博士論文

Machine Learning with Weak Supervision from Risk Minimization Perspective

（リスク最小化の観点からの弱教師付き機械学習）

坂井 智哉

Contents

Abstract

Recent advances in the Internet have resulted in the generation of an enormous amount of data every day. By extracting useful rules and information from the data, we can utilize them for various purposes. However, since handling a large amount of data is a laborious task for humans, people want computers to conduct human-like information processing. To this end, *machine learning*, which is aimed at obtaining human-like intelligent systems, has been gathering growing attention and has achieved remarkable success in several tasks.

The success of machine learning mainly comes from supervised learning with a large amount of *labeled* data. However, a sufficient amount of labeled data is not always available in practice. To address this problem, researchers have explored a machine learning approach that can learn from *limited* or *weak supervision*. For example, semi-supervised learning, which utilizes abundant *unlabeled* data in addition to a small amount of labeled data, has been studied. However, most of the existing semi-supervised learning methods proposed so far rely on particular distributional assumptions, such as the *cluster assumption*, that are rarely satisfied in real-world applications. Once such distributional assumptions are violated, the performance of the existing methods degrades and sometimes becomes even worse than that of the plain supervised learning method using only a small amount of labeled data. Thus, it is desired to have semi-supervised learning methods that do not rely on strong distributional assumptions.

On the other hand, in real-world applications, it is conceivable that only positive and unlabeled data are available. Learning from such a situation is called *positive-unlabeled learning* (PU learning), which can also be regarded as a form of learning from weak supervision since labeled data is available only from the positive class. For example, in online ad click prediction problems, when collecting data, merely regarding an unclicked advertisement as negative data introduces a *bias*; users may have an interest in the advertisement, but may not click on the ad because of lack of time. PU learning provides a solution that cancels the bias. Since the PU learning situation often occurs in real-world applications, it is practically desirable to have PU learning methods for various machine learning tasks such as imbalanced classification and statistical dependency analysis.

To address the above issues, this dissertation is devoted to developing machine learning approaches under weak supervision. In particular, we focus on a risk minimization approach. That is, based on a learning criterion called a *risk* or *error*, machine learning methods learn decision rules and statistical dependency from data. The contributions of this dissertation are as follows: i) We propose a PU classification method for imbalanced data, where the number of positive and negative samples in underlying distributions is far from even. ii) We develop statistical dependency analysis tools for PU data. iii) We propose semi-supervised learning methods that do not rely on strong distributional assumptions. Below, we briefly explain these contributions.

Firstly, in addition to the PU learning scenario, we consider a class-imbalance situation. In this situation, maximizing the *area under the receiver operating characteristic curve* (AUC) is a standard approach rather than ordinary misclassification rate minimization. We propose a risk function based on AUC for the PU learning situation and prove that minimization of the risk estimator corresponds to minimization of the generalization error bound in AUC maximization. We experimentally demonstrate the practical effectiveness of the AUC maximization method from PU data.

Next, we develop statistical dependency analysis tools for PU data. As a statistical dependency measure, we employ *squared-loss mutual information* (SMI). We present an SMI estimation method from only PU data and prove its optimal convergence to true SMI. Then, we apply the SMI estimator to dimension reduction and independence test. Through numerical experiments, we investigate the behavior of the SMI estimator and demonstrate the effectiveness of the SMI-based statistical dependency analysis methods.

Then, we discuss a semi-supervised learning approach based on PU learning. Unlike most of the existing methods, our methods can leverage unlabeled data without the restrictive distributional assumptions such as the cluster assumption. We prove that the variance of the proposed risk estimators is smaller than that of their supervised counterparts, and generalization error bounds decrease with respect to the number of positive, negative, and unlabeled data with the optimal parametric rate without restrictive distributional assumptions. We experimentally show the effectiveness of the semi-supervised learning methods.

Finally, we summarize the conclusions of this dissertation and discuss future directions.

Acknowledgements

First of all, I am deeply grateful to my supervisor, Prof. Masashi Sugiyama. Without his support, my Ph.D. studies might have reached a dead end. Every time I was confronted with a difficulty or challenge, he encouraged me and provided valuable comments to help me overcome them. He patiently supported me as I progressed, and his unwavering confidence in me has helped relieve my anxiety while pursuing my education. I would also like to express my gratitude to Prof. Issei Sato and Prof. Junya Honda. They frankly communicated with me and helped provide refreshing ideas and advice. Moreover, I am very grateful to Prof. Masato Okada, Prof. Noboru Kunihiro, and Prof. Taiji Suzuki for reviewing and evaluating my thesis.

Next, I would like to express my great appreciation to Dr. Marthinus Christoffel du Plessis and Prof. Gang Niu. I greatly appreciated their advice, as it was based on their research experience and it opened my eyes to numerous essential points for conducting research and writing papers. Without them, my Ph.D. studies would have been almost impossible to complete.

Furthermore, my gratitude goes out all the members of Sugiyama-Sato-Honda Laboratory. In particular, I am grateful to Ikko Yamane. His insight helped not only my Ph.D. studies but also setting up all servers for numerical experiments from scratch. I am also grateful to Prof. Hiroaki Sasaki, Dr. Kiyoshi Irie, Prof. Hideko Kawakubo, Dr. Voot Tangkaratt, Inbal Horev, Hiroaki Shiino, Mina Ashizawa, Soma Yokoi, Takashi Ishida, Futoshi Futami, Weihua Hu, Ryuichi Kiryo, Masayoshi Hayashi, Ryosuke Kamesawa, Han Bao, and Yusuke Konno. The secretaries, Ms. Ayako Tamai and Ms. Yuko Kawashima, kindly supported me. I deeply appreciate their help.

My research project was financially supported by the Japan Society for the Promotion of Science (JSPS) Fellowship Program (the grant number 15J09111). The financial support helped me in various aspects: preparing research environment, conducting research, and attending conferences. Without this support, the life of my Ph.D. studies would have been harder.

Last but not least, I would like to thank my parents and grandparents for their lasting support to me.

List of Figures

List of Tables

Notation

$\mathbb{R}$	The set of real numbers
$\boldsymbol{x} \in \mathbb{R}^d$	A d-dimensional vector of features
$\boldsymbol{x}^{(i)}$	An i-th dimension of a vector $\boldsymbol{x}$
y	An output or class label
$p(\boldsymbol{x}, y)$	The joint probability density function
$p(\boldsymbol{x})$	The marginal probability density of $\boldsymbol{x}$
$p(y)$	The marginal probability mass of y
$\mathcal{O}_p(\cdot)$	The probabilistic order
$\lVert \cdot \rVert_2$ or $\lVert \cdot \rVert$	The ℓ_2 norm or Euclidean norm
$\lVert \cdot \rVert_1$	The ℓ_1 norm
$I(c)$	The indicator function takes 1 if the condition c is true; otherwise 0
$\mathbf{1}_b$	The b-dimensional vector whose elements are all one
$\boldsymbol{I}_b$	The $b \times b$ identity matrix
$\top$	Transpose of vectors and matrices
$\log(x)$	Natural logarithm of x

Chapter 1

Introduction

A recent success of machine learning such as (super) human-level performance in image classification (He et al., 2016) is based on supervised learning with a large amount of annotated data. However, such a large number of annotated data is not always available in real-world applications. This dissertation is devoted to developing machine learning algorithms without sufficient amount of supervision from humans.

1.1 Learning from Data

Since ancient Greece, human beings have wondered about their intelligence and the possibility of creating intelligence. In ancient mythology, the robot behaving like a human has already appeared. When mechanical machines made by gears emerged, some people thought that the machines could think/act like humans. Indeed, people have created intelligence with several gears, and a machine was applied to decrypting codes in Germany (Singh, 2000). Based on these studies, researchers are trying to implement human intelligence using computers composed of billions/trillions of semiconductor transistors instead of gears.

Since the 2000s, the Internet has been connecting computers worldwide via cables. Along with the popularization of the Internet, a vast amount of data is being created every day. These days, researchers are studying a way of acquiring intelligence from this data. *Artificial intelligence* is intended to mimic human intelligence; in particular, *machine learning* implements it based on probability and statistics, which enables computers to conduct human-like information processing through learning from data.

As machine learning has been successfully applied to various tasks, it has been gaining growing attention from several fields. One of the successful examples is image classification. The goal of classification is to identify category or class of data points. For example,

suppose that we are given pairs of a photo (input) and an animal name in the photo (output). With machine learning methods, computers can determine the name of an animal in a new photo. Furthermore, from the input-output data, we can analyze an input-output dependency to reveal the properties of the data by using machine learning methods. In the following section, we explain machine learning approaches to solve related problems.

1.2 Machine Learning with Weak Supervision

In this section, we introduce four machine learning approaches from the perspective of supervision.

1.2.1 Supervised Learning

In a mathematics lecture at school, we first learn the rules and method for solving a problem from a teacher. During the lecture, we may further practice solving problems with several exercises. Sometime after the lecture, we may take an exam where we are faced with new unseen problems that are related to previous exercises but not the same. If we want to obtain a high score on the exam, we need to learn the rule from the teacher properly.

From the viewpoint of machine learning, learning with full supervision (from humans), is called *supervised learning* (Hastie et al., 2009; Mohri et al., 2012; Shalev-Shwartz and Ben-David, 2014). Formally, in supervised learning, we are given pairs of input and output samples

$$\{(\boldsymbol{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^n, \tag{1.1}$$

where $\mathcal{X}$ and $\mathcal{Y}$ are domains of the input and output, respectively (see Figure 1.1 for a binary case, i.e., $|\mathcal{Y}| = 2$). For example, the inputs are images taking animals and the outputs are names of animals in the images. By using supervised learning, a mapping function from inputs and outputs can be obtained by

$$g\colon \mathcal{X} \to \mathcal{Y}. \tag{1.2}$$

Then, we can answer a name of an animal on a new image by using a learned mapping function.

FIGURE 1.1: Illustration of a supervised learning situation for a binary class case. In a binary class case, one class is called *positive* class and the other is called *negative* class. Samples belonging positive and negative classes are respectively called positive and negative data, denoted by "○" and "×".

Supervised learning is a strong approach to obtain accurate input-output relationships. Moreover, in classification problems, supervised learning was proved that it achieves optimal classification accuracy as the size of data increases. Roughly speaking, we can expect high classification accuracy if we have a large amount of labeled data. Recent advances in deep learning partly come from this fact, i.e., due to the progress of the Internet, the large size of the dataset available and it enables us to utilize for learning an accurate mapping function.

From the perspective of supervision, to execute supervised learning methods, we need to prepare annotated samples in advance, e.g., in the example of animal images classification, we need to assign a name of animals for each image. However, collecting labeled samples requires laborious annotation task and a large budget to hire specialists for annotation tasks. For each input, we need to give labels appropriately so that supervised learning methods can learn input-output relation correctly. Thus, to reduce labeling costs, researchers have studied alternative learning approach.

1.2.2 Unsupervised Learning

In contrast to supervised learning, supervision is not provided in unsupervised learning (Hastie et al., 2009; Mohri et al., 2012; Shalev-Shwartz and Ben-David, 2014). That is, the available information is inputs only as shown in Figure 1.2:

$$\{\boldsymbol{x}_i\}_{i=1}^n. \tag{1.3}$$

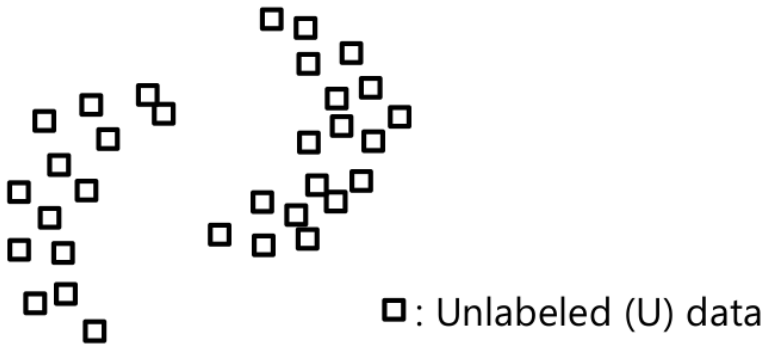


FIGURE 1.2: Illustration of a unsupervised learning situation.

The data without supervision are called *unlabeled data*. From obtained unlabeled data, unsupervised learning is designed to extract useful information.

For example, finding several groups from unlabeled data based on some characteristics is useful for people to understand properties of the data. This is called the *clustering*, and several algorithms were proposed. Another example is (unsupervised) *dimensionality reduction*; its goal is to obtain low-dimensional representation from (redundant) high-dimensional data for visualizing and analyzing the data without supervision.

Unsupervised learning does not require costly labeled data. In this sense, unsupervised learning is preferable to supervised learning. To extract information from unlabeled data, we often presume a particular assumption on the data, e.g., a sort of cluster structure is assumed for clustering methods. However, once the data distribution is far from the prespecified structure or assumptions, unsupervised learning methods do not work well as we expected. Moreover, since it is difficult to measure the quality of unsupervised learning, we often need some heuristic or quality measure based on a particular assumption to choose results obtained by unsupervised learning. If classification is the final goal, the labels would be precious information to achieve high performance.

1.2.3 Semi-Supervised Learning

Semi-supervised learning aims at obtaining high performance from both unlabeled and limited size of labeled data (see Figure 1.3), which can be regarded as a combination of supervised and unsupervised learning. From the perspective of supervision, supervision provided to semi-supervised learning is limited (weak) since the size of labeled data is

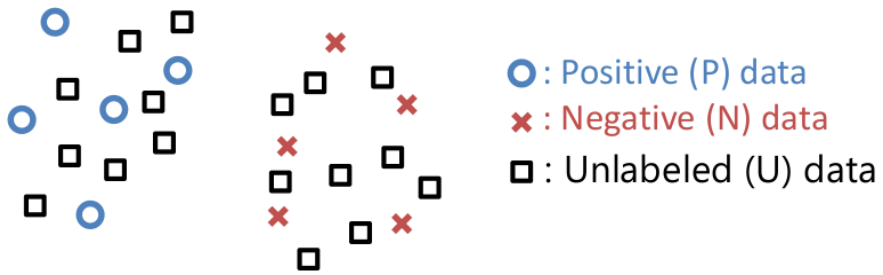


FIGURE 1.3: Illustration of a semi-supervised learning situation.

often assumed to be rather small compared with that of supervised learning. If semi-supervised learning methods can achieve high accuracy with a small amount of labeled data, we could manage both cost and accuracy.

In the beginning, most of the researchers tried to incorporate knowledge and techniques from unsupervised learning. For instance, Chapelle et al. (2002) proposed the *cluster kernel* based on the cluster assumption: "similar" data points tend to have the same label. Another example is the *manifold assumption*, which supposes that samples are distributed on a (near) low-dimensional manifold in the data space (Belkin et al., 2006). These assumptions can be summarized as the *low-density separation principle* that samples in different classes tend to be separated in a region with low data density (Chapelle et al., 2006). Although hundreds of studies have been devoted to developing semi-supervised learning methods based on the low-density separation principle, once the distributional assumptions are not satisfied, the performance of existing semi-supervised learning methods degrades or even worse than that of supervised learning methods with a limited amount of labeled data (Cozman et al., 2003; Sokolovska et al., 2008).

To cope with this problem, recently several methods have been proposed to mitigate undesired performance degeneration of semi-supervised learning methods (Li and Zhou, 2015; Loog, 2016; Krijthe and Loog, 2017). For instance, a safe approach (Li and Zhou, 2015) aims at designing semi-supervised learning methods so that its performance does not become worse than the supervised learning methods. However, the method proposed by Li and Zhou (2015) still requires the distributional assumption for leveraging unlabeled data. Thus, if the distributional assumptions are not satisfied, the performance improvement is not expected even though the performance degeneration does not occur.

Therefore, it is practically desirable to have semi-supervised learning methods which do not strongly rely on the distributional assumptions.

FIGURE 1.4: Illustration of a PU learning situation.

1.2.4 Positive-Unlabeled Learning (PU Learning)

In real-world applications, we are sometimes faced with the situation in which labeled data for the positive class are relatively easy to obtain but that for the negative class are difficult or almost impossible. In *positive-unlabeled learning* (PU learning) (Letouzey et al., 2000), we are only given positive and unlabeled data, but not given negative data (supervision is available only for the positive class), unlike ordinary supervised learning. Figure 1.4 illustrates a PU learning situation.

For example, suppose that we would like to classify cat photos out of animal photos. To this end, we assign cats and non-cats labels to some portion of photos. In this situation, assigning non-cats label would be more difficult than assigning cats label to the photos because we need to cover the several possibilities of non-cat photos such as dogs, lions, monkeys, ..., etc. This problem is known as the diversity of negative classes; the negative class contains several classes.

Another example is the case that labels for the negative class are hardly obtained. For instance, in recommendation system on social service, a service provider can observe a user's favorite articles through "like" button. In contrast, it is difficult to observe the user's disliked article because "dislike" button is usually not implemented. Moreover, whole articles that the user did not push the "like" button cannot be regarded merely as disliked articles since the user may not have read them yet: these articles are a mixture of liked and disliked articles. For predicting user ' s preference, classification problem in PU learning is needed to be solved.

One naive way of dealing with such a PU classification task is to regard unlabeled data as negative data and apply an ordinary supervised classification method to train a classifier. However, this approach causes bias because unlabeled data consist of both positive and negative data. To address this problem, several methods have been proposed to learn a decision rule from only PU data (Lee and Liu, 2003; Liu et al., 2003; Li and

Liu, 2005; Elkan and Noto, 2008). Recently, a PU learning method based on an empirical risk minimization framework was proposed (du Plessis et al., 2014, 2015) and shown that its risk function computed from only PU data is equivalent to the risk in supervised classification. Furthermore, a generalization property of PU learning was theoretically investigated, revealing that PU learning outperforms an ordinary supervised classification method under some conditions.

As the PU learning situation is perceived, it is practically desirable to have accurate classification and statistical data analysis methods for PU data.

1.3 Contributions

In this section, we describe the contributions of this dissertation.

1.3.1 Imbalanced Classification from Positive and Unlabeled Data

We develop a PU classification method for imbalanced classification, where the number of positive and negative samples in underlying distributions is far from even. For such a imbalanced classification problem, optimizing (maximizing) the *area under the receiver operating characteristic curve* (AUC) is a standard approach.

We derive the risk based on AUC optimization from only PU data (the PU-AUC risk) and reveal that the existing AUC optimization method uses a biased risk. We prove that minimization of the PU-AUC risk estimator corresponds to minimization of upper bound of the generalization error in AUC optimization. The result shows that PU data contribute to reducing the upper bound without any distributional assumptions.

We experimentally show the effectiveness of the method based on the PU-AUC risk on various datasets.

1.3.2 Statistical Dependency Analysis from Positive and Unlabeled Data

We present statistical dependency analysis tools for PU data.

To evaluate input-output dependency, we employ *squared-loss mutual information* (SMI) as a statistical dependency measure. With only PU data, we reformulate SMI (PU-SMI) that is equivalent to SMI from positive-negative data and show an estimation method. We establish a convergence rate of the PU-SMI estimator, revealing that PU data

help converge to true SMI. We then apply the PU-SMI estimator to two statistical data analysis tasks: dimension reduction and independence test. In particular, the SMI-based dimension reduction method can be executed without an estimate of class-prior that is a critical quantity for the existing PU classification method.

Through numerical experiments, we confirm that the SMI-based dimension reduction method gives a lower class-prior estimation error compared with other dimension reduction method. Additionally, we evaluate the performance of the SMI-based independence test, showing that the method can detect statistical dependency of input-output from only PU data.

1.3.3 Learning from Positive, Negative, and Unlabeled Data

We present a semi-supervised learning approach that does not rely on strong distributional assumptions.

An advantage of the semi-supervised learning approach is that it can employ any classification performance measures as long as a risk in PU learning is available, which is equivalent to that in PN learning. We show the classification methods based on the misclassification rate and AUC. Without strong distributional assumptions, we derive generalization error bounds, revealing that both labeled and unlabeled contribute to reducing the upper bound of the generalization error. Furthermore, we show the variance of the risk estimators based on PU learning becomes smaller than that of the plain supervised learning.

Through extensive experiments, we investigate the behavior of the semi-supervised learning methods and confirm its effectiveness.

1.4 Organization

This dissertation is organized into six chapters shown in Figure 1.5.

Chapter 1 introduces machine learning approach from the perspective of supervision and describes the contributions of this dissertation.

Chapter 2 provides the background of this dissertation. First of all, we review basic concepts of classification in machine learning. Based on the knowledge, we review existing methods in supervised learning, positive-unlabeled learning, and semi-supervised learning methods. Then, we introduce class-prior estimation methods used in experiments

and statistical dependency measures. The notations used in Sections 2.3 and Section 2.5 are frequently used in subsequent chapters.

In Chapter 3, we introduce a method for imbalanced classification in the PU learning setting. We show a learning criterion with a measure for imbalanced classification in a PU learning scenario. Then, we report theoretical and empirical properties of the imbalanced classification method for PU data.

Next, we discuss statistical dependency analysis for PU data in Chapter 4. We present an SMI estimation method from only PU data and apply the estimators to statistical dependency analysis tasks. We confirm the effectiveness of the SMI-based statistical dependency analysis methods.

In Chapter 5, we introduce a semi-supervised learning framework based on positive-unlabeled learning. The framework can be applied to both balanced and imbalanced classification. We investigate the theoretical property and practical effectiveness of the semi-supervised learning approach.

Finally, we present conclusions and discuss future perspective in Chapter 6.

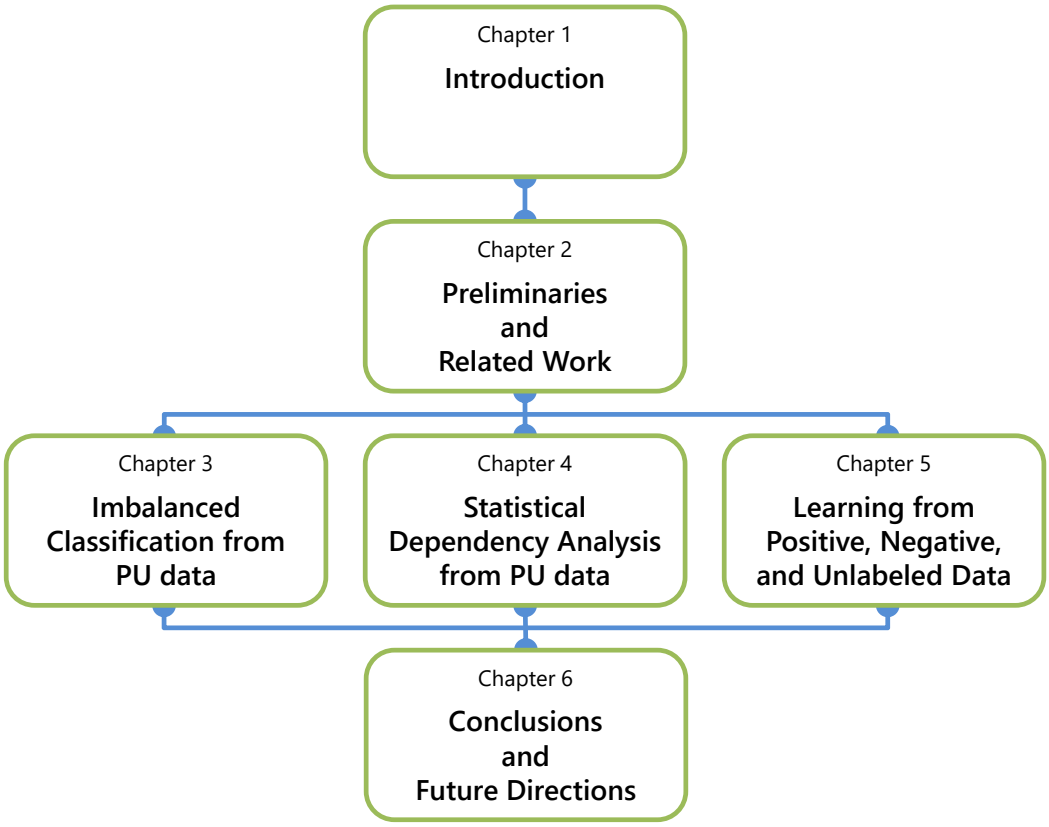


FIGURE 1.5: Organization of this Dissertation

Chapter 2

Preliminaries and Related Work

In this chapter, we review basic concepts of machine learning and related work of this dissertation. First, we describe the common notation in Section 2.1, and explain empirical risk minimization in Section 2.2. In Section 2.3, 2.4, and 2.5, we review basic components of classification, supervised classification, and PU classification. The notation introduced in these sections are frequently used in the subsequent chapters. Then, we review existing semi-supervised learning methods and class-prior estimation methods in Section 2.6 and 2.7, respectively, which are used in numerical experiments. In Section 2.8, we review statistical dependency measures.

2.1 Common Notation

Unless otherwise noted, in this chapter, we focus on binary classification tasks. Let random variables $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{+1, -1\}$ be equipped with probability density $p(\boldsymbol{x}, y)$, where d is a positive integer.

Let us define positive (P), negative (N), and unlabeled (U) data as

$$\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x} \mid y = +1), \tag{2.1}$$

$$\{\boldsymbol{x}_j^{\mathrm{N}}\}_{j=1}^{n_{\mathrm{N}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x} \mid y = -1), \tag{2.2}$$

$$\{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x}) = \theta_{\mathrm{P}} p(\boldsymbol{x} \mid y = +1) + \theta_{\mathrm{N}} p(\boldsymbol{x} \mid y = -1), \tag{2.3}$$

where $\theta_{\mathrm{P}} := p(y = +1)$ and $\theta_{\mathrm{N}} := p(y = -1)$ are the class-prior probabilities. Moreover, let E_{P}, E_{N}, and E_{U} be the expectations over $p(\boldsymbol{x} \mid y = +1)$, $p(\boldsymbol{x} \mid y = -1)$, and $p(\boldsymbol{x})$, respectively.

We use this notation to review the basic concepts of machine learning and existing methods in the subsequent sections.

2.2 Empirical Risk Minimization (ERM)

Here, we explain a learning approach based on minimizing a *risk* or *error*, called *risk minimization*. In classification tasks, a possible choice for the risk is the ratio between the number of data points misclassified by a model and the total number of data points, where the model is assumed for approximating a target function such as a discrimination function.[1] A small risk means that a model is somehow close to the *true* or *optimal* target function which achieves the minimum of the risk.

Let $g\colon \mathbb{R}^d \to \mathbb{R}$ be a specific model and $R\colon \mathbb{R} \to \mathbb{R}$ be a risk function in a certain task.[2] In the risk function, the quality of the model is evaluated as averaged or expected performance, the expectation is computed by the underlying distribution. Then, the learned model is obtained by minimizing the risk function:

$$g^* := \operatorname*{argmin}_{g} \; R(g), \tag{2.4}$$

where g^* is called the *true* or *optimal* model. However, since the underlying distribution is not accessible in practice, we approximate the risk function by using samples collected from the distribution. Let $\widehat{R}(g)$ be the *empirical* risk function that is an approximation of the true risk $R(g)$. Practically, we obtain the learned model by minimizing the empirical risk:

$$\widehat{g} := \operatorname*{argmin}_{g} \; \widehat{R}(g), \tag{2.5}$$

where $\widehat{g}$ is the estimated model. This procedure is called *empirical risk minimization* (ERM).

In this dissertation, we design a risk function for a specific machine learning task such as classification from PU data, and approximate the risk function by using obtained samples. In the PU learning scenario, the risk function can be approximated by P and U data, and in the semi-supervised learning scenario, it can be approximated by P, N, and U data.

In the next section, we see how to apply ERM to classification tasks.

[1] Another example is a sum of squared errors of outputs between the model and the target function.

[2] For the sake of simplicity, we focus on $g\colon \mathbb{R}^d \to \mathbb{R}$ and $R\colon \mathbb{R} \to \mathbb{R}$, but the definition can be extended to more general settings, e.g., the domain of g can be a set and the range of g can be categorical.

2.3 Classification in Statistical Machine Learning

In this section, we review basic components in classification. There are three main components: a function to classify a data point, a measure to evaluate the quality of the function, and a criterion to train the function.

2.3.1 Classifier

A classifier plays a central role in classification. Mathematically, a classifier is a function mapping d-dimensional vector $\boldsymbol{x} \in \mathbb{R}^d$ (input) to a scalar (output):

$$g\colon \mathbb{R}^d \to \mathbb{R}. \tag{2.6}$$

Under a specific rule, we classify data points into each class by outputs of the classifier. In binary classification, we often use a sign of the classifier $\mathrm{sign}(g(\boldsymbol{x}))$ for the decision rule. Below, we introduce typical classifiers used in machine learning.

The linear-in-input classifier is a simple and widely used classifier defined as

$$g(\boldsymbol{x}) = \sum_{\ell=1}^{d} v_\ell x^{(\ell)} + v_0 = \boldsymbol{v}^\top \overline{\boldsymbol{x}}, \tag{2.7}$$

where $\boldsymbol{v} := (v_0, v_1, \ldots, v_d)^\top$ and $\overline{\boldsymbol{x}} := (1, x^{(1)}, \ldots, x^{(d)})^\top$. The linear-in-input classifier is assumed that a hyperplane can separate most of data points. Thus, it is useful when a decision boundary of data points is almost linear, which is likely to happen in high-dimensional data. However, the data points are not always separated linearly. In such a case, applying a non-linear transformation to data points, the data can be classified more correctly in transformed space.

