other model on the unlabeled data, thereby enhancing the accuracy of both classifiers. This iterative process significantly improves the utilization of unlabeled data, pushing it particularly valuable when labeled data is scarce.

A pivotal contribution to the field is the research conducted by Charles Elkan and Keith Noto, which tackles the challenge of developing effective classifiers in scenarios where only positive and unlabeled data are available [8]. Their methodology relies on the Selected Completely At Random (SCAR) assumption, which suggests that positive examples are randomly

chosen from the entire set of positive instances. This assumption is crucial as it facilitates the creation of a classifier that can estimate the probability of an example being positive based solely on its features and whether it is labeled or unlabeled.

Bayesian classifiers have also been developed specifically for the context of PU learning. These classifiers utilize Bayesian principles to estimate the likelihood of an instance being positive based on its features and the prior probabilities derived from the observed positive and unlabeled data [11]. By incorporating a probabilistic framework, Bayesian classifiers offer a robust method for handling the uncertainty inherent in unlabeled datasets and improving classification accuracy under PU learning conditions.

The effective classification of data in cases where solely positive and unlabeled examples are available is a recurring scenario in numerous real-world applications. Acquiring labeled negative instances in such cases is often expensive or impractical. To address this, innovative approaches such as interactive learning with policy gradient methods have been developed to enhance the learning process under these conditions. Traditionally, classifier training relies on both positive and negative labeled examples to differentiate between the two categories accurately. However, in many practical applications, a balanced dataset is not available, necessitating the use of datasets composed solely of positive examples and a mixture of unlabeled data, which may contain both positive and negative instances.

The foundational work by Elkan and Noto (2008) [8] provides a comprehensive exploration of this issue, presenting methods for training classifiers from positive and unlabeled data. Building on the foundational concepts of PU learning, Tianyu Li and Chien-Chih Wang (2020) [1] introduce an innovative policy gradient approach that dynamically integrates unlabeled data into the classifier training process. This methodology employs reinforcement learning principles where a policy network is tasked with inferring label assignments for the unlabeled data. This policy network and the classifier engage in a dynamic, iterative process: the policy network assigns labels to the unlabeled data, and the classifier, based on these assignments, predicts labels and provides feedback in the form of rewards. These rewards are used by the policy network to refine its labeling strategy continually. The goal is to maximize a reward function that carefully balances the accurate identification of positive examples with minimizing the occurrence of false positives.

Another notable advancement in PU learning is the development of non-negative risk estimators, as discussed by Kiryo et al. (2017) [9]. This approach addresses the overfitting problem commonly associated with PU learning by ensuring that the risk estimators are non-negative, thereby enhancing the stability and performance of the learning algorithms.

Furthermore, recent research by Luo et al. (2022) [10] introduces the Positive-Unlabeled Learning with Effective Negative Sample Selector (PULNS) [10] approach, which employs reinforcement learning to optimize the selection of negative instances from the unlabeled data. This method enhances the classifier's performance by iteratively updating the selector and the classifier, demonstrating significant improvements over traditional PU learning methods.

In addition to these techniques, the use of tree-based models in conjunction with neural networks has shown promise in handling tabular data, which is often encountered in real-world scenarios. Tree-based models, like decision trees and random forests, are particularly effective for tabular datasets due to their ability to capture complex interactions between features [6, 7]. Integrating these models within an interactive learning framework can further enhance the classifier's performance by iteratively refining predictions based on feedback.

The development of ensemble methods, such as the Random Forest classifier, also plays a significant role in PU learning. Ensemble methods combine multiple classifiers to improve overall performance and robustness. These methods can be particularly effective in PU learning contexts, where the integration of diverse models helps to mitigate the limitations of individual classifiers and better handle the uncertainties associated with unlabeled data.

Additionally, the application of semi-supervised learning techniques, where a small amount of labeled data is combined with a larger pool of unlabeled data, offers another promising avenue for PU learning. Semi-supervised learning leverages the structure inherent in the unlabeled data to improve classifier performance. Techniques such as self-training, where the model iteratively labels the most confident unlabeled examples and retrains on the expanded labeled set, have been shown to be effective in various PU learning scenarios.

In conclusion, the literature on PU learning reveals a diverse range of methodologies and theoretical advancements aimed at addressing the unique challenges posed by datasets with only positive and unlabeled examples. These approaches, from co-training and Bayesian classifiers to policy gradient methods, non-negative risk estimators, tree-based models, ensemble methods, and semi-supervised learning techniques, collectively contribute to the development of robust classifiers capable of operating effectively in environments where labeled data is limited. The ongoing research and innovations in this field continue to expand the applicability and efficacy of PU learning across various domains.

## 2.2 Imbalanced Datasets and Their Challenges

Imbalanced datasets are a pervasive issue in the area of machine learning, where the distribution of classes is significantly skewed. In such datasets, one class, referred to as the minority class, is underrepresented reached to the majority class. This imbalance poses substantial challenges to the development of effective machine learning models, as traditional algorithms till to be biased towards the large class, leading to suboptimal performance on the minority class. This section explores the nature of imbalanced datasets, the challenges they present, and various strategies to address these issues, with a particular focus on fraud detection.

### 2.2.1 Nature of Imbalanced Datasets

Imbalanced datasets are characterized by a disproportionate ratio of the number of instances in different classes. For instance, in a binary classification problem involving Positive and Unlabeled (PU) learning, the number of instances belonging to the positive class (minority class) might be significantly lower than those in the unlabeled class (majority class). This disparity can severely impact the learning process of machine learning models, as they are designed to minimize overall error and may achieve this by predominantly predicting the majority class. Consequently, the model's performance on the minority class, which is often the class of primary interest, suffers.

### 2.2.2 Challenges of Imbalanced Datasets

The primary challenges associated with imbalanced datasets include:

- **Model Bias:** Training of machine learning algorithms on imbalanced datasets often exhibits bias towards the majority class, resulting in high accuracy for the majority class but poor performance in predicting the minority class. This bias results in a high number of false negatives, which is particularly problematic in critical applications like fraud detection and medical diagnosis.
- **Evaluation Metrics:** Traditional evaluation metrics, such as accuracy, are inadequate for imbalanced datasets because they fail to capture the model's performance on the minority class. Metrics such as precision, recall, F1-score, and the area under the ROC curve (AUC-ROC) are more suitable for assessing models trained on imbalanced datasets.
- **Data Sparsity:** The underrepresentation of the minority class leads to sparsity in the dataset, making it hard for the model to learn meaningful patterns and generalize well to unseen data.



- **Overfitting:** Models trained on imbalanced datasets are prone to overfitting the minority class if the minority instances are treated as equally important as the majority instances without appropriate handling. This overfitting can reduce the model's ability to perform well on new, unseen data.

### 2.2.3 Fraud Detection

Fraud detection is a prime example of a domain plagued by imbalanced datasets. In the context of financial transactions, fraudulent activities (the minority class) are significantly less frequent than legitimate transactions (the majority class). Despite their rarity, fraudulent transactions can result in substantial financial losses, making their detection crucial.

#### Traditional Fraud Detection Methods

Historically, businesses relied on rule-based methods for fraud detection. These methods employed a set of predefined logical rules to flag potentially fraudulent transactions. For example, transactions exceeding a certain amount or originating from high-risk locations might be flagged as suspicious. While rule-based systems are straightforward and easy to implement, they come with several drawbacks:

- **False Positives:** Rigid rules can result in a high number of false positives, where fair transactions are wrongly flagged as fraudulent. This can lead to customer dissatisfaction and loss of revenue.
- **Fixed Outcomes:** Rule-based systems are limited and may not adjust well to evolving fraud patterns. As fraudsters develop new tactics, the predefined rules may become obsolete, reducing the system's effectiveness.
- **Scalability Issues:** Maintaining and updating a large set of rules is labor-intensive and may not scale well with increasing transaction volumes. As the complexity of fraud detection grows, the rule set must be continually expanded, increasing the system's maintenance burden.

#### Machine Learning for Fraud Detection

Machine learning presents a more sophisticated technique to fraud detection by leveraging patterns and trends in transaction data to identify fraudulent activities. However, the inherent class imbalance in fraud detection datasets poses significant challenges to traditional machine

learning methods. To address these challenges, several strategies have been developed:

- **Resampling Techniques:** Resampling involves adjusting the training dataset to achieve a balanced class distribution. This can be accomplished by oversampling the minority class, such as using the Synthetic Minority Over-sampling Technique (SMOTE), or by under-sampling the majority class. Although these methods can be effective, they may also introduce noise or result in the loss of crucial information.
- **Cost-Sensitive Learning:** Cost-sensitive learning applies different misclassification penalties to various classes, with higher penalties for misclassifying the minority class. This method encourages the model to focus more on accurately predicting instances of the minority class.
- **Anomaly Detection:** Anomaly detection methods treat fraudulent transactions as anomalies or outliers. These methods can be particularly effective in identifying rare events like fraud, as they are designed to detect deviations from normal patterns.
- **Ensemble Methods:** Ensemble methods, such as boosting and bagging, merge multiple classifiers to improve overall performance. Methods like Random Forests and Gradient Boosting Machines (GBM) are generally used in fraud detection to enhance the model's ability to capture intricate patterns in imbalanced datasets.

#### 2.2.4 Conclusion

Imbalanced datasets present significant challenges in machine learning, particularly in critical applications like fraud detection. While traditional rule-based systems have limitations, machine learning offers a robust solution by leveraging advanced techniques to handle class imbalances. By combining resampling methods, cost-sensitive learning, anomaly detection, and ensemble approaches, machine learning models can effectively detect fraud and minimize financial losses. As the landscape of fraud continues to evolve, ongoing research and innovation in handling imbalanced datasets will be crucial in developing more effective and efficient fraud detection systems [12].

## 2.3 Positive and Unlabeled Learning

Positive and Unlabeled (PU) learning is a specialized subset of machine learning focused on scenarios where the available training data consists primarily of positive examples and a large pool of unlabeled examples. This situation frequently arises in real-world applications where obtaining negative examples is difficult, expensive, or impractical. PU learning provides a robust framework for building classifiers in these challenging environments.

For example, in medical diagnosis, the presence of a disease (positive class) is often confirmed through tests, while the absence of the disease (negative class) is not typically documented. Similarly, in personalized advertising, user interactions like clicks or purchases are positive examples, while non-interactions are unlabeled. These scenarios necessitate methods that can effectively leverage the abundant unlabeled data to improve classification performance.

### 2.3.1 Challenges in PU Learning

PU learning presents unique challenges that necessitate specialized approaches:

- **Class Imbalance:** The inherent inequality between the positive and unlabeled classes can bias the learning process. Traditional algorithms may be overwhelmed by the majority unlabeled class, resulting in inferior performance on the minority positive class.
- **Label Noise:** Unlabeled data may contain a mix of positive and negative instances, introducing noise that complicates the learning process. Accurately distinguishing between these hidden classes is critical for effective model performance.
- **Lack of Negative Examples:** The absence of explicitly labeled negative examples poses a significant challenge, as the model must infer the negative class indirectly. This requires innovative strategies to specify reliable negative samples from the unlabeled pool.

### 2.3.2 Strategies for PU Learning

Several strategies have been developed to address the challenges inherent in PU learning:

#### Two-Step Approaches

Two-step approaches are common in PU learning. The first step involves identifying a subset of dedicated negative samples from the unlabeled data. This can be achieved through techniques such as clustering, heuristic rules, or statistical methods. The second step uses

these identified negatives along with the labeled positives to train a traditional classifier. While effective, this method relies heavily on the accuracy of the initial negative identification process.