Let $\phi(\boldsymbol{x})$ be a basis function, a specific non-linear transformation for inputs. The linear-in-parameter classifier is defined as

$$g(\boldsymbol{x}) = \sum_{\ell=1}^{b} w_\ell \phi_\ell(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}), \tag{2.8}$$

where $\boldsymbol{w} := (w_1, \ldots, w_b)^\top$ is the vector of the parameters, $\boldsymbol{\phi}(\boldsymbol{x}) := (\phi_1(\boldsymbol{x}), \ldots, \phi_b(\boldsymbol{x}))^\top$ is the vector of the basis functions, and b is the number of basis functions. By setting $b = n$, where n is the number of training samples, and using the *kernel* function $K(\boldsymbol{x}, \boldsymbol{x}')$,

the linear-in-parameter classifier can be regarded as the *kernel classifier*:

$$g(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i), \tag{2.9}$$

where $\{\alpha_i\}_{i=1}^n$ is a set of parameters. For the kernel function, a popular choice is to use the *Gaussian kernel* function

$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp\Big(- \frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\sigma^2} \Big), \tag{2.10}$$

where $\|\cdot\|$ denotes the Euclidean distance and σ is the Gaussian width. The Gaussian kernel can be interpreted as that it maps data points to infinite dimensional space (Schölkopf and Smola, 2001). In the infinite dimensional space, transformed data points can be separated easily.

2.3.2 Loss Function

To evaluate the quality of a classifier, a loss function also plays a vital role in machine learning. So far, many loss functions have been proposed and studied from several aspects such as tasks and algorithms. Here, we briefly review loss functions used in classification.

In binary classification, if a pattern $\boldsymbol{x}$ with label $y \in \{\pm 1\}$ is correctly classified by a classifier g, it satisfies $yg(\boldsymbol{x}) > 0$; otherwise $yg(\boldsymbol{x}) \leq 0$. To evaluate the quality of the classifier, the loss functions map $yg(\boldsymbol{x})$ to some value

$$\ell \colon \mathbb{R} \to \mathbb{R}. \tag{2.11}$$

The *zero-one* loss function is the most basic loss function in classification tasks and can be defined as

$$\ell_{\text{0-1}}(m) := \begin{cases} 0 & (m > 0), \\ 1 & (m \leq 0). \end{cases} \tag{2.12}$$

If a pattern is correctly classified, i.e., $yg(\boldsymbol{x}) > 0$, the zero-one loss function does not give any loss; otherwise, it gives 1 as the loss. This corresponds to checking whether a pattern is correctly classified or not. Thus, an accurate classifier can be found by searching a classifier that can correctly classify a pattern in terms of the zero-one loss function.

Although obtaining a classifier in terms of the zero-one loss function is our ultimate goal, due to the discrete nature of the zero-one loss function, we use *surrogate* loss functions, in practice. So far, several surrogate loss functions have been proposed for classification. One famous loss function is the *hinge loss* function, which is used for the *support vector machine/classifier* (Vapnik, 1995; Cortes and Vapnik, 1995), defined as

$$\ell_{\mathrm{H}}(m) := \max(0, 1 - m). \tag{2.13}$$

In statistics, the *logistic loss* function,

$$\ell_{\mathrm{L}}(m) := \log(1 + \exp(-m)), \tag{2.14}$$

which is used for the *logistic regression* (Hastie et al., 2009), has been used for classification. The squared loss is also often used:

$$\ell_{\mathrm{S}}(m) := (1 - m)^2. \tag{2.15}$$

In addition to the above, there are many other loss functions listed below:

- Squared hinge loss: $\ell_{\mathrm{SH}}(m) := (\max(0, 1 - m))^2$,
- Ramp loss: $\ell_{\mathrm{R}}(m) := \min(2, \max(0, 1 - m))$,
- Absolute loss: $\ell_{\mathrm{A}}(m) := |1 - m|$,
- Exponential loss: $\ell_{\mathrm{E}}(m) := \exp(-m)$,
- Double hinge loss: $\ell_{\mathrm{DH}}(m) := \max(-2m, \max(0, 1 - m))$,
- Sigmoid loss: $\ell_{\mathrm{SG}}(m) := \dfrac{1}{1 + \exp(-m)}$.

These loss functions can be chosen from the viewpoint of tasks, algorithms, and a property of loss functions.

2.3.3 Performance Measure

Now, we have classifiers and loss functions. The final component is performance measure for classification.

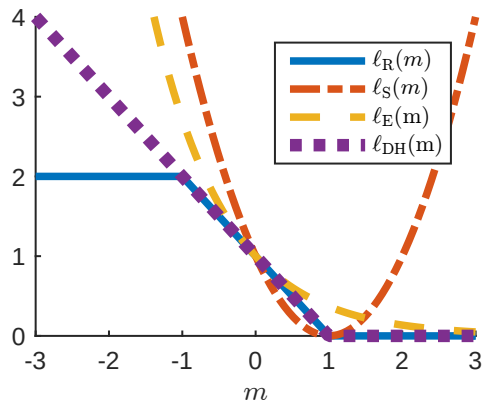


FIGURE 2.1: Shape of Loss functions. The ramp loss ℓ_{R}, squared-loss ℓ_{S}, exponential loss ℓ_{E}, and double-hinge loss ℓ_{DH}.

Misclassification Rate: The misclassification rate is defined as

$$\mathrm{MR}(g) := \sum_{y=\pm 1} \int I(yg(\boldsymbol{x}) \leq 0) p(\boldsymbol{x}, y) \mathrm{d}\boldsymbol{x} = \mathrm{E}_{p(\boldsymbol{x},y)}[I(yg(\boldsymbol{x}) \leq 0)], \tag{2.16}$$

where $\mathrm{E}_{p(\boldsymbol{x},y)}$ is the expectation over joint distribution, and $I(c)$ is the *indicator* function taking 1 if c is true; otherwise 0. As its name indicates, the misclassification rate computes the rate of the number of misclassified samples. To obtain an accurate classifier regarding the misclassification rate, we train a classifier by minimizing the following risk (the MR risk):

$$R^{\mathrm{MR}}(g) = \mathrm{E}_{p(\boldsymbol{x},y)}[\ell(yg(\boldsymbol{x}))]. \tag{2.17}$$

If we adopt the zero-one loss function for evaluating the MR risk, we recover the original misclassification rate. However, as mentioned, the surrogate loss functions are used for ease of optimization.

Since the underlying *true* distribution $p(\boldsymbol{x}, y)$ is unknown in practice, with *independently and identically distributed* (i.i.d.) *paired* samples drawn from the joint distribution

$$\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x}, y), \tag{2.18}$$

where n is the number of samples, we compute an *empirical* MR risk given by

$$\widehat{R}^{\mathrm{MR}}(g) := \frac{1}{n} \sum_{i=1}^n \ell(y_i g(\boldsymbol{x}_i)). \tag{2.19}$$

Based on empirical risk minimization (ERM), we find a classifier that minimizes the empirical risk.

Let

$$I^{\mathrm{MR}}(g) := \mathrm{E}_{p(\boldsymbol{x},y)}[\ell_{0\text{-}1}(yg(\boldsymbol{x}))] \tag{2.20}$$

be the MR risk with the zero-one loss function over the joint distribution, called the *generalization error* in misclassification rate minimization. Our ultimate goal is to obtain the classifier that minimizes the generalization error, but there is a gap since we use the surrogate loss functions in practice. Although the shape of loss functions is different, Bartlett et al. (2006) elucidated that minimization of the MR risk with a loss function belonging the *classification-calibrated loss* is proved to achieve that with the zero-one loss function asymptotically. For example, the hinge loss is the classification calibrated. Thus, minimization of the MR risk with the hinge loss asymptotically achieves minimization of the generalization error even though the hinge loss is a surrogate of the zero-one loss function.

Area under the Receiver Operating Characteristic Curve (AUC): The misclassification rate has been widely used and studied so far. However, the misclassification rate is not a general metric for classification performance. When the two classes are imbalanced, where the class-prior θ_{P} is far from 0.5, the use of the *area under the receiver operating characteristic curve* (AUC) (Hanley and McNeil, 1982) is a preferable approach (Cortes and Mohri, 2003). AUC is defined as

$$\mathrm{AUC}(g) := \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[I(g(\boldsymbol{x}^{\mathrm{P}}) \geq g(\boldsymbol{x}^{\mathrm{N}}))]], \tag{2.21}$$

where $\mathrm{E}_{\mathrm{P}}[g(\boldsymbol{x})] := \int g(\boldsymbol{x})p(\boldsymbol{x} \mid y = +1)\mathrm{d}\boldsymbol{x}$, $\mathrm{E}_{\mathrm{N}}[g(\boldsymbol{x})] := \int g(\boldsymbol{x})p(\boldsymbol{x} \mid y = -1)\mathrm{d}\boldsymbol{x}$, and $\boldsymbol{x}^{\mathrm{P}}$ and $\boldsymbol{x}^{\mathrm{N}}$ are random variables drawn from $p(\boldsymbol{x} \mid y = +1)$ and $p(\boldsymbol{x} \mid y = -1)$, respectively.

Unlike the misclassification rate, AUC takes into account the order of estimates of a classifier. This property is, in turn, the important for imbalanced classification. For example, when the class-prior is 0.1,[3] the classifier can achieve the higher classification

[3] 10% of samples drawn from the distribution $p(\boldsymbol{x}, y)$ tends to be positive samples and the rest 90% tends to be negative samples.

accuracy 90% by simply classifying all test samples into negative class. However, the classifier cannot necessarily achieve higher AUC.

Based on the above principle, a classifier trained by maximizing AUC has been explored. Firstly, AUC can be transformed as follows:

$$\begin{aligned}\mathrm{AUC}(g) &= \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[I(g(\boldsymbol{x}^{\mathrm{P}}) \geq g(\boldsymbol{x}^{\mathrm{N}}))]] \\ &= 1 - \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[I(g(\boldsymbol{x}^{\mathrm{P}}) < g(\boldsymbol{x}^{\mathrm{N}}))]]. \end{aligned} \tag{2.22}$$

Then, the risk for AUC maximization (the AUC risk) is defined as

$$R^{\mathrm{AUC}}(g) := \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}}))]]. \tag{2.23}$$

If the zero-one loss is used, the risk coincides with $1 - \mathrm{AUC}(g)$. Maximization of AUC corresponds to minimization of the AUC risk R^{AUC}. As well as misclassification rate minimization, a classifier can be trained based on AUC through the ERM framework.

Similarly to the classification-calibrated loss (Bartlett et al., 2006), the consistency of ERM with the surrogate loss in AUC optimization has been studied (Gao and Zhou, 2015; Gao et al., 2016), i.e., whether the AUC risk minimization with a specific surrogate loss function is asymptotically consistent with that with the zero-one loss. It revealed that the AUC risk minimization with the squared loss, exponential loss, and logistic loss functions shown to be consistent, while the hinge loss and absolute loss functions are *not* consistent.

2.3.4 Regularization

As discussed in the previous section, the empirical risk minimization with an appropriate loss function is asymptotically equivalent to the true risk minimization. However, the size of data is finite in practice, applying only ERM leads to a classifier fitted to the observed data completely, but it does not work well on unseen test data, which is known as *overfitting*. To avoid overfitting, several *regularization* techniques has been investigated (Hastie et al., 2009; Shalev-Shwartz and Ben-David, 2014).

The ℓ_2-regularizer is a standard choice. For a linear-in-parameter model $g(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x})$, the ℓ_2-regularizer is defined as

$$W_2(\boldsymbol{w}) := \sum_{\ell=1}^{b} w_\ell^2 = \|\boldsymbol{w}\|_2^2. \tag{2.24}$$

The ℓ_2-regularizer controls complexity of the model in terms of the ℓ_2-norm of the parameter vector.

The ℓ_1-regularizer is also widely used in machine learning, which forces most of the elements of parameter vector to be zero, and leads to a *sparse* parameter vector. The ℓ_1-regularizer is defined as

$$W_1(\boldsymbol{w}) := \sum_{\ell=1}^{b} |w_\ell| = \|\boldsymbol{w}\|_1. \tag{2.25}$$

With the regularizational functional $W(g)$, we obtain the trained classifier by solving the following optimization problem:

$$\widehat{g} := \operatorname*{argmin}_{g} \Big[\widehat{R}(g) + W(g)\Big]. \tag{2.26}$$

2.4 PN Classification (Supervised Classification)

To solve binary or positive-negative (PN) classification problems, supervised classification can be applied with positive and negative (labeled) samples.

Recall that the MR risk is defined as

$$R^{\mathrm{MR}}(g) = \mathrm{E}_{p(\boldsymbol{x},y)}[\ell(yg(\boldsymbol{x}))]. \tag{2.27}$$

Let us define the MR risk in PN classification (the PN-MR risk) by decomposing the MR risk in terms of class:

$$R^{\mathrm{MR}}(g) = \theta_{\mathrm{P}}\,\mathrm{E}_{\mathrm{P}}[\ell(g(\boldsymbol{x}))] + \theta_{\mathrm{N}}\,\mathrm{E}_{\mathrm{N}}[\ell(-g(\boldsymbol{x}))] := R^{\mathrm{MR}}_{\mathrm{PN}}(g). \tag{2.28}$$

The goal of PN classification is to obtain a classifier that minimizes the PN-MR risk as small as it can. In ERM, we try to accomplish such a classifier by using samples collected from the distributions.

Suppose we are given positive and negative samples drawn from class-conditional distributions:

$$\{\boldsymbol{x}^{\mathrm{P}}_i\}_{i=1}^{n_{\mathrm{P}}} \overset{i.i.d.}{\sim} p(\boldsymbol{x} \mid y=+1), \tag{2.29}$$

$$\{\boldsymbol{x}^{\mathrm{N}}_j\}_{j=1}^{n_{\mathrm{N}}} \overset{i.i.d.}{\sim} p(\boldsymbol{x} \mid y=-1). \tag{2.30}$$

With these samples, the empirical PN-MR risk can be obtained by

$$\widehat{R}_{\mathrm{PN}}^{\mathrm{MR}}(g) := \frac{\theta_{\mathrm{P}}}{n_{\mathrm{P}}} \sum_{i=1}^{n_{\mathrm{P}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}})) + \frac{\theta_{\mathrm{N}}}{n_{\mathrm{N}}} \sum_{j=1}^{n_{\mathrm{N}}} \ell(-g(\boldsymbol{x}_i^{\mathrm{N}})). \tag{2.31}$$

We obtain a classifier by minimizing the empirical PN risk with, e..g, the ℓ_2-regularizer.

2.5 PU Classification

In this section, we review an existing PU learning method based on empirical risk minimization and theoretical comparisons of PU classification against PN classification.

2.5.1 Empirical Risk Minimization Approach

Recently, du Plessis et al. (2014, 2015) proposed methods based on an empirical risk minimization framework for PU classification. Their idea is to derive a risk equivalent to the PN-MR risk from only PU data without any distributional assumptions on the data. Note that they focused on the misclassification rate as a classification performance measure.

Suppose that we are given *positive* (P) and *unlabeled* (U) data:

$$\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}} \overset{i.i.d.}{\sim} p(\boldsymbol{x} \mid y = +1), \tag{2.32}$$

$$\{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}} \overset{i.i.d.}{\sim} p(\boldsymbol{x}) := \theta_{\mathrm{P}} p(\boldsymbol{x} \mid y = +1) + \theta_{\mathrm{N}} p(\boldsymbol{x} \mid y = -1), \tag{2.33}$$

where $\theta_{\mathrm{P}} := p(y = +1)$ and $\theta_{\mathrm{N}} := p(y = -1)$ are the class-prior probabilities. By definition of the marginal density, we obtain

$$\mathrm{E}_{\mathrm{U}}[\ell(-g(\boldsymbol{x}))] = \theta_{\mathrm{P}}\, \mathrm{E}_{\mathrm{P}}[\ell(-g(\boldsymbol{x}))] + \theta_{\mathrm{N}}\, \mathrm{E}_{\mathrm{N}}[\ell(-g(\boldsymbol{x}))]. \tag{2.34}$$

Equivalently,

$$\theta_{\mathrm{N}}\, \mathrm{E}_{\mathrm{N}}[\ell(-g(\boldsymbol{x}))] = \mathrm{E}_{\mathrm{U}}[\ell(-g(\boldsymbol{x}))] - \theta_{\mathrm{P}}\, \mathrm{E}_{\mathrm{P}}[\ell(-g(\boldsymbol{x}))]. \tag{2.35}$$

Recall the PN-MR risk

$$R_{\mathrm{PN}}^{\mathrm{MR}}(g) = \theta_{\mathrm{P}}\, \mathrm{E}_{\mathrm{P}}[\ell(g(\boldsymbol{x}))] + \theta_{\mathrm{N}}\, \mathrm{E}_{\mathrm{N}}[\ell(-g(\boldsymbol{x}))]. \tag{2.36}$$

By plugging Eq. (2.35) into the second term of the PN-MR risk, we obtain the risk in PU classification (the PU-MR risk):

$$\begin{aligned} R_{\mathrm{PN}}^{\mathrm{MR}}(g) &= \theta_{\mathrm{P}}\,\mathrm{E}_{\mathrm{P}}[\ell(g(\boldsymbol{x})) - \ell(-g(\boldsymbol{x}))] + \mathrm{E}_{\mathrm{U}}[\ell(-g(\boldsymbol{x}))] \\ &= \theta_{\mathrm{P}}\,\mathrm{E}_{\mathrm{P}}[\widetilde{\ell}(g(\boldsymbol{x}))] + \mathrm{E}_{\mathrm{U}}[\ell(-g(\boldsymbol{x}))] := R_{\mathrm{PU}}^{\mathrm{MR}}(g), \end{aligned} \tag{2.37}$$

where

$$\widetilde{\ell}(m) := \ell(m) - \ell(-m) \tag{2.38}$$

is a composite loss function. Thus, without distributional assumptions such as the cluster assumption, we have the following lemma:

Lemma 1. *The MR, PN-MR, and PU-MR risks are equivalent:*

$$R^{\mathrm{MR}}(g) = R_{\mathrm{PN}}^{\mathrm{MR}}(g) = R_{\mathrm{PN}}^{\mathrm{PU}}(g). \tag{2.39}$$

The above lemma also shows that the minimizers of the PN-MR and PU-MR risks are equivalent even though their expressions are different. In contrast to the PN-MR risk, the PU-MR risk can be computed by the expectations over P and U data. Therefore, we can train a classifier by using only PU data. In practice, we use the empirical PU-MR risk:

$$\widehat{R}_{\mathrm{PU}}^{\mathrm{MR}}(g) := \frac{\theta_{\mathrm{P}}}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\widetilde{\ell}(g(\boldsymbol{x}_i^{\mathrm{P}})) + \frac{1}{n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(-g(\boldsymbol{x}_k^{\mathrm{U}})), \tag{2.40}$$

where the expectations are replaced with the corresponding sample averages. The empirical PU-MR risk converges to the PU-MR risk, i.e., the MR risk in the order of $\mathcal{O}_p(1/\sqrt{n_{\mathrm{P}}} + 1/\sqrt{n_{\mathrm{U}}})$, which is the optimal convergence rate without any additional distributional assumptions (Vapnik, 1998).

Non-Convex Approach: If the loss function satisfies

$$\ell(m) + \ell(-m) = 1, \tag{2.41}$$

the composite loss function becomes $\widetilde{\ell}(m) = 2\ell(m) - 1$. We thus obtain the *non-convex* PU-MR risk and the empirical risk as

$$R^{\mathrm{MR}}_{\text{N-PU}}(g) := 2\theta_{\mathrm{P}}\,\mathrm{E}_{\mathrm{P}}[\ell(g(\boldsymbol{x}))] + \mathrm{E}_{\mathrm{U}}[\ell(-g(\boldsymbol{x}))] - \theta_{\mathrm{P}}, \tag{2.42}$$

$$\widehat{R}^{\mathrm{MR}}_{\text{N-PU}}(g) := \frac{2\theta_{\mathrm{P}}}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}^{\mathrm{P}}_i)) + \frac{1}{n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(-g(\boldsymbol{x}^{\mathrm{U}}_k)) - \theta_{\mathrm{P}}. \tag{2.43}$$

This formulation can be seen as cost-sensitive classification of P and U data with weight $2\theta_{\mathrm{P}}$ (du Plessis et al., 2014).

The ramp loss used in the robust support vector machine (Collobert et al., 2006),

$$\ell_{\mathrm{R}}(m) := \frac{1}{2}\max(0, \min(2, 1 - m)), \tag{2.44}$$

satisfies the condition (2.41). However, the use of the ramp loss (and any other losses that satisfy the condition (2.41)) yields a non-convex optimization problem, which may be solved locally by the *concave-convex procedure* (CCCP) (Yuille and Rangarajan, 2002; Collobert et al., 2006; du Plessis et al., 2014).

Convex Approach: If a convex surrogate loss function satisfies

$$\ell(m) - \ell(-m) = -m, \tag{2.45}$$

the composite loss function becomes a linear function $\widetilde{\ell}(m) = -m$ (du Plessis et al., 2015). For example, the logistic loss $\ell_{\mathrm{L}}(m) = \log(1 + \exp(-m))$, double-hinge loss $\ell_{\mathrm{DH}}(m) = \max(-2m, \max(0, 1 - m))$, scaled squared loss $(m - 1)^2/4$ satisfy the condition. We thus obtain the *convex* PU-MR risk and the empirical risk as

$$R^{\mathrm{MR}}_{\text{C-PU}}(g) := \theta_{\mathrm{P}}\,\mathrm{E}_{\mathrm{P}}[-g(\boldsymbol{x})] + \mathrm{E}_{\mathrm{U}}[\ell(-g(\boldsymbol{x}))], \tag{2.46}$$

$$\widehat{R}^{\mathrm{MR}}_{\text{C-PU}}(g) := -\frac{\theta_{\mathrm{P}}}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}}g(\boldsymbol{x}^{\mathrm{P}}_i) + \frac{1}{n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(-g(\boldsymbol{x}^{\mathrm{U}}_k)). \tag{2.47}$$

This formulation yields the convex optimization problem that can be solved efficiently.

NU Classification: As a mirror of PU classification, we can consider NU classification. The risk in NU classification (the NU-MR risk) and its empirical version are given by

$$R_{\mathrm{NU}}^{\mathrm{MR}}(g) := \theta_{\mathrm{N}}\,\mathrm{E}_{\mathrm{N}}[\widetilde{\ell}(-g(\boldsymbol{x}))] + \mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}))], \tag{2.48}$$

$$\widehat{R}_{\mathrm{NU}}^{\mathrm{MR}}(g) := \frac{\theta_{\mathrm{N}}}{n_{\mathrm{N}}}\sum_{j=1}^{n_{\mathrm{N}}}\widetilde{\ell}(-g(\boldsymbol{x}_j^{\mathrm{N}})) + \frac{1}{n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(g(\boldsymbol{x}_k^{\mathrm{U}})). \tag{2.49}$$

NU classification is used in theoretical analysis in the next section and semi-supervised learning methods in Chapter 5.

Here, without distributional assumptions, we have the following lemma:

Lemma 2. *The MR, PN-MR, PU-MR, and NU-MR risks are equivalent:*

$$R^{\mathrm{MR}}(g) = R_{\mathrm{PN}}^{\mathrm{MR}}(g) = R_{\mathrm{PN}}^{\mathrm{PU}}(g) = R_{\mathrm{PN}}^{\mathrm{NU}}(g). \tag{2.50}$$

In contrast to the PN-MR risk, the NU-MR risk can be computed by the expectations over N and U data. Therefore, we can train a classifier by using only NU data. Additionally, Lemma 2 shows that the PU-MR and NU-MR risks are equivalent. However, the practical behavior is different. Indeed, in a theoretical analysis in the subsequent section, we compare PN, PU, and NU classification methods and see that theoretically expected performances are different depending on the number of P, N, and U samples and the class-prior θ_{P}.

Similarly to PU classification, the non-convex and convex NU-MR risks are expressed as

$$R_{\text{N-NU}}^{\mathrm{MR}}(g) := 2\theta_{\mathrm{N}}\,\mathrm{E}_{\mathrm{N}}[\ell(-g(\boldsymbol{x}))] + \mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}))] - \theta_{\mathrm{N}}, \tag{2.51}$$

$$R_{\text{C-NU}}^{\mathrm{MR}}(g) := \theta_{\mathrm{N}}\,\mathrm{E}_{\mathrm{N}}[g(\boldsymbol{x})] + \mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}))]. \tag{2.52}$$

In practice, the empirical versions are used for training a classifier:

$$\widehat{R}_{\text{N-NU}}^{\mathrm{MR}}(g) := \frac{2\theta_{\mathrm{N}}}{n_{\mathrm{N}}}\sum_{j=1}^{n_{\mathrm{N}}}\ell(-g(\boldsymbol{x}_j^{\mathrm{N}})) + \frac{1}{n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(g(\boldsymbol{x}_k^{\mathrm{U}})) - \theta_{\mathrm{N}}, \tag{2.53}$$

$$\widehat{R}_{\text{C-NU}}^{\mathrm{MR}}(g) := \frac{\theta_{\mathrm{N}}}{n_{\mathrm{N}}}\sum_{j=1}^{n_{\mathrm{N}}}g(\boldsymbol{x}_j^{\mathrm{N}}) + \frac{1}{n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(g(\boldsymbol{x}_k^{\mathrm{U}})). \tag{2.54}$$

2.5.2 Theoretical Comparisons of PU Classification against Supervised Classification

du Plessis et al. (2014, 2015) proposed the MR risk in PU classification equivalent to the MR risk in PN (supervised) classification. Even though those two risks are equivalent, from a qualitative point of view, those risks utilize different data. One may ask a question that which learning approach, PU or PN classification, is better? To answer this question, Niu et al. (2016) theoretically compared the classification performance between PU and PN classification regarding estimation error bounds.

We assume that $\mathcal{G}$ is a set of functions for the classifiers defined as

$$\mathcal{G} := \{g(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}) \mid \|\boldsymbol{w}\| \leq C_{\boldsymbol{w}};\ \forall \boldsymbol{x}\colon \|\boldsymbol{\phi}(\boldsymbol{x})\| \leq C_{\boldsymbol{\phi}}\}, \tag{2.55}$$

where $C_{\boldsymbol{w}} > 0$ and $C_{\boldsymbol{\phi}} > 0$ are certain positive constants. This assumption is reasonable because the ℓ_2-regularizer included in training and the use of bounded basis functions, e.g., the Gaussian kernel basis, ensures that the minimizer of the empirical risk belongs to such the function class $\mathcal{G}$. Let us define the PN, PU, and NU classifiers as

$$\widehat{g}_{\mathrm{PN}} := \operatorname*{argmin}_{g \in \mathcal{G}}\ \widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(g), \tag{2.56}$$

$$\widehat{g}_{\mathrm{PU}} := \operatorname*{argmin}_{g \in \mathcal{G}}\ \widehat{R}^{\mathrm{MR}}_{\mathrm{PU}}(g), \tag{2.57}$$

$$\widehat{g}_{\mathrm{NU}} := \operatorname*{argmin}_{g \in \mathcal{G}}\ \widehat{R}^{\mathrm{MR}}_{\mathrm{NU}}(g), \tag{2.58}$$

respectively.

The estimation error bounds can be obtained separately with probability at least $1-\delta$:

$$R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PN}}) - R^{\mathrm{MR}}(g^*) \leq h^{\mathrm{MR}}(\delta)\Big(\frac{\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}}\Big), \tag{2.59}$$

$$R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PU}}) - R^{\mathrm{MR}}(g^*) \leq h^{\mathrm{MR}}(\delta)\Big(\frac{2\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{2.60}$$

$$R^{\mathrm{MR}}(\widehat{g}_{\mathrm{NU}}) - R^{\mathrm{MR}}(g^*) \leq h^{\mathrm{MR}}(\delta)\Big(\frac{2\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{2.61}$$

where $g^* := \operatorname{argmin}_{g \in \mathcal{G}}\ R^{\mathrm{MR}}(g)$ is the optimal decision function and $h^{\mathrm{MR}}(\delta) := 4L_\ell C_{\boldsymbol{w}} C_{\boldsymbol{\phi}} + \sqrt{2\log(4/\delta)}$.

We then compare the bounds between PN and PU (NU) classification. Since $h(\delta)\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}}$ (resp., $h(\delta)\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}}$) appears both of Eq. (2.59) and Eq. (2.60) (resp.,

Eq. 2.61), we subtract the term from both bounds and compute the ratio of the two bounds:

$$\rho_{\mathrm{PU,PN}} := \frac{\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}} + 1/\sqrt{n_{\mathrm{U}}}}{\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}}}, \tag{2.62}$$

$$\rho_{\mathrm{NU,PN}} := \frac{\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}} + 1/\sqrt{n_{\mathrm{U}}}}{\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}}}. \tag{2.63}$$

When $\rho_{\mathrm{PU,PN}} < 1$, PU classification is more promising than PN classification; on the other hand, $\rho_{\mathrm{PU,PN}} > 1$ means that PN classification is more promising than PU classification. Similarly, we can compare NU and PN classification by $\rho_{\mathrm{NU,PN}}$.

Furthermore, when vast amounts of unlabeled data are available ($n_{\mathrm{U}} \to \infty$ which is faster than $n_{\mathrm{P}} \to \infty$ and $n_{\mathrm{N}} \to \infty$), we have the results of *asymptotic comparison* (Niu et al., 2016):

$$\bar{\rho}_{\mathrm{PU,PN}} := \lim_{n_{\mathrm{U}}\to\infty} \rho_{\mathrm{PU,PN}} = \frac{\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}}}{\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}}}, \tag{2.64}$$

$$\bar{\rho}_{\mathrm{NU,PN}} := \lim_{n_{\mathrm{U}}\to\infty} \rho_{\mathrm{NU,PN}} = \frac{\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}}}{\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}}}. \tag{2.65}$$

Since

$$\bar{\rho}_{\mathrm{PU,PN}} \times \bar{\rho}_{\mathrm{NU,PN}} = 1, \tag{2.66}$$

there are only three cases needed to be considered:

$$\bar{\rho}_{\mathrm{PU,PN}} < 1 \text{ and } \bar{\rho}_{\mathrm{NU,PN}} > 1, \tag{2.67}$$

$$\bar{\rho}_{\mathrm{PU,PN}} > 1 \text{ and } \bar{\rho}_{\mathrm{NU,PN}} < 1, \tag{2.68}$$

$$\bar{\rho}_{\mathrm{PU,PN}} = 1 \text{ and } \bar{\rho}_{\mathrm{NU,PN}} = 1. \tag{2.69}$$

The first and second cases indicate that the classifier based on either PU or NU classification is more promising than that based on PN classification although PU or NU classification can not access negative or positive data, respectively. The exception is the third case which happens when $n_{\mathrm{P}}/n_{\mathrm{N}} = \theta_{\mathrm{P}}^2/\theta_{\mathrm{N}}^2$, but the probability of occurring the third case is close to zero because n_{P} and n_{N} are independently determined by θ_{P} and θ_{N}, in practice.

2.6 Semi-Supervised Classification

In this section, we review the existing semi-supervised learning algorithms, which are compared with our methods in experiments.

To simplify the notation, we introduce the followings:

$$\{(\boldsymbol{x}_i^{\mathrm{L}}, y_i)\}_{i=1}^{n_{\mathrm{L}}} = \{(\boldsymbol{x}_i^{\mathrm{P}}, +1)\}_{i=1}^{n_{\mathrm{P}}} \cup \{(\boldsymbol{x}_j^{\mathrm{N}}, -1)\}_{j=1}^{n_{\mathrm{N}}},$$

where $n_{\mathrm{L}} = n_{\mathrm{P}} + n_{\mathrm{N}}$ and $\{c_i\}_{i=1}^{n_{\mathrm{L}}}$ is a set of weights taking $c_i = \theta_{\mathrm{P}}/n_{\mathrm{P}}$ if $y_i = +1$ and $\theta_{\mathrm{N}}/n_{\mathrm{N}}$ if $y_i = -1$.

2.6.1 Entropy Regularization

Based on the *entropy minimization principle*, *entropy regularization* (ER) (Grandvalet and Bengio, 2004) was proposed. From unlabeled samples. ER computes the conditional entropy of class labels conditioned on the inputs.

Specifically, let $q_{\boldsymbol{w}}(y \mid \boldsymbol{x})$ be a model of posterior distribution $p(y \mid \boldsymbol{x})$. The objective function consists of the likelihood function and the entropy term:

$$\max_{\boldsymbol{w} \in \mathbb{R}^b} \left[\sum_{i=1}^{n_{\mathrm{L}}} c_i \log q_{\boldsymbol{w}}(y_i \mid \boldsymbol{x}_i^{\mathrm{L}}) + \lambda_{\mathrm{E}} \sum_{k=1}^{n_{\mathrm{U}}} \sum_{y=\pm 1} q_{\boldsymbol{w}}(y \mid \boldsymbol{x}_k^{\mathrm{U}}) \log q_{\boldsymbol{w}}(y \mid \boldsymbol{x}_k^{\mathrm{U}}) \right], \tag{2.70}$$

where $\lambda_{\mathrm{E}} \geq 0$ is the regularization parameter. The first term can be regarded as (weighted) conditional likelihood, and the second term can be regarded as negative entropy. The entropy regularization is known as that it favors a decision boundary lying on a low-density region (Chapelle et al., 2006). Therefore, if the low-density separation principle does not hold on data points, the regularization scheme does not work well.

From the optimization viewpoint, the objective function in Eq. (2.70) is non-convex. Hence, the solution which we obtain is usually local optima. Following to Grandvalet and Bengio (2006), we first minimize the objective without entropy regularization, i.e., ordinary logistic regression, and then minimize the objective in Eq. (2.70) using the first solution as an initial solution for a quasi-newton method (Nocedal and Wright, 2006).

2.6.2 Manifold Regularization

The *manifold assumption* presumes that data points are supported on a (near) low-dimensional manifold (Belkin et al., 2006). Based on the manifold assumption, Belkin

et al. (2006) proposed the *manifold regularization* approach, encoding a manifold structure in the data to a regularizer.

Laplacian support vector machines (LapSVM) (Belkin et al., 2006) is the *support vector machine* (SVM) (Cortes and Vapnik, 1995) with the manifold regularization. Let us define a kernel model as $g(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i)$, where α is dual variable, $K(\boldsymbol{x}, \boldsymbol{x}')$ is the kernel function. The problem of LapSVM (in primal) (Melacci and Belkin, 2011) can be expressed as

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \left[\sum_{i=1}^{n_{\mathrm{L}}} c_i \ell_{\mathrm{SH}}(1 - y_i \boldsymbol{k}_i^\top \boldsymbol{\alpha}) + C_{\mathrm{A}} \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha} + C_{\mathrm{I}} \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{L} \boldsymbol{K} \boldsymbol{\alpha} \right], \tag{2.71}$$

where C_{A} is the weight of the norm of the function (or *ambient* norm), C_{I} is the weight of the norm of function in the low dimensional manifold (or *intrinsic* norm), $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, $\boldsymbol{W}$ is the adjacency matrix of the data graph, $\boldsymbol{D}$ is the diagonal matrix whose element is $D_{i,i} = \sum_{j=1}^{n} W_{i,j}$, $k_i^{(j)} = K(\boldsymbol{x}_i^{\mathrm{L}}, \boldsymbol{x}_j)$, and $K_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Compared with the originally proposed LapSVM objective function (Belkin et al., 2006), the above objective employs the squared hinge loss ℓ_{SH} for labeled samples so that its gradient can be computed, unlike the hinge loss. Melacci and Belkin (2011) showed that solving the above objective function by preconditioned conjugate gradient along with early stopping significantly reduces the computation time.

2.6.3 Squared-Loss Mutual Information Regularization

Squared-Loss Mutual Information Regularization (SMIR) (Niu et al., 2013) is based on the *information maximization principle* (Linsker, 1988). The idea of SMIR is to learn the class-posterior probability $p(y \mid \boldsymbol{x})$ while maximizing *squared-loss mutual information* (SMI) (Suzuki et al., 2009) defined as

$$\mathrm{SMI} := \frac{1}{2} \sum_{y=\pm 1} \int p(\boldsymbol{x}) p(y) \Big(\frac{p(\boldsymbol{x}, y)}{p(\boldsymbol{x}) p(y)} - 1 \Big)^2 \mathrm{d}\boldsymbol{x}. \tag{2.72}$$

Unlike maximization of *mutual information* (Cover and Thomas, 2006), the optimization problem of SMIR is strictly convex under mild conditions. It thus can be solved efficiently and achieves the unique globally optimal solution.

Specifically, let $q_{\boldsymbol{\alpha}}(y \mid \boldsymbol{x}) := \boldsymbol{\alpha}_y^\top \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{K}^{-\frac{1}{2}} \boldsymbol{\phi}_n(\boldsymbol{x})$ be the model of the class-posterior probability, where $K_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\boldsymbol{D} = \mathrm{diag}(d_1, \ldots, d_n)$, and $d_i = \sum_{j=1}^{n} K(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

The optimization problem of SMIR can be formulated as

$$\min_{\boldsymbol{A}\in\mathbb{R}^{n\times 2}}\left[\widehat{\Delta}_2(p,q_{\boldsymbol{\alpha}})-\gamma_S\widehat{\mathrm{SMI}}+\frac{\lambda_S}{2}\operatorname{tr}(\boldsymbol{A}^\top\boldsymbol{A})\right], \tag{2.73}$$

where $\boldsymbol{A}:=(\boldsymbol{\alpha}_{+1},\boldsymbol{\alpha}_{-1})$ is the matrix representation of model parameters, γ_S and λ_S are parameters for trading off between the SMI regularization and ℓ_2-regularization. $\Delta_2(p,q)$ is the squared difference between p and q defined as

$$\Delta_2(p,q):=\frac{1}{2}\sum_{y=\pm1}\int(p(y\mid\boldsymbol{x})-q(y\mid\boldsymbol{x}))^2p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}. \tag{2.74}$$

As explained, if the condition $\lambda_S > \eta/(n\min_{y=\pm1}p(y))$ is satisfied, the optimization problem can be strictly convex with respect to $\boldsymbol{A}$.

2.6.4 Maximum-Margin Principle

SVM has been widely used in many applications and known as obtaining a large margin decision boundary. Based on the *maximum-margin principle*, *semi-supervised SVM* (S3VM) aims at learning a classifier by minimizing the risk computed from labeled data while maximizing margin on unlabeled data (Chapelle et al., 2008).

While the original S3VM solves the non-convex optimization problem, weakly labeled support vector machines (WellSVM) proposed by Li et al. (2013) solve the tightly relaxed convex optimization problem, which can be solved efficiently.

The optimization problem of WellSVM can be expressed as

$$\min_{\{\mu_t\}\in\mathcal{M}}\max_{\boldsymbol{\alpha}\in\mathcal{A}}\left[\mathbf{1}^\top\boldsymbol{\alpha}-\frac{1}{2}\boldsymbol{\alpha}^\top\Big[\sum_{t:\widehat{\boldsymbol{y}}_t\in\mathcal{B}}\mu_t\Big(\boldsymbol{K}\odot(\widehat{\boldsymbol{y}}_t\widehat{\boldsymbol{y}}_t^\top)\Big)\Big]\boldsymbol{\alpha}\right], \tag{2.75}$$

where

$$\mathcal{A}=\{\boldsymbol{\alpha}\mid 0\le\alpha_i\le C_{\mathrm{L}}c_i,\ 0\le\alpha_j\le C_{\mathrm{U}},\ i\in\mathcal{L},\ j\in\mathcal{U}\}, \tag{2.76}$$

$$\mathcal{B}=\{\widehat{\boldsymbol{y}}\mid\widehat{y}_i=y_i,\ \widehat{y}_j\in\{\pm1\},\ \frac{1}{n_{\mathrm{U}}}\sum_{j}^{n_{\mathrm{U}}}I(\widehat{y}_j=+1)=\theta_{\mathrm{p}},\ i\in\mathcal{L},\ j\in\mathcal{U}\}, \tag{2.77}$$

$\odot$ denotes the element-wise multiplication (a.k.a. *Hadamard* product), C_{L} and C_{U} are the regularization parameter on the losses for labeled and unlabeled data, $\mathcal{L}$ and $\mathcal{U}$ are

the index sets for labeled and unlabeled data, and $I(\cdot)$ is the indicator function. To solve Eq.(2.75), the authors utilized the *cutting plane algorithm* (Kelley, 1960) by iteratively generating label. More specifically, we first initialize a label vector $\widehat{\boldsymbol{y}}$ and the working set $\mathcal{C}$ to $\widehat{\boldsymbol{y}}$. We then solve Eq.(2.75) with respect to $\boldsymbol{\alpha}$ by standard supervised learning methods. After that, we generate a violated vector $\widehat{\boldsymbol{y}}$ (Li et al., 2013), and add it into $\mathcal{C}$. We repeat the above procedure until the decrease of objective function is smaller than a threshold.

2.6.5 Safe Approach

As discussed in Section 1.2.3, once the distributional assumptions are not satisfied, the performance of the existing methods becomes even worse than the method based on supervised learning. To mitigate this problem, a *safe* semi-supervised learning approach was proposed to avoid undesirable performance degeneration (Li and Zhou, 2015).

Based on the safe approach, *safe S3VM* (S4VM) (Li and Zhou, 2015) was proposed so that its performance is not worse than the ordinary supervised SVM. Specifically, let $\boldsymbol{y}^* \in \mathbb{R}^{n_\mathrm{U}}$ be the ground-truth label assignment and $\boldsymbol{y}^{\mathrm{svm}} \in \mathbb{R}^{n_\mathrm{U}}$ be the predictive labels of SVM on unlabeled instances. The $\mathrm{gain}(\boldsymbol{y}, \boldsymbol{y}^*, \boldsymbol{y}^{\mathrm{svm}})$ and $\mathrm{loss}(\boldsymbol{y}, \boldsymbol{y}^*, \boldsymbol{y}^{\mathrm{svm}})$ respectively measure the gained and lost accuracies compared to SVM, where $\boldsymbol{y} := (y_1, \ldots, y_{n_\mathrm{U}})^\top$. The goal of S4VM is to maximize the improvement against SVM:

$$\widehat{\boldsymbol{y}} = \underset{\boldsymbol{y} \in \{\pm 1\}^{n_\mathrm{U}}}{\mathrm{argmax}} \ \min_{\bar{\boldsymbol{y}} \in \mathcal{M}} \left[\mathrm{gain}(\boldsymbol{y}, \bar{\boldsymbol{y}}, \boldsymbol{y}^{\mathrm{svm}}) - \lambda_{\mathrm{S4VM}} \cdot \mathrm{loss}(\boldsymbol{y}, \bar{\boldsymbol{y}}, \boldsymbol{y}^{\mathrm{svm}}) \right], \tag{2.78}$$

where

$$\mathrm{gain}(\boldsymbol{y}, \bar{\boldsymbol{y}}, \boldsymbol{y}^{\mathrm{svm}}) = \sum_{j=n_\mathrm{L}+1}^{n_\mathrm{L}+n_\mathrm{U}} \left(c_\mathrm{P} \frac{1+y_j}{2} + c_\mathrm{N} \frac{1-y_j}{2} \right) \frac{1+y_j \bar{y}_j}{2} \frac{1-y_j^{\mathrm{svm}} \bar{y}_j}{2}, \tag{2.79}$$

$$\mathrm{loss}(\boldsymbol{y}, \bar{\boldsymbol{y}}, \boldsymbol{y}^{\mathrm{svm}}) = \sum_{j=n_\mathrm{L}+1}^{n_\mathrm{L}+n_\mathrm{U}} \left(c_\mathrm{P} \frac{1+y_j}{2} + c_\mathrm{N} \frac{1-y_j}{2} \right) \frac{1-y_j \bar{y}_j}{2} \frac{1+y_j^{\mathrm{svm}} \bar{y}_j}{2}, \tag{2.80}$$

$c_\mathrm{P} = \theta_\mathrm{P}/n_\mathrm{P}$, $c_\mathrm{N} = \theta_\mathrm{N}/n_\mathrm{N}$, and λ_{S4VM} is a parameter for controlling the trade-off between gain and loss functions. S4VM assumes that the ground-truth is realized by a low-density separator, i.e., $\boldsymbol{y}^* \in \mathcal{M}$, where $\mathcal{M} := \{\bar{\boldsymbol{y}}_t\}_{t=1}^T$ is a pool of low-density separators obtained by, e.g., S3VM. If the class-prior of unlabeled data is θ_P different from that of labeled data,

we restrict that the assigned labels $\boldsymbol{y}$ are in $\mathcal{B} = \{\boldsymbol{y} \in \{\pm 1\}^{n_{\mathrm{U}}} \mid -\beta \leq \frac{1}{n_{\mathrm{U}}} \sum_{i=1}^{n_{\mathrm{U}}} y_i - \theta_{\mathrm{P}} \leq \beta\}$, where β is a small constant controlling the inconsistency of class prior.

Although S4VM can avoid unexpected performance degeneration, its core mechanism for utilizing unlabeled data still relies on the low-density separation principle. Thus, the performance degeneration may not happen, but S4VM cannot leverage information of unlabeled data.

2.7 Class-Prior Estimation Methods

In this section, we review class-prior estimation methods used in numerical experiments in this dissertation.

2.7.1 Class-Prior Shift

In practice, we are often faced with *dataset shift* or *distribution change*, where distributions between training data and test data are different:

$$p_{\mathrm{tr}}(\boldsymbol{x}, y) \neq p_{\mathrm{te}}(\boldsymbol{x}, y), \tag{2.81}$$

where p_{tr} and p_{te} are training and test densities, respectively. When class-conditional densities are the same but class-priors are different between training and test distributions, the situation is called *class-prior shift* (Quiñonero Candela et al., 2009).[4] Formally, class-prior shift can be expressed as

$$p_{\mathrm{tr}}(\boldsymbol{x} \mid y) = p_{\mathrm{te}}(\boldsymbol{x} \mid y), \tag{2.82}$$

$$p_{\mathrm{tr}}(y) \neq p_{\mathrm{te}}(y). \tag{2.83}$$

Class-prior shift occurs in many real-world applications. For example, suppose that we would like to obtain a classifier for discriminating a face image into male or female. The classifier can be trained with, for instance, face images collected from a lab. However, the face images often contains mostly face images from males and smaller size of face images from females. Thus, if we simply apply an ordinary classification method, a learned classifier will be reflected by the proportion of male and female faces in the training data. On the other hand, in real-world, the proportion of male and female is almost equal. Thus, the learned classifier on training data will perform poorly on test distribution.

[4] There are other situations studied such as *covariate shift* and *target shift*.

To cope with the class-prior shift situation, we train a classifier with training samples weighted by class-priors of test data. For example, if we have domain knowledge about class-priors of test distribution, we can train a classifier with the class-priors, e.g., the use of a weighted SVM with prespecified weights. However, it is not always the case; we need an estimate of class-prior of test data.

For estimating a class-prior, several class-prior estimation methods have been proposed for both PU and semi-supervised learning situations, and shown its usefulness in experiments (Kawakubo et al., 2016; Ramaswamy et al., 2016; du Plessis et al., 2017).

2.7.2 Class-Prior Estimation under Semi-Supervised Learning Setting

Here, we review a class-prior estimation method based on *energy distance* minimization (Kawakubo et al., 2016), which employs the energy distance (Székely and Rizzo, 2013) for measuring the difference between two distributions.

The energy distance (ED) between densities p and q is defined as

$$\mathrm{ED}(p,q) := \int \|\psi_p(\boldsymbol{t}) - \psi_q(\boldsymbol{t})\|^2 \Big(\frac{\pi^{\frac{d+1}{2}}}{\Gamma(\frac{d+1}{2})} \|\boldsymbol{t}\|^{d+1} \Big) \mathrm{d}\boldsymbol{t}, \tag{2.84}$$

where $\|\cdot\|$ denotes the Euclidean distance, Γ is the *gamma* function, and ψ_p and ψ_q are the characteristic functions of p and q, respectively. An advantage of the energy distance is that it can be expressed as expectations of Euclid distance:

$$\mathrm{ED}(p,q) = 2\,\mathrm{E}_{\boldsymbol{x}\sim p,\bar{\boldsymbol{x}}\sim q}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|] - \mathrm{E}_{\boldsymbol{x},\bar{\boldsymbol{x}}\sim p}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|] - \mathrm{E}_{\boldsymbol{x},\bar{\boldsymbol{x}}\sim q}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|]. \tag{2.85}$$

Let us define a distribution model parameterized by class-priors as

$$q_\theta(\boldsymbol{x}) = \theta_{\mathrm{P}} p(\boldsymbol{x} \mid y=+1) + \theta_{\mathrm{N}} p(\boldsymbol{x} \mid y=-1). \tag{2.86}$$

To obtain estimated class-priors, we minimize the energy distance between a marginal density p and the distribution model $q_{\theta_{\mathrm{P}}}$. The estimated class-prior can be obtained by

minimizing the energy distance with respect to θ_{P} and θ_{N}:

$$\widehat{\theta}_{\mathrm{P}}, \widehat{\theta}_{\mathrm{N}} = \underset{\theta=\{\theta_{\mathrm{P}},\theta_{\mathrm{N}}\}}{\operatorname{argmin}} \ \mathrm{ED}(p, q_\theta), \tag{2.87}$$

$$\begin{aligned}\mathrm{ED}(p, q_\theta) = {} & 2\theta_{\mathrm{P}} \, \mathrm{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}),\bar{\boldsymbol{x}}\sim p(\boldsymbol{x}|y=+1)}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|] + 2\theta_{\mathrm{N}} \, \mathrm{E}_{\boldsymbol{x}\sim p,\bar{\boldsymbol{x}}\sim p(\boldsymbol{x}|y=-1)}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|] \\ & - \mathrm{E}_{\boldsymbol{x},\bar{\boldsymbol{x}}\sim p}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|] - \theta_{\mathrm{P}}^2 \, \mathrm{E}_{\boldsymbol{x},\bar{\boldsymbol{x}}\sim p(\boldsymbol{x}|y=+1)}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|] \\ & - \theta_{\mathrm{N}}^2 \, \mathrm{E}_{\boldsymbol{x},\bar{\boldsymbol{x}}\sim p(\boldsymbol{x}|y=-1)}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|] \\ & + \theta_{\mathrm{P}}\theta_{\mathrm{N}} \, \mathrm{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|y=+1),\bar{\boldsymbol{x}}\sim p(\boldsymbol{x}|y=-1)}[\|\boldsymbol{x}-\bar{\boldsymbol{x}}\|].\end{aligned} \tag{2.88}$$

2.7.3 Class-Prior Estimation under PU Learning Setting

In this section, we review class-prior estimation methods for the PU learning situation (du Plessis and Sugiyama, 2014; du Plessis et al., 2017). To estimate a class-prior from PU data, du Plessis and Sugiyama (2014) proposed a *partial distribution matching* approach.

Let us define a *partial model* as

$$q'_{\theta_{\mathrm{P}}}(\boldsymbol{x}) := \theta_{\mathrm{P}} p(\boldsymbol{x} \mid y = +1). \tag{2.89}$$

Then, we obtain an estimate of a class-prior by minimizing a discrepancy measure, such as the *Kullback-Leibler* (KL) divergence, between the partial model and the true marginal density with respect to θ_{P}:

$$\widehat{\theta}_{\mathrm{P}} = \underset{0\leq\theta_{\mathrm{P}}\leq 1}{\operatorname{argmin}} \ \mathrm{Div}_f[p, q'_{\theta_{\mathrm{P}}}], = \underset{0\leq\theta_{\mathrm{P}}\leq 1}{\operatorname{argmin}} \int p(\boldsymbol{x}) f\Big(\frac{q'_{\theta_{\mathrm{P}}}(\boldsymbol{x})}{p(\boldsymbol{x})}\Big) \mathrm{d}\boldsymbol{x}, \tag{2.90}$$

where Div_f is the f-divergence (Ali and Slivey, 1966; Csizár, 1967) and $f(t)$ is a convex function with $f(1) = 0$.

Although the partial distribution matching via f-divergences had been shown its effectiveness on various datasets experimentally, it was known that the partial matching via f-divergences overestimates the true class-prior unless distributions has no overlap. To cope with this problem, a partial distribution matching via *penalized* f-divergences was proposed. In penalized f-divergences, we replace the function f in Eq. (2.90) with the function satisfying the following condition:

$$\widetilde{f}(t) = \begin{cases} f(t) & 0 \leq t \leq 1, \\ \infty & \text{otherwise.} \end{cases} \tag{2.91}$$

The penalized f-divergence is restricting the range of the density-ratio of $q'_{\theta_{\mathrm{P}}}/p(\boldsymbol{x})$ so as to be $[0,1]$. We then obtain an estimate of the class-prior by solving the following optimization problem:

$$\widehat{\theta}_{\mathrm{P}} = \underset{0\leq\theta_{\mathrm{P}}\leq 1}{\operatorname{argmin}} \int p(\boldsymbol{x})\widetilde{f}\Big(\frac{q'_{\theta_{\mathrm{P}}}(\boldsymbol{x})}{p(\boldsymbol{x})}\Big)\mathrm{d}\boldsymbol{x}. \tag{2.92}$$

du Plessis et al. (2017) prove that consistency, stability, and an estimation error of their class-prior estimation method. Furthermore, they showed an estimate of class-prior can be computed efficiently with the partial matching via the penalized L_1-distance.

2.8 Statistical Dependency

In this section, we review two statistical dependency measures: *mutual information* (MI) (Cover and Thomas, 2006) and *squared-loss mutual information* (SMI) (Suzuki et al., 2009). Note that in this dissertation, since our focus is a binary output, the definition of these statistical dependency measures will be given by the binary classification case.

2.8.1 Mutual Information (MI)

Mutual information (MI), originally developed in *information theory* (Shannon, 1948), is a vital tools for statistical dependency analysis. MI is defined as

$$\mathrm{MI} := \sum_{y=\pm 1}\int p(\boldsymbol{x},y)\log\Big(\frac{p(\boldsymbol{x},y)}{p(\boldsymbol{x})p(y)}\Big)\mathrm{d}\boldsymbol{x}. \tag{2.93}$$

MI can be regarded as the *Kullback-Leibler divergence* from $p(\boldsymbol{x},y)$ to $p(\boldsymbol{x})p(y)$. Thus, MI is non-negative and takes zero if and only if $p(\boldsymbol{x},y) = p(\boldsymbol{x})p(y)$, i.e., $\boldsymbol{x}$ and y are statistically independent. This property allows us to measure the dependency between $\boldsymbol{x}$ and y.

Based on the *information maximization principle*, empirical estimators of MI have been employed for solving various machine learning tasks (Torkkola, 2003; Krause et al., 2010). However, since such MI estimators are known to be sensitive to outliers (Basu et al., 1998), its practical usefulness is limited. Thus, more practical dependency measures are desirable.

2.8.2 Squared-Loss Mutual Information (SMI)

To cope with the sensitivity issue of MI, the squared-loss variant of MI has been proposed and its supervised estimator was developed (Suzuki et al., 2009).

Squared-loss MI (SMI) is defined as

$$\mathrm{SMI} := \frac{1}{2}\sum_{y=\pm 1}\int p(\boldsymbol{x})p(y)\Big(\frac{p(\boldsymbol{x},y)}{p(\boldsymbol{x})p(y)}-1\Big)^2 \mathrm{d}\boldsymbol{x}. \tag{2.94}$$

SMI can be regarded as the *Pearson divergence* (Pearson, 1990) from $p(\boldsymbol{x},y)$ to $p(\boldsymbol{x})p(y)$. That is, SMI is also non-negative and takes zero if and only if $p(\boldsymbol{x},y)=p(\boldsymbol{x})p(y)$.

Unlike MI, SMI is more robust to outliers and has superior numerical properties (Kanamori et al., 2012). So far, various SMI-based machine learning algorithms have been proposed and demonstrated their effectiveness: independence test (Sugiyama and Suzuki, 2011), dimension reduction (Suzuki and Sugiyama, 2013), clustering (Sugiyama et al., 2014), object matching (Yamada et al., 2015), and image registration (Sakai et al., 2015).

SMI Estimation: In practice, since $p(\boldsymbol{x},y)$ is unknown, SMI cannot be computed directly. Here, we review an SMI estimation method from samples. Suppose that we are given a set of *independent and identically distributed* (i.i.d.) paired samples from underlying joint density:

$$\{(\boldsymbol{x}_i,y_i)\}_{i=1}^n \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x},y). \tag{2.95}$$

A naive approach to estimate SMI is as follows: first densities $p(\boldsymbol{x},y)$, $p(\boldsymbol{x})$, and $p(y)$ are estimated from the samples, and then the estimated densities $\widehat{p}(\boldsymbol{x},y)$, $\widehat{p}(\boldsymbol{x})$, and $\widehat{p}(y)$ are plugged into the definition of SMI. However, such a two-step approach is not reliable since estimation error of densities can be magnified, in particular, when division by estimated densities is performed in $\frac{\widehat{p}(\boldsymbol{x},y)}{\widehat{p}(\boldsymbol{x})\widehat{p}(y)}$. To cope with this problem, the SMI estimation method via the direct density-ratio estimation was proposed (Suzuki et al., 2009; Sakai and Sugiyama, 2014).

Let

$$r(\boldsymbol{x},y) := \frac{p(\boldsymbol{x},y)}{p(\boldsymbol{x})p(y)} \tag{2.96}$$

be the density-ratio function and $q(\boldsymbol{x}, y)$ be a model of the density-ratio. The model is learned so that it minimizes the squared error (the PN-SMI risk) defined as

$$\begin{aligned} R_{\mathrm{PN}}^{\mathrm{SMI}}(q) &= \frac{1}{2}\sum_{y=\pm 1}\int (q(\boldsymbol{x}, y) - r(\boldsymbol{x}, y))^2 p(\boldsymbol{x})p(y)\mathrm{d}\boldsymbol{x} - C \\ &= \frac{1}{2}\sum_{y=\pm 1}\int q^2(\boldsymbol{x}, y)p(\boldsymbol{x})p(y)\mathrm{d}\boldsymbol{x} - \sum_{y=\pm 1}\int q(\boldsymbol{x}, y)p(\boldsymbol{x}, y)\mathrm{d}\boldsymbol{x}, \end{aligned} \tag{2.97}$$

where we used $r(\boldsymbol{x}, y)p(\boldsymbol{x})p(y) = p(\boldsymbol{x}, y)$ to obtain the second term and $C = \frac{1}{2}\sum_{y=\pm 1}\int r^2(\boldsymbol{x}, y)p(\boldsymbol{x})p(y)\mathrm{d}\boldsymbol{x}$. By replacing the expectations with corresponding sample averages, we obtain an empirical squared error (the empirical PN-SMI risk) as

$$\widehat{R}_{\mathrm{PN}}^{\mathrm{SMI}}(q) = \sum_{y=\pm 1}\frac{n_y}{2n^2}\sum_{i=1}^{n} q^2(\boldsymbol{x}_i, y) - \frac{1}{n}\sum_{i=1}^{n} q(\boldsymbol{x}_i, y_i), \tag{2.98}$$

where n_y is the number of samples for each class in $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$. The learned model $\widehat{g}$ can be obtained by minimizing the regularized empirical PN-SMI risk:

$$\widehat{q} := \operatorname*{argmin}_{q}\left[\widehat{R}_{\mathrm{PN}}^{\mathrm{SMI}}(q) + \lambda W(q)\right], \tag{2.99}$$

where W is a regularizational functional and $\lambda \geq 0$ is the regularization parameter. With the estimated parameter, an SMI approximator can be obtained by

$$\widehat{\mathrm{SMI}} := \frac{1}{n}\sum_{i=1}^{n}\widehat{q}(\boldsymbol{x}_i, y_i) - \sum_{y=\pm 1}\frac{n_y}{2n^2}\sum_{i=1}^{n}\widehat{q}^2(\boldsymbol{x}_i, y) - \frac{1}{2}, \tag{2.100}$$

since SMI can be expressed as

$$\mathrm{SMI} = \frac{1}{2}\sum_{y=\pm 1}\int r^2(\boldsymbol{x}, y)p(\boldsymbol{x})p(y)\mathrm{d}\boldsymbol{x} - \frac{1}{2} \tag{2.101}$$

$$= \sum_{y=\pm 1}\int r(\boldsymbol{x}, y)p(\boldsymbol{x}, y)\mathrm{d}\boldsymbol{x} - \frac{1}{2}\sum_{y=\pm 1}\int r^2(\boldsymbol{x}, y)p(\boldsymbol{x})p(y)\mathrm{d}\boldsymbol{x} - \frac{1}{2}, \tag{2.102}$$

where the first equation can be obtained by expanding the definition of SMI with the relation $r(\boldsymbol{x}, y)p(\boldsymbol{x}, y) = p(\boldsymbol{x})p(y)$, and the equality between the first and second equations can be confirmed by using $r(\boldsymbol{x}, y)p(\boldsymbol{x}, y) = p(\boldsymbol{x})p(y)$ for the second term in the second equation.