### **Cost-Sensitive Learning**

Cost-sensitive learning assigns distinct miss-classification costs to positive and unlabeled classes. By penalizing miss-classifications of the positive class more heavily, the model can be guided to focus more on correctly identifying positive instances. This approach is particularly useful in balancing the impact of the majority unlabeled class.

### **PU Bagging**

PU Bagging is an ensemble method adapted for PU learning. It involves creating multiple training sets by randomly sampling the unlabeled data and treating each subset as negative in turn. Multiple classifiers are trained on these varied datasets, and their outputs are aggregated to make final predictions. This technique helps in reducing the bias towards the unlabeled class and improves generalization.

### **Positive-Unlabeled (PU) Learning with Policy Gradient**

Reinforcement learning techniques have been adapted for PU learning, as demonstrated by Tianyu Li et al. (2020) [1]. Their method involves a policy network that iteratively adjusts its assumptions about the labels of unlabeled data based on rewards received from a classifier. This dynamic interaction helps in better exploiting the unlabeled data and improving classification performance. The policy network generates actions that label the unlabeled data, which are then used to train the classifier. The classifier's performance provides feedback to the policy network, forming a continuous learning loop that refines both components.

### **2.3.3 Conclusion**

Positive and Unlabeled learning describes a crucial advancement in the field of machine learning, addressing the limitations posed by imbalanced and incomplete datasets. By employing innovative strategies such as two-step approaches, cost-sensitive learning, and reinforcement learning, PU learning models can effectively use unlabeled data to improve their performance. As research in this area continues to evolve, PU learning is poised to play a pivotal role in diverse applications, from medical diagnosis to fraud detection, driving significant improvements in predictive accuracy and operational efficiency.

## 2.4 Co-training and Semi-Supervised Learning

Co-training and semi-supervised learning are powerful paradigms within machine learning designed to leverage both labeled and unlabeled data. These techniques are particularly effective when getting labeled data is expensive or time-taking, while unlabeled data is readily known.

### 2.4.1 Co-training

Co-training, introduced by Blum and Mitchell in 1998 [3], is a semi-supervised learning technique that involves training two classifiers on two distinct views of the data. The core idea is to iteratively train these classifiers to label the unlabeled data, which is then used to enhance each other's training set.

#### Process

1. **Initialization:** Two classifiers are initially trained on separate labeled datasets derived from different views of the same data.
2. **Labeling Unlabeled Data:** Each classifier labels a portion of the unlabeled data, selecting the examples it is most confident about.
3. **Mutual Training:** The confidently labeled examples from one classifier are added to the training set of the other classifier.
4. **Iteration:** Steps 2 and 3 are repeated iteratively, allowing each classifier to benefit from the other's labeling, progressively improving their accuracy.

#### Advantages

- **Exploiting Multiple Views:** Co-training is effective when the data can be divided into two distinct, conditionally independent views given the class label.
- **Increased Training Data:** By employing unlabeled data, co-training increases the amount of data available for training, which can improve model performance.

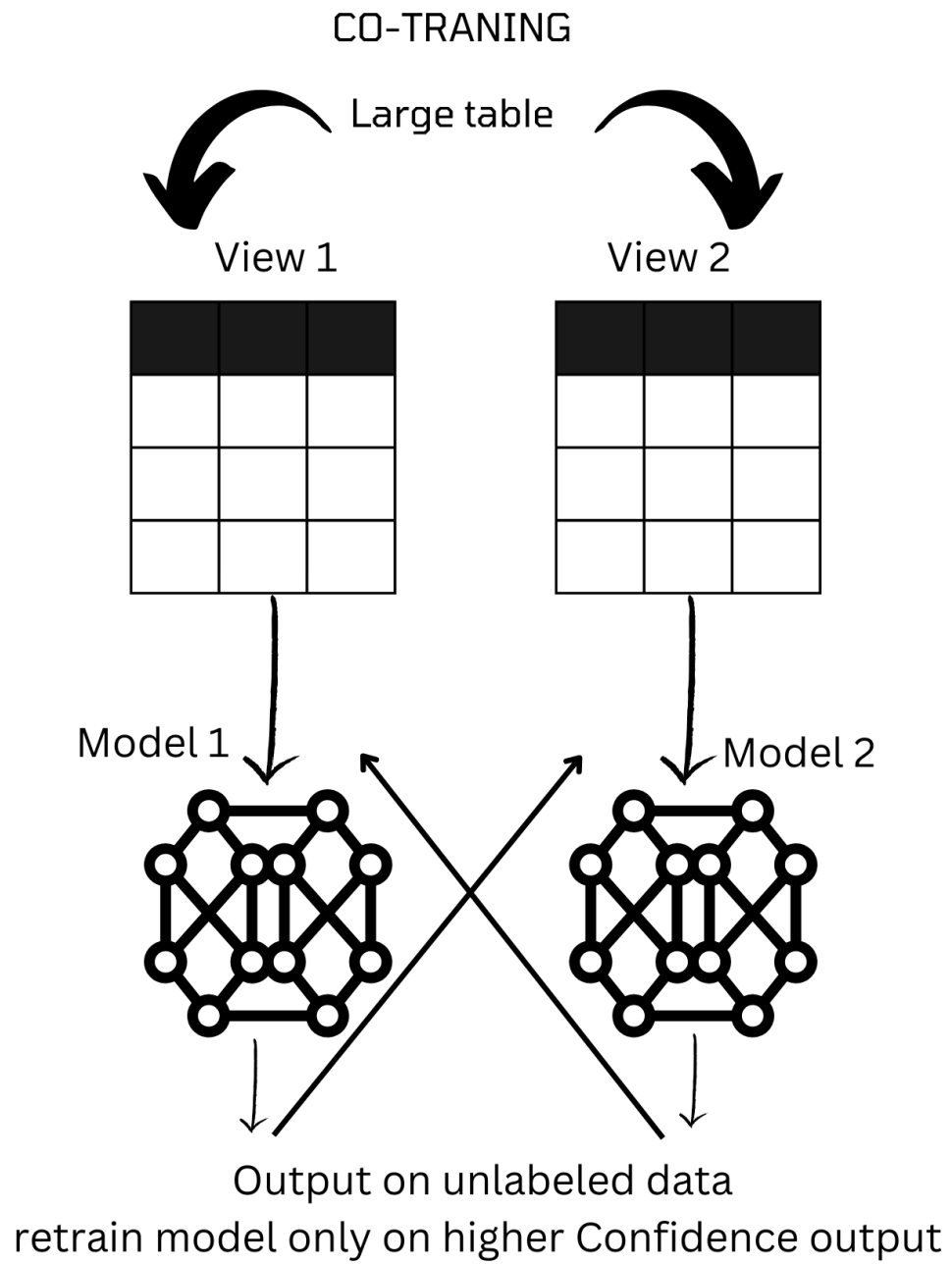


Figure 2.1: Pictorial representation of CO-training

### 2.4.2 Semi-Supervised Learning

Semi-supervised learning (SSL) is an extensive framework that integrates a small portion of labeled data with a larger portion of unlabeled data during the training process. The aim is to enhance learning accuracy by leveraging the unlabeled data.

#### Techniques

- **Self-Training:** To predict labels for the unlabeled data, the model is first trained using labeled data. The model is retrained using this larger dataset after the predictions with the greatest confidence are added to the training set.
- **Graph-Based Methods:** In a graph, data points are represented as nodes, with edges denoting similarities between them. Label information spreads through the graph, leveraging its structure to infer labels for the unlabeled nodes.
- **Consistency Regularization:** Models are trained to produce consistent predictions for unlabeled data under various perturbations, enhancing robustness and leveraging unlabeled data effectively.

#### Advantages

- **Efficiency:** SSL leverages the abundance of unlabeled data, making it more efficient in terms of data utilization.
- **Cost-Effective:** Reduces the need for extensive labeled datasets, lowering the cost and time involved in data annotation.

#### Applications

- **Natural Language Processing (NLP):** SSL is widely used in NLP tasks like language modeling, where vast amounts of text data are available but labeled data is scarce.
- **Image Recognition:** In image recognition, SSL helps improve performance by using large collections of unlabeled images along with a smaller labeled dataset.

## 2.5 Bayesian Approaches to PU Learning

Bayesian methods offer a probabilistic framework for handling uncertainty in machine learning, making them particularly suitable for Positive and Unlabeled (PU) learning scenarios.

These methods apply Bayes' theorem to revise the probability estimate for a hypothesis as new evidence or data is introduced.

### 2.5.1 Bayesian Classifiers in PU Learning

In PU learning, Bayesian classifiers function by estimating the likelihood that an example is positive based on its features and prior knowledge. The main elements of Bayesian approaches in PU learning include:

- **Prior Probability:** The initial probability estimate of the classes before observing the data.
- **Likelihood:** The probability of the observed data given the class labels.
- **Posterior Probability:** The updated probability estimate of the classes after observing the data, calculated using Bayes' theorem.

#### Process

1. **Model Initialization:** Define prior probabilities for the positive and unlabeled classes based on domain knowledge or initial data analysis.
2. **Evidence Incorporation:** Collect evidence from the data to calculate the likelihood of each class.
3. **Posterior Calculation:** Use Bayes' theorem to update the posterior probabilities of the classes.

#### Techniques

- **Naive Bayes:** It assumes that features are conditionally independent given the class label, which simplifies the calculation of likelihoods.
- **Bayesian Networks:** Represents dependencies among variables using a directed acyclic graph, allowing for more complex interactions between features.
- **(MCMC) Markov Chain Monte Carlo:** Is a technique used for sampling from the posterior distribution when direct computation is not feasible.

#### Advantages

- **Probabilistic Interpretation:** Bayesian methods provide a clear probabilistic interpretation of classification results, which is useful for decision-making under uncertainty.
- **Flexibility:** Can incorporate prior knowledge and update beliefs as more data becomes available, making them adaptable to changing data distributions.



### Applications

- **Medical Diagnosis:** Bayesian methods are used to identify disease genes from positive and unlabeled data in biomedical research, leveraging prior knowledge about gene functions and interactions [5].
- **Text Classification:** Effective in spam detection and sentiment analysis, where labeled data is limited but a large corpus of unlabeled text is available.

## 2.6 Policy Gradient Methods in PU Learning

Policy gradient methods, rooted in reinforcement learning, have recently been adapted for Positive and Unlabeled (PU) learning to address the challenges of class inequality and label uncertainty. These methods involve optimizing a policy network that guides the labeling process for unlabeled data.

### 2.6.1 Core Concepts

- **Policy Network:** A neural network that outputs actions (label assignments) established on the input features of unlabeled data.
- **Reward Function:** Measures the quality of the policy's actions, guiding the optimization process to improve label assignments.

### 2.6.2 Methodology

1. **Initialization:** Begin with a policy network and classifier. The policy network assigns either soft or hard labels to the unlabeled data.
2. **Interaction:** The classifier is trained using both the labeled data and the outputs from the policy network. The classifier's performance provides rewards to the policy network.
3. **Optimization:** The policy network is updated using policy gradient methods to maximize the expected reward, refining its labeling strategy iteratively.

### 2.6.3 Advantages

- **Dynamic Learning:** The policy network adapts to new data and evolving patterns, improving the classifier's performance over time.
- **Exploitation of Unlabeled Data:** By treating unlabeled data as a dynamic component, policy gradient methods can effectively leverage large amounts of unlabeled data.