Note that the above SMI approximator can be obtained by *Legendre-Fenchel* convex

duality (Boyd and Vandenberghe, 2004) as in the f-divergence estimation (Keziou, 2003; Nguyen et al., 2007):

$$\mathrm{SMI} = -\inf_{g} \; R_{\mathrm{PN}}^{\mathrm{SMI}}(g) - \frac{1}{2}. \tag{2.103}$$

This implies that minimizing the PN-SMI risk corresponds to maximizing lower bound of SMI.

Practical Implementation: In practice, the model g can be either parametric or non-parametric. A simple choice is to use a linear-in-parameter model:

$$q(\boldsymbol{x}, y) := \sum_{\ell=1}^{b} \alpha_\ell \psi_\ell(\boldsymbol{x}, y) = \boldsymbol{\alpha}^\top \boldsymbol{\psi}(\boldsymbol{x}, y), \tag{2.104}$$

where $\boldsymbol{\alpha} := (\alpha_1, \ldots, \alpha_b)^\top$ is the parameter vector, $\boldsymbol{\psi}(\boldsymbol{x}, y) := (\psi_1(\boldsymbol{x}, y), \ldots, \psi_b(\boldsymbol{x}, y))$ is the basis function vector, b is the number of basis functions, and $^\top$ denotes the transpose of vectors and matrices. Then, with the ℓ_2-regularizer $W(q) = \boldsymbol{\alpha}^\top \boldsymbol{\alpha}/2$, the optimization problem yields

$$\widehat{\boldsymbol{\alpha}} := \operatorname*{argmin}_{\boldsymbol{\alpha} \in \mathbb{R}^b} \left[\frac{1}{2} \boldsymbol{\alpha}^\top \widehat{\boldsymbol{H}} \boldsymbol{\alpha} - \boldsymbol{\alpha}^\top \widehat{\boldsymbol{h}} + \frac{\lambda}{2} \boldsymbol{\alpha}^\top \boldsymbol{\alpha} \right], \tag{2.105}$$

where

$$\widehat{H}_{\ell,\ell'} := \sum_{y=\pm 1} \frac{n_y}{n^2} \sum_{i=1}^{n} \psi_\ell(\boldsymbol{x}_i, y) \psi_{\ell'}(\boldsymbol{x}_i, y)^\top, \tag{2.106}$$

$$\widehat{h}_\ell := \frac{1}{n} \sum_{i=1}^{n} \psi_\ell(\boldsymbol{x}_i, y_i). \tag{2.107}$$

The solution can be obtained analytically by differentiating the objective function with respect to $\boldsymbol{\alpha}$ and set it to zero:

$$\widehat{\boldsymbol{\alpha}} = (\widehat{\boldsymbol{H}} + \lambda \boldsymbol{I}_b)^{-1} \widehat{\boldsymbol{h}}, \tag{2.108}$$

where $\boldsymbol{I}_b$ is the $b \times b$ identity matrix. Then, the density-ratio can be estimated by

$$\widehat{q}(\boldsymbol{x}, y) = \widehat{\boldsymbol{\alpha}}^\top \boldsymbol{\psi}(\boldsymbol{x}, y). \tag{2.109}$$

With the estimated density-ratio, the SMI approximator can be expressed as

$$\widehat{\mathrm{SMI}} = \widehat{\boldsymbol{\alpha}}^\top \widehat{\boldsymbol{h}} - \frac{1}{2}\widehat{\boldsymbol{\alpha}}^\top \widehat{\boldsymbol{H}}\widehat{\boldsymbol{\alpha}} - \frac{1}{2}. \tag{2.110}$$

Chapter 3

Imbalanced Classification from Positive and Unlabeled Data

In this chapter, we discuss learning from positive and unlabeled data in an imbalanced classification scenario.

3.1 Introduction

Supervised learning is widely used and studied in many fields. However, in practice, positive data can be easily accessed, but negative data are not due to diverse representations of negative class or additional expensive costs to collect negative labels. For instance, a friend in a social network service is easy to observe through users' declaration, while a non-friend is difficult unless we conduct a user study. To address this issue, learning from positive and unlabeled data (PU learning) has been studied and revealed useful results.

In addition to the PU learning scenario, we are sometimes faced with the situation where the number of positive data is significantly smaller than that of negative data, i.e., the class-ratio is *imbalanced*. In such an imbalanced classification problem, maximizing the *area under the receiver operating characteristic curve* (AUC) (Hanley and McNeil, 1982) is a standard approach (Cortes and Mohri, 2003). While the misclassification rate relies on the sign of the score of a single sample, AUC is governed by the ranking of the scores of *two* samples. Based on this principle, various supervised methods for directly optimizing AUC have been developed so far and demonstrated to be useful (Herschtal and Raskutti, 2004; Zhao et al., 2011; Rakhlin et al., 2012; Kotlowski et al., 2011; Ying et al., 2016). Previously, a *pairwise ranking* method for PU data has been developed (Sundararajan et al., 2011), which can be regarded as an AUC optimization method for PU data. However, it merely regards unlabeled data as negative data, and thus the obtained classifier is biased.

In this chapter, we present a PU learning method for imbalanced classification. In contrast to Sundararajan et al. (2011), the method is based on an unbiased risk estimator. Furthermore, we prove the generalization error bound of the method and show that unlabeled data contribute to reducing an upper bound on the generalization error with the optimal parametric convergence rate. Experiments confirm the effectiveness of the PU learning method for imbalanced classification.

3.2 Problem Setting

Let random variables $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{+1, -1\}$ be equipped with probability density $p(\boldsymbol{x}, y)$, where d is a positive integer. Let us consider a binary classification problem from $\boldsymbol{x}$ to y, given two sets of samples called the *positive* (P) and *unlabeled* (U) data:

$$\mathcal{X}_{\mathrm{P}} := \{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x} \mid y = +1), \tag{3.1}$$

$$\mathcal{X}_{\mathrm{U}} := \{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x}) = \theta_{\mathrm{P}} p(\boldsymbol{x} \mid y = +1) + \theta_{\mathrm{N}} p(\boldsymbol{x} \mid y = -1), \tag{3.2}$$

where

$$\theta_{\mathrm{P}} := p(y = +1) \ \text{ and } \ \theta_{\mathrm{N}} := p(y = -1).$$

Let E_{P}, E_{N}, and E_{U} be the expectations over $p(\boldsymbol{x} \mid y = +1)$, $p(\boldsymbol{x} \mid y = -1)$, and $p(\boldsymbol{x})$, respectively.

Moreover, let $g \colon \mathbb{R}^d \to \mathbb{R}$ be an arbitrary real-valued decision function for binary classification, and classification is performed based on its sign. Let $\ell \colon \mathbb{R} \to \mathbb{R}$ be a loss function. The goal is to obtain a better classifier in the imbalanced classification scenario.

3.3 PU-AUC Optimization

In this section, we explain the AUC optimization method for the PU learning setting.

Similarly to du Plessis et al. (2014, 2015), the risk function is derived for AUC optimization in PU learning setting. That is, the risk function is equivalent to the risk of its supervised counterpart, and it depends only on positive and unlabeled data distributions. In the derivation and theoretical analysis, the class-priors θ_{P} and θ_{N} are assumed to be known. In practice, they are replaced with their estimates obtained, e.g., by Ramaswamy et al. (2016) and du Plessis et al. (2017), and references therein.

Recall the PN-AUC risk in Eq. (2.23):

$$R_{\mathrm{PN}}^{\mathrm{AUC}}(g) := \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}}))]]. \tag{3.3}$$

By definition of the marginal density $p(\boldsymbol{x}) = \theta_{\mathrm{P}} p(\boldsymbol{x} \mid y = +1) + \theta_{\mathrm{N}} p(\boldsymbol{x} \mid y = -1)$, we have

$$\mathrm{E}_{\mathrm{U}}[\cdot] = \theta_{\mathrm{P}}\, \mathrm{E}_{\mathrm{P}}[\cdot] + \theta_{\mathrm{N}}\, \mathrm{E}_{\mathrm{N}}[\cdot]. \tag{3.4}$$

Thus, we obtain the following relation:

$$\begin{aligned}
&\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] \\
&= \theta_{\mathrm{P}}\, \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] + \theta_{\mathrm{N}}\, \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] \\
&= \theta_{\mathrm{P}}\, \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] + \theta_{\mathrm{N}} R_{\mathrm{PN}}^{\mathrm{AUC}}(g),
\end{aligned} \tag{3.5}$$

where $\mathrm{E}_{\overline{\mathrm{P}}}$ denotes the expectation over $p(\overline{\boldsymbol{x}}^{\mathrm{P}} \mid y = +1)$. Dividing the above equation by θ_{N} and rearranging it, we can express the PN-AUC risk based on PU data (the PU-AUC risk) as

$$\begin{aligned}
R_{\mathrm{PN}}^{\mathrm{AUC}}(g) = &\frac{1}{\theta_{\mathrm{N}}}\, \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] \\
&- \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\, \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] := R_{\mathrm{PU}}^{\mathrm{AUC}}(g).
\end{aligned} \tag{3.6}$$

Thus, we immediately obtain the following lemma.

Lemma 3. *The AUC, PN-AUC, and PU-AUC risks are equivalent:*

$$R^{\mathrm{AUC}}(g) = R_{\mathrm{PN}}^{\mathrm{AUC}}(g) = R_{\mathrm{PU}}^{\mathrm{AUC}}(g). \tag{3.7}$$

In contrast to the PN-AUC risk, the PU-AUC risk can be computed by the expectations over P and U data, but the minimizers are equivalent, meaning that the performance of the classifier obtained by using the PU-AUC risk is the same as the one obtained by using the PN-AUC risk, i.e., supervised learning. We refer to the method minimizing the PU-AUC risk as *PU-AUC optimization*. We explain a practical implementation in Section 3.4 and investigate the superiority of $R_{\mathrm{PU}}^{\mathrm{AUC}}$ in Section 3.5.

As discussed in Section 3.1, Sundararajan et al. (2011) proposed a pairwise ranking method for PU data, which can be regarded as an AUC optimization method for PU data.

Their approach simply regards unlabeled data as negative data, and the ranking SVM (Joachims, 2002) is applied to PU data so that the score of positive data tends to be higher than that of unlabeled data. Although this approach is simple and shown computationally efficient in experiments, the obtained classifier is biased. From the mathematical viewpoint, the existing method ignores the second term in Eq. (3.6) and maximizes only the first term with the hinge loss function. However, the effect of ignoring the second term is not negligible when the class prior, θ_{P}, is not sufficiently small. In contrast, the PU-AUC risk includes the second term so that it is equivalent to its supervised counterpart, i.e., the PN-AUC risk.

For the next chapter, we will present the symmetry of the PU-AUC risk, that is, the risk in AUC optimization from negative and unlabeled data (the NU-AUC risk). From the definition of the marginal density, we have

$$\begin{aligned}
&\mathrm{E}_{\mathrm{U}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{U}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] \\
&= \theta_{\mathrm{P}}\,\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] + \theta_{\mathrm{N}}\,\mathrm{E}_{\mathrm{N}}[\mathrm{E}_{\overline{\mathrm{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\overline{\boldsymbol{x}}^{\mathrm{N}}))]] \\
&= \theta_{\mathrm{P}} R_{\mathrm{PN}}^{\mathrm{AUC}}(g) + \theta_{\mathrm{N}}\,\mathrm{E}_{\mathrm{N}}[\mathrm{E}_{\overline{\mathrm{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\overline{\boldsymbol{x}}^{\mathrm{N}}))]],
\end{aligned} \tag{3.8}$$

where $\mathrm{E}_{\overline{\mathrm{N}}}$ denotes the expectation over $p(\overline{\boldsymbol{x}}^{\mathrm{N}} \mid y = -1)$. Rearranging the above equation, we can obtain the PN-AUC risk in Eq. (2.23) based on negative and unlabeled data (the NU-AUC risk):

$$\begin{aligned}
R_{\mathrm{PN}}^{\mathrm{AUC}}(g) = &\frac{1}{\theta_{\mathrm{P}}}\,\mathrm{E}_{\mathrm{U}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{U}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] \\
&- \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}}}\,\mathrm{E}_{\mathrm{N}}[\mathrm{E}_{\overline{\mathrm{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\overline{\boldsymbol{x}}^{\mathrm{N}}))]] := R_{\mathrm{NU}}^{\mathrm{AUC}}(g).
\end{aligned} \tag{3.9}$$

We refer to the method minimizing the NU-AUC risk as *NU-AUC optimization*. Finally, we have the following lemma.

Lemma 4. *The AUC, PN-AUC, PU-AUC, and NU-AUC risks are equivalent:*

$$R^{\mathrm{AUC}}(g) = R_{\mathrm{PN}}^{\mathrm{AUC}}(g) = R_{\mathrm{PU}}^{\mathrm{AUC}}(g) = R_{\mathrm{NU}}^{\mathrm{AUC}}(g). \tag{3.10}$$

This property of equivalence will be utilized for developing semi-supervised learning methods in Chapter 5.

3.4 Practical Implementation

In this section, we present the implementation details of the PU-AUC optimization method.

3.4.1 General Case

In practice, the AUC risk R^{AUC} is replaced with its empirical version $\widehat{R}^{\mathrm{AUC}}$, i.e., the expectations in R^{AUC} are replaced with the corresponding sample averages.

Here, we focus on the linear-in-parameter model given by

$$g(\boldsymbol{x}) = \sum_{\ell=1}^{b} w_\ell \phi(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}), \tag{3.11}$$

where $^\top$ denotes the transpose of vectors and matrices, b is the number of basis functions, $\boldsymbol{w} = (w_1, \ldots, w_b)^\top$ is a parameter vector, and $\boldsymbol{\phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \ldots, \phi_b(\boldsymbol{x}))^\top$ is a basis function vector. The linear-in-parameter model allows us to express the difference of two classifiers also in a linear-in-parameter form as

$$g(\boldsymbol{x}) - g(\boldsymbol{x}') = \boldsymbol{w}^\top (\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{\phi}(\boldsymbol{x}')) = \boldsymbol{w}^\top \bar{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{x}'), \tag{3.12}$$

where

$$\bar{\boldsymbol{\phi}}(\boldsymbol{x}, \boldsymbol{x}') := \boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{\phi}(\boldsymbol{x}') \tag{3.13}$$

is a composite basis function vector. This property makes the implementation simple with certain loss functions. Then, we train the classifier by minimizing the ℓ_2-regularized empirical AUC risk:

$$\min_{\boldsymbol{w}} \left[\widehat{R}^{\mathrm{AUC}}(g) + \lambda \|\boldsymbol{w}\|^2 \right], \tag{3.14}$$

where $\lambda \geq 0$ is the regularization parameter.

3.4.2 Analytical Solution for Squared Loss

For the squared loss $\ell_S(m) := (1-m)^2$, the empirical PU-AUC risk (See Section 3.7 for the derivation) can be expressed as

$$\begin{aligned}\widehat{R}_{\mathrm{PU}}^{\mathrm{AUC}}(g) &= \frac{1}{\theta_{\mathrm{N}} n_{\mathrm{P}} n_{\mathrm{U}}} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{k=1}^{n_{\mathrm{U}}} \ell_{\mathrm{S}}(\boldsymbol{w}^\top \bar{\boldsymbol{\phi}}(\boldsymbol{x}_i^{\mathrm{P}}, \boldsymbol{x}_k^{\mathrm{U}})) \\ &\quad - \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}} n_{\mathrm{P}}(n_{\mathrm{P}}-1)} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{i'=1}^{n_{\mathrm{P}}} \ell_{\mathrm{S}}(\boldsymbol{w}^\top \bar{\boldsymbol{\phi}}(\boldsymbol{x}_i^{\mathrm{P}}, \boldsymbol{x}_{i'}^{\mathrm{P}})) + \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}(n_{\mathrm{P}}-1)} \\ &= 1 - 2\boldsymbol{w}^\top \widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{AUC}} + \boldsymbol{w}^\top \widehat{\boldsymbol{H}}_{\mathrm{PU}}^{\mathrm{AUC}} \boldsymbol{w} - \boldsymbol{w}^\top \widehat{\boldsymbol{H}}_{\mathrm{PP}}^{\mathrm{AUC}} \boldsymbol{w}, \end{aligned} \tag{3.15}$$

where

$$\widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{AUC}} := \frac{1}{\theta_{\mathrm{N}} n_{\mathrm{P}}} \boldsymbol{\Phi}_{\mathrm{P}}^\top \mathbf{1}_{n_{\mathrm{P}}} - \frac{1}{\theta_{\mathrm{N}} n_{\mathrm{U}}} \boldsymbol{\Phi}_{\mathrm{U}}^\top \mathbf{1}_{n_{\mathrm{U}}}, \tag{3.16}$$

$$\begin{aligned}\widehat{\boldsymbol{H}}_{\mathrm{PU}}^{\mathrm{AUC}} &:= \frac{1}{\theta_{\mathrm{N}} n_{\mathrm{P}}} \boldsymbol{\Phi}_{\mathrm{P}}^\top \boldsymbol{\Phi}_{\mathrm{P}} - \frac{1}{\theta_{\mathrm{N}} n_{\mathrm{P}} n_{\mathrm{U}}} \boldsymbol{\Phi}_{\mathrm{U}}^\top \mathbf{1}_{n_{\mathrm{U}}} \mathbf{1}_{n_{\mathrm{P}}}^\top \boldsymbol{\Phi}_{\mathrm{P}} \\ &\quad - \frac{1}{\theta_{\mathrm{N}} n_{\mathrm{P}} n_{\mathrm{U}}} \boldsymbol{\Phi}_{\mathrm{P}}^\top \mathbf{1}_{n_{\mathrm{P}}} \mathbf{1}_{n_{\mathrm{U}}}^\top \boldsymbol{\Phi}_{\mathrm{U}} + \frac{1}{\theta_{\mathrm{N}} n_{\mathrm{U}}} \boldsymbol{\Phi}_{\mathrm{U}}^\top \boldsymbol{\Phi}_{\mathrm{U}}, \end{aligned} \tag{3.17}$$

$$\widehat{\boldsymbol{H}}_{\mathrm{PP}}^{\mathrm{AUC}} := \frac{2\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}(n_{\mathrm{P}}-1)} \boldsymbol{\Phi}_{\mathrm{P}}^\top \boldsymbol{\Phi}_{\mathrm{P}} - \frac{2\theta_{\mathrm{P}}}{\theta_{\mathrm{N}} n_{\mathrm{P}}(n_{\mathrm{P}}-1)} \boldsymbol{\Phi}_{\mathrm{P}}^\top \mathbf{1}_{n_{\mathrm{P}}} \mathbf{1}_{n_{\mathrm{P}}}^\top \boldsymbol{\Phi}_{\mathrm{P}}, \tag{3.18}$$

$$\boldsymbol{\Phi}_{\mathrm{P}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{P}}), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{P}}}^{\mathrm{P}}))^\top, \tag{3.19}$$

$$\boldsymbol{\Phi}_{\mathrm{U}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{U}}), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{U}}}^{\mathrm{U}}))^\top, \tag{3.20}$$

and $\mathbf{1}_b$ is the b-dimensional vector whose elements are all one. With the ℓ_2-regularizer, we can analytically obtain the solution[1] by

$$\widehat{\boldsymbol{w}}_{\mathrm{PU}}^{\mathrm{AUC}} := (\widehat{\boldsymbol{H}}_{\mathrm{PU}}^{\mathrm{AUC}} - \widehat{\boldsymbol{H}}_{\mathrm{PP}}^{\mathrm{AUC}} + \lambda \boldsymbol{I}_b)^{-1} \widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{AUC}}, \tag{3.21}$$

where $\boldsymbol{I}_b$ is the b-dimensional identity matrix.

The computational complexity of computing $\widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{AUC}}$, $\widehat{\boldsymbol{H}}_{\mathrm{PU}}^{\mathrm{AUC}}$, and $\widehat{\boldsymbol{H}}_{\mathrm{PP}}^{\mathrm{AUC}}$ are $\mathcal{O}((n_{\mathrm{P}} + n_{\mathrm{U}})b)$, $\mathcal{O}((n_{\mathrm{P}} + n_{\mathrm{U}})b^2)$, and $\mathcal{O}(n_{\mathrm{P}} b^2)$, respectively. Then, solving a system of linear equations to obtain the solution $\widehat{\boldsymbol{w}}_{\mathrm{PU}}^{\mathrm{AUC}}$ requires the computational complexity of $\mathcal{O}(b^3)$.

[1] To be precise, the obtained solution is a critical point (stationary point) of the optimization problem. However, with the squared loss, since the first term in Eq. (3.6) and ℓ_2-regularizer are strongly convex functions, the optimization problem will become convex if the regularization parameter λ is sufficiently large.

In total, the computational complexity of this PU-AUC optimization method is $\mathcal{O}((n_{\mathrm{P}} + n_{\mathrm{U}})b^2 + b^3)$.

From the viewpoint of computational complexity, the squared loss and the exponential loss are more efficient than the logistic loss because these loss functions reduce the nested summations to individual ones. More specifically, for example, in the PU-AUC optimization method, the logistic loss requires $\mathcal{O}(n_{\mathrm{P}} n_{\mathrm{U}})$ operations for evaluating the first term in the PU-AUC risk, i.e., the loss over positive and unlabeled samples. In contrast, the squared loss and exponential loss reduce the number of operations for loss evaluation to $\mathcal{O}(n_{\mathrm{P}} + n_{\mathrm{U}})$. For example, the exponential loss over positive and unlabeled data can be computed as follows:

$$\begin{aligned}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{k=1}^{n_{\mathrm{U}}} \ell_{\mathrm{E}}(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_k^{\mathrm{U}})) &= \sum_{i=1}^{n_{\mathrm{P}}}\sum_{k=1}^{n_{\mathrm{U}}} \exp(-g(\boldsymbol{x}_i^{\mathrm{P}}) + g(\boldsymbol{x}_k^{\mathrm{U}})) \\ &= \sum_{i=1}^{n_{\mathrm{P}}} \exp(-g(\boldsymbol{x}_i^{\mathrm{P}})) \sum_{k=1}^{n_{\mathrm{U}}} \exp(g(\boldsymbol{x}_k^{\mathrm{U}})). \end{aligned} \tag{3.22}$$

Thus, the number of operations for loss evaluation is reduced to $n_{\mathrm{P}} + n_{\mathrm{U}} + 1$ rather than $n_{\mathrm{P}} n_{\mathrm{U}}$. This property is beneficial especially when we handle large scale datasets.

3.4.3 Model Selection

In practice, the performance of the PU-AUC optimization method depends on hyperparameters. To tune the hyperparameters, cross-validation is available. Although labeled data for all the classes are not available unlike supervised learning, we can conduct cross-validation by only using positive and unlabeled data.

Specifically, we first split the positive and unlabeled samples, $\mathcal{S}^{\mathrm{P}}$ and $\mathcal{S}^{\mathrm{U}}$, into K disjoint subsets, $\{\mathcal{S}_k^{\mathrm{P}}\}_{k=1}^K$ and $\{\mathcal{S}_k^{\mathrm{U}}\}_{k=1}^K$ of almost the same size; we construct K disjoint subsets $\{\mathcal{S}_k := \mathcal{S}_k^{\mathrm{P}} \cup \mathcal{S}_k^{\mathrm{U}}\}_{k=1}^K$. Then, we obtain the classifier $\widehat{g}_k$ with training samples $\mathcal{S}\backslash\mathcal{S}_k$. The approximation error is computed by the hold-out samples $\mathcal{S}_k$. We repeat this procedure for $k = 1, \ldots, K$, and compute its mean $\widehat{J}_{\mathrm{PU}}^{\mathrm{CV}}$. Finally, we choose the best model that minimizes $\widehat{J}_{\mathrm{PU}}^{\mathrm{CV}}$ from all model candidates.

3.5 Generalization Error Bounds

As the classifier g, we assume the linear-in-parameter model. Let $\mathcal{G}$ be a function class of bounded hyperplanes:

$$\mathcal{G} := \{g(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}) \mid \|\boldsymbol{w}\| \leq C_{\boldsymbol{w}};\ \forall \boldsymbol{x}\colon \|\boldsymbol{\phi}(\boldsymbol{x})\| \leq C_{\boldsymbol{\phi}}\}, \tag{3.23}$$

where $C_{\boldsymbol{w}} > 0$ and $C_{\boldsymbol{\phi}} > 0$ are certain positive constants. This assumption is reasonable because the ℓ_2-regularizer included in training and the use of bounded basis functions, e.g., the Gaussian kernel basis, ensure that the minimizer of the empirical AUC risk belongs to such function class $\mathcal{G}$. Furthermore, we assume that a surrogate loss is bounded from above by C_ℓ and denote the Lipschitz constant by L. For simplicity, we focus on a surrogate loss satisfying $\ell_{0\text{-}1}(m) \leq \ell(m)$, but the theoretical analysis can be easily extended to the loss satisfying $\ell_{0\text{-}1}(m) \leq M\ell(m)$ with a certain $M > 0$. For example, the squared loss and the exponential loss satisfy the condition $M = 1$. Moreover, these losses are bounded in our setting, since the input to $\ell(m)$, i.e., g is bounded.

For the sake of simplicity, we assume that another set of positive samples $\{\overline{\boldsymbol{x}}_{i'}^{\mathrm{P}}\}_{i'=1}^{n_{\mathrm{P}}}$ is available for computing the nested expectations over $p(\boldsymbol{x}^{\mathrm{P}} \mid y = +1)$ and $p(\overline{\boldsymbol{x}}^{\mathrm{P}} \mid y = +1)$ in the second term of Eq. (3.6), unlike the practical implementation. On the other hand, for approximating the PU-AUC risk function, we can split positive samples $\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}}$ into two parts, e.g., $\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}/2}$ and $\{\overline{\boldsymbol{x}}_{i'}^{\mathrm{P}}\}_{i'=1}^{n_{\mathrm{P}}/2} = \{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=n_{\mathrm{P}}/2}^{n_{\mathrm{P}}}$ provided that n_{P} is even. Approximating the risk function in the latter way loosens the upper bound compared with the one obtained by the former approach. In our implementation (see Sections 3.4 and 3.7), we approximate the nested expectations without largely reducing the number of samples, unlike the latter approach. Thus, the practical performance of our risk estimator might be in between the former and the later approaches.

Let

$$I^{\mathrm{AUC}}(g) = \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[\ell_{0\text{-}1}(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] \tag{3.24}$$

be the generalization error of g in AUC optimization. For convenience, we define

$$c^{\mathrm{AUC}}(\delta) := 2\sqrt{2}LC_\ell C_{\boldsymbol{w}} C_{\boldsymbol{\phi}} + \frac{3}{2}\sqrt{2\log(2/\delta)}. \tag{3.25}$$

The following generalization error bounds are proved for the PU-AUC/NU-AUC risks (its proof is available in Section 3.8):

Theorem 5. *For any $\delta > 0$, the following inequalities hold separately with probability at least $1 - \delta$ for all $g \in \mathcal{G}$:*

$$I^{\mathrm{AUC}}(g) \leq \widehat{R}_{\mathrm{PU}}^{\mathrm{AUC}}(g) + c^{\mathrm{AUC}}(\delta/2)\Big(\frac{1}{\theta_{\mathrm{N}}\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})}} + \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}\sqrt{n_{\mathrm{P}}}}\Big), \tag{3.26}$$

$$I^{\mathrm{AUC}}(g) \leq \widehat{R}_{\mathrm{NU}}^{\mathrm{AUC}}(g) + c^{\mathrm{AUC}}(\delta/2)\Big(\frac{1}{\theta_{\mathrm{P}}\sqrt{\min(n_{\mathrm{N}}, n_{\mathrm{U}})}} + \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}}\sqrt{n_{\mathrm{N}}}}\Big), \tag{3.27}$$

where $\widehat{R}_{\mathrm{PU}}^{\mathrm{AUC}}$ and $\widehat{R}_{\mathrm{NU}}^{\mathrm{AUC}}$ are unbiased empirical risk estimators corresponding to $R_{\mathrm{PU}}^{\mathrm{AUC}}$ and $R_{\mathrm{NU}}^{\mathrm{AUC}}$, respectively.

Theorem 5 guarantees that $I^{\mathrm{AUC}}(g)$ can be bounded from above by the empirical risk, $\widehat{R}^{\mathrm{AUC}}(g)$, plus the confidence terms of order

$$\mathcal{O}_p\Big(\frac{1}{\sqrt{n_{\mathrm{P}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big) \text{ and } \mathcal{O}_p\Big(\frac{1}{\sqrt{n_{\mathrm{N}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big).$$

Since n_{P} (n_{N}) and n_{U} can increase independently in our setting, this is the optimal convergence rate without any additional assumptions (Vapnik, 1998; Mendelson, 2008).

3.6 Experiments

In this section, we compare the PU-AUC optimization method against the existing AUC optimization method based on the ranking SVM (PU-RSVM) (Sundararajan et al., 2011). We used 8 benchmark datasets from the *IDA Benchmark Repository* (Rätsch et al., 2001), the *LIBSVM* (Chang and Lin, 2011), and the *UCI Machine Learning Repository* (Lichman, 2013).

As the classifier, we used the linear-in-parameter model with the Gaussian kernel basis function:

$$\phi_\ell(\boldsymbol{x}) = \exp\Big(-\frac{\|\boldsymbol{x} - \boldsymbol{x}_\ell\|^2}{2\sigma^2}\Big),$$

where $\sigma > 0$ is the Gaussian bandwidth, $\{\boldsymbol{x}_\ell\}_{\ell=1}^{b}$ are the samples randomly selected from unlabeled samples $\{\boldsymbol{x}_i^{\mathrm{U}}\}_{i=1}^{n_{\mathrm{U}}}$. The number of basis functions was set at $b = \min(n_{\mathrm{U}}, 200)$. The candidates of the Gaussian bandwidth were

$$\mathrm{median}(\{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\}_{i,j=1}^{n}) \times \Big\{\frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2\Big\} \tag{3.28}$$

TABLE 3.1: Average and standard error of the estimated class-prior over 50 trials on benchmark datasets in PU learning setting.

Dataset	d	$\theta_{\mathrm{P}} = 0.1$	$\theta_{\mathrm{P}} = 0.2$
Banana	2	0.15 (0.01)	0.24 (0.01)
skin_nonskin	3	0.10 (0.00)	0.20 (0.01)
cod-rna	8	0.32 (0.01)	0.40 (0.02)
Magic	10	0.34 (0.01)	0.39 (0.01)
Image	18	0.13 (0.01)	0.24 (0.01)
Twonorm	20	0.27 (0.00)	0.34 (0.00)
Waveform	21	0.31 (0.00)	0.38 (0.01)
mushrooms	112	0.19 (0.00)	0.28 (0.00)

and those of the regularization parameter were $\{10^{-3},\ 10^{-2},\ 10^{-1},\ 10^{0},\ 10^{1}\}$. The bandwidth and regularization parameter were determined by five-fold cross-validation. We trained a classifier with samples of size $n_{\mathrm{P}} = 100$ and $n_{\mathrm{U}} = 1000$ under the different class-priors $\theta_{\mathrm{P}} = 0.1$ and 0.2. For the PU-AUC optimization method, the squared loss function was used and the class-prior was estimated by the distribution matching method (du Plessis et al., 2017). The results of the estimated class-prior are summarized in Table 3.1.

Table 3.2 lists the average with standard error of the AUC over 50 trials, showing that the PU-AUC optimization method achieves larger AUC than PU-RSVM. In particular, when $\theta_{\mathrm{P}} = 0.2$, the difference between PU-RSVM and our method becomes larger compared with the difference when $\theta_{\mathrm{P}} = 0.1$. This is because PU-RSVM can be interpreted as regarding unlabeled data as negative; it causes larger bias when θ_{P} increases.

Figure 3.1 summarizes the average computation time over 50 trials. The computation time of the PU-AUC optimization method includes both the class-prior estimation and the empirical risk minimization. The results show that the PU-AUC optimization method requires almost twice computation time as that of PU-RSVM, but it would be acceptable in practice to obtain better performance.

TABLE 3.2: Average and standard error of the AUC over 50 trials on benchmark datasets. The boldface denotes the best and comparable methods in terms of the average AUC according to the t-test at the significance level 5%. The bottom row shows the number of best/comparable cases of each method.

Dataset	d	$\theta_{\mathrm{P}} = 0.1$		$\theta_{\mathrm{P}} = 0.2$	
		PU-AUC	PU-RSVM	PU-AUC	PU-RSVM
Banana	2	**95.4 (0.1)**	88.5 (0.2)	**95.0 (0.1)**	85.9 (0.2)
skin_nonskin	3	**99.9 (0.0)**	86.2 (0.3)	**99.8 (0.0)**	73.8 (0.5)
cod-rna	8	**98.1 (0.1)**	62.8 (0.3)	**97.6 (0.1)**	59.4 (0.4)
Magic	10	**87.4 (0.2)**	82.8 (0.2)	**86.4 (0.2)**	81.9 (0.1)
Image	18	**97.5 (0.1)**	82.2 (0.2)	**96.7 (0.2)**	77.0 (0.4)
Twonorm	20	**99.6 (0.0)**	85.5 (0.2)	**99.5 (0.0)**	79.2 (0.4)
Waveform	21	**96.6 (0.1)**	73.9 (0.6)	**96.3 (0.1)**	59.5 (1.0)
mushrooms	112	**99.8 (0.0)**	93.9 (0.3)	**99.6 (0.1)**	84.7 (0.3)
#Best/Comp.		8	0	8	0

3.7 Derivation of PU-AUC Risk Estimator

Here, we present the derivation of the PU-AUC risk estimator. Recall that the PU-AUC risk in Eq. (3.6) is defined as

$$R_{\mathrm{PU}}^{\mathrm{AUC}}(g) = \frac{1}{\theta_{\mathrm{N}}} \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] - \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}} \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]]. \quad (3.29)$$

If one additional set of positive samples $\{\overline{\boldsymbol{x}}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}}$ is available, we obtain the unbiased PU-AUC risk estimator by

$$\widehat{R}_{\mathrm{PU}}^{\mathrm{AUC}}(g) = \frac{1}{\theta_{\mathrm{N}}} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{k=1}^{n_{\mathrm{U}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_k^{\mathrm{U}})) - \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}} {n_{\mathrm{P}}}^2} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{i'=1}^{n_{\mathrm{P}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}_{i'}^{\mathrm{P}})). \quad (3.30)$$

We used this estimator in our theoretical analyses because learning is not involved. However, obtaining one additional set of samples is not always possible in practice. Thus,

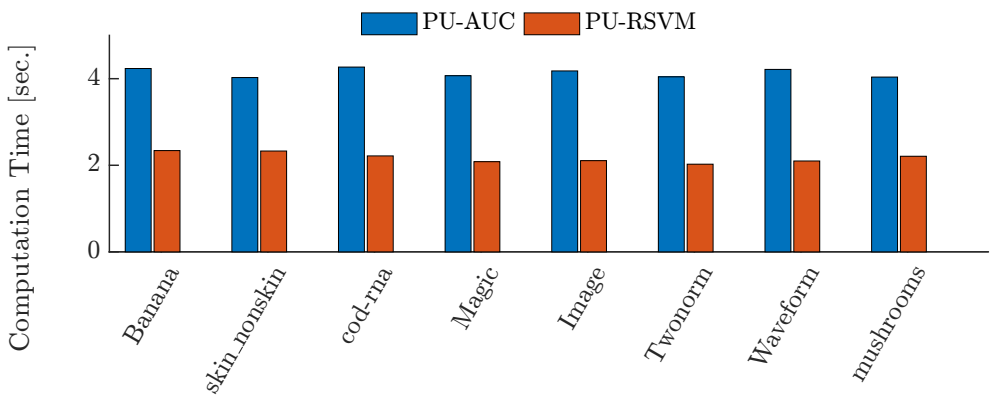


FIGURE 3.1: Average computation time on benchmark datasets when $\theta_{\mathrm{P}} = 0.1$ over 50 trials. The computation time of the PU-AUC optimization method includes the class-prior estimation and the empirical risk minimization.

instead of the above risk estimator, we use the following risk estimator in our implementation:

$$\begin{aligned}\widehat{R}_{\mathrm{PU}}^{\mathrm{AUC}}(g) = \frac{1}{\theta_{\mathrm{N}}} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{k=1}^{n_{\mathrm{U}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_k^{\mathrm{U}})) \\ - \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}} \left(\frac{1}{n_{\mathrm{P}}(n_{\mathrm{P}}-1)} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{i'=1}^{n_{\mathrm{P}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_{i'}^{\mathrm{P}})) - \frac{\ell(0)}{n_{\mathrm{P}}-1} \right).\end{aligned} \tag{3.31}$$

This estimator is also unbiased. To show unbiasedness of this estimator, let us rewrite the second term of the PU-AUC risk without coefficient in Eq. (3.6) as

$$\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] = \mathrm{E}_{\boldsymbol{x}^{\mathrm{P}}, \overline{\boldsymbol{x}}^{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]. \tag{3.32}$$

The unbiased estimator can be expressed as

$$\frac{1}{n_{\mathrm{P}}(n_{\mathrm{P}}-1)} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{i'=1}^{n_{\mathrm{P}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_{i'}^{\mathrm{P}})) - \frac{\ell(0)}{n_{\mathrm{P}}-1}, \tag{3.33}$$

because the expectation of the above estimator can be computed as follows:

$$
\begin{aligned}
&\mathrm{E}_{\boldsymbol{x}_1^{\mathrm{P}},\ldots,\boldsymbol{x}_{n_{\mathrm{P}}}^{\mathrm{P}}}\Big[\frac{1}{n_{\mathrm{P}}(n_{\mathrm{P}}-1)}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'=1}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}})-g(\boldsymbol{x}_{i'}^{\mathrm{P}}))-\frac{\ell(0)}{n_{\mathrm{P}}-1}\Big]\\
&=\mathrm{E}_{\boldsymbol{x}_1^{\mathrm{P}},\ldots,\boldsymbol{x}_{n_{\mathrm{P}}}^{\mathrm{P}}}\Big[\frac{1}{n_{\mathrm{P}}(n_{\mathrm{P}}-1)}\Big(\sum_{i=1}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}})-g(\boldsymbol{x}_i^{\mathrm{P}}))+\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'\neq i}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}})-g(\boldsymbol{x}_{i'}^{\mathrm{P}}))\Big)\\
&\quad-\frac{\ell(0)}{n_{\mathrm{P}}-1}\Big]\\
&=\mathrm{E}_{\boldsymbol{x}_1^{\mathrm{P}},\ldots,\boldsymbol{x}_{n_{\mathrm{P}}}^{\mathrm{P}}}\Big[\frac{1}{n_{\mathrm{P}}(n_{\mathrm{P}}-1)}\Big(\sum_{i=1}^{n_{\mathrm{P}}}\ell(0)+\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'\neq i}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}})-g(\boldsymbol{x}_{i'}^{\mathrm{P}}))\Big)-\frac{\ell(0)}{n_{\mathrm{P}}-1}\Big]\\
&=\frac{1}{n_{\mathrm{P}}(n_{\mathrm{P}}-1)}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'\neq i}^{n_{\mathrm{P}}}\mathrm{E}_{\boldsymbol{x}_i^{\mathrm{P}},\boldsymbol{x}_{i'}^{\mathrm{P}}}\Big[\ell(f(\boldsymbol{x}_i^{\mathrm{P}},\boldsymbol{x}_{i'}^{\mathrm{P}}))\Big]\\
&=\frac{1}{n_{\mathrm{P}}(n_{\mathrm{P}}-1)}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'\neq i}^{n_{\mathrm{P}}}\mathrm{E}_{\boldsymbol{x}^{\mathrm{P}},\bar{\boldsymbol{x}}^{\mathrm{P}}}\Big[\ell(g(\boldsymbol{x}^{\mathrm{P}})-g(\bar{\boldsymbol{x}}^{\mathrm{P}}))\Big]\\
&=\mathrm{E}_{\boldsymbol{x}^{\mathrm{P}},\bar{\boldsymbol{x}}^{\mathrm{P}}}\Big[\ell(g(\boldsymbol{x}^{\mathrm{P}})-g(\bar{\boldsymbol{x}}^{\mathrm{P}}))\Big].
\end{aligned}
\tag{3.34}
$$

If the squared loss function $\ell(m)=(1-m)^2$ is used, $\ell(0)=1$ (cf. the implementation with the squared loss in Section 3.4.2). Therefore, the PU-AUC risk estimator is unbiased.

3.8 Proof of Theorem 5

Finally, we show the proof of generalization error bounds in Section 3.5. The proof is based on Usunier et al. (2006).

Let

$$\{\boldsymbol{x}_i\}_{i=1}^{n}\overset{\text{i.i.d.}}{\sim}q(\boldsymbol{x})\quad\text{and} \tag{3.35}$$

$$\{\boldsymbol{x}'_j\}_{j=1}^{n'}\overset{\text{i.i.d.}}{\sim}q'(\boldsymbol{x}) \tag{3.36}$$

be two sets of samples drawn from the distribution equipped with densities $q(\boldsymbol{x})$ and $q'(\boldsymbol{x})$, respectively. Let $\mathcal{G}$ be a function class of bounded hyperplanes:

$$\mathcal{G}:=\{g(\boldsymbol{x})=\boldsymbol{w}^\top\boldsymbol{\phi}(\boldsymbol{x})\mid\|\boldsymbol{w}\|\leq C_{\boldsymbol{w}};\ \forall\boldsymbol{x}\colon\|\boldsymbol{\phi}(\boldsymbol{x})\|\leq C_{\boldsymbol{\phi}}\}, \tag{3.37}$$

where $C_{\boldsymbol{w}} > 0$ and $C_{\boldsymbol{\phi}} > 0$ are certain positive constants. The AUC risk over distributions q and q' and its empirical version can be expressed as

$$R^{\mathrm{AUC}}(g) := \mathrm{E}_{\boldsymbol{x}\sim q}[\mathrm{E}_{\boldsymbol{x}'\sim q'}[\ell(g(\boldsymbol{x}) - g(\boldsymbol{x}'))]], \tag{3.38}$$

$$\widehat{R}^{\mathrm{AUC}}(g) := \frac{1}{nn'}\sum_{i=1}^{n}\sum_{j=1}^{n'}\ell(g(\boldsymbol{x}_i) - g(\boldsymbol{x}'_j)). \tag{3.39}$$

For convenience, let us define

$$c^{\mathrm{AUC}}(\delta) := 2\sqrt{2}LC_{\ell}C_{\boldsymbol{w}}C_{\boldsymbol{\phi}} + \frac{3}{2}\sqrt{2\log(2/\delta)}. \tag{3.40}$$

Firstly, we have the following theorem:

Theorem 6. *For any $\delta > 0$, the following inequality holds with probability at least $1-\delta$ for any $g \in \mathcal{G}$:*

$$R^{\mathrm{AUC}}(g) - \widehat{R}^{\mathrm{AUC}}(g) \leq c^{\mathrm{AUC}}(\delta)\frac{1}{\sqrt{\min(n,n')}}. \tag{3.41}$$

Proof. By slightly modifying Theorem 7 in Usunier et al. (2006) to fit our setting, for any $\delta > 0$, with probability at least $1-\delta$ for any $g \in \mathcal{G}$, we have

$$\begin{aligned} R^{\mathrm{AUC}}(g) - \widehat{R}^{\mathrm{AUC}}(g) \leq{}& \frac{2LC_{\ell}C_{\boldsymbol{w}}\sqrt{\max(n,n')}}{nn'}\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n'}\|\boldsymbol{\phi}(\boldsymbol{x}_i) - \boldsymbol{\phi}(\boldsymbol{x}_j)\|^2} \\ &+ 3\sqrt{\frac{\log(2/\delta)}{2\min(n,n')}}. \end{aligned} \tag{3.42}$$

Then, applying the triangle inequality to the fist term in Eq. (3.42), we have

$$\begin{aligned} \sum_{i=1}^{n}\sum_{j=1}^{n'}\|\boldsymbol{\phi}(\boldsymbol{x}_i) - \boldsymbol{\phi}(\boldsymbol{x}_j)\|^2 &\leq n'\sum_{i=1}^{n}\|\boldsymbol{\phi}(\boldsymbol{x}_i)\|^2 + n\sum_{j=1}^{n'}\|\boldsymbol{\phi}(\boldsymbol{x}_j)\|^2 \\ &\leq 2nn'C_{\boldsymbol{\phi}}^2. \end{aligned} \tag{3.43}$$

Plugging the above into Eq. (3.42), we conclude the proof of the theorem. □

Next, we prove the risk bounds of the PU-AUC and NU-AUC risks. For the sake of simplicity, we assume that the expectations of E_{P} (E_{N}) and $\mathrm{E}_{\overline{\mathrm{P}}}$ ($\mathrm{E}_{\overline{\mathrm{N}}}$) in the PU-AUC (NU-AUC) risks are estimated by using the same number of samples n_{P} (n_{N}), i.e., $\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}}$

and $\{\overline{\boldsymbol{x}}_{i'}^{\mathrm{P}}\}_{i'=1}^{n_{\mathrm{P}}}$ are used for approximating E_{P} and $\mathrm{E}_{\overline{\mathrm{P}}}$.[2] By using Theorem 6, we have the following lemma:

Lemma 7. *For any $\delta > 0$, the following inequalities hold separately with probability at least $1-\delta$ for any $g \in \mathcal{G}$:*

$$R_{\mathrm{PU}}^{\mathrm{AUC}}(g) - \widehat{R}_{\mathrm{PU}}^{\mathrm{AUC}}(g) \leq c^{\mathrm{AUC}}(\delta/2)\Big(\frac{1}{\theta_{\mathrm{N}}\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})}} + \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}\sqrt{n_{\mathrm{P}}}}\Big), \tag{3.44}$$

$$R_{\mathrm{NU}}^{\mathrm{AUC}}(g) - \widehat{R}_{\mathrm{NU}}^{\mathrm{AUC}}(g) \leq c^{\mathrm{AUC}}(\delta/2)\Big(\frac{1}{\theta_{\mathrm{P}}\sqrt{\min(n_{\mathrm{N}}, n_{\mathrm{U}})}} + \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}}\sqrt{n_{\mathrm{N}}}}\Big). \tag{3.45}$$

Proof. Recall that the PU-AUC and NU-AUC risks are expressed as

$$R_{\mathrm{PU}}^{\mathrm{AUC}}(g) = \frac{1}{\theta_{\mathrm{N}}}\,\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] - \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\,\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]], \tag{3.46}$$

$$R_{\mathrm{NU}}^{\mathrm{AUC}}(g) = \frac{1}{\theta_{\mathrm{P}}}\,\mathrm{E}_{\mathrm{U}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{U}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] - \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}}}\,\mathrm{E}_{\mathrm{N}}[\mathrm{E}_{\overline{\mathrm{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\overline{\boldsymbol{x}}^{\mathrm{N}}))]]. \tag{3.47}$$

[2] Otherwise, we can consider the obtained samples are divided into two parts, e.g., $n_{\mathrm{P}}/2$, and the bound in Eq. (3.50) are loosen by two times.

From Theorem 6, for any $\delta > 0$, we have these uniform deviation bounds with probability at least $1 - \delta/2$:

$$\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] - \frac{1}{n_{\mathrm{P}}n_{\mathrm{U}}}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_k^{\mathrm{U}}))\Big) \leq \frac{c^{\mathrm{AUC}}(\delta/2)}{\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})}}, \tag{3.48}$$

$$\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{U}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{U}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] - \frac{1}{n_{\mathrm{N}}n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}}\sum_{j=1}^{n_{\mathrm{N}}}\ell(g(\boldsymbol{x}_k^{\mathrm{U}}) - g(\boldsymbol{x}_j^{\mathrm{N}}))\Big) \leq \frac{c^{\mathrm{AUC}}(\delta/2)}{\sqrt{\min(n_{\mathrm{N}}, n_{\mathrm{U}})}}, \tag{3.49}$$

$$\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] - \frac{1}{{n_{\mathrm{P}}}^2}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'=1}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}_{i'}^{\mathrm{P}}))\Big) \leq \frac{c^{\mathrm{AUC}}(\delta/2)}{\sqrt{n_{\mathrm{P}}}}, \tag{3.50}$$

$$\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{N}}[\mathrm{E}_{\overline{\mathrm{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\overline{\boldsymbol{x}}^{\mathrm{N}}))]] - \frac{1}{{n_{\mathrm{N}}}^2}\sum_{j=1}^{n_{\mathrm{N}}}\sum_{j'=1}^{n_{\mathrm{N}}}\ell(g(\boldsymbol{x}_j^{\mathrm{N}}) - g(\overline{\boldsymbol{x}}_{j'}^{\mathrm{N}}))\Big) \leq \frac{c^{\mathrm{AUC}}(\delta/2)}{\sqrt{n_{\mathrm{N}}}}. \tag{3.51}$$

Then, simple calculation showed that for any $\delta > 0$, with probability $1 - \delta$, we have

$$\begin{aligned}
&\sup_{g\in\mathcal{G}}\Big(R_{\mathrm{PU}}^{\mathrm{AUC}}(g) - \widehat{R}_{\mathrm{PU}}^{\mathrm{AUC}}(g)\Big) \\
&\leq \frac{1}{\theta_{\mathrm{N}}}\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] - \frac{1}{n_{\mathrm{P}}n_{\mathrm{U}}}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_k^{\mathrm{U}}))\Big) \\
&\quad + \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] - \frac{1}{{n_{\mathrm{P}}}^2}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'=1}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}_{i'}^{\mathrm{P}}))\Big) \\
&\leq c^{\mathrm{AUC}}(\delta/2)\Big(\frac{1}{\theta_{\mathrm{N}}\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})}} + \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}\sqrt{n_{\mathrm{P}}}}\Big),
\end{aligned} \tag{3.52}$$

where we used

$$\sup(x + y) \leq \sup(x) + \sup(y), \tag{3.53}$$

$$R_{\mathrm{PU}}^{\mathrm{AUC}}(g) \leq \frac{1}{\theta_{\mathrm{N}}}\,\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] + \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\,\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]]. \tag{3.54}$$

Similarly, for the NU-AUC risk, we have

$$\sup_{g\in\mathcal{G}}\Big(R_{\mathrm{NU}}^{\mathrm{AUC}}(g)-\widehat{R}_{\mathrm{NU}}^{\mathrm{AUC}}(g)\Big)\leq c^{\mathrm{AUC}}(\delta/2)\Big(\frac{1}{\theta_{\mathrm{P}}\sqrt{\min(n_{\mathrm{N}},n_{\mathrm{U}})}}+\frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}}\sqrt{n_{\mathrm{N}}}}\Big). \tag{3.55}$$

Eqs. (3.52) and (3.55) conclude the lemma. □

Finally, we give the proof of Theorem 5.

Proof. Assume the loss satisfying $\ell_{0\text{-}1}(m)\leq M\ell(m)$. Then, we have $I^{\mathrm{AUC}}(g)\leq M\cdot R^{\mathrm{AUC}}(g)$. When $M=1$ such as the squared loss $\ell_{\mathrm{S}}(m)$ and the exponential loss $\ell_{\mathrm{E}}(m)$, $I^{\mathrm{AUC}}(g)\leq R^{\mathrm{AUC}}(g)$ holds. This observation yields Theorem 5. □

Chapter 4

Statistical Dependency Analysis from Positive and Unlabeled Data

In this chapter, we develop statistical dependency analysis tools for positive and unlabeled data.

4.1 Introduction

Capturing relation between input and output is an important task in statistical data analysis. To this end, a statistical dependency measure plays a central role. Among statistical dependency measures, based on the *information maximization principle*, *mutual information* (MI) (Cover and Thomas, 2006), in particular, its empirical estimator has been employed to solve various machine learning tasks (Torkkola, 2003; Krause et al., 2010). However, since the MI estimator is known to be sensitive to outliers (Basu et al., 1998), it limits the range of applications of the MI-based methods. To cope with this problem, *squared-loss MI* (SMI) and its empirical estimator were proposed (Suzuki et al., 2009). Unlike MI, the SMI estimator is robust to noise and outliers, and having superior numerical properties (Kanamori et al., 2009), allowing us to apply the estimator to various statistical dependency analysis (Sugiyama, 2013).

On the other hand, as discussed throughout this dissertation, it is conceivable that only positive and unlabeled data are available (the PU learning situation) in real-world applications. So far, attention in PU learning has mostly been paid to classification (Elkan and Noto, 2008; du Plessis et al., 2015), and there are few studies on statistical dependency analysis for PU data. Recently, a PU-based independence test was proposed (Sechidis et al., 2014), but its range of applications is limited since the method can only be applied

to a categorical univariate input. Furthermore, there seems no other statistical dependency analysis, such as dimension reduction, for PU data. To analyze PU data from many aspects, it is practically desirable to have statistical dependency analysis for PU data.

In this chapter, we discuss the problem of estimating SMI and SMI-based statistical data analyses under the PU learning setting. We first derive an SMI expression over PU data (PU-SMI), which is equivalent to its supervised counterpart, i.e., an SMI expression over PN data. Then, we develop an estimation method for PU-SMI and show that the PU-SMI estimator converges to true SMI at the optimal parametric rate. We apply the PU-SMI estimator to dimension reduction and independence test. In particular, dimension reduction can be executed without estimating the class-prior, which is indispensable for most of the existing PU classification methods. However, the class-prior estimation from PU data is known to be challenging (du Plessis and Sugiyama, 2014; Ramaswamy et al., 2016), especially for high-dimensional data. Thus, our dimension reduction can mitigate the issue by providing informative low-dimensional representations.

4.2 Problem Setting

Let $\boldsymbol{x} \in \mathbb{R}^d$ be an input pattern, $y \in \{\pm 1\}$ be a corresponding class label, and $p(\boldsymbol{x}, y)$ be the underlying joint density, where d is a positive integer. In PU learning, we cannot access negatively labeled data, but we can utilize unlabeled data. Let us define *positive* (P) and *unlabeled* (U) data as

$$\mathcal{X}_{\mathrm{P}} := \{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x} \mid y = +1), \tag{4.1}$$

$$\mathcal{X}_{\mathrm{U}} := \{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x}) = \theta_{\mathrm{P}} p(\boldsymbol{x} \mid y = +1) + \theta_{\mathrm{N}} p(\boldsymbol{x} \mid y = -1), \tag{4.2}$$

where

$$\theta_{\mathrm{P}} := p(y = +1) \ \text{ and } \ \theta_{\mathrm{N}} := p(y = -1).$$

The goal is to develop statistical data analysis tools for PU data. To this end, we first show an SMI estimation method from PU data and apply the estimator to data analysis tasks.

4.3 SMI Estimation from Positive and Unlabeled Data

In this section, we first reformulate SMI with only positive and unlabeled data, which is equivalent to SMI with positive and negative data, and then describe an SMI estimation method.

4.3.1 SMI with Positive and Unlabeled Data (PU-SMI)

Let us express SMI in Eq. (2.94) as

$$\mathrm{SMI} = \frac{\theta_{\mathrm{P}}}{2}\int\Big(\frac{p(\boldsymbol{x}\mid y=+1)}{p(\boldsymbol{x})}-1\Big)^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + \frac{\theta_{\mathrm{N}}}{2}\int\Big(\frac{p(\boldsymbol{x}\mid y=-1)}{p(\boldsymbol{x})}-1\Big)^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}. \tag{4.3}$$

From the definition of the marginal density $p(\boldsymbol{x})$, we have

$$\begin{aligned}
\theta_{\mathrm{N}}\frac{p(\boldsymbol{x}\mid y=-1)}{p(\boldsymbol{x})} &= 1-\theta_{\mathrm{P}}\frac{p(\boldsymbol{x}\mid y=+1)}{p(\boldsymbol{x})},\\
\theta_{\mathrm{N}}\Big(\frac{p(\boldsymbol{x}\mid y=-1)}{p(\boldsymbol{x})}-1\Big) &= \theta_{\mathrm{P}}\Big(1-\frac{p(\boldsymbol{x}\mid y=+1)}{p(\boldsymbol{x})}\Big),\\
\Big(\frac{p(\boldsymbol{x}\mid y=-1)}{p(\boldsymbol{x})}-1\Big)^2 &= \frac{\theta_{\mathrm{P}}^2}{\theta_{\mathrm{N}}^2}\Big(\frac{p(\boldsymbol{x}\mid y=+1)}{p(\boldsymbol{x})}-1\Big)^2,
\end{aligned} \tag{4.4}$$

where the equality between the first and second equations can be confirmed by using $\theta_{\mathrm{P}}+\theta_{\mathrm{N}}=1$. Plugging the last equation into the second term of Eq. (4.3), we then obtain an expression of SMI only with positive and unlabeled data (PU-SMI) as

$$\mathrm{SMI} = \frac{\theta_{\mathrm{P}}}{2\theta_{\mathrm{N}}}\int\Big(\frac{p(\boldsymbol{x}\mid y=+1)}{p(\boldsymbol{x})}-1\Big)^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} := \text{PU-SMI}. \tag{4.5}$$

This immediately leads to the following lemma.

Lemma 8. *SMI and PU-SMI are equivalent.*

Compared with the original definition of SMI, PU-SMI can be expressed as only the probability densities of positive and unlabeled data. In the next section, we explain the estimation method for PU-SMI.

4.3.2 PU-SMI Estimation

Next, we show an SMI estimation method from only positive and unlabeled data.