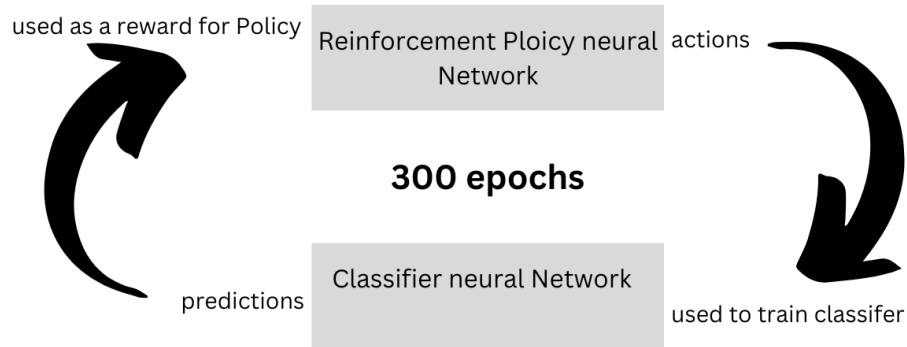


Figure 2.2: Policy gradient interactive learning process. Inspired by the work in [1]

#### 2.6.4 Applications

- **Fraud Detection:** These methods are used to dynamically identify fraudulent transactions from a mix of positive and unlabeled data, improving detection accuracy [1].
- **Personalized Advertising:** Helps in predicting user behavior and preferences by adaptively labeling user interactions as positive or negative.

## 2.7 Differentiating Techniques for Interactive Learning

Interactive learning involves continuous feedback loops between the model and the environment, enhancing learning efficiency and adaptability. In the context of PU learning, several interactive learning techniques are employed to improve model performance.

### 2.7.1 Active Learning

Active learning involves the model selectively querying the most informative examples for labeling. This technique is particularly useful in PU learning to identify the most ambiguous or uncertain examples from the unlabeled pool.

1. **Query Strategy:** Selects instances where the model's confidence is lowest.
2. **Label Acquisition:** These instances are labeled by an oracle, such as a human annotator, and incorporated into the training set.

3. **Model Update:** The model is retrained with the newly labeled data, improving its performance iteratively.

### 2.7.2 Reinforcement Learning

Reinforcement learning techniques, such as policy gradient methods, are used to dynamically adapt the model's strategy for labeling unlabeled data based on feedback from the classifier's performance.

### 2.7.3 Semi-Supervised Learning

Semi-supervised learning integrates a limited amount of labeled data with a large amount of unlabeled data. Techniques like co-training, self-training, and graph-based methods fall under this category, each leveraging different aspects of the data to enhance learning.

### 2.7.4 Human-in-the-Loop

Integrating human feedback into the learning process refines the model's predictions. Human specialists can inspect and correct the model's outputs, providing high-quality labels that enhance the model's accuracy and robustness.

## 2.8 Limitations of SVM for PU Learning

Support Vector Machines (SVMs) are widely utilized for classification tasks but face several limitations when applied to Positive and Unlabeled (PU) learning scenarios.

### 2.8.1 Challenges

#### Class Imbalance

SVMs are exposed to class imbalance, models often resulting in bias towards the majority class. In PU learning, where the majority of data is unlabeled, SVMs struggle to effectively identify the minority positive class.

#### Non-Probabilistic Output

SVMs do not inherently provide probabilistic outputs, which are crucial for estimating the likelihood of class membership in PU learning. This limitation hinders the application of SVMs in scenarios requiring probabilistic interpretation of predictions, such as identifying positive instances among the unlabeled data.

### Limited Adaptability

SVMs are less adaptable to evolving data distributions compared to methods like neural networks or ensemble models. In PU learning, where the characteristics of unlabeled data can change, SVMs struggle to maintain their performance over time.

### 2.8.2 Alternatives to SVM in PU Learning

To overcome these limitations, alternative approaches are preferred in PU learning:

- **Decision Trees and Random Forests:** These methods offer better handling of imbalanced data and provide probabilistic interpretations of classifications.
- **Neural Networks:** Qualified of learning complicated patterns and adjusting to dynamic data distributions, these models are well-suited for PU learning..
- **Cost-Sensitive Learning:** Adjusts the training process to account for class imbalance, improving performance on the minority positive class.

### 2.8.3 Conclusion

While SVMs are powerful tools for many classification tasks, their limitations in handling PU learning scenarios necessitate the use of alternative methods. Decision trees, neural networks, and cost-sensitive learning approaches provide more robust solutions for leveraging unlabeled data and addressing class imbalance in PU learning applications.

## 3 Methodology

In this chapter, we provide an exhaustive overview of the methods and procedures employed to investigate the Positive and Unlabeled (PU) learning effectiveness using the Tree-based algorithm with reinforcement learning. This chapter serves as a detailed guide for the research process undertaken in this study.

### 3.1 Data Collection and Preprocessing

#### 3.1.1 Datasets

For this study, we have selected two primary datasets: the MNIST [2] dataset and a credit card transaction [13] dataset from Kaggle. The MNIST dataset is used as a benchmark dataset, commonly employed in previous studies for Positive and Unlabeled (PU) learning. The credit card transaction dataset (<https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>) serves as our real-world testing dataset, providing a practical application context for fraud detection.

#### 3.1.2 MNIST Dataset Preprocessing

The MNIST dataset comprises handwritten digit images, originally a balanced multiclass classification problem. To align it with our PU learning framework, we performed the following preprocessing steps:

#### 3.1.3 Construction of Imbalanced Dataset from Balanced Dataset

To simulate a PU learning environment, the original MNIST dataset, which consists of ten classes of digits (0-9), was converted into a binary classification problem. The even digits (0, 2, 4, 6, 8) were labeled as 0 (unlabeled), and the odd digits (1, 3, 5, 7, 9) were labeled as 1 (positive). This conversion was necessary to create a scenario where the minority class (positive instances) is significantly underrepresented.

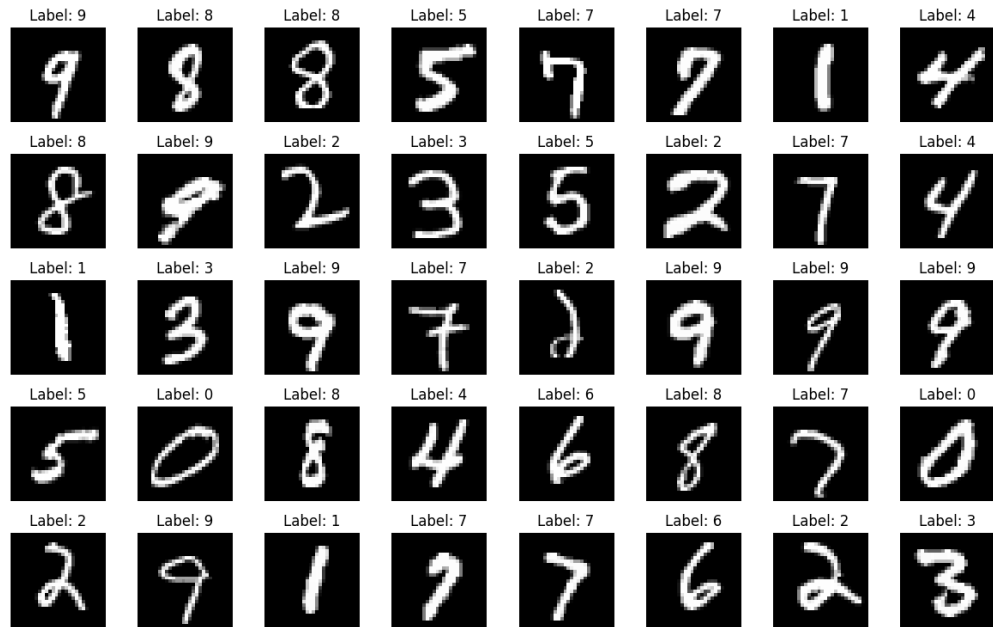


Figure 3.1: Labels in the original MNIST dataset [2].

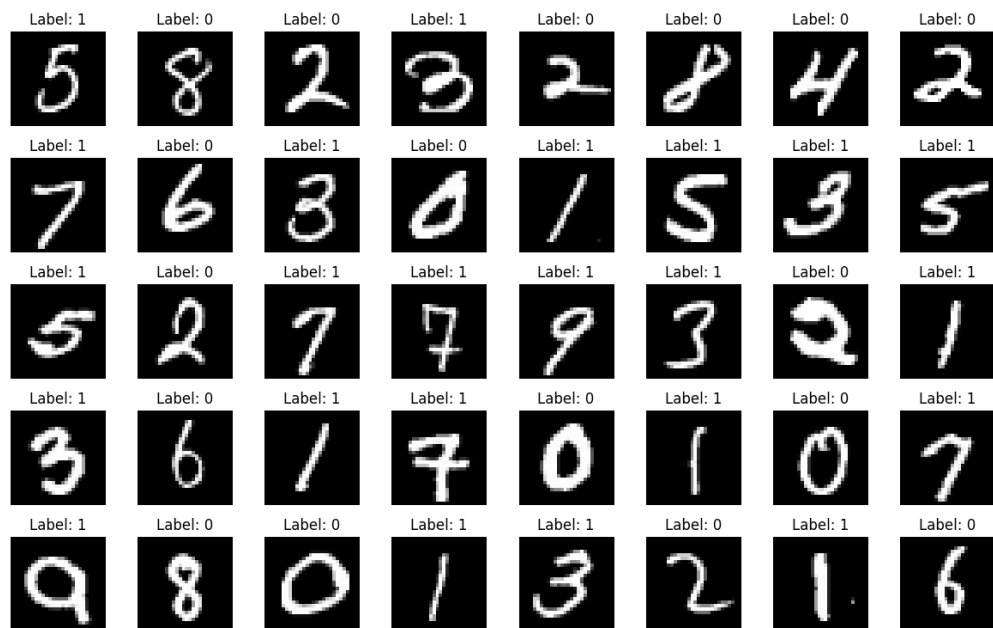


Figure 3.2: Sample of MNIST images labeled as 0 and 1 after preprocessing.

### 3.1.4 Label Distribution Before and After Preprocessing

After converting the labels to 0s and 1s, the dataset was further processed to hide some positive instances within the unlabeled data. This step mimics the real-world scenario of PU learning, where the unlabeled collection may contain both positive and negative instances. The label distribution before and after preprocessing is shown in Figures 3.3 and 3.4.

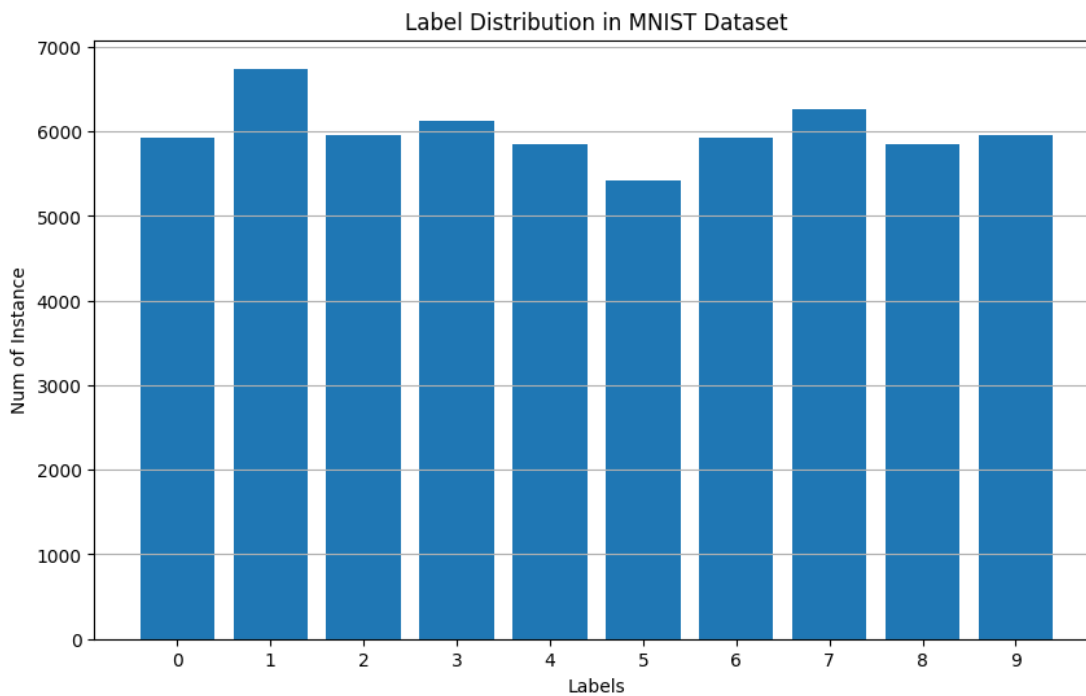


Figure 3.3: Label Distribution in MNIST Dataset Before Preprocessing

PU dataset distribution.