Let

$$s(\boldsymbol{x}) := \frac{p(\boldsymbol{x} \mid y=+1)}{p(\boldsymbol{x})} \tag{4.6}$$

be the density-ratio function and g be a model of s, and we learn s so that the squared error (the PU-SMI risk) between model g and true ratio s is minimized:

$$R_{\mathrm{PU}}^{\mathrm{SMI}}(g) := \frac{1}{2}\int \Big(g(\boldsymbol{x}) - \frac{p(\boldsymbol{x} \mid y=+1)}{p(\boldsymbol{x})}\Big)^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} - C_{\mathrm{PU}} \tag{4.7}$$

$$= \frac{1}{2}\int g^2(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} - \int g(\boldsymbol{x})p(\boldsymbol{x} \mid y=+1)\mathrm{d}\boldsymbol{x}, \tag{4.8}$$

where $C_{\mathrm{PU}} := \frac{1}{2}\int \frac{p^2(\boldsymbol{x}|y=+1)}{p(\boldsymbol{x})}\mathrm{d}\boldsymbol{x}$. An empirical PU-SMI risk can be obtained by

$$\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}(g) := \frac{1}{2n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}} g^2(\boldsymbol{x}_k^{\mathrm{U}}) - \frac{1}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}} g(\boldsymbol{x}_i^{\mathrm{P}}). \tag{4.9}$$

Similarly to the existing SMI estimation method, we formulate our optimization problem as

$$\widehat{g} := \operatorname*{argmin}_{g} \Big[\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}(g) + \lambda W(g)\Big], \tag{4.10}$$

where $\lambda \geq 0$ is the regularization parameter.

With the obtained density-ratio estimator, a PU-SMI estimator can be given by

$$\widehat{\text{PU-SMI}} = \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\Big(\frac{1}{n_{\mathrm{P}}}\sum_{i=1}^{n_{\mathrm{P}}} \widehat{g}(\boldsymbol{x}_i^{\mathrm{P}}) - \frac{1}{2n_{\mathrm{U}}}\sum_{k=1}^{n_{\mathrm{U}}} \widehat{g}^2(\boldsymbol{x}_k^{\mathrm{U}}) - \frac{1}{2}\Big), \tag{4.11}$$

since PU-SMI can be expressed as

$$\begin{aligned}\text{PU-SMI} &= \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\Big(\frac{1}{2}\int s(\boldsymbol{x})p(\boldsymbol{x} \mid y=+1)\mathrm{d}\boldsymbol{x} - \frac{1}{2}\Big) \\ &= \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\Big(\int s(\boldsymbol{x})p(\boldsymbol{x} \mid y=+1)\mathrm{d}\boldsymbol{x} - \frac{1}{2}\int s^2(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} - \frac{1}{2}\Big),\end{aligned} \tag{4.12}$$

where the first equation can be obtained by expanding Eq. (4.5) and using $s(\boldsymbol{x})p(\boldsymbol{x}) = p(\boldsymbol{x}|y=+1)$, and the equality of the first and second equations can be confirmed by plugging $s(\boldsymbol{x})p(\boldsymbol{x}) = p(\boldsymbol{x}|y=+1)$ into the second term of the second equation.

Similarly to Suzuki and Sugiyama (2013), the minimization of the empirical PU-SMI

risk $\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}$ corresponds to the maximization of $-\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}$ which is a lower-bound of PU-SMI as in the f-divergence estimation (Keziou, 2003; Nguyen et al., 2007). That is, the PU-SMI estimator can also be obtained by maximizing the lower-bound:

$$\widehat{\mathrm{PU\text{-}SMI}} = \max_{g} -\frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\Big(\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}(g) + \frac{1}{2}\Big) = -\frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\Big(\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}(\widehat{g}) + \frac{1}{2}\Big). \tag{4.13}$$

Based on this fact, we derive convergence results of the parameters of the density-ratio model and the PU-SMI estimator in Section 4.5.

The PU-SMI risk minimization for estimating density-ratio does not involve class-prior θ_{P} (and θ_{N}), and PU-SMI is proportional to class-prior ratio $\theta_{\mathrm{P}}/\theta_{\mathrm{N}}$, meaning that the class-prior estimation is not necessary as long as our purpose is maximizing (minimizing) PU-SMI. For instance, in SMI-based dimensionality reduction discussed in Section 4.6.1, the ratio of the class-prior ratio $\theta_{\mathrm{P}}/\theta_{\mathrm{N}}$ can be safely ignored from the PU-SMI maximization. Since class-prior estimation only from PU data is known to be challenging (du Plessis and Sugiyama, 2014; Ramaswamy et al., 2016; Jain et al., 2016; du Plessis et al., 2017), being able to avoid class-prior estimation is a significant advantage of the SMI-based method in practice.

4.4 Practical Implementation

In this section, we explain a practical implementation of the PU-SMI estimation method.

4.4.1 Neural Network and Linear-in-Parameter Models

When we use a neural network model, a solution can be typically obtained by backpropagation (Goodfellow et al., 2016). To this end, various deep learning implementations (Jia et al., 2014; Abadi et al., 2015; Tokui et al., 2015; Chen et al., 2015) may be used for obtaining a learned model.

Another candidate of the density-ratio model is a linear-in-parameter model:

$$g(\boldsymbol{x}) = \sum_{\ell=1}^{b} w_\ell \phi_\ell(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}), \tag{4.14}$$

where $\boldsymbol{w} := (w_1, \ldots, w_b)^\top$ is a vector of parameters, and $\boldsymbol{\phi}(\boldsymbol{x}) := (\phi_1(\boldsymbol{x}), \ldots, \phi_b(\boldsymbol{x}))^\top$ is a vector of basis functions. This model allows us to obtain an analytic-form solution with the ℓ_2-regularizer and a PU-SMI estimator, similarly to the existing SMI estimation

method. Furthermore, the optimal convergence is theoretically guaranteed as shown in Section 4.5.

More specifically, with the ℓ_2-regularizer, the optimization problem is given by

$$\widehat{\boldsymbol{w}} := \operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^\top \widehat{\boldsymbol{H}}_{\mathrm{U}}\boldsymbol{w} - \boldsymbol{w}^\top \widehat{\boldsymbol{h}}_{\mathrm{P}} + \frac{\lambda}{2}\boldsymbol{w}^\top \boldsymbol{w}, \tag{4.15}$$

where $\lambda \geq 0$ is the regularization parameter and

$$\widehat{\boldsymbol{H}}_{\mathrm{U}} := \frac{1}{n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^\top \boldsymbol{\Phi}_{\mathrm{U}}, \tag{4.16}$$

$$\widehat{\boldsymbol{h}}_{\mathrm{P}} := \frac{1}{n_{\mathrm{P}}}\boldsymbol{\Phi}_{\mathrm{P}}^\top \mathbf{1}_{n_{\mathrm{P}}}, \tag{4.17}$$

$$\boldsymbol{\Phi}_{\mathrm{P}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{P}}), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{P}}}^{\mathrm{P}}))^\top, \tag{4.18}$$

$$\boldsymbol{\Phi}_{\mathrm{U}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{U}}), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{U}}}^{\mathrm{U}}))^\top, \tag{4.19}$$

where $\mathbf{1}_b$ is the b-dimensional vector whose elements are all ones. The solution can be obtained analytically by differentiating the objective function with respect to $\boldsymbol{w}$ and set it to zero:

$$\widehat{\boldsymbol{w}} = (\widehat{\boldsymbol{H}}_{\mathrm{U}} + \lambda \boldsymbol{I}_b)^{-1}\widehat{\boldsymbol{h}}_{\mathrm{P}}, \tag{4.20}$$

where $\boldsymbol{I}_b$ is the $b \times b$ identity matrix. Finally, with the obtained estimator, we can compute an SMI approximator only from positive and unlabeled data:

$$\widehat{\mathrm{PU\text{-}SMI}} = \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\Big(\widehat{\boldsymbol{w}}^\top \widehat{\boldsymbol{h}}_{\mathrm{P}} - \frac{1}{2}\widehat{\boldsymbol{w}}^\top \widehat{\boldsymbol{H}}_{\mathrm{U}}\widehat{\boldsymbol{w}} - \frac{1}{2}\Big). \tag{4.21}$$

4.4.2 Model Selection

In practice, the performance of the PU-SMI estimator depends on hyperparameters such as the regularization parameter and the number of epochs for backpropagation.

One way is to prepare a validation dataset and tune hyperparameters based on it. With positive and unlabeled samples for validation, we compute the empirical PU-SMI risk in Eq. (4.9) and choose the best model that minimizes the empirical PU-SMI risk from all model candidates.

Another way is to use cross-validation. The procedure is the same as cross-validation for the PU-AUC optimization in Section 3.4.3. The notable difference is that cross-validation for the PU-SMI estimation does not require an estimate of the class-prior (see

the discussion in Section 4.3.2).

4.5 Theoretical Analysis

In this section, we derive the convergence rate of the parameter of the density-ratio model and the PU-SMI approximator based on the *perturbation analysis of optimization problems* (Bonnans and Cominetti, 1996; Bonnans and Shapiro, 1998).

Assume the linear-in-parameter model in Eq.(4.14) for the density-ratio $\frac{p(\boldsymbol{x}|y=+1)}{p(\boldsymbol{x})}$. Without loss of generality, we assume that the basis functions $\{\phi_\ell(\boldsymbol{x})\}_{\ell=1}^b$ are linearly independent over the marginal density $p(\boldsymbol{x}) > 0$ and satisfy $0 \leq \phi_\ell(\boldsymbol{x}) \leq 1$ for all $\ell = 1, \ldots, b$ and $\boldsymbol{x} \in \mathbb{R}^d$. Moreover, we assume that there exists a constant M such that $\|\boldsymbol{w}\|_2 \leq M$, i.e., $\boldsymbol{w}$ is regularized.

Let

$$\boldsymbol{w}^* := \operatorname*{argmin}_{\boldsymbol{w}} \; R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}) \tag{4.22}$$

be the ideal estimate of the parameter of the density-ratio model. Let $\mathcal{O}_p$ denote the order in probability. Then we have the following convergence result (its proof is given in Section 4.8):

Theorem 9. *As $n_{\mathrm{P}}, n_{\mathrm{U}} \to \infty$, we have*

$$\|\widehat{\boldsymbol{w}} - \boldsymbol{w}^*\|_2 = \mathcal{O}_p\Big(\frac{1}{\sqrt{n_{\mathrm{P}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{4.23}$$

$$|\mathrm{PU\text{-}SMI} - \widehat{\mathrm{PU\text{-}SMI}}| = \mathcal{O}_p\Big(\frac{1}{\sqrt{n_{\mathrm{P}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{4.24}$$

provided that $\lambda = \mathcal{O}_p(1/\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})})$.[1]

Theorem 9 guarantees that the convergence of the density-ratio estimator and the PU-SMI estimator. In our setting, since n_{P} and n_{U} can increase independently, this is the optimal convergence rate without any additional assumption (Kanamori et al., 2009, 2012).

Note that Theorem 9 shows that both positive and unlabeled data contribute to convergence, implying that unlabeled data is directly used in the estimation rather than extracting the information of a data structure, such as the cluster structure frequently assumed in semi-supervised learning (Chapelle et al., 2006).

[1] The regularization parameter λ vanishes with the order of $\min(n_{\mathrm{P}}, n_{\mathrm{U}})$, i.e., the convergence rate is dominated by the smaller size of positive and unlabeled data.

4.6 Application based on PU-SMI

In this section, we show two applications of PU-SMI, along the line of existing SMI-based applications.

4.6.1 Dimension Reduction

Here, we extend the existing SMI-based dimension reduction (Suzuki and Sugiyama, 2013), called *least-squares dimension reduction* (LSDR), to the PU learning setting.

Algorithm: Let $\boldsymbol{v}\colon \mathbb{R}^d \to \mathbb{R}^m$, where $m < d$, be a mapping function from an input vector to its low-dimensional representation. Following the information-maximization principle (Linsker, 1988), we maximize PU-SMI with respect to the mapping function to find low-dimensional representation that maximally preserves dependency between input and output.

To obtain such a mapping function, we propose an alternating optimization procedure to maximize SMI—First, we estimate PU-SMI, and then we maximize the estimated PU-SMI with respect to the mapping function. More specifically, we solve the objective function in Eq. (4.9) to approximate PU-SMI while fixing the mapping function $\widehat{\boldsymbol{v}}$:

$$\min_{g} \ \widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}(g \circ \widehat{\boldsymbol{v}}), \tag{4.25}$$

where $\widehat{\boldsymbol{v}}$ is a current estimate of the mapping function and "$\circ$" denotes the function composition, i.e., $(g \circ \boldsymbol{v})(\boldsymbol{x}) = g(\boldsymbol{v}(\boldsymbol{x}))$. Note that the input dimension of g is redefined as m, i.e., $g\colon \mathbb{R}^m \to \mathbb{R}$, accordingly. Then, we maximize the estimated PU-SMI with respect to $\boldsymbol{v}$ while fixing $\widehat{g}$:

$$\max_{\boldsymbol{v}} \ \widehat{\mathrm{PU\text{-}SMI}}(\widehat{g} \circ \boldsymbol{v}), \tag{4.26}$$

where $\widehat{g}$ is a current estimate of the density ratio.

As discussed in Section 4.3.2, an advantage of our dimension reduction method is that the whole procedure can be executed without estimating the class-prior, unlike existing PU learning methods (Elkan and Noto, 2008; du Plessis et al., 2015). This can be easily confirmed by Eq. (4.13):

$$\operatorname*{argmax}_{\boldsymbol{v}} \ \widehat{\mathrm{PU\text{-}SMI}}(\widehat{g} \circ \boldsymbol{v}) = \operatorname*{argmax}_{\boldsymbol{v}} \ -\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}(\widehat{g} \circ \boldsymbol{v}). \tag{4.27}$$

Algorithm 1 PUDR

Require: $\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}}$, $\{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}}$, g, and $\boldsymbol{v}$

1: Initialize g and $\boldsymbol{v}$
2: **repeat**
3: Update $\widehat{g} \leftarrow \widehat{g} - \varepsilon_g \nabla_g \widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}(g \circ \widehat{\boldsymbol{v}})$
4: Update $\widehat{\boldsymbol{v}} \leftarrow \widehat{\boldsymbol{v}} - \varepsilon_{\boldsymbol{v}} \nabla_{\boldsymbol{v}} \widehat{R}_{\mathrm{PU}}^{\mathrm{SMI}}(\widehat{g} \circ \boldsymbol{v})$
5: **until** stopping conditions meet

Moreover, minimization of the squared error does not require an estimate of the class-prior. Thus, our dimension reduction method does not need for a class-prior estimation in advance.

We refer to the dimension reduction method for PU data as *PUDR*. A pseudo-code of PUDR is summarized in Algorithm 1.

Linear Transformation: A choice of a mapping function is to use a linear transformation similarly to the existing SMI-based dimension reduction method (Suzuki and Sugiyama, 2013). As in the existing method, we search a projection matrix $\boldsymbol{W} \in \mathbb{R}^{d \times m}$ that satisfies $\boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I}_m$ to avoid degenerated solutions. Note that if a linear transformation is used, the maximization of SMI corresponds to *sufficient dimension reduction* (Li, 1991).

Non-Linear Transformation: Another choice is to use non-linear transformation for capturing complex structures of the data. To obtain low-dimensional representation from a non-linear mapping, we use a neural network because we can use several useful techniques developed recently. Specifically, we regard $g \circ \boldsymbol{v}$ as a neural network. We set the last layer as a fully-connected layer, which corresponds to g and the parameter $\boldsymbol{w}$ of the linear-in-parameter model. Layers before the last layer can be regarded as the basis function $\boldsymbol{\phi}$ and it can be anything; we could use, for instance, a convolution layer or a pretrained network.

4.6.2 Independence Test

Here, we extend the existing SMI-based independence test, called *least-squares independence test* (LSIT), to the PU learning setting.

Algorithm 2 PUIT

Require: $\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}}$, $\{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}}$, and T

Prepare a memory h of size T
Compute $\widehat{\mathrm{PU\text{-}SMI}}$ with $\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}}$ and $\{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}}$
$i \leftarrow 0$
repeat
    Generate random labels $\{\widetilde{y}_k\}_{k=1}^{n_{\mathrm{U}}}$
    Compute $\widetilde{\mathrm{SMI}}$ with $\{(\boldsymbol{x}_k^{\mathrm{U}}, \widetilde{y}_k)\}_{k=1}^{n_{\mathrm{U}}}$
    Store $\widetilde{\mathrm{SMI}}$ in $h[i]$
    $i \leftarrow i + 1$
until $i \leq T$
Construct a distribution of $\widetilde{\mathrm{SMI}}$ with h
Approximate p-value with $\widehat{\mathrm{PU\text{-}SMI}}$ and the distribution of $\widetilde{\mathrm{SMI}}$

In the independence test, we test whether a pair of random variables are independent or not. Since SMI takes zero if and only if two random variables are independent, we utilize the property to detect independence of variables. Similarly to Sugiyama and Suzuki (2011), we employ a permutation test (Efron and Tibshirani, 1994) for numerically computing a p-*value* for the test.

In the original SMI-based test, we first estimate SMI from paired samples; then we repeatedly compute SMI with samples whose labels are randomly shuffled to construct a distribution of the SMI approximator when two random variables are independent. However, unlike the fully supervised case, we do not have labels to be shuffled in our setting. To conduct the permutation test, we propose the following procedure. We assign random labels $\{\widetilde{y}_k\}_{k=1}^{n_{\mathrm{U}}}$ to unlabeled data $\{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}}$ and compute $\widetilde{\mathrm{SMI}}$ from the randomly labeled data $\{(\boldsymbol{x}_k^{\mathrm{U}}, \widetilde{y}_k)\}_{k=1}^{n_{\mathrm{U}}}$. We compute $\widetilde{\mathrm{SMI}}$ many times and then construct the distribution of $\widetilde{\mathrm{SMI}}$. Then, we approximate a p-value from the constructed distribution by comparing the estimate of $\widehat{\mathrm{PU\text{-}SMI}}$ computed from the obtained data. Finally, we test the null hypothesis (the pair of random variables are independent) with the obtained p-value. We refer to the above independence test for PU data as *PUIT*. A pseudo-code of PUDR is summarized in Algorithm 2.

Note that the class proportion of random labels $\{\widetilde{y}_k\}_{k=1}^{n_{\mathrm{U}}}$ is determined by the estimated class-prior from PU data so that the class-prior of randomly paired samples $\{(\boldsymbol{x}_k^{\mathrm{U}}, \widetilde{y}_k)\}_{k=1}^{n_{\mathrm{U}}}$ is the same as that of the marginal distribution $p(\boldsymbol{x})$, i.e., $p(\boldsymbol{x} \mid y)p(y) = p(\boldsymbol{x})p(y)$.

4.7 Experiments

In this section, we experimentally investigate the behavior of the PU-SMI estimator and evaluate the performance of the dimension reduction and independence testing methods on various benchmark datasets: the MNIST (LeCun et al., 1998), the CIFAR10 (Torralba et al., 2008), the SVHN (Netzer et al., 2011) datasets, and several datasets obtained from the *LIBSVM* webpage (Chang and Lin, 2011).

4.7.1 Experimental Analysis

First, we investigate the estimation accuracy of the PU-SMI estimator.

We used the linear-in-parameter model with the Gaussian basis function

$$\phi_\ell(\boldsymbol{x}) := \exp\Big(-\frac{\|\boldsymbol{x}-\boldsymbol{x}_\ell\|^2}{2\sigma^2}\Big), \tag{4.28}$$

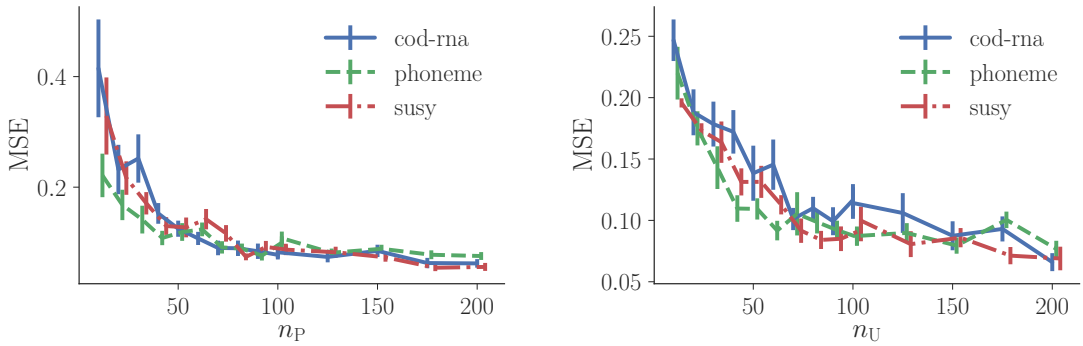
where $\sigma > 0$ is the bandwidth and $\{\boldsymbol{x}_\ell\}_{\ell=1}^b$ are the Gaussian centers randomly sampled from $\{\boldsymbol{x}_k^{\mathrm{U}}\}_{k=1}^{n_{\mathrm{U}}}$. The number of basis functions was set at $b = \min(n_{\mathrm{U}}, 200)$. The candidates of the Gaussian bandwidth were

$$\mathrm{median}(\{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\}_{i,j=1}^n) \times \Big\{\frac{1}{8},\ \frac{1}{4},\ \frac{1}{2},\ 1,\ 2\Big\} \tag{4.29}$$

and that of the regularization parameter were $\{10^{-3},\ 10^{-2},\ 10^{-1},\ 10^{0},\ 10^{1}\}$. The bandwidth and regularization parameter were determined by five-fold cross-validation. We varied the number of positive (unlabeled) samples from 10 to 200, with the number of unlabeled (positive) samples fixed. The class-prior was assumed to be known in this illustrative experiment and set at $\theta_{\mathrm{P}} = 0.5$.

Figure 4.1 summarizes the average and standard error of the squared error of PU-SMI over 50 trials[2]. The results show that the mean squared error decreased both when the number of positive samples was increased and the number of unlabeled samples was increased. Thus, both positive and unlabeled data contributed to improving the estimation accuracy of SMI, which well agrees with the theoretical result in Section 4.5.

[2]True SMI was computed by a supervised SMI estimator (Suzuki et al., 2009) with a sufficiently large number of positive and negative samples.



(a) $n_{\mathrm{U}} = 400$ (b) $n_{\mathrm{P}} = 200$

FIGURE 4.1: Average and standard error of the squared error of PU-SMI over 50 trials. (a) n_{P} is increased while $n_{\mathrm{U}} = 400$ is fixed. (b) n_{U} is increased while $n_{\mathrm{P}} = 200$ is fixed. The results show that both positive and unlabeled samples contribute to improving the estimation accuracy of SMI.

4.7.2 Dimension Reduction

Next, we evaluate the performance of the dimension reduction method based on PU-SMI (PUDR).

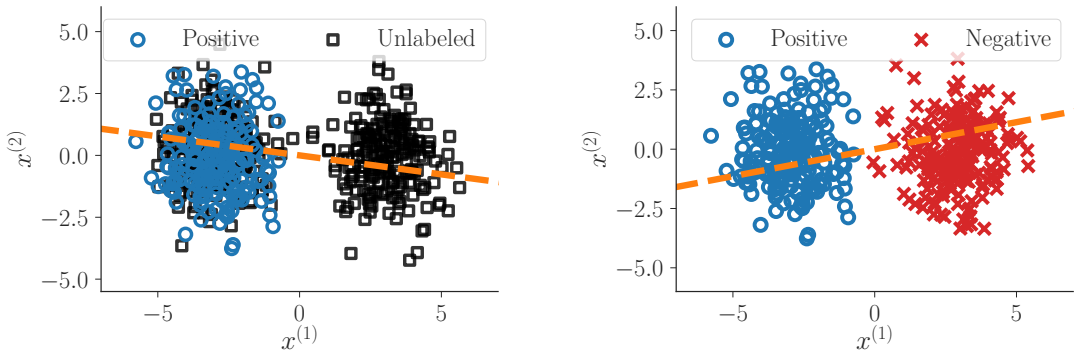
Illustration: We first illustrate how the PUDR method works on an artificial dataset. As a reference, we also show the result of the existing SMI-based dimension reduction method (LSDR) (Suzuki and Sugiyama, 2013) that uses both positive and negative data. For this illustrative experiment, we used a linear transformation as a projection matrix (i.e., linear dimension reduction).

For the artificial data, we generated samples from the following densities:

$$p(\boldsymbol{x} \mid y = +1) = N\left(\boldsymbol{x}; \begin{pmatrix} -3 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}\right), \tag{4.30}$$

$$p(\boldsymbol{x} \mid y = -1) = N\left(\boldsymbol{x}; \begin{pmatrix} 3 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}\right), \tag{4.31}$$

where $N(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the normal density with the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$. From the densities, we drew $n_{\mathrm{P}} = 200$ positive and $n_{\mathrm{U}} = 400$ unlabeled samples for PUDR, and $n_{\mathrm{P}} = 200$ positive and $n_{\mathrm{N}} = 200$ negative samples for LSDR. Other experimental settings were the same as those in Section 4.7.1.



(a) PUDR (b) LSDR

FIGURE 4.2: Illustration of linear dimension reduction. (a) Estimated subspace obtained by PUDR, where positive "○" and unlabeled samples "□" are used. (b) Estimated subspace obtained by LSDR, where positive "○" and negative samples "×" are used. The results show that from only PU data, the PUDR method gave a subspace similar to that obtained by its supervised counterpart.

We plot the estimated subspaces obtained by the PUDR and LSDR methods in Figure 4.2, showing that from only PU samples, the PUDR method obtained a subspace similar to that obtained by LSDR using positive and negative data.

Benchmark Data: Here, we apply the PUDR method to benchmark datasets. To obtain low-dimensional representation, we used a fully-connected neural network with four layers (d−60−20−1), the sigmoid function was used as the activation function, and *batch normalization* (Ioffe and Szegedy, 2015) was applied to all hidden layers. Stochastic gradient descent was used for optimization with learning rate 0.01. Also, weight decay with 0.001 and gradient noise with 0.01 were applied. We iteratively updated g (the last layer) with four minibatches and $\boldsymbol{v}$ (layers before the last layer) with one minibatch.

We compared the accuracy of class-prior estimation with and without dimension reduction. For comparison, we also considered principal component analysis (PCA) with different numbers of components: $\lfloor d/4 \rfloor$, $\lfloor d/2 \rfloor$, and $\lfloor 3d/4 \rfloor$, where $\lfloor \cdot \rfloor$ is the floor function. As a class-prior estimation method, we used the method based on the *kernel mean embedding* (KMM) method proposed by Ramaswamy et al. (2016). We used the mushrooms, the a9a, the MNIST, the CIFAR10, and the SVHN datasets. For multi-class classification datasets, we divided the whole classes into 2 groups to make binary classification

TABLE 4.1: Mean absolute error (with standard error) between the estimated class-prior and the true value over 20 trials. KM means the class-prior estimation method based on kernel mean embedding, PCA is the principal component analysis. The boldface denotes the best and comparable approach in terms of the average absolute error according to the t-test at the significance level 5%.

Dataset	θ_{P}	KM	PCA+KM			PUDR+KM
			$\lfloor d/4 \rfloor$	$\lfloor d/2 \rfloor$	$\lfloor 3d/4 \rfloor$	
mushrooms	0.3	0.05 (0.00)	**0.03 (0.00)**	0.05 (0.00)	0.05 (0.00)	**0.03 (0.00)**
	0.5	**0.04 (0.01)**	**0.04 (0.00)**	**0.04 (0.01)**	**0.04 (0.01)**	**0.04 (0.01)**
	0.7	**0.03 (0.01)**	**0.03 (0.00)**	**0.03 (0.01)**	**0.03 (0.01)**	0.05 (0.01)
a9a	0.3	0.11 (0.01)	0.11 (0.01)	0.11 (0.01)	0.11 (0.01)	**0.04 (0.01)**
	0.5	0.10 (0.01)	0.10 (0.01)	0.10 (0.01)	0.10 (0.01)	**0.04 (0.01)**
	0.7	0.08 (0.01)	0.08 (0.01)	0.08 (0.01)	0.08 (0.01)	**0.03 (0.01)**
MNIST	0.3	0.10 (0.00)	0.10 (0.00)	0.10 (0.00)	0.10 (0.00)	**0.06 (0.01)**
	0.5	0.12 (0.01)	0.12 (0.01)	0.12 (0.01)	0.12 (0.01)	**0.07 (0.01)**
	0.7	0.53 (0.07)	0.53 (0.07)	0.53 (0.07)	0.53 (0.07)	**0.10 (0.01)**
CIFAR10	0.3	0.30 (0.02)	0.57 (0.00)	0.56 (0.01)	0.50 (0.02)	**0.17 (0.01)**
	0.5	0.48 (0.01)	0.37 (0.00)	0.34 (0.01)	0.35 (0.01)	**0.06 (0.01)**
	0.7	0.65 (0.03)	0.17 (0.00)	**0.15 (0.01)**	**0.16 (0.01)**	**0.12 (0.01)**
SVHN	0.3	**0.08 (0.03)**	0.57 (0.00)	0.52 (0.03)	0.53 (0.01)	0.32 (0.02)
	0.5	**0.04 (0.00)**	0.37 (0.00)	0.29 (0.02)	0.34 (0.01)	0.19 (0.02)
	0.7	0.36 (0.08)	0.17 (0.00)	0.13 (0.02)	0.18 (0.02)	**0.06 (0.01)**

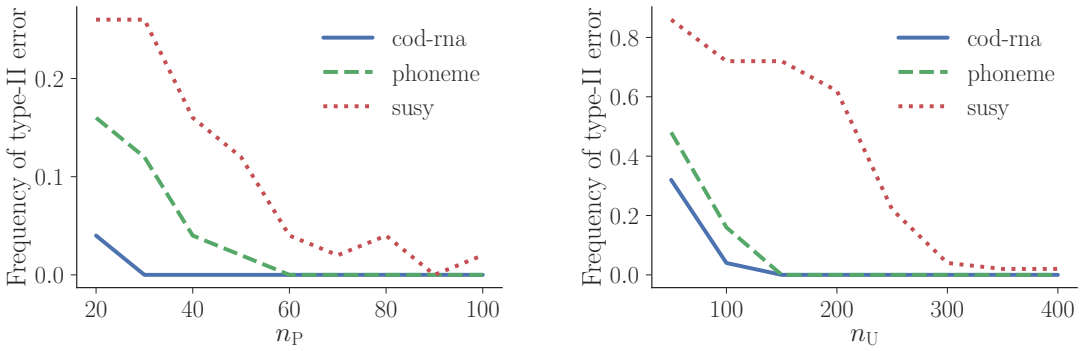
tasks. From the datasets, we drew $n_{\mathrm{P}} = 1000$ positive and $n_{\mathrm{U}} = 2000$ unlabeled samples. For model selection, we used validation samples of size $n_{\mathrm{P}} = 50$ and $n_{\mathrm{U}} = 200$.