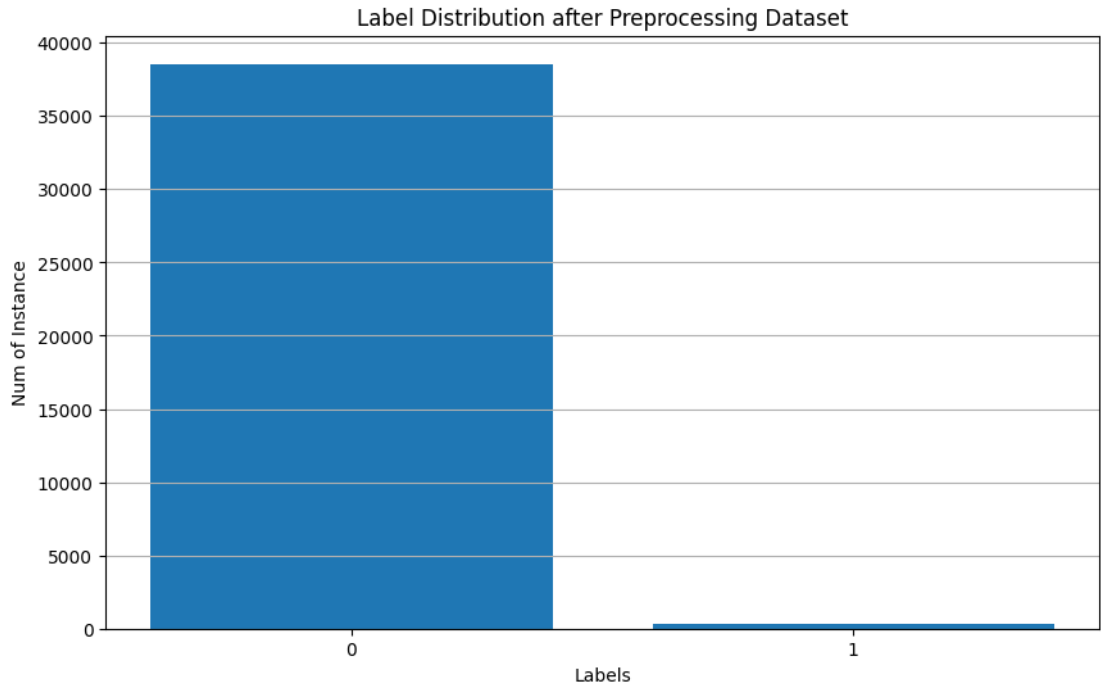


Figure 3.4: Label Distribution in MNIST Dataset After Preprocessing

### 3.1.5 Preprocessing steps

The following steps are being done respectively to achieve the mimic PU dataset.

- **Construction of Imbalanced Dataset:** We transformed the balanced MNIST dataset into an imbalanced dataset by converting it into a binary classification problem. Even digits were labeled as 0 (Unlabeled), and odd digits were labeled as 1 (Positive).
- **Conversion of Labels:** Labels were converted to 0s and 1s to facilitate binary classification.
- **Hiding Positive Instances:** To simulate a realistic PU learning scenario, a subset of positive instances (odd digits) was randomly selected and marked as 0 (Unlabeled).



### 3.1.6 Credit Card Transaction Dataset Preprocessing

For the credit card transaction dataset, the preprocessing involved the following steps:

- **Handling Missing Values:** Any absent values in the dataset were handled using proper imputation techniques.
- **Normalization:** Features were normalized to guarantee that contribution from all variables is equal to the model's learning process.
- **Label Conversion:** Similar to the MNIST dataset, labels were converted to binary, with fraudulent transactions labeled as 1 (Positive) and legitimate transactions labeled as 0 (Unlabeled).

## 3.2 Reinforcement Learning in PU Learning

In the field of machine learning known as reinforcement learning (RL), an agent gains decision-making skills by acting and then getting feedback in the form of rewards or penalties. Reinforcement learning may be applied to PU learning to iteratively improve the classifier by dynamically modifying how unlabeled input is treated. This dynamic adjustment is crucial in dealing with the inherent uncertainty and imbalance of PU datasets.

### 3.2.1 Policy Network

The policy network in our framework is designed using 5-layer Dense Neural Network layers with ReLU activation. This network takes actions (predictions) on a batch of 128 instances from the dataset. The actions are used to train a Tree-based algorithm, and the outcomes from the Tree-based algorithm are fed back into the policy network as rewards.

### 3.2.2 Reward Function

The reward function is critical in reinforcement learning. For our PU learning scenario, we designed a reward function as follows:

- If  $y = 1$  (Positive), then  $y_{\hat{}}$  (predicted outcome) is the reward.
- If  $y = 0$  (Unlabeled) and  $y_{\hat{}} \geq \text{threshold}$ , then  $y_{\hat{}}$  is the reward.
- If  $y = 0$  (Unlabeled) and  $y_{\hat{}} < \text{threshold}$ , then  $y_{\hat{}} - 1$  is the reward.

This approach ensures that positive outcomes are rewarded, while incorrect predictions receive negative rewards, guiding the policy network towards better performance.

#### Reward function Code:

```
def calculate_rewards(lgb_probabilities, y_batch, threshold):
    rewards = []
    for prob, actual in zip(lgb_probabilities.flatten(), y_batch):
        if actual == 1 or (actual == 0 and prob >= threshold):
            reward = prob
        else:
            reward = prob - 1
        rewards.append(reward)
    return rewards
```

### 3.2.3 Calculation of the Threshold

The concept of the threshold is pivotal in our reward function, particularly when dealing with unlabeled data. Calculating this threshold dynamically during each reward computation allows for adaptive learning based on the current data distribution and model performance. The threshold calculation is performed using a function named `calculate_threshold()`, which processes the classifier or Gradient Boosting Machine (GMB) probabilities. Here's an in-depth explanation of the threshold calculation:

1. **Classifier Probabilities Input:** The `calculate_threshold()` function accepts the probabilities output by the classifier or GMB. These probabilities represent the model's confidence in its predictions.
2. **Identification of Positive Instances:** The function first isolates the instances where the actual label  $y = 1$  (Positive). This step is crucial as it focuses on the subset of data that the model has identified with certainty.

3. **Minimum Probability (Min Threshold):** Among these positive instances, the function identifies the minimum probability value. This minimum probability is termed as *min\_threshold*. It serves as a baseline, ensuring that any probability lower than this is considered less reliable.
4. **Mean Probability for Unlabeled Data:** Next, the function considers the instances where the actual label  $y = 0$  (Unlabeled). It calculates the mean of the probabilities associated with these unlabeled instances. This average probability reflects the model's overall confidence level for the unlabeled data.
5. **Setting the Threshold:** The threshold is dynamically set based on these calculations. The mean of the unlabeled instances probabilities then used as a threshold. The use of both the minimum probability of positive instances and the mean probability of unlabeled instances ensures a balanced approach. It accommodates the variability in the data and adjusts the threshold to optimize the model's performance continuously.

**Threshold function Code:**

```
def calculate_threshold(clf_probabilities , y_batch):
    positive_indices = (y_batch == 1)
    if np.any(positive_indices):
        threshmin = np.min(clf_probabilities[positive_indices])
    else:
        threshmin = 0

    U0_indices = (clf_probabilities >= threshmin)
    if np.any(U0_indices):
        threshold = np.mean(clf_probabilities[U0_indices])
    else:
        threshold = threshmin

    return threshold
```

The dynamic nature of this threshold calculation allows our reinforcement learning framework to adapt in real-time, fostering a robust learning process. By incorporating both the *min\_threshold* and the average probability, our model can distinguish between high-confidence predictions and those that require more cautious consideration. This method not only improves the precision of positive forecasts but also mitigates the risk of false positives, thereby refining the overall performance of the PU learning scenario.

### 3.3 Tree-Based Algorithm in PU Learning

Tree-based algorithms are powerful tools in machine learning, particularly effective for classification and regression tasks on tabular data. In our interactive learning framework, we utilize LightGBM, a highly efficient and scalable gradient boosting framework, as the Tree-based algorithm.

#### 3.3.1 LightGBM

Tree-based learning methods are used in the gradient boosting framework LightGBM (Light Gradient Boosting Machine). Because of its efficiency and scalability, it can perform well while managing massive amounts of data. LightGBM is particularly suitable for our PU learning framework due to its following features:

- **Speed and Efficiency:** LightGBM is optimized for both memory usage and computational speed, making it ideal for large datasets.
- **Accuracy:** It can achieve high accuracy by leveraging gradient boosting, which builds a sequence of trees where individually tree corrects the errors of the previous one.
- **Scalability:** LightGBM can handle large datasets and supports parallel learning, improving scalability and efficiency.
- **Support for Sparse Data:** It efficiently takes sparse data, which is typical in real-world scenarios like fraud detection.

### 3.4 Proposed Interactive Learning Framework

Our proposed framework integrates reinforcement learning with traditional machine learning Tree-base algorithms to enhance the effectiveness of PU learning.

#### 3.4.1 Pre-Training Phase

Both the policy network (Dense layers with ReLU activation) and the Tree-based algorithm undergo a pre-training phase:

- **Tree-Based Algorithm:** Pre-trained for 15 epochs on positive and negative (unlabeled treated as negative) datasets.
- **Policy Neural Network:** Pre-trained for 15 epochs on the predictive outcomes of the Tree-based algorithm.

### 3.4.2 Interactive Training Process

The interactive training process involves alternating between the policy network and the Tree-based algorithm:

- The policy network predicts actions on a batch of 128 instances.
- These actions are used to train the Tree-based algorithm.
- Predictions from the Tree-based algorithm are fed back to the policy network as rewards.
- The policy network is updated every three epochs.

## 3.5 Model Selection and Justification

The choice of models is crucial for the success of the proposed framework. We selected a tree-based algorithm and a neural network with Dense layers for the following reasons:

### 3.5.1 Tree-Based Algorithm

Algorithms such as Random Forests and Gradient Boosting Machines (GBM) are tree-based and well-suited for handling tabular data. They are capable of capturing complex interactions between features and are robust to overfitting when properly tuned.

### 3.5.2 Dense Neural Network

Dense neural networks, specifically those using ReLU activation functions, are highly effective in recognizing patterns and structures in data. In our framework, the policy network is composed of 5-layer Dense layers, chosen for their ability to learn from the predictive outcomes of the tree-based algorithm and refine its actions iteratively.

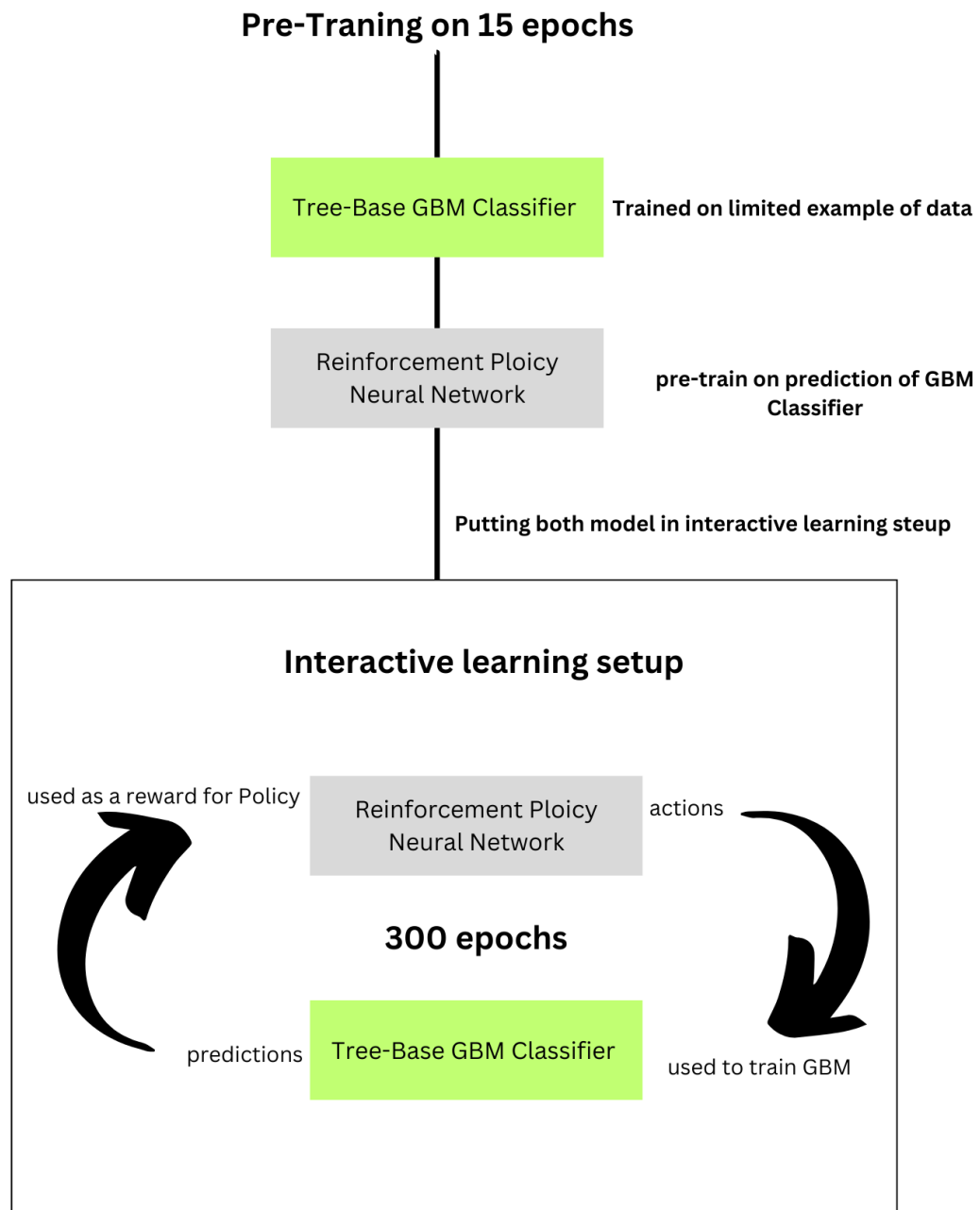


Figure 3.5: Proposed framework for PU learning, first we trained GBM classifier on available positive and negative balanced examples, then Policy Network trained on prediction of GBM Classifier, then add both models in interactive learning setup where training each on unlabeled dataset by their prediction.

### 3.5.3 Why Tree-Based Algorithm over Neural Network for classification

In previous research, neural networks have primarily been used for PU learning due to their strong performance in various domains. However, for tabular datasets, tree-based algorithms have shown superior performance compared to neural networks. This observation is supported by several studies indicating that tree-based models, such as LightGBM, often outperform neural networks in scenarios involving structured data. This potentially give me a direction to test the effectiveness of a tree-base algorithm on tabular dataset in an interactive learning setup.

## 3.6 Evaluation Metrics

Evaluating the performance of models trained on imbalanced datasets requires appropriate metrics that reflect the model's effectiveness on the minority class.

### 3.6.1 Precision and Recall

Precision and recall are crucial metrics for assessing model performance, particularly in imbalanced datasets such as those encountered in fraud detection.

**Precision** The ratio of true positive predictions to all positive predictions is measured by Precision. Precision in fraud detection refers to the proportion of accurately detected fraudulent transactions among all transactions flagged as fraudulent. High accuracy is important because it guarantees that a transaction is probably correct when the model flags it as fraudulent. By doing this, the frequency of false positives—when valid transactions are inadvertently reported as fraudulent—is decreased. Because valid transactions may be unfairly denied as a result of false positives, there may be consumer unhappiness and possible revenue loss. As a result, a very precise model minimizes the inconvenience to real customers while accurately recognizing fraudulent transactions.

**Recall**, in contrast, measures the ratio of true positive predictions to all actual positives. In fraud detection, recall indicates the number of correctly identified fraudulent transactions out of the total actual fraudulent transactions. High recall is essential because it ensures that the model can identify as many fraudulent transactions as possible, thus reducing the risk of undetected fraud. Failing to identify fraudulent transactions (false negatives) can result in considerable financial losses and undermine the effectiveness of the fraud detection system. Therefore, a model with high recall is sensitive to capturing all instances of fraud, even if it means flagging more transactions that require further verification.

Balancing precision and recall is critical in fraud detection. While high precision minimizes the inconvenience to legitimate customers, high recall ensures that the model does not miss any fraudulent activity. The F1-score, which is calculated as the harmonic mean of precision

and recall, is often used to find an optimal balance between these two metrics, ensuring that the fraud detection system is both accurate and comprehensive.

### 3.6.2 F1-Score

As the harmonic mean of recall and accuracy, the F1-score provides a fair measure that takes care of both issues. It is particularly helpful in situations where the distribution of classes is wildly unbalanced.

### 3.6.3 Area Under the ROC Curve (AUC-ROC)

The model's ability to distinguish between positive and negative classes at different threshold settings is assessed using the AUC-ROC measure. Better performance is indicated by a higher AUC-ROC value.

### 3.6.4 Area Under the Precision-Recall Curve (AUC-PR)

The AUC-PR metric focuses on the performance of the model concerning precision and recall, offering a more informative view in the context of imbalanced datasets.

### 3.6.5 Implementation

Calculating these metrics at each epoch during training helps in tracking the model's performance and making necessary adjustments.

## 3.7 Conclusion

The methodology presented in this chapter outlines a comprehensive approach to tackling the challenges of PU learning through the integration of reinforcement learning and traditional machine learning algorithms. By leveraging the strengths of both approaches, our framework aims to improve the detection of minority class instances in imbalanced datasets, with a specific application to fraud detection.



## 4 Experimental Setup

### 4.1 Hardware and Software Environment

The hardware and software environment for running the proposed PU learning framework, integrating both Tree-based and Policy Gradient algorithms, is essential to ensure efficient and effective performance. Below are the key aspects of the hardware and software setup used in this research:

#### 4.1.1 Hardware

- **CPU:** Intel remote Windows system with a high-performance processor to handle extensive computational tasks efficiently.
- **RAM:** 1 TB RAM, which provides ample memory for handling large datasets and complex model computations without running into memory bottlenecks.
- **Storage:** 2 TB of hard disk space, ensuring sufficient storage for datasets, intermediate results, and model checkpoints.
- **GPU:** NVIDIA TESLA T4 with 8 GB GPU memory. The GPU significantly accelerates the training process of deep learning models, making it suitable for handling the computationally intensive Policy Gradient algorithm.

#### 4.1.2 Software

The software environment is configured to support the execution of machine learning models and data preprocessing tasks efficiently. The following are the key software components and libraries used:

- **Operating System:** Windows 10
- **Coding Environment:** Visual Studio Code
- **Programming Language:** Python 3.9

- **Dependencies and Libraries:**

- **Conda:** A cross-platform, open-source package management and environment control system for Windows, macOS, and Linux. It facilitates the easy installation of libraries and management of dependencies.
- **TensorFlow:** A popular open-source library for numerical computation and machine learning, used particularly for implementing and training the Policy Gradient neural network.
- **Numpy:** A fundamental package for scientific computing with Python, used for handling numerical operations and arrays.
- **Pandas:** A powerful data manipulation library for Python, essential for handling structured data and preprocessing tasks.
- **nvidia-smi:** A command-line utility, shipped with the NVIDIA GPU driver, used for monitoring and managing the GPU state.
- **Cuda:** An API concept and parallel computing platform created by NVIDIA that lets programmers use GPUs with CUDA support for general-purpose processing jobs.
- **cuDNN:** A GPU-accelerated library for deep neural networks, offering optimized implementations for common operations such as forward and backward convolution, pooling, normalization, and activation layers.
- **sklearn:** For computing assessment metrics, a Python machine learning package that provides simple and effective tools for data mining and analysis is specifically used.
- **matplotlib:** For the Python programming language, a charting package that is connected with NumPy, the numerical extension, is utilized to visualize performance measures such as accuracy.
- **lightgbm:** A rapid, distributed, high-efficiency gradient boosting framework based on decision tree algorithms, utilized for implementing the tree-based model.

## 4.2 Model Configuration

### 4.2.1 Tree-Based Algorithm

The Tree-based algorithm is implemented using LightGBM, known for its efficiency and scalability. The following parameters were configured for optimal performance:

- **boosting\_type:** 'gbdt' (Gradient Boosting Decision Tree) - This specifies the boosting type to be used.
- **objective:** 'binary' (Binary classification) - This sets the objective function to binary classification, suitable for PU learning.
- **metric:** 'binary\_logloss' (Evaluation metric) - This defines the evaluation metric used for evaluating the performance of the model.
- **num\_leaves:** 31 (Number of leaves in full tree) - This parameter prevents the complexity of the individual trees.
- **learning\_rate:** 0.0001 (Learning rate) - This parameter defines the step size at each iteration during the optimization process, guiding the model toward minimizing the loss function.
- **feature\_fraction:** 0.9 (Fraction of features to be used at each iteration) - This parameter is used to specify the fraction of features to be randomly selected at each iteration of tree building.
- **bagging\_fraction:** 0.8 (Fraction of data to be utilized for each iteration) - This defines the fraction of data to be utilized for each boosting round.
- **bagging\_freq:** 5 (Frequency for bagging) - This specifies the frequency for performing bagging.
- **device:** 'gpu' (GPU acceleration) - This enables the usage of GPU for training, accelerating the process.
- **verbose:** 1 (Verbosity level) - This sets the verbosity level of the LightGBM output.

### 4.2.2 Reinforcement Learning Algorithm

The reinforcement learning component of the framework is based on a Policy Gradient method implemented using TensorFlow. The policy network is configured as a Dense Neural Network with the following setup:

- **Optimizer:** Adam - An optimization algorithm capable of managing sparse gradients in noisy problems.
- **Learning Rate:** 0.00001 - This parameter sets the learning rate for the optimizer, dictating the extent to which the model is adjusted in response to the estimated error with each update to the model weights.
- **Loss Function:** 'binary\_crossentropy' - This specifies the loss function used for binary classification problems.
- **Activation Function:** Sigmoid - This activation function is used in the output layer to map predictions to probabilities.
- **Weight Decay:** 0.5 - This is used to penalize large weights and thus control overfitting.

## 4.3 Conclusion

The experimental setup detailed in this chapter outlines the comprehensive framework utilized to evaluate the proposed Positive and Unlabeled (PU) learning methodology. By leveraging a high-performance hardware and software environment, we ensure that our experiments are conducted efficiently and produce reliable results. This setup provides a solid foundation for implementing and testing the proposed interactive learning framework, which integrates reinforcement learning with traditional tree-based machine learning algorithms. In the subsequent chapters, we will dive into the experimental results and discuss the implications of our findings.