Table 4.1 lists the mean absolute error (with standard error) between an estimated class-prior and the true value. Overall, the PUDR method tends to outperform other methods, meaning that the PUDR method provided useful low-dimensional representation except the SVHN dataset when $\theta_{\mathrm{P}} = 0.3$ and 0.5. For the mushrooms, a9a, and MNIST datasets, applying the unsupervised dimension reduction method, PCA, did not improve the estimation accuracy. In fact, PCA performed poorly on the CIFAR10 and SVHN datasets, in particular, when $\theta_{\mathrm{P}} = 0.3$.

4.7.3 Independence Test

Finally, we evaluate the performance of the SMI-based independence test (PUIT).

We computed the frequency of the *type-II* error, i.e., the number of times the null



(a) $n_{\mathrm{U}} = 400$ (b) $n_{\mathrm{P}} = 100$

FIGURE 4.3: Frequency of the type-II error (the number of times the null hypothesis is accepted when the alternative hypothesis is true) of the PU-SMI based independence test with the significance level 5% over 50 trials. (a) n_{P} is increased while $n_{\mathrm{U}} = 400$ is fixed. (b) n_{U} is increased while $n_{\mathrm{P}} = 100$ is fixed. In both cases, the frequency of the type-II error decreased as the number of positive/unlabeled samples increases.

hypothesis (two random variables are independent) is accepted when the alternative hypothesis is true. Similarly to Section 4.7.1, we used classification datasets. Thus, the two random variables (pattern $\boldsymbol{x}$ and class label y) are highly dependent, i.e., the frequency of the type-II error is expected to be low. Other experimental settings were the same as those in Section 4.7.1.

Figure 4.3 summarizes the frequency of the type-II error by the PUIT method at the significance level 5% over 50 trials. In both cases, the frequency decreased as the number of positive and unlabeled samples increased, showing that the PUIT method could detect statistical dependency well only from PU data.

4.8 Proof of Theorem 9

The idea of the proof is to regard the approximated squared error as perturbed optimization of expected one. Recall the linear-in-parameter model $g(\boldsymbol{x}) = \sum_{\ell=1}^{b} w_\ell \phi_\ell(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x})$, where $0 \leq \phi_\ell \leq 1$ for all $\ell = 1, \ldots, b$ and $\boldsymbol{x} \in \mathbb{R}^d$, and there exists a constant such that $\|\boldsymbol{w}\|_2 \leq M$. Moreover, we assume that the basis functions $\{\phi_\ell(\boldsymbol{x})\}_{\ell=1}^{b}$ are linearly independent over the marginal density $p(\boldsymbol{x}) > 0$.

Let

$$R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}) := \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{H}_{\mathrm{U}}\boldsymbol{w}^\top - \boldsymbol{w}^\top \boldsymbol{h}_{\mathrm{P}}, \tag{4.32}$$

$$\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\boldsymbol{w}) := \frac{1}{2}\boldsymbol{w}^\top \widehat{\boldsymbol{H}}_{\mathrm{U}}\boldsymbol{w}^\top - \boldsymbol{w}^\top \widehat{\boldsymbol{h}}_{\mathrm{P}} + \frac{\lambda_{\mathrm{PU}}}{2}\boldsymbol{w}^\top \boldsymbol{w} \tag{4.33}$$

be the expected squared error and the ℓ_2-regularized empirical squared error, respectively.

To begin with, the following second order growth condition is proved:

Lemma 10. *Let ϵ be the smallest eigenvalue of $\boldsymbol{H}_{\mathrm{U}}$. The second order growth condition holds*

$$R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}) \geq R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*) + \epsilon\|\boldsymbol{w} - \boldsymbol{w}^*\|_2^2. \tag{4.34}$$

Proof. Since $\boldsymbol{H}_{\mathrm{U}}$ is positive definite provided the linearly independent basis functions over $p(\boldsymbol{x})$, $R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w})$ is strongly convex with parameter at least ϵ. Thus, we have

$$\begin{aligned} R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}) &\geq R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*) + \nabla R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*)^\top(\boldsymbol{w} - \boldsymbol{w}^*) + \epsilon\|\boldsymbol{w} - \boldsymbol{w}^*\|_2^2 \\ &\geq R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*) + \epsilon\|\boldsymbol{w} - \boldsymbol{w}^*\|_2^2, \end{aligned} \tag{4.35}$$

where the optimality condition $\nabla R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*) = \boldsymbol{0}$ is used. □

Next, let us define a set of perturbation parameters as

$$\boldsymbol{u} := \{\boldsymbol{U}_{\mathrm{U}},\ \boldsymbol{u}_{\mathrm{P}} \mid \boldsymbol{U}_{\mathrm{U}} \in \mathbb{S}_+^b,\ \boldsymbol{u}_{\mathrm{P}} \in \mathbb{R}^b\}, \tag{4.36}$$

where $\mathbb{S}_+^b \subset \mathbb{R}^{b\times b}$ is the cone of $b \times b$ positive semi-definite matrices. Our perturbed objective function and the solution are given by

$$R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}, \boldsymbol{u}) := \frac{1}{2}\boldsymbol{w}^\top(\boldsymbol{H}_{\mathrm{U}} + \boldsymbol{U}_{\mathrm{U}})\boldsymbol{w} - \boldsymbol{w}^\top(\boldsymbol{h}_{\mathrm{P}} + \boldsymbol{u}_{\mathrm{P}}), \tag{4.37}$$

$$\boldsymbol{w}(\boldsymbol{u}) := \operatorname*{argmin}_{\boldsymbol{w}}\ R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}, \boldsymbol{u}), \tag{4.38}$$

where

$$\boldsymbol{H}_{\mathrm{U}} := \int \boldsymbol{\phi}(\boldsymbol{x})\boldsymbol{\phi}(\boldsymbol{x})^\top p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}, \tag{4.39}$$

$$\boldsymbol{h}_{\mathrm{P}} := \int \boldsymbol{\phi}(\boldsymbol{x})p(\boldsymbol{x} \mid y = +1)\mathrm{d}\boldsymbol{x}. \tag{4.40}$$

It can be easily confirmed $R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}) = R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}, \boldsymbol{0})$. Then, the following Lemma holds:

Lemma 11. *$R_{\mathrm{PU}}^{\mathrm{SMI}}(\cdot, \boldsymbol{u}) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\cdot)$ is Lipschitz continuous modulus $\omega(\boldsymbol{u}) = \mathcal{O}(\|\boldsymbol{U}_{\mathrm{U}}\|_{\mathrm{Fro}} + \|\boldsymbol{u}_{\mathrm{P}}\|_2)$ on a sufficiently small neighborhood of $\boldsymbol{w}^*$, where $\|\cdot\|_{\mathrm{Fro}}$ is the Frobenius norm.*

Proof. Firstly, we have

$$R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}, \boldsymbol{u}) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{U}_{\mathrm{U}} \boldsymbol{w} - \boldsymbol{w}^\top \boldsymbol{u}_{\mathrm{P}}. \tag{4.41}$$

The partial gradient is given by

$$\frac{\partial}{\partial \boldsymbol{w}}(R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}, \boldsymbol{u}) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w})) = \boldsymbol{U}_{\mathrm{U}} \boldsymbol{w} - \boldsymbol{u}_{\mathrm{P}}. \tag{4.42}$$

Let us define the δ-ball of $\boldsymbol{w}^*$ as $B_\delta(\boldsymbol{w}^*) := \{\boldsymbol{w} \mid \|\boldsymbol{w} - \boldsymbol{w}^*\|_2 \leq \delta\}$. For any $\boldsymbol{w} \in B_\delta(\boldsymbol{w}^*)$, we can immediately show

$$\|\boldsymbol{w}\|_2 \leq \|\boldsymbol{w} - \boldsymbol{w}\|_2 + \|\boldsymbol{w}^*\|_2 \leq \delta + M. \tag{4.43}$$

Therefore, we have

$$\left\| \frac{\partial}{\partial \boldsymbol{w}}(R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}, \boldsymbol{u}) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w})) \right\|_2 \leq (\delta + M)\|\boldsymbol{U}_{\mathrm{U}}\|_{\mathrm{Fro}} + \|\boldsymbol{u}_{\mathrm{P}}\|_2. \tag{4.44}$$

This means that $R_{\mathrm{PU}}^{\mathrm{SMI}}(\cdot, \boldsymbol{u}) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\cdot)$ is Lipschitz continuous on $B_\delta(\boldsymbol{w}^*)$ with a Lipschitz constant of order $\mathcal{O}(\|\boldsymbol{U}_{\mathrm{U}}\|_{\mathrm{Fro}} + \|\boldsymbol{u}_{\mathrm{P}}\|_2)$. □

Finally, we prove Theorem 9.

Proof. According to the *central limit theorem*, and our assumption about λ_{PU}, we have

$$\|\boldsymbol{U}_{\mathrm{U}}\|_{\mathrm{Fro}} = \mathcal{O}_p(1/\sqrt{n_{\mathrm{U}}}), \quad \|\boldsymbol{u}_{\mathrm{P}}\|_2 = \mathcal{O}_p(1/\sqrt{n_{\mathrm{P}}})$$

as $n_{\mathrm{P}}, n_{\mathrm{U}} \to \infty$. Thus, by using Lemma 10, Lemma 11, and Proposition 6.1 in Bonnans and Shapiro (1998), we have the first half of Theorem 9:

$$\begin{aligned} \|\widehat{\boldsymbol{w}} - \boldsymbol{w}^*\|_2 &\leq \epsilon^{-1}\omega(\boldsymbol{u}) \\ &= \mathcal{O}(\|\boldsymbol{U}_{\mathrm{U}}\|_{\mathrm{Fro}} + \|\boldsymbol{u}_{\mathrm{P}}\|_2) \\ &= \mathcal{O}_p(1/\sqrt{n_{\mathrm{P}}} + 1/\sqrt{n_{\mathrm{U}}}). \end{aligned} \tag{4.45}$$

Next, we prove the latter half of Theorem 9. For the squared errors, we have

$$|\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\widehat{\boldsymbol{w}}) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*)| \leq |\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\widehat{\boldsymbol{w}}) - \widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\boldsymbol{w}^*) + |\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\boldsymbol{w}^*) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*)|. \tag{4.46}$$

Here, we have

$$\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\widehat{\boldsymbol{w}}) - \widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\boldsymbol{w}^*) = \frac{1}{2}(\widehat{\boldsymbol{w}} + \boldsymbol{w}^*)^\top \widehat{\boldsymbol{H}}_{\mathrm{U}}(\widehat{\boldsymbol{w}} - \boldsymbol{w}^*) - (\widehat{\boldsymbol{w}} - \boldsymbol{w}^*)^\top \widehat{\boldsymbol{h}}_{\mathrm{P}}, \tag{4.47}$$

$$\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\boldsymbol{w}^*) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*) = \frac{1}{2}\boldsymbol{w}^{*\top}\boldsymbol{U}_{\mathrm{U}}\boldsymbol{w}^* - \boldsymbol{u}_{\mathrm{P}}\boldsymbol{w}^* + \frac{\lambda_{\mathrm{PU}}}{2}\boldsymbol{w}^{*\top}\boldsymbol{w}^*. \tag{4.48}$$

Since $0 \leq \phi_\ell(\boldsymbol{x}) \leq 1$, $\|\boldsymbol{w}^*\|_2 \leq M$, and $\lambda_{\mathrm{PU}} = \mathcal{O}_p(1/\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})})$, it leads to

$$\begin{aligned}
|\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\widehat{\boldsymbol{w}}) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*)| &\leq |\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\widehat{\boldsymbol{w}}) - \widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\boldsymbol{w}^*)| + |\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\boldsymbol{w}^*) - R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*)| \\
&\leq \mathcal{O}_p(\|\widehat{\boldsymbol{w}} - \boldsymbol{w}^*\|_2) + \mathcal{O}_p(\|\boldsymbol{U}_{\mathrm{U}}\|_{\mathrm{Fro}} + \|\boldsymbol{u}_{\mathrm{P}}\|_2 + \lambda_{\mathrm{PU}}) \\
&= \mathcal{O}_p(1/\sqrt{n_{\mathrm{P}}} + 1/\sqrt{n_{\mathrm{U}}}).
\end{aligned} \tag{4.49}$$

As discussed in Section 4.3.2, we have

$$\text{PU-SMI} = -\frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\Big(R_{\mathrm{PU}}^{\mathrm{SMI}}(\boldsymbol{w}^*) + \frac{1}{2}\Big), \tag{4.50}$$

$$\widehat{\text{PU-SMI}} = -\frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\Big(\widehat{R}_{\mathrm{PU}}^{\mathrm{SMI},\lambda}(\widehat{\boldsymbol{w}}) + \frac{1}{2}\Big). \tag{4.51}$$

The final result indicates

$$|\text{PU-SMI} - \widehat{\text{PU-SMI}}| = \mathcal{O}_p(1/\sqrt{n_{\mathrm{P}}} + 1/\sqrt{n_{\mathrm{U}}}). \tag{4.52}$$

□

Chapter 5

Learning from Positive, Negative, and Unlabeled Data

In this chapter, we discuss a classification problem in a semi-supervised learning setup.

5.1 Introduction

In real-world machine learning applications, collecting a large amount of labeled data is a critical bottleneck owing to laborious manual annotation. On the other hand, unlabeled data can often be collected automatically and abundantly, e.g., by a web crawler. This fact has led to the development of various semi-supervised learning algorithms over the past decades.

To leverage unlabeled data in training, most of the existing semi-supervised learning methods rely on particular assumptions on the data distribution (Chapelle et al., 2006). For example, the *cluster assumption* requires that samples in the same cluster tend to have the same label. However, such strong distributional assumptions are rarely satisfied in practice. Once the data do not meet the required distributional assumption, the performance of the existing methods degrades, and sometimes it is even worse than that of supervised learning methods (Cozman et al., 2003; Sokolovska et al., 2008).

In this chapter, we introduce a semi-supervised learning approach based on PU learning. Unlike most of the existing methods, our semi-supervised learning approach does not require the strong distributional assumptions. Furthermore, this approach can be applied any learning scenario as long as a risk function in PU learning is available. As the concrete implementation, we present two methods based on the misclassification rate and AUC. Additionally, without the distributional assumptions, we derive the generalization error bounds and show that the variance of the empirical risk estimator becomes smaller than its supervised counterpart. Experimental results report the usefulness of the methods.

5.2 Problem Setting

Let random variables $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{+1, -1\}$ be equipped with probability density $p(\boldsymbol{x}, y)$, where d is a positive integer. Let us consider a binary classification problem from $\boldsymbol{x}$ to y, given three sets of samples called the *positive* (P), *negative* (N), and *unlabeled* (U) data:

$$\{\boldsymbol{x}_i^{\mathrm{P}}\}_{i=1}^{n_{\mathrm{P}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x} \mid y = +1), \tag{5.1}$$

$$\{\boldsymbol{x}_i^{\mathrm{N}}\}_{i=1}^{n_{\mathrm{N}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x} \mid y = -1), \tag{5.2}$$

$$\{\boldsymbol{x}_i^{\mathrm{U}}\}_{i=1}^{n_{\mathrm{U}}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x}) = \theta_{\mathrm{P}} p(\boldsymbol{x} \mid y = +1) + \theta_{\mathrm{N}} p(\boldsymbol{x} \mid y = -1), \tag{5.3}$$

where

$$\theta_{\mathrm{P}} := p(y = +1) \ \text{ and } \ \theta_{\mathrm{N}} := p(y = -1)$$

are the class-prior probabilities for the positive and negative classes such that $\theta_{\mathrm{P}} + \theta_{\mathrm{N}} = 1$. Let E_{P}, E_{N}, and E_{U} be the expectations over $p(\boldsymbol{x} \mid y = +1)$, $p(\boldsymbol{x} \mid y = -1)$, and $p(\boldsymbol{x})$, respectively.

Furthermore, let $g \colon \mathbb{R}^d \to \mathbb{R}$ be an arbitrary real-valued decision function for binary classification, and classification is performed based on its sign. Let $\ell \colon \mathbb{R} \to \mathbb{R}$ be a loss function. The goal is to obtain an accurate classifier from P, N, and U samples.

Note that positive and negative (labeled) data are assumed to be drawn from corresponding conditional densities compared with standard supervised and semi-supervised learning settings in which labeled data are assumed to be drawn from the joint density $p(\boldsymbol{x}, y)$. In fact, our setting would be more practical than the existing one. Imagine that you will annotate samples collected from your domain of interest. Then, you may annotate some portion of the obtained data, e.g., 100 positive and 100 negative samples; then, the class-ratio of labeled data would be different from that of the underlying joint distribution. In this sense, our problem setting would be more practical.

5.3 Semi-Supervised Learning based on PU Learning

In this section, we introduce a semi-supervised learning approach based on PU learning.

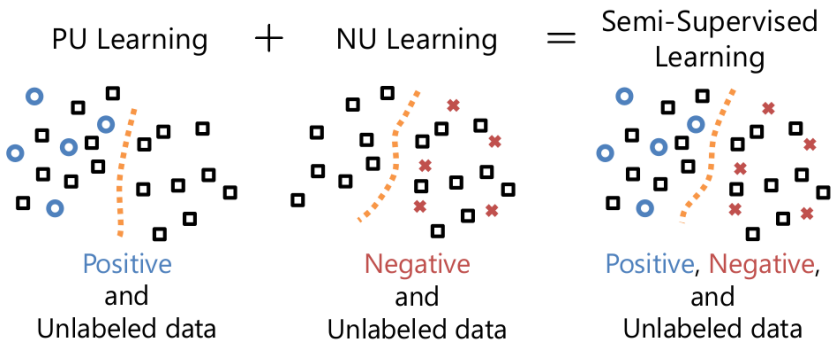


FIGURE 5.1: Illustration of PUNU Learning

5.3.1 PUNU Learning

The first idea to utilize positive, negative, and unlabeled data is to combine the PU and NU risks (see Figure 5.1). Let R_{PU} and R_{NU} be the risks in PU and NU learning, respectively.[1] For $\gamma \in [0, 1]$, let us consider a linear combination of the PU and NU risks (the PUNU risk):

$$R^{\gamma}_{\mathrm{PUNU}}(g) := (1-\gamma)R_{\mathrm{PU}}(g) + \gamma R_{\mathrm{NU}}(g). \tag{5.4}$$

We refer to this combined approach as *PUNU learning*.

Since this approach is independent of a classification measure, we can utilize any measures as long as the risk in PU (NU) learning is equivalent to that in PN learning. Below, we show two realizations of PUNU learning.

PUNU Classification: The PUNU risk in misclassification rate minimization (the PUNU-MR risk) is expressed as

$$R^{\mathrm{MR},\gamma}_{\mathrm{PUNU}}(g) := (1-\gamma)R^{\mathrm{MR}}_{\mathrm{PU}}(g) + \gamma R^{\mathrm{MR}}_{\mathrm{NU}}(g), \tag{5.5}$$

where $R^{\mathrm{MR}}_{\mathrm{PU}}$ and $R^{\mathrm{MR}}_{\mathrm{NU}}$ are the PU and NU risks in misclassification rate minimization. We refer to the method minimizing the PUNU-MR risk as *PUNU classification*.

[1] Depending on tasks, R_{PU} (R_{NU}) takes a specific form. For example, R_{PU} is replaced with $R^{\mathrm{MR}}_{\mathrm{PU}}$ in misclassification rate minimization, $R^{\mathrm{AUC}}_{\mathrm{PU}}$ in AUC optimization, $R^{\mathrm{SMI}}_{\mathrm{PU}}$ in an SMI estimation, respectively.

PUNU-AUC Optimization: The PUNU risk in AUC maximization (the PUNU-AUC risk) is expressed as

$$R_{\mathrm{PUNU}}^{\mathrm{AUC},\gamma}(g) := (1-\gamma)R_{\mathrm{PU}}^{\mathrm{AUC}}(g) + \gamma R_{\mathrm{NU}}^{\mathrm{AUC}}(g), \tag{5.6}$$

where $R_{\mathrm{PU}}^{\mathrm{AUC}}$ and $R_{\mathrm{NU}}^{\mathrm{AUC}}$ are the PU and NU risks in AUC maximization. We refer to the method minimizing the PUNU-AUC risk as *PUNU-AUC optimization*.

5.3.2 PNU Learning

Another possibility of using positive, negative, and unlabeled data is to combine the PN and PU/NU risks (see Figure 5.2). Let R_{PN} be the risk in PN learning.[2] For $\gamma \in [0,1]$, let us consider linear combinations of the PN and PU/NU risks (the PNPU and PNNU risks):

$$R_{\mathrm{PNPU}}^{\gamma}(g) := (1-\gamma)R_{\mathrm{PN}}(g) + \gamma R_{\mathrm{PU}}(g), \tag{5.7}$$

$$R_{\mathrm{PNNU}}^{\gamma}(g) := (1-\gamma)R_{\mathrm{PN}}(g) + \gamma R_{\mathrm{NU}}(g). \tag{5.8}$$

We refer to the method minimizing the PNPU (resp., PNNU) risks as *PNPU (resp., PNNU) learning*. In practice, we combine the PNPU and PNNU risks and adaptively choose one of them with a new trade-off parameter $\eta \in [-1,1]$ as

$$R_{\mathrm{PNU}}^{\eta}(g) := \begin{cases} R_{\mathrm{PNPU}}^{\eta}(g) & (\eta \geq 0), \\ R_{\mathrm{PNNU}}^{-\eta}(g) & (\eta < 0). \end{cases} \tag{5.9}$$

We refer to this combined approach as *PNU learning*. Clearly, PNU learning with $\eta = -1, 0, +1$ corresponds to NU, PN, and PU learning. As η gets large (resp., small), the effect of the positive (resp., negative) class is more emphasized.

Similarly to PUNU learning, we show two realizations of PNU learning.

PNU Classification: The PNU risk in misclassification rate minimization (the PNU-MR risk) is expressed as

$$R_{\mathrm{PNU}}^{\mathrm{MR},\eta}(g) := \begin{cases} R_{\mathrm{PNPU}}^{\mathrm{MR},\eta}(g) & (\eta \geq 0), \\ R_{\mathrm{PNNU}}^{\mathrm{MR},-\eta}(g) & (\eta < 0), \end{cases} \tag{5.10}$$

[2] Depending on tasks, R_{PN} takes a specific form. For example, R_{PN} is replaced with $R_{\mathrm{PU}}^{\mathrm{MR}}$ in misclassification rate minimization, $R_{\mathrm{PN}}^{\mathrm{AUC}}$ in AUC optimization, $R_{\mathrm{PN}}^{\mathrm{SMI}}$ in an SMI estimation, respectively.

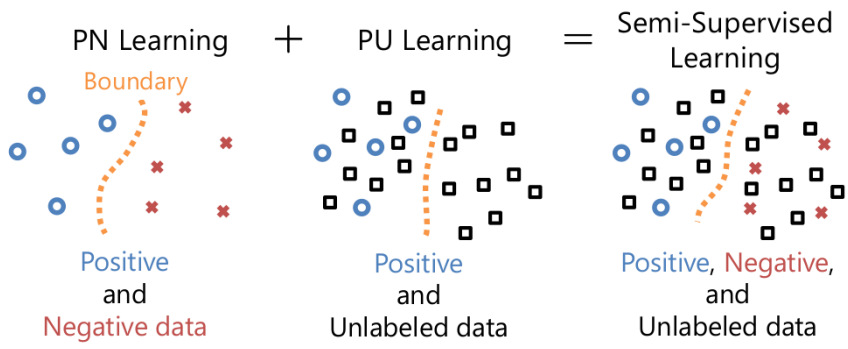


FIGURE 5.2: Illustration of PNU Learning

where

$$R_{\mathrm{PNPU}}^{\mathrm{MR},\gamma}(g) := (1-\gamma)R_{\mathrm{PN}}^{\mathrm{MR}}(g) + \gamma R_{\mathrm{PU}}^{\mathrm{MR}}(g), \tag{5.11}$$

$$R_{\mathrm{PNNU}}^{\mathrm{MR},\gamma}(g) := (1-\gamma)R_{\mathrm{PN}}^{\mathrm{MR}}(g) + \gamma R_{\mathrm{NU}}^{\mathrm{MR}}(g). \tag{5.12}$$

We refer to the method minimizing the PNU-MR risk as *PNU classification*.

PNU-AUC Optimization: The PNU risk in AUC maximization (the PNU-AUC risk) is expressed as

$$R_{\mathrm{PNU}}^{\mathrm{AUC},\eta}(g) := \begin{cases} R_{\mathrm{PNPU}}^{\mathrm{AUC},\eta}(g) & (\eta \geq 0), \\ R_{\mathrm{PNNU}}^{\mathrm{AUC},-\eta}(g) & (\eta < 0), \end{cases} \tag{5.13}$$

where

$$R_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g) := (1-\gamma)R_{\mathrm{PN}}^{\mathrm{AUC}}(g) + \gamma R_{\mathrm{PU}}^{\mathrm{AUC}}(g), \tag{5.14}$$

$$R_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}(g) := (1-\gamma)R_{\mathrm{PN}}^{\mathrm{AUC}}(g) + \gamma R_{\mathrm{NU}}^{\mathrm{AUC}}(g). \tag{5.15}$$

We refer to the method minimizing the PNU-AUC risk as *PNU-AUC optimization*.

In summary, we have the following lemma.

Lemma 12. *The following semi-supervised learning risks are equivalent:*

$$R^{\mathrm{MR}}(g) = R_{\mathrm{PUNU}}^{\mathrm{MR},\gamma}(g) = R_{\mathrm{PNPU}}^{\mathrm{MR},\gamma}(g) = R_{\mathrm{PNNU}}^{\mathrm{MR},\gamma}(g) = R_{\mathrm{PNU}}^{\mathrm{MR},\eta}(g), \tag{5.16}$$

$$R^{\mathrm{AUC}}(g) = R_{\mathrm{PUNU}}^{\mathrm{AUC},\gamma}(g) = R_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g) = R_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}(g) = R_{\mathrm{PNU}}^{\mathrm{AUC},\eta}(g). \tag{5.17}$$

It might seem that there is no reason for using different risk functions. However, the performance among risk estimators is different. In practice, the use of one risk function is better than others depending on, e.g., the size of samples in practice, similarly to PN, PU, and NU learning cases discussed in Section 2.5.2. In the next section, we discuss which approach, PUNU or PNU learning, is promising.

5.3.3 PUNU vs. PNU Learning

Here, we discuss which approach, PUNU or PNU learning, is more promising. Specifically, we focus on the misclassification rate as the classification measure and compare PUNU and PNU classification based on state-of-the-art theoretical comparisons (Niu et al., 2016), which are gave on estimation error bounds.

Let $\widehat{g}_{\mathrm{PN}}$, $\widehat{g}_{\mathrm{PU}}$, and $\widehat{g}_{\mathrm{NU}}$ be the minimizers of $\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(g)$, $\widehat{R}^{\mathrm{MR}}_{\mathrm{PU}}(g)$, and $\widehat{R}^{\mathrm{MR}}_{\mathrm{NU}}(g)$, respectively. Let

$$\rho_{\mathrm{PU,PN}} := \frac{\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}} + 1/\sqrt{n_{\mathrm{U}}}}{\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}}}, \tag{5.18}$$

$$\rho_{\mathrm{NU,PN}} := \frac{\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}} + 1/\sqrt{n_{\mathrm{U}}}}{\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}}}. \tag{5.19}$$

The finite-sample comparisons (Niu et al., 2016) state that if $\rho_{\mathrm{PU,PN}} > 1$ (resp., $\rho_{\mathrm{NU,PN}} > 1$), PN classification is more promising than PU (resp., NU) classification in the sense that $R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PN}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PU}})$ (resp., $R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PN}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{NU}})$); otherwise PU (resp., NU) classification is more promising than PN classification.