## 5 Results

In this chapter, we detail the results from a series of experiments designed to assess the effectiveness of tree-based algorithms within an interactive learning framework. Specifically, we utilized a policy-gradient reinforcement learning algorithm (Policy Network) combined with different classifiers: a Neural Network Classifier and a Tree-Based Classifier (GBM, implemented using LightGBM). Our goal was to assess the performance of these models on both image and tabular datasets, represented by the MNIST Image Dataset and the Credit Card Fraud Detection Dataset, respectively.

### 5.0.1 Experimental Setting

We performed four distinct experiments to evaluate the performance of the classifiers in combination with the Policy Network:

- Policy Network with Neural Network Classifier on the MNIST Image Dataset.
- Policy Network with Neural Network Classifier on the Credit Card Fraud Detection Dataset.
- Policy Network with Tree-Based Classifier GBM on the MNIST Image Dataset.
- Policy Network with Tree-Based Classifier GBM on the Credit Card Fraud Detection Dataset.

For each experiment, we measured the following metrics to gauge model performance: ROC-AUC, Accuracy, and PR-AUC. The results are summarized in Table 5.1.

Interactive Model	MNIST Image Dataset			Credit Card Tabular		
	ROC_AUC	Accuracy	PR_AUC	ROC_AUC	Accuracy	PR_AUC
<b>Policy Network with Neural Network Classifier</b>	0.982	0.961	0.983	0.717	0.670	0.70
<b>Policy Network with Tree Based Classifier GBM</b>	0.975	0.868	0.981	<b>0.996</b>	<b>0.950</b>	<b>0.997</b>

Figure 5.1: Performance metrics for different interactive models on MNIST Image Dataset and Credit Card Tabular Dataset

### 5.0.2 Performance Analysis on MNIST Image Dataset

The Policy Network combined with the Neural Network Classifier performed exceptionally well on the MNIST Image Dataset, achieving a ROC-AUC of 0.982, an accuracy of 0.961, and a PR-AUC of 0.983. These results indicate that the model was able to effectively learn and distinguish between different digit classes.

When the Policy Network was paired with the Tree-Based Classifier GBM, it also performed robustly, though slightly lower than the Neural Network Classifier in terms of ROC-AUC (0.975) and PR-AUC (0.981). The accuracy, however, was notably lower at 0.868. This difference in performance highlights the strengths of neural networks in image classification tasks, possibly due to their capability to capture complex patterns and spatial hierarchies inherent in image data.

### 5.0.3 Performance Analysis on Credit Card Tabular Dataset

The results for the Credit Card Fraud Detection Dataset reveal a different trend. The Policy Network with the Neural Network Classifier achieved a ROC-AUC of 0.717, an accuracy of 0.670, and a PR-AUC of 0.70. These metrics suggest that the Neural Network Classifier struggled to handle the tabular nature of the dataset, resulting in suboptimal performance.

In contrast, the Policy Network with the Tree-Based Classifier GBM significantly outperformed on this credit card tabular dataset, achieving a ROC-AUC of 0.996, an accuracy of 0.950, and a PR-AUC of 0.997. These results demonstrate the effectiveness of tree-based algorithms, like GBM, in handling tabular data, where they can better capture the feature interactions and

handle categorical variables.

## 5.1 Policy Gradient with Neural Network Classifier on MNIST Dataset: Detailed Analysis

In this section, we delve into the performance of the Policy Gradient algorithm combined with a Neural Network classifier on the MNIST dataset. The accompanying plots illustrate the trends observed in ROC AUC, Epoch Accuracy, and PR AUC over the course of the training epochs. Each of these metrics delivers valuable insights into the model's learning dynamics and generalization capabilities.

### 5.1.1 Overview of the Learning Curves

The learning curves, as shown in Figure 5.2, depict the evolution of ROC AUC, Accuracy, and PR AUC over 300 epochs. The initial phase shows high performance due to pre-training, followed by a significant dip when the model is exposed to unseen/unlabeled data during interactive learning. Eventually, the metrics recover and stabilize, reflecting the model's ability to learn and adapt.

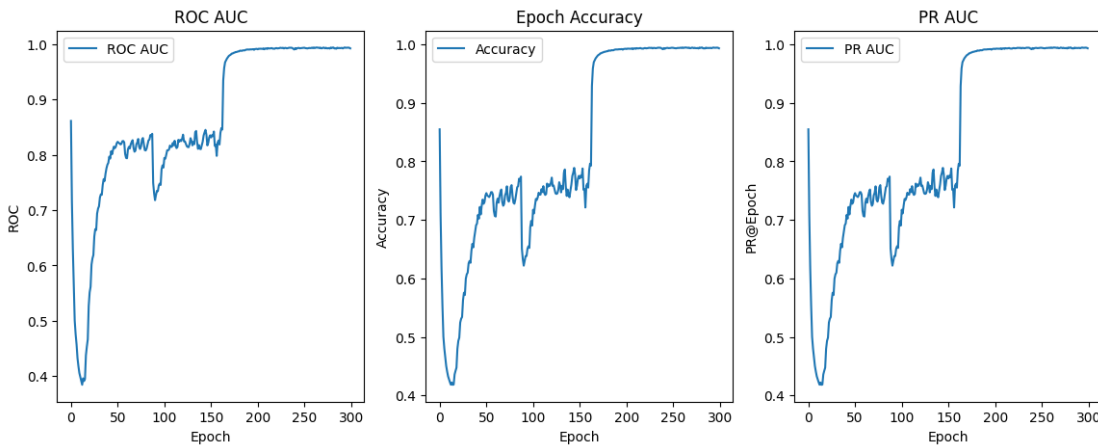


Figure 5.2: Learning curves for Policy Gradient with Neural Network Classifier on MNIST Dataset. The plots display ROC AUC, Epoch Accuracy, and PR AUC over 300 epochs.

### 5.1.2 Initial High Performance and Subsequent Dip

At the beginning of the training process, we observe that all three metrics (ROC AUC, Accuracy, and PR AUC) start at relatively high values. This initial high performance can be attributed to the pre-training phase, where the model has already been exposed to a subset of the data and has learned to identify patterns effectively.

However, as the training progresses and the model is introduced to unseen data through interactive learning, there is a noticeable dip in performance across all metrics. This drop indicates that the model initially struggles to generalize from the pre-training data to the new, unrecognized data. The interactive learning phase challenges the model with a wider variety of examples, exposing its limitations and prompting further learning.

### 5.1.3 Influence of High Weight Decay

Another critical factor influencing the model's performance is the high weight decay parameter. Weight decay is a regularization technique that penalizes large weights in the model, effectively reducing overfitting by encouraging simpler models. In this experiment, the high weight decay parameter has a dual effect:

1. **Penalizing High Weights:** By penalizing higher weight values, the model is prevented from depending excessively on specific features or patterns within the data. This regularization helps prevent overfitting but can initially lead to a decrease in performance as the model adjusts its weights.
2. **Encouraging Generalization:** Over time, this penalty helps the model to generalize better to unseen data. This effect is evident in the later stages of training, where the performance metrics gradually recover and stabilize, indicating improved generalization.

### 5.1.4 Recovery and Stabilization

Despite the initial dip, the learning curves show a significant recovery after approximately 160 epochs. Both ROC AUC and PR AUC metrics, along with accuracy, improve steadily, eventually stabilizing close to their initial high values. This recovery phase highlights the effectiveness of the interactive learning approach combined with high weight decay in enhancing the model's capability to generalize.

- **ROC AUC:** The ROC AUC curve shows that the model's capability to differentiate between positive and negative classes improves over time, reaching close to 0.98, which indicates excellent performance.



- **Epoch Accuracy:** The accuracy plot reflects the model's overall performance in forecasting the accurate labels. The final accuracy stabilizes at a high value, demonstrating the model's effectiveness.
- **PR\_AUC:** The PR\_AUC curve, which is particularly useful for imbalanced datasets, indicates that the model maintains high precision and recall throughout the training process.

### 5.1.5 Insights and Discussion

The observed trends in the learning curves provide several insights into the dynamics of the Policy Gradient algorithm combined with a Neural Network classifier:

1. **Pre-training Benefits:** The initial high performance underscores the benefits of pre-training, which provides the model with a strong starting point.
2. **Challenges of Interactive Learning:** The performance dip highlights the challenges posed by interactive learning, where the model is exposed to a wider range of data, testing its adaptability.
3. **Role of Regularization:** High weight decay plays a crucial role in preventing overfitting and promoting generalization, as evidenced by the recovery in performance metrics.
4. **Model Adaptability:** The eventual recovery and stabilization of the metrics demonstrate the model's ability to adapt and improve with continued exposure to diverse data.

In conclusion, the combination of Policy Gradient reinforcement learning and a Neural Network classifier shows promising results on the MNIST dataset. The interplay between pre-training, interactive learning, and regularization leads to a robust model capable of generalizing well to unseen data. These findings reinforce the importance of balanced training strategies and appropriate regularization in developing effective machine-learning models.

## 5.2 Policy Gradient with LightGBM Classifier on MNIST Dataset: Detailed Analysis

In this section, we examine the performance of the Policy Gradient algorithm combined with a LightGBM classifier on the MNIST dataset. The accompanying plots illustrate the trends observed in ROC\_AUC, Epoch Accuracy, and PR\_AUC over several epochs. Each of these

metrics delivers helpful insights into the model's learning dynamics and its generalization capabilities.

### 5.2.1 Overview of the Learning Curves

The learning curves, as shown in Figure 5.3, depict the evolution of ROC\_AUC, Accuracy, and PR\_AUC over epochs. The plots reveal a steady improvement in all metrics, indicating that the model consistently learns and adapts as training progresses.

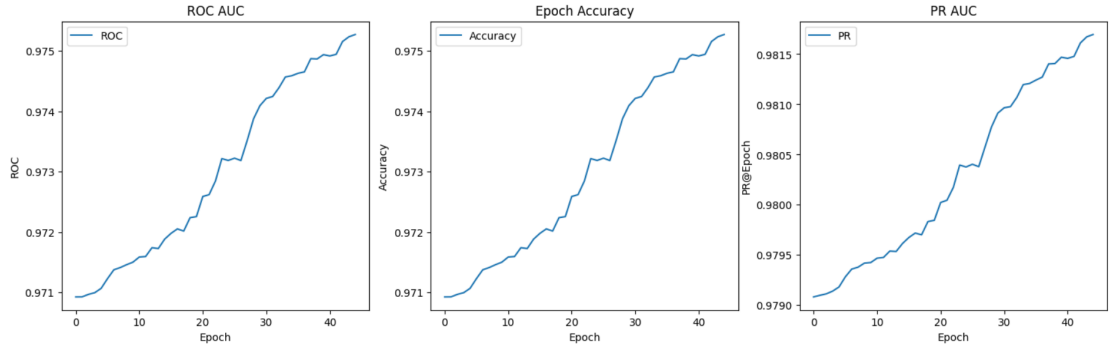


Figure 5.3: Learning curves for Policy Gradient with LightGBM Classifier on MNIST Dataset. The plots display ROC\_AUC, Epoch Accuracy, and PR\_AUC.

### 5.2.2 Steady Improvement and Model Learning

The learning curves display a consistent upward trend in ROC\_AUC, Accuracy, and PR\_AUC. This steady improvement suggests that the LightGBM classifier, in conjunction with the Policy Gradient algorithm, is effectively learning the underlying patterns in the MNIST data:

- **ROC\_AUC:** The ROC\_AUC metric gradually increment indicates that the model's capability to differentiate between the positive and negative classes improves steadily over time.
- **Epoch Accuracy:** The accuracy metric follows a similar trend, rising to about 0.97. This shows that the model's overall performance in predicting the accurate labels improves consistently throughout the training process.
- **PR\_AUC:** The PR\_AUC metric, which is particularly important for imbalanced datasets, also shows a steady till 0.98. This indicates that the model maintains high precision and

recall, even as it continues to learn from the data.

### 5.2.3 Detailed Insights

The steady improvement in the learning curves highlights several important aspects of the model's performance:

1. **Effectiveness of LightGBM:** LightGBM, a tree-based gradient boosting framework, is renowned for its efficiency and superior performance on tabular data. Its application to the MNIST dataset demonstrates that it can also perform well on image data, particularly when combined with a robust reinforcement learning framework like Policy Gradient.
2. **Learning Stability:** The gradual and consistent improvement in all three metrics suggests that the model is learning in a stable manner. There are no significant drops or fluctuations in performance, indicating that the learning process is smooth and the model is not overfitting or underfitting significantly.
3. **Adaptability:** The consistent upward trend across all metrics shows the model's adaptability to the training data. The LightGBM classifier, supported by the Policy Gradient reinforcement learning algorithm, is able to continuously refine its understanding of the data, leading to improved performance over time.
4. **Impact of Interactive Learning:** The interactive learning setting, where the model is revealed to a combination of data points and continuously learns from its environment, appears to be beneficial. The steady improvement indicates that the model effectively uses feedback from the environment to enhance its predictive capabilities.
5. **Model Robustness:** The robustness of the model is evident from the stable improvement across all metrics. The LightGBM classifier's ability to handle various features and interactions, combined with the Policy Gradient's optimization, leads to a robust learning process.

### 5.2.4 Discussion

Comparatively, the Neural Network classifier initially performs well due to pre-training but experiences a significant performance dip when exposed to new data, which eventually recovers. In contrast, the LightGBM classifier shows a stable and consistent improvement without such a dip. The high weight decay's impact is specifically noted in the Neural Network classifier, helping to prevent overfitting and improve generalization over time, a detail not highlighted for the LightGBM classifier. The results from the Policy Gradient with LightGBM

classifier on the MNIST dataset provide several key insights:

1. **Balanced Performance:** The model achieves a balanced performance across ROC\_AUC, Accuracy, and PR\_AUC, indicating a well-rounded ability to distinguish classes, predict correctly, and maintain precision and recall.
2. **Generalization Capability:** The consistent improvement in metrics indicates that the model is not only effectively learning from the training data but is also likely to generalize well to new, unseen data. This is crucial for practical applications where models must perform well on new, unseen data points.
3. **Efficiency of LightGBM:** LightGBM's efficiency and ability to handle complex interactions between features make it a suitable choice for reinforcement learning scenarios. Its performance on the MNIST dataset showcases its versatility beyond traditional tabular data.

In conclusion, the combination of Policy Gradient reinforcement learning and a LightGBM classifier shows promising results on the MNIST dataset. The steady improvement in ROC\_AUC, Accuracy, and PR\_AUC underscores the effectiveness of this approach. These findings highlight the importance of selecting appropriate model architectures and training strategies to achieve optimal performance.

## 5.3 Policy Gradient with Neural Network Classifier on Credit Card Transaction Dataset: Detailed Analysis

In this section, we examine the performance of the Policy Gradient algorithm combined with a Neural Network classifier on the Credit Card Transaction data. The accompanying plots illustrate the trends observed in ROC\_AUC, Epoch Accuracy, and PR\_AUC over the course of 300 epochs. Each of these metrics delivers valuable insights into the model's learning dynamics and its generalization capabilities.

### 5.3.1 Overview of the Learning Curves

The learning curves, as shown in Figure 5.4, depict the evolution of ROC\_AUC, Accuracy, and PR\_AUC over 300 epochs. The plots reveal a high degree of fluctuation in all metrics, indicating instability in the learning process.

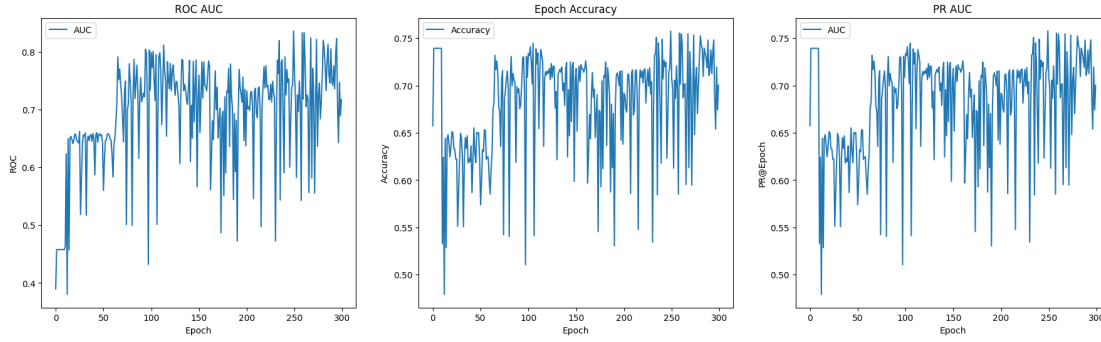


Figure 5.4: Learning curves for Policy Gradient with Neural Network Classifier on Credit Card Transaction Dataset. The plots display ROC\_AUC, Epoch Accuracy, and PR\_AUC over 300 epochs.

### 5.3.2 Fluctuating Performance and Learning Instability

The learning curves display significant fluctuations in ROC\_AUC, Accuracy, and PR\_AUC. This instability suggests that the Neural Network classifier, in conjunction with the Policy Gradient algorithm, is struggling to learn effectively from the Credit Card Transaction data:

- **ROC\_AUC:** The ROC\_AUC metric starts around 0.4, gradually increases, but shows frequent and erratic drops and spikes throughout the training process. This indicates that the model's capability to differentiate between the positive and negative classes is inconsistent.
- **Epoch Accuracy:** The accuracy metric follows a similar trend, starting around 0.5 and rising to about 0.76, but with numerous fluctuations. This inconsistency reflects the model's unstable performance in forecasting the correct labels.
- **PR\_AUC:** The PR\_AUC metric also shows a high degree of fluctuation, starting around 0.5 and reaching up to 0.75, but with frequent drops. This indicates variability in the model's precision and recall, particularly for the minority class.

### 5.3.3 Detailed Insights

The fluctuating performance in the learning curves highlights several important aspects of the model's performance:

1. **Data Complexity:** The Credit Card Transaction dataset is inherently complex and highly imbalanced, with a significant disparity between fraudulent and non-fraudulent

transactions This complexity poses a significant challenge for the Neural Network classifier, leading to unstable learning.

2. **Model Instability:** The significant fluctuations in all three metrics suggest that the model is not learning in a stable manner. This could be due to several factors, including insufficient regularization, inappropriate learning rate, or the inherent difficulty of the dataset.
3. **Overfitting and Underfitting:** It appears that the model may be fluctuating between being overfitting and underfitting based on the unpredictable changes in performance indicators. Underfitting arises when the model is unable to detect the underlying patterns, whereas overfitting happens when the model detects noise in the training data.
4. **Impact of Interactive Learning:** The interactive learning setting, while beneficial in some scenarios, may be contributing to the instability observed here. The constant exposure to a variety of data points and the dynamic adjustment of the policy may be causing the model to struggle in finding a consistent learning path.

#### 5.3.4 Discussion

The results from the Policy Gradient with Neural Network classifier on the Credit Card Transaction dataset provide several key insights:

1. **Challenges of Imbalanced Data:** The model's fluctuating performance underscores the challenges associated with learning from highly imbalanced datasets. Special strategies such as SMOTE (Synthetic Minority Over-sampling Technique), adjusted class weights, or anomaly detection methods might be necessary to improve stability.
2. **Need for Regularization:** The instability suggests a need for stronger regularization techniques. Adjusting the weight decay parameter, using dropout, or employing other regularization methods might help in stabilizing the learning process. We might need to spend more time to find out the fine parameters for this.
3. **Learning Rate Adjustments:** It could be necessary to adjust the learning rate more precisely. While a low learning rate may make the learning process unduly lengthy and prone to becoming stuck in local minima, a high learning rate may cause the model to converge too rapidly on a suboptimal solution.

In conclusion, the combination of Policy Gradient reinforcement learning and a Neural Network classifier faces significant challenges when used to the Credit Card Transaction data. The observed instability and fluctuating performance metrics highlight the need for improved data handling, regularization, and learning rate tuning. These results highlight how crucial it is to customize model architectures and training plans to the particular features of the dataset in order to attain peak performance.

## 5.4 Policy Gradient with LightGBM Classifier on Credit Card Transaction Dataset: Detailed Analysis

This section examines the Credit Card Transaction dataset’s performance when the Policy Gradient algorithm and a LightGBM classifier are used together. Plots that accompany the text show the trends in ROC\_AUC, PR\_AUC, and epoch accuracy over a span of 300 epochs. Each of these measures offers important information about the generalization and learning dynamics of the model.

### 5.4.1 Overview of the Learning Curves

The learning curves, as shown in Figure 5.5, depict the evolution of ROC\_AUC, Accuracy, and PR\_AUC over 300 epochs. The plots reveal a steady and continuous improvement in all metrics, indicating that the model effectively learns and adapts as training progresses.

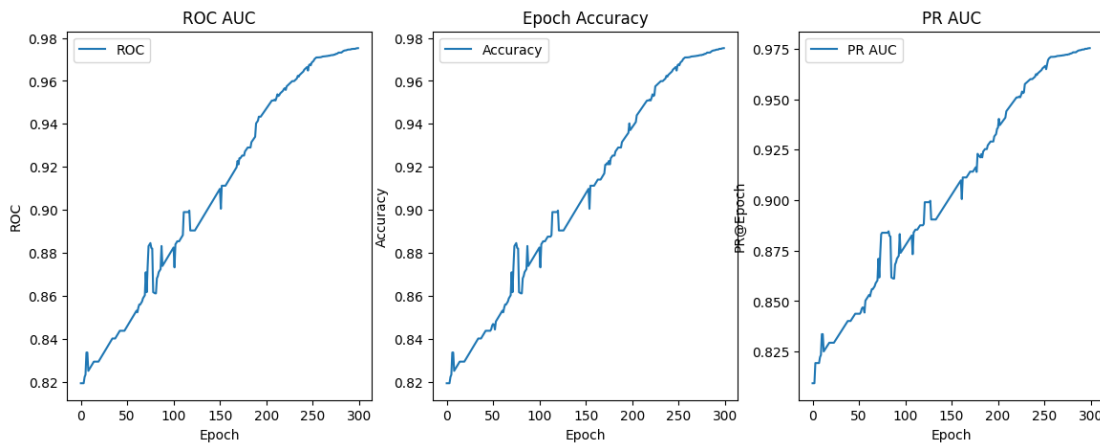


Figure 5.5: Learning curves for Policy Gradient with LightGBM Classifier on Credit Card Transaction Dataset. The plots display ROC\_AUC, Epoch Accuracy, and PR\_AUC over 300 epochs.

### 5.4.2 Steady Improvement and Model Learning

The learning curves display a consistent upward trend in ROC\_AUC, Accuracy, and PR\_AUC. This steady improvement suggests that the LightGBM classifier, in conjunction with the Policy Gradient algorithm, is effectively learning the underlying patterns in the Credit Card Transaction data:

- **ROC\_AUC:** The ROC\_AUC metric starts around 0.82 and gradually increases to approximately 0.98 by the 300th epoch. This indicates that the model's ability to distinguish between the positive and negative classes improves steadily over time.
- **Epoch Accuracy:** The accuracy metric follows a similar trend, starting around 0.82 and rising to about 0.98. This demonstrates how the model continuously gets better at predicting the right labels as it goes through the training phase.
- **PR\_AUC:** The PR\_AUC metric, which is particularly important for imbalanced datasets, also shows a steady increase from around 0.82 to approximately 0.98. This indicates that the model maintains high precision and recall, even as it continues to learn from the data.

### 5.4.3 Detailed Insights

The steady improvement in the learning curves highlights several important aspects of the model's performance:

1. **Effectiveness of LightGBM:** One tree-based gradient boosting system that has gained popularity is LightGBM, which performs exceptionally well on tabular data. Its application to the Credit Card Transaction dataset demonstrates that it can effectively handle the complexities of this type of data, particularly when combined with a robust reinforcement learning framework like Policy Gradient.
2. **Learning Stability:** The gradual and consistent improvement in all three metrics suggests that the model is learning in a stable manner. There are no significant drops or fluctuations in performance, indicating that the learning process is smooth and the model is not overfitting or underfitting significantly.
3. **Adaptability:** The way that all of the indices are constantly rising indicates how well the model can adjust to the training set. Through the use of the Policy Gradient reinforcement learning technique, the LightGBM classifier may be able to enhance its understanding of the data over time, leading to continuous performance improvements.



4. **Impact of Interactive Learning:** The model seems to benefit from the interactive learning environment, as it is exposed to a range of data points and continuously learns from its surroundings. The steady improvement indicates that the model effectively uses feedback from the environment to enhance its predictive capabilities.
5. **Model Robustness:** The robustness of the model is evident from the stable improvement across all metrics. The LightGBM classifier's ability to handle various features and interactions, combined with the Policy Gradient's optimization, leads to a robust learning process.

#### 5.4.4 Discussion

The results from the Policy Gradient with LightGBM classifier on the Credit Card Transaction dataset provide several key insights:

1. **Effectiveness of Tree-Based Algorithms:** Our hypothesis that tree-based algorithms perform well on tabular data is strongly supported by these results. LightGBM's performance in this interactive learning setup demonstrates its capability to effectively handle and learn from the intricacies of credit card transaction data.
2. **Balanced Performance:** The model achieves a balanced performance across ROC\_AUC, Accuracy, and PR\_AUC, indicating a well-rounded ability to distinguish classes, predict correctly, and maintain precision and recall.
3. **Generalization Capability:** Based on the measures' consistent improvement, it looks that the model is learning well from the training data and has the ability to generalize well to new data. This is crucial for practical applications where models must perform well on new, unseen data points.
4. **Efficiency of LightGBM:** LightGBM's efficiency and ability to handle complex interactions between features make it a suitable choice for reinforcement learning scenarios. Its performance on the Credit Card Transaction dataset showcases its versatility beyond traditional tabular data.

In conclusion, the combination of Policy Gradient reinforcement learning and a LightGBM classifier shows promising results on the Credit Card Transaction dataset. The steady improvement in Accuracy, ROC\_AUC, and PR\_AUC underscores the effectiveness of this approach. These findings highlight the importance of selecting appropriate model architectures and training strategies to achieve optimal performance.



## 6 Conclusion

This thesis' main goal was to assess the performance of tree-based algorithms in an interactive learning environment, with a focus on tabular datasets. In order to accomplish this, we ran a number of tests using a Policy Gradient reinforcement learning method in conjunction with several classifiers, such as LightGBM and Neural Network classifiers. The key findings from these experiments are summarized below:

### 6.0.1 Policy Network with Neural Network Classifier on MNIST Dataset

The first experiment involved a Policy Network with a Neural Network classifier applied to the MNIST Image Dataset. The results demonstrated that this combination performed exceptionally well, achieving high scores in ROC\_AUC, accuracy, and PR\_AUC. The initial high performance was attributed to the pre-training phase, which provided a strong starting point. The subsequent interactive learning phase further refined the model, highlighting the effectiveness of neural networks in handling image data by capturing complex patterns and spatial hierarchies.

### 6.0.2 Policy Network with Neural Network Classifier on Credit Card Transaction Dataset

The second experiment applied the Policy Network with a Neural Network classifier to the Credit Card Transaction Dataset. Unlike the first experiment, the learning curves exhibited high fluctuations in ROC\_AUC, accuracy, and PR\_AUC, indicating instability and inconsistent learning. This instability was primarily due to the complexity and imbalance inherent in the dataset, which posed significant challenges for the neural network classifier. The results underscored the necessity of specialized techniques to manage imbalanced tabular data, including enhanced regularization and tailored network architectures.

### 6.0.3 Policy Network with LightGBM Classifier on MNIST Dataset

In the third experiment, we used the MNIST Image Dataset to combine a Policy Network and a LightGBM classifier. Over the training epochs, the findings demonstrated a constant and steady gain in accuracy, PR\_AUC, and ROC\_AUC. Although LightGBM is typically associated with tabular data, its application to the MNIST dataset in this context demonstrated its

versatility. The stable learning process and gradual enhancement of performance metrics highlighted LightGBM's capability to handle image data effectively within an interactive learning setup.

#### 6.0.4 Policy Network with LightGBM Classifier on Credit Card Transaction Dataset

The final experiment, which was central to our thesis, involved a Policy Network with a LightGBM classifier on the Credit Card Transaction Dataset. This combination yielded highly promising results, with continuous and substantial improvements in all performance metrics, and outperformed across all conducted experiments, achieving high scores in ROC\_AUC, accuracy, and PR\_AUC. These results validated our hypothesis that tree-based algorithms, particularly LightGBM, excel in handling tabular data. The success of combining LightGBM with reinforcement learning was demonstrated by the model's resilience and flexibility in handling the intricate details of credit card transaction data.

#### 6.0.5 Overall Insights and Future Directions

The experiments provided several key insights into the effectiveness of tree-based algorithms in an interactive learning framework on tabular datasets:

1. **Superiority of Tree-Based Algorithms on Tabular Data:** LightGBM significantly outperformed the Neural Network classifier on the tabular Credit Card Transaction dataset, confirming its suitability for this type of data.
2. **Learning Stability and Generalization:** The consistent and stable improvements in performance metrics for the LightGBM classifier indicated its robust learning and generalization capabilities.
3. **Interactive Learning Benefits:** The interactive learning setup effectively enhanced model performance by continuously exposing the model to diverse data points and incorporating feedback, leading to improved learning dynamics.

#### 6.0.6 Overall Insights and Future Directions

The experiments provided several key insights into the effectiveness of tree-based algorithms in an interactive learning framework on tabular datasets:

1. **Superiority of Tree-Based Algorithms on Tabular Data:** LightGBM significantly outperformed on the tabular Credit Card Transaction dataset, confirming its suitability for this type of data.
2. **Learning Stability and Generalization:** The consistent and stable improvements in performance metrics for the LightGBM classifier indicated its robust learning and generalization capabilities.
3. **Interactive Learning Benefits:** The interactive learning setup effectively enhanced model performance by continuously exposing the model to diverse data points and incorporating feedback, leading to improved learning dynamics.

## 6.1 Future Work

Future work should explore advanced techniques to further enhance the performance of tree-based algorithms in interactive learning setups. This could include:

- **Fine-Tuning Hyperparameters:** Investigating optimal hyperparameters to further improve model performance and stability.
- **Experimenting with Different Network Architectures:** Exploring deeper neural networks, convolutional neural networks, or hybrid architectures to handle the complexities of different datasets more effectively.
- **Incorporating Advanced Regularization Methods:** Employing techniques such as dropout, L1/L2 regularization, or other forms of regularization to manage overfitting and improve generalization.
- **Leveraging Large Language Models (LLMs):** Experimenting with training Large Language Models (LLMs) using Positive-Unlabeled (PU) datasets. LLMs have shown proficiency in one-shot learning and could potentially enhance the learning process in PU scenarios by leveraging their ability to generalize from limited labeled data.

In conclusion, this thesis has demonstrated the effectiveness of tree-based algorithms, specifically LightGBM, in an interactive learning framework on tabular datasets. The findings highlight the importance of selecting appropriate model architectures and training strategies to achieve optimal performance in machine learning tasks. Future research should continue to explore innovative techniques and leverage advanced models like LLMs to further enhance the capabilities of interactive learning systems.



# Bibliography

- [1] T. Li, C.-C. Wang, Y. Ma, P. Ortal, Q. Zhao, B. Stenger, and Y. Hirate, “Learning classifiers on positive and unlabeled data with policy gradient,” <https://arxiv.org/pdf/1910.06535>, 2020.
- [2] Y. LeCun, Courant Institute, NYU Corinna Cortes, Google Labs, New York Christopher J.C. Burges, Microsoft Research, and Redmond., “The mnist database of handwritten digits,” <https://yann.lecun.com/exdb/mnist/>, 1998.
- [3] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” <https://www.cs.cmu.edu/~avrim/Papers/cotrain.pdf>, 1998.
- [4] J. Bekker and J. Davis, “Learning from positive and unlabeled data: A survey,” <https://arxiv.org/pdf/1811.04820>, Nov 2018.
- [5] P. Yang, X.-L. Li, J.-P. Mei, C.-K. Kwoh, and S.-K. Ng, “Positive-unlabeled learning for disease gene identification,” *Bioinformatics*, October 15 2012. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3467748/>
- [6] L. Grinsztajn, E. Oyallon, and G. Varoquaux, “Why do tree-based models still outperform deep learning on tabular data?” 2022. [Online]. Available: <https://arxiv.org/pdf/2207.08815>
- [7] R. Shwartz-Ziv and A. Armon, “Tabular data: Deep learning is not all you need,” *arXiv preprint arXiv:2106.03253*, 2021. [Online]. Available: <https://arxiv.org/abs/2106.03253>
- [8] C. Elkan and K. Noto, “Learning classifiers from only positive and unlabeled data,” <https://cseweb.ucsd.edu/~elkan/posonly.pdf>, 2008.
- [9] R. Kiryo, G. Niu, M. C. du Plessis, and M. Sugiyama, “Positive-unlabeled learning with non-negative risk estimator,” <https://arxiv.org/pdf/1703.00593>, 2017.
- [10] C. Luo, P. Zhao, C. Chen, B. Qiao, C. Du, H. Zhang, W. Wu, S. Cai, B. He, S. Rajmohan, and Q. Lin, “Pulns: Positive-unlabeled learning with effective negative sample selector,” in *Proceedings of the 2022 Conference on Machine Learning Research*. Microsoft Research, China; Microsoft 365, United States; The University of Newcastle, Australia; L3S Research Center, Leibniz University Hannover, Germany; State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China; School of Computer Science and Technology, University of Chinese Academy of Sciences, China, 2022, <https://ojs.aaai.org/index.php/AAAI/article/view/17064>.

- [11] J. He, Y. Zhang, X. Li, and Y. Wang, “Bayesian classifiers for positive unlabeled learning,” pp. 81–93, 09 2011. [Online]. Available: [https://www.researchgate.net/publication/221509548\\_Bayesian\\_Classifiers\\_for\\_Positive\\_Unlabeled\\_Learning](https://www.researchgate.net/publication/221509548_Bayesian_Classifiers_for_Positive_Unlabeled_Learning)
- [12] Ravelin, “Machine learning for fraud detection,” <https://www.ravelin.com/insights/machine-learning-for-fraud-detection>, 2024, accessed: 2024-07-09.
- [13] Worldline and the Machine Learning Group of ULB (Université Libre de Bruxelles), “Credit card fraud detection,” 2016, database: Open Database, Contents: Database Contents. The dataset has been collected and analysed during a research collaboration on big data mining and fraud detection. More details are available on <https://www.researchgate.net/project/Fraud-detection-5> and the page of the DefeatFraud project. [Online]. Available: <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>