Suppose that n_{U} is not sufficiently large against n_{P} and n_{N}. According to the finite-sample comparisons, PN classification is most promising, and either PU or NU classification is the second best, i.e.,

$$R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PN}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PU}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{NU}}) \tag{5.20}$$

or

$$R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PN}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{NU}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PU}}). \tag{5.21}$$

On the other hand, if n_{U} is sufficiently large ($n_{\mathrm{U}} \to \infty$, which is faster than $n_{\mathrm{P}}, n_{\mathrm{N}} \to \infty$), we have the asymptotic comparisons:

$$\rho^*_{\mathrm{PU,PN}} = \lim_{n_{\mathrm{P}},n_{\mathrm{N}},n_{\mathrm{U}}\to\infty} \rho_{\mathrm{PU,PN}} = \frac{\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}}}{\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}}}, \tag{5.22}$$

$$\rho^*_{\mathrm{NU,PN}} = \lim_{n_{\mathrm{P}},n_{\mathrm{N}},n_{\mathrm{U}}\to\infty} \rho_{\mathrm{NU,PN}} = \frac{\theta_{\mathrm{N}}/\sqrt{n_{\mathrm{N}}}}{\theta_{\mathrm{P}}/\sqrt{n_{\mathrm{P}}}}. \tag{5.23}$$

From the above results, we have

$$\rho^*_{\mathrm{PU,PN}} \cdot \rho^*_{\mathrm{NU,PN}} = 1. \tag{5.24}$$

If $\rho^*_{\mathrm{PU,PN}} < 1$, then $\rho^*_{\mathrm{NU,PN}} > 1$, implying that PU (resp., PN) classification is more promising than PN (resp., NU) classification, i.e.,

$$R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PU}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PN}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{NU}}). \tag{5.25}$$

Similarly, when $\rho^*_{\mathrm{PU,PN}} > 1$ and $\rho^*_{\mathrm{NU,PN}} < 1$,

$$R^{\mathrm{MR}}(\widehat{g}_{\mathrm{NU}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PN}}) < R^{\mathrm{MR}}(\widehat{g}_{\mathrm{PU}}). \tag{5.26}$$

In real-world applications, since we do not know whether the number of unlabeled samples is sufficiently large or not, a practical approach would be to combine the best methods in both the finite-sample and asymptotic cases (See Eqs. (5.20), (5.21), (5.25), and (5.26)). From this viewpoint, PNU classification is the combination of the best methods in both cases, but PUNU classification is not. Additionally, PUNU classification includes the worst one in its combination in both cases.

In summary, PNU classification would be more promising than PUNU classification, as demonstrated in the experiments in Section 5.6.2.

5.4 Practical Implementation

In this section, practical implementation of the semi-supervised learning methods is explained.

5.4.1 General Case

We have so far only considered the true risk R (with respect to the expectations over true data distributions). When a classifier is trained with samples in practice, we use the empirical risk $\widehat{R}$, where the expectations are replaced with corresponding sample averages.

More specifically, we use a linear-in-parameter model given by

$$g(\boldsymbol{x}) = \sum_{j=1}^{b} w_j \phi_j(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}),$$

where $^\top$ denotes the transpose, b is the number of basis functions, $\boldsymbol{w} = (w_1, \ldots, w_b)^\top$ is a parameter vector, and $\boldsymbol{\phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \ldots, \phi_b(\boldsymbol{x}))^\top$ is a basis function vector. The parameter vector $\boldsymbol{w}$ is learned in order to minimize the ℓ_2-regularized empirical risk:

$$\min_{\boldsymbol{w}} \left[\widehat{R}(g) + \lambda \boldsymbol{w}^\top \boldsymbol{w} \right], \tag{5.27}$$

where $\lambda \geq 0$ is the regularization parameter.

5.4.2 PNU Classification

As given by Eq. (5.10), the PNU classification method consists of the PNPU-MR risk and the PNNU-MR risk. For the squared loss $\ell_S(m) := (1-m)^2$, the empirical PNPU-MR risk can be expressed as

$$\begin{aligned}
\widehat{R}_{\mathrm{PNPU}}^{\mathrm{MR},\gamma}(g) &= \frac{(1-\gamma)\theta_{\mathrm{P}}}{n_{\mathrm{P}}} \sum_{i=1}^{n_{\mathrm{P}}} \ell_{\mathrm{S}}(\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}_i^{\mathrm{P}})) + \frac{(1-\gamma)\theta_{\mathrm{N}}}{n_{\mathrm{N}}} \sum_{j=1}^{n_{\mathrm{N}}} \ell_{\mathrm{S}}(-\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}_j^{\mathrm{N}})) \\
&\quad + \frac{\gamma\theta_{\mathrm{P}}}{n_{\mathrm{P}}} \sum_{i=1}^{n_{\mathrm{P}}} \widetilde{\ell}_{\mathrm{S}}(\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}_i^{\mathrm{P}})) + \frac{\gamma}{n_{\mathrm{U}}} \ell_{\mathrm{S}}(-\boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}_k^{\mathrm{U}})) \\
&= (1-\gamma)\boldsymbol{w}^\top \widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{MR}} \boldsymbol{w} - 2(1-\gamma)\boldsymbol{w}^\top \widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{MR}} \\
&\quad + \gamma \boldsymbol{w}^\top \widehat{\boldsymbol{H}}_{\mathrm{U}}^{\mathrm{MR}} \boldsymbol{w} - 2\gamma \boldsymbol{w}^\top \widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{MR}} + 1,
\end{aligned} \tag{5.28}$$

where

$$\widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{MR}} := \frac{\theta_{\mathrm{P}}}{n_{\mathrm{P}}}\boldsymbol{\Phi}_{\mathrm{P}}^{\top}\mathbf{1}_{n_{\mathrm{P}}} - \frac{\theta_{\mathrm{N}}}{n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{N}}^{\top}\mathbf{1}_{n_{\mathrm{N}}}, \tag{5.29}$$

$$\widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{MR}} := \frac{\theta_{\mathrm{P}}}{n_{\mathrm{P}}}\boldsymbol{\Phi}_{\mathrm{P}}^{\top}\boldsymbol{\Phi}_{\mathrm{P}} + \frac{\theta_{\mathrm{N}}}{n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{N}}^{\top}\boldsymbol{\Phi}_{\mathrm{N}}, \tag{5.30}$$

$$\widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{MR}} := \frac{2\theta_{\mathrm{P}}}{n_{\mathrm{P}}}\boldsymbol{\Phi}_{\mathrm{P}}^{\top}\mathbf{1}_{n_{\mathrm{P}}} - \frac{1}{n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^{\top}\mathbf{1}_{n_{\mathrm{U}}}, \tag{5.31}$$

$$\widehat{\boldsymbol{H}}_{\mathrm{U}}^{\mathrm{MR}} := \frac{1}{n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^{\top}\boldsymbol{\Phi}_{\mathrm{U}}, \tag{5.32}$$

$$\boldsymbol{\Phi}_{\mathrm{P}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{P}}), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{P}}}^{\mathrm{P}}))^{\top}, \tag{5.33}$$

$$\boldsymbol{\Phi}_{\mathrm{N}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{N}}), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{N}}}^{\mathrm{N}}))^{\top}, \tag{5.34}$$

$$\boldsymbol{\Phi}_{\mathrm{U}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{U}}), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{U}}}^{\mathrm{U}}))^{\top}, \tag{5.35}$$

where $\mathbf{1}_b$ is a b-dimension vector whose elements are all one. The solution for the ℓ_2-regularized PNPU classification can be analytically obtained by

$$\widehat{\boldsymbol{w}}_{\mathrm{PNPU}}^{\mathrm{MR},\gamma} := \Big((1-\gamma)\widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{MR}} + \gamma\widehat{\boldsymbol{H}}_{\mathrm{U}}^{\mathrm{MR}} + \lambda\boldsymbol{I}_b\Big)^{-1}\Big((1-\gamma)\widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{MR}} + \gamma\widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{MR}}\Big), \tag{5.36}$$

where $\lambda \geq 0$ is the regularization parameter. Similarly, the solution for the ℓ_2-regularized PNNU classification can be obtained by

$$\widehat{\boldsymbol{w}}_{\mathrm{PNNU}}^{\mathrm{MR},\gamma} := \Big((1-\gamma)\widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{MR}} + \gamma\widehat{\boldsymbol{H}}_{\mathrm{U}}^{\mathrm{MR}} + \lambda\boldsymbol{I}_b\Big)^{-1}\Big((1-\gamma)\widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{MR}} + \gamma\widehat{\boldsymbol{h}}_{\mathrm{NU}}^{\mathrm{MR}}\Big), \tag{5.37}$$

where

$$\widehat{\boldsymbol{h}}_{\mathrm{NU}}^{\mathrm{MR}} := \frac{1}{n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^{\top}\mathbf{1}_{n_{\mathrm{U}}} - \frac{2\theta_{\mathrm{N}}}{n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{N}}^{\top}\mathbf{1}_{n_{\mathrm{N}}}. \tag{5.38}$$

5.4.3 PNU-AUC Optimization

As given by Eq. (5.13), the PNU-AUC optimization method consists of the PNPU-AUC and PNNU-AUC risks. For the squared loss $\ell_S(m) := (1-m)^2$, the empirical PNPU-AUC risk can be expressed as

$$\begin{aligned}
\widehat{R}_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g) &= \frac{1-\gamma}{n_{\mathrm{P}}n_{\mathrm{N}}}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{j=1}^{n_{\mathrm{N}}}\ell_{\mathrm{S}}(\boldsymbol{w}^\top\bar{\boldsymbol{\phi}}(\boldsymbol{x}_i^{\mathrm{P}},\boldsymbol{x}_j^{\mathrm{N}})) + \frac{\gamma}{\theta_{\mathrm{N}}n_{\mathrm{P}}n_{\mathrm{U}}}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell_{\mathrm{S}}(\boldsymbol{w}^\top\bar{\boldsymbol{\phi}}(\boldsymbol{x}_i^{\mathrm{P}},\boldsymbol{x}_k^{\mathrm{U}})) \\
&\quad - \frac{\gamma\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}n_{\mathrm{P}}(n_{\mathrm{P}}-1)}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'=1}^{n_{\mathrm{P}}}\ell_{\mathrm{S}}(\boldsymbol{w}^\top\bar{\boldsymbol{\phi}}(\boldsymbol{x}_i^{\mathrm{P}},\boldsymbol{x}_{i'}^{\mathrm{P}})) + \frac{\gamma\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}(n_{\mathrm{P}}-1)} \\
&= (1-\gamma) - 2(1-\gamma)\boldsymbol{w}^\top\widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{AUC}} + (1-\gamma)\boldsymbol{w}^\top\widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{AUC}}\boldsymbol{w} \\
&\quad + \gamma - 2\gamma\boldsymbol{w}^\top\widehat{\boldsymbol{h}}_{\mathrm{PU}}\mathrm{AUC} + \gamma\boldsymbol{w}^\top\widehat{\boldsymbol{H}}_{\mathrm{PU}}^{\mathrm{AUC}}\boldsymbol{w} - \gamma\boldsymbol{w}^\top\widehat{\boldsymbol{H}}_{\mathrm{PP}}^{\mathrm{AUC}}\boldsymbol{w},
\end{aligned} \tag{5.39}$$

where

$$\widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{AUC}} := \frac{1}{n_{\mathrm{P}}}\boldsymbol{\Phi}_{\mathrm{P}}^\top\boldsymbol{1}_{n_{\mathrm{P}}} - \frac{1}{n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{N}}^\top\boldsymbol{1}_{n_{\mathrm{N}}}, \tag{5.40}$$

$$\begin{aligned}
\widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{AUC}} &:= \frac{1}{n_{\mathrm{P}}}\boldsymbol{\Phi}_{\mathrm{P}}^\top\boldsymbol{\Phi}_{\mathrm{P}} - \frac{1}{n_{\mathrm{P}}n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{P}}^\top\boldsymbol{1}_{n_{\mathrm{P}}}\boldsymbol{1}_{n_{\mathrm{N}}}^\top\boldsymbol{\Phi}_{\mathrm{N}} \\
&\quad - \frac{1}{n_{\mathrm{P}}n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{N}}^\top\boldsymbol{1}_{n_{\mathrm{N}}}\boldsymbol{1}_{n_{\mathrm{P}}}^\top\boldsymbol{\Phi}_{\mathrm{P}} + \frac{1}{n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{N}}^\top\boldsymbol{\Phi}_{\mathrm{N}},
\end{aligned} \tag{5.41}$$

$$\widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{AUC}} := \frac{1}{\theta_{\mathrm{N}}n_{\mathrm{P}}}\boldsymbol{\Phi}_{\mathrm{P}}^\top\boldsymbol{1}_{n_{\mathrm{P}}} - \frac{1}{\theta_{\mathrm{N}}n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^\top\boldsymbol{1}_{n_{\mathrm{U}}}, \tag{5.42}$$

$$\begin{aligned}
\widehat{\boldsymbol{H}}_{\mathrm{PU}}^{\mathrm{AUC}} &:= \frac{1}{\theta_{\mathrm{N}}n_{\mathrm{P}}}\boldsymbol{\Phi}_{\mathrm{P}}^\top\boldsymbol{\Phi}_{\mathrm{P}} - \frac{1}{\theta_{\mathrm{N}}n_{\mathrm{P}}n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^\top\boldsymbol{1}_{n_{\mathrm{U}}}\boldsymbol{1}_{n_{\mathrm{P}}}^\top\boldsymbol{\Phi}_{\mathrm{P}} \\
&\quad - \frac{1}{\theta_{\mathrm{N}}n_{\mathrm{P}}n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{P}}^\top\boldsymbol{1}_{n_{\mathrm{P}}}\boldsymbol{1}_{n_{\mathrm{U}}}^\top\boldsymbol{\Phi}_{\mathrm{U}} + \frac{1}{\theta_{\mathrm{N}}n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^\top\boldsymbol{\Phi}_{\mathrm{U}},
\end{aligned} \tag{5.43}$$

$$\widehat{\boldsymbol{H}}_{\mathrm{PP}}^{\mathrm{AUC}} := \frac{2\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}(n_{\mathrm{P}}-1)}\boldsymbol{\Phi}_{\mathrm{P}}^\top\boldsymbol{\Phi}_{\mathrm{P}} - \frac{2\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}n_{\mathrm{P}}(n_{\mathrm{P}}-1)}\boldsymbol{\Phi}_{\mathrm{P}}^\top\boldsymbol{1}_{n_{\mathrm{P}}}\boldsymbol{1}_{n_{\mathrm{P}}}^\top\boldsymbol{\Phi}_{\mathrm{P}}, \tag{5.44}$$

$$\boldsymbol{\Phi}_{\mathrm{P}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{P}}),\ldots,\boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{P}}}^{\mathrm{P}}))^\top, \tag{5.45}$$

$$\boldsymbol{\Phi}_{\mathrm{U}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{U}}),\ldots,\boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{U}}}^{\mathrm{U}}))^\top, \tag{5.46}$$

$$\boldsymbol{\Phi}_{\mathrm{N}} := (\boldsymbol{\phi}(\boldsymbol{x}_1^{\mathrm{N}}),\ldots,\boldsymbol{\phi}(\boldsymbol{x}_{n_{\mathrm{N}}}^{\mathrm{N}}))^\top. \tag{5.47}$$

The solution for the ℓ_2-regularized PNPU-AUC optimization can be analytically obtained by

$$\widehat{\boldsymbol{w}}_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma} := \Big((1-\gamma)\widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{AUC}} + \gamma\widehat{\boldsymbol{H}}_{\mathrm{PU}}^{\mathrm{AUC}} - \gamma\widehat{\boldsymbol{H}}_{\mathrm{PP}}^{\mathrm{AUC}} + \lambda\boldsymbol{I}_b\Big)^{-1}\Big((1-\gamma)\widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{AUC}} + \gamma\widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{AUC}}\Big). \tag{5.48}$$

Similarly, the solution for the ℓ_2-regularized PNNU-AUC optimization can be obtained by

$$\widehat{\boldsymbol{w}}_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma} := \Big((1-\gamma)\widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{AUC}} + \gamma\widehat{\boldsymbol{H}}_{\mathrm{NU}}^{\mathrm{AUC}} - \gamma\widehat{\boldsymbol{H}}_{\mathrm{NN}}^{\mathrm{AUC}} + \lambda\boldsymbol{I}_b\Big)^{-1}\Big((1-\gamma)\widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{AUC}} + \gamma\widehat{\boldsymbol{h}}_{\mathrm{NU}}^{\mathrm{AUC}}\Big), \tag{5.49}$$

where

$$\widehat{\boldsymbol{h}}_{\mathrm{NU}}^{\mathrm{AUC}} := \frac{1}{\theta_{\mathrm{P}} n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^{\top}\boldsymbol{1}_{n_{\mathrm{U}}} - \frac{1}{\theta_{\mathrm{P}} n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{N}}^{\top}\boldsymbol{1}_{n_{\mathrm{N}}}, \tag{5.50}$$

$$\begin{aligned}\widehat{\boldsymbol{H}}_{\mathrm{NU}}^{\mathrm{AUC}} &:= \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}} n_{\mathrm{N}}}\boldsymbol{\Phi}_{\mathrm{N}}^{\top}\boldsymbol{\Phi}_{\mathrm{N}} - \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}} n_{\mathrm{N}} n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^{\top}\boldsymbol{1}_{n_{\mathrm{U}}}\boldsymbol{1}_{n_{\mathrm{N}}}^{\top}\boldsymbol{\Phi}_{\mathrm{N}} \\ &\quad - \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}} n_{\mathrm{N}} n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{N}}^{\top}\boldsymbol{1}_{n_{\mathrm{N}}}\boldsymbol{1}_{n_{\mathrm{U}}}^{\top}\boldsymbol{\Phi}_{\mathrm{U}} + \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}} n_{\mathrm{U}}}\boldsymbol{\Phi}_{\mathrm{U}}^{\top}\boldsymbol{\Phi}_{\mathrm{U}},\end{aligned} \tag{5.51}$$

$$\widehat{\boldsymbol{H}}_{\mathrm{NN}}^{\mathrm{AUC}} := \frac{2\theta_{\mathrm{N}}}{\theta_{\mathrm{P}}(n_{\mathrm{N}}-1)}\boldsymbol{\Phi}_{\mathrm{N}}^{\top}\boldsymbol{\Phi}_{\mathrm{N}} - \frac{2\theta_{\mathrm{N}}}{\theta_{\mathrm{P}} n_{\mathrm{N}}(n_{\mathrm{N}}-1)}\boldsymbol{\Phi}_{\mathrm{N}}^{\top}\boldsymbol{1}_{n_{\mathrm{N}}}\boldsymbol{1}_{n_{\mathrm{N}}}^{\top}\boldsymbol{\Phi}_{\mathrm{N}}. \tag{5.52}$$

The computational complexity of computing $\widehat{\boldsymbol{h}}_{\mathrm{PN}}^{\mathrm{AUC}}$ and $\widehat{\boldsymbol{H}}_{\mathrm{PN}}^{\mathrm{AUC}}$ are $\mathcal{O}((n_{\mathrm{P}}+n_{\mathrm{N}})b)$ and $\mathcal{O}((n_{\mathrm{P}}+n_{\mathrm{N}})b^2)$, respectively. Then, obtaining the solution $\widehat{\boldsymbol{w}}_{\mathrm{PNPU}}^{\mathrm{AUC}}$ ($\widehat{\boldsymbol{w}}_{\mathrm{PNNU}}^{\mathrm{AUC}}$) requires the computational complexity of $\mathcal{O}(b^3)$. Including the computational complexity of computing $\widehat{\boldsymbol{h}}_{\mathrm{PU}}^{\mathrm{AUC}}$, $\widehat{\boldsymbol{H}}_{\mathrm{PU}}^{\mathrm{AUC}}$, and $\widehat{\boldsymbol{H}}_{\mathrm{PP}}^{\mathrm{AUC}}$, the total computational complexity of the PNPU-AUC optimization method is $\mathcal{O}((n_{\mathrm{P}}+n_{\mathrm{N}}+n_{\mathrm{U}})b^2+b^3)$. Similarly, the total computational complexity of the PNNU-AUC optimization method is $\mathcal{O}((n_{\mathrm{P}}+n_{\mathrm{N}}+n_{\mathrm{U}})b^2+b^3)$. In total, the computational complexity of the PNU-AUC optimization method is $\mathcal{O}((n_{\mathrm{P}}+n_{\mathrm{N}}+n_{\mathrm{U}})b^2+b^3)$.

5.4.4 Model Selection

To tune the hyperparameters such as the regularization parameter λ, we use cross-validation. Unlike most of the semi-supervised learning methods, the semi-supervised learning approach based on PU learning can utilize unlabeled data for the risk estimation, meaning that unlabeled data can be utilized for computing a score of cross-validation.

PNU Classification: For the PNU classification method, we use the PNU risk in Eq. (5.10) with the zero-one loss as the score. To this end, however, we need to fix the combination parameter η in advance and then, we tune the hyperparameters including the combination parameter η. More specifically, let $\bar{\eta} \in [-1, 1]$ be the predefined combination parameter. We conduct cross-validation with respect to $R_{\mathrm{PNU}}^{\mathrm{MR},\bar{\eta}}(g)$ for tuning the hyperparameters. Since the PNU risk is equivalent to the PN risk for any $\bar{\eta}$, we can choose any $\bar{\eta}$ in principle. However, when the empirical PNU risk is used in practice, choice of $\bar{\eta}$ may affect the performance of cross-validation.

Here, based on theoretical results of variance reduction given in Section 5.5.2, we give a practical method to determine $\bar{\eta}$. The variance reduction implies that the empirical PNU risk is more reliable than the PN risk because the variance of the empirical PNU risk becomes smaller than the PN risk, given a combination parameter in a certain interval. This means that the empirical PNU risk approximates the expected risk more accurately than the risk computed by only labeled data; therefore, hyperparameters will be chosen more reliably by the empirical PNU risk.

Assuming the standard deviations (in Section 5.5.2) satisfying $\sigma_{\mathrm{P}}(g) = \sigma_{\mathrm{N}}(g)$, we can obtain simpler expressions of Eqs. (5.74) and (5.75) as

$$\bar{\gamma}_{\mathrm{PNPU}}^{\mathrm{MR}} := \frac{1 - \theta_{\mathrm{P}}^2 n_{\mathrm{N}}/(\theta_{\mathrm{N}}^2 n_{\mathrm{P}})}{1 + \theta_{\mathrm{P}}^2 n_{\mathrm{N}}/(\theta_{\mathrm{N}}^2 n_{\mathrm{P}})}, \tag{5.53}$$

$$\bar{\gamma}_{\mathrm{PNNU}}^{\mathrm{MR}} := \frac{1 - \theta_{\mathrm{N}}^2 n_{\mathrm{P}}/(\theta_{\mathrm{P}}^2 n_{\mathrm{N}})}{1 + \theta_{\mathrm{N}}^2 n_{\mathrm{P}}/(\theta_{\mathrm{P}}^2 n_{\mathrm{N}})}. \tag{5.54}$$

They can be computed simply from the number of samples and the (estimated) class-prior. Finally, to select the combination parameter η for training a classifier, we use $\widehat{R}_{\mathrm{PNPU}}^{\bar{\gamma}_{\mathrm{PNPU}}^{\mathrm{MR}}}$ for $\eta \geq 0$, and $\widehat{R}_{\mathrm{PNNU}}^{\bar{\gamma}_{\mathrm{PNNU}}^{\mathrm{MR}}}$ for $\eta < 0$.

PNU-AUC Optimization: Similarly to the misclasficaition rate case, to conduct cross-validation, we need to fix the combination parameter in advance. Assuming the covariances (in Section 5.5.2), e.g., $\tau_{\mathrm{PN,PP}}(g)$, are small enough to be neglected and $\tau_{\mathrm{PN}}(g) = \tau_{\mathrm{PP}}(g) = \tau_{\mathrm{NN}}(g)$, we can obtain simpler expressions of Eqs. (5.85) and (5.86) as

$$\bar{\gamma}_{\mathrm{PNPU}}^{\mathrm{AUC}} := \frac{1}{1 + \theta_{\mathrm{P}}^2 n_{\mathrm{N}}/(\theta_{\mathrm{N}}^2 n_{\mathrm{P}})}, \tag{5.55}$$

$$\bar{\gamma}_{\mathrm{PNNU}}^{\mathrm{AUC}} := \frac{1}{1 + \theta_{\mathrm{N}}^2 n_{\mathrm{P}}/(\theta_{\mathrm{P}}^2 n_{\mathrm{N}})}. \tag{5.56}$$

Similarly to the misclassification rate case, they can be computed simply from the number of samples and the (estimated) class-prior. Finally, to select the combination parameter η, we use $\widehat{R}_{\mathrm{PNPU}}^{\widetilde{\gamma}_{\mathrm{PNPU}}^{\mathrm{AUC}}}$ for $\eta \geq 0$, and $\widehat{R}_{\mathrm{PNNU}}^{\widetilde{\gamma}_{\mathrm{PNNU}}^{\mathrm{AUC}}}$ for $\eta < 0$.

5.5 Theoretical Analyses

In this section, we theoretically analyze the behavior of the empirical risks of our semi-supervised learning methods. We first derive generalization error bounds and then discuss variance reduction.

5.5.1 Generalization Error Bounds

Rigollet (2007) proved generalization error bounds for semi-supervised learning by mathematically modeling the cluster assumption. As a consequence, the generalization error bounds hold when the cluster assumption is satisfied; otherwise, the generalization error bounds cannot be guaranteed. In contrast, we give generalization error bounds to our risk estimators without such a strong distributional assumption.

Similarly to Section 3.5, we assume that a classifier g is the linear-in-parameter model and it belongs to a function class of bounded hyperplanes $\mathcal{G}$:

$$\mathcal{G} := \{g(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\phi}(\boldsymbol{x}) \mid \|\boldsymbol{w}\| \leq C_{\boldsymbol{w}};\ \forall \boldsymbol{x}\colon \|\boldsymbol{\phi}(\boldsymbol{x})\| \leq C_{\boldsymbol{\phi}}\}, \tag{5.57}$$

where $C_{\boldsymbol{w}} > 0$ and $C_{\boldsymbol{\phi}} > 0$ are certain positive constants.

Misclassification Rate Case: Denote by

$$I^{\mathrm{MR}}(g) = \mathrm{E}_{p(\boldsymbol{x},y)}[\ell_{\text{0-1}}(yg(\boldsymbol{x}))] \tag{5.58}$$

the risk of g for binary classification, i.e., the generalization error of g. In the following, we study upper bounds of $I^{\mathrm{MR}}(g)$ holding uniformly for all $g \in \mathcal{G}$. We respectively focus on the *(scaled) ramp and squared losses* for the non-convex and convex methods. Similar results can be obtained with a little more effort if other eligible losses are used.

First, we consider the case that the scaled ramp loss function is used. Let $R_{\text{N-PUNU}}^{\mathrm{MR},\gamma}$, $R_{\text{N-PNPU}}^{\mathrm{MR},\gamma}$, and $R_{\text{N-PNNU}}^{\mathrm{MR},\gamma}$ be the PUNU-MR, PNPU-MR, and PNNU-MR risks equipped with the scaled ramp loss. A key observation is that $\ell_{\text{0-1}}(m) \leq 2\ell_{\mathrm{R}}(m)$, and consequently

$I^{\mathrm{MR}}(g) \leq 2R^{\mathrm{MR}}(g)$. Note that by definition we have

$$R^{\mathrm{MR},\gamma}_{\text{N-PUNU}}(g) = R^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g) = R^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g) = R^{\mathrm{MR}}(g).$$

The theorem below can be proven using the Rademacher analysis (see, for example, Mohri et al., 2012; Ledoux and Talagrand, 1991). The proof is available in Section 5.8.1.

Theorem 13. *Let $\ell_{\mathrm{R}}(m)$ be the loss for defining the empirical risks. For any $\delta > 0$, the following inequalities hold separately with probability at least $1-\delta$ for all $g \in \mathcal{G}$:*

$$I^{\mathrm{MR}}(g) \leq 2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PUNU}}(g) + c_1^{\mathrm{MR}}(\delta) \cdot \Big(\frac{2(1-\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{2\gamma\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{|2\gamma-1|}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{5.59}$$

$$I^{\mathrm{MR}}(g) \leq 2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g) + c_1^{\mathrm{MR}}(\delta) \cdot \Big(\frac{(1+\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{(1-\gamma)\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{\gamma}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{5.60}$$

$$I^{\mathrm{MR}}(g) \leq 2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g) + c_1^{\mathrm{MR}}(\delta) \cdot \Big(\frac{(1-\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{(1+\gamma)\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{\gamma}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{5.61}$$

where $c_1^{\mathrm{MR}}(\delta) = 2C_w C_\phi + \sqrt{2\ln(3/\delta)}$.

Theorem 13 guarantees that when $\ell_{\mathrm{R}}(m)$ is used, $I^{\mathrm{MR}}(g)$ can be bounded from above by two times the empirical risks, i.e., $2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PUNU}}(g)$, $2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)$, and $2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)$, plus the corresponding confidence terms of order

$$\mathcal{O}_p\Big(\frac{1}{\sqrt{n_{\mathrm{P}}}} + \frac{1}{\sqrt{n_{\mathrm{N}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big).$$

Since n_{P}, n_{N}, and n_{U} can increase independently, this is already the optimal convergence rate without any additional assumption (Vapnik, 1998; Mendelson, 2008).

Next, we prove the generalization error bounds when the squared loss function is used. Let $R^{\mathrm{MR},\gamma}_{\text{C-PUNU}}$, $R^{\mathrm{MR},\gamma}_{\text{C-PNPU}}$, and $R^{\mathrm{MR},\gamma}_{\text{C-PNNU}}$ be the PUNU-MR, PNPU-MR, and PNNU-MR risks equipped with the squared loss. Analogously, we have $\ell_{\text{0-1}}(m) \leq 4\ell_{\mathrm{S}}(m)$ for the squared loss. However, it is too loose when $|m| \gg 0$. Fortunately, we do not have to use $\ell_{\mathrm{S}}(m)$ if we work on the generalization error rather than the estimation error. To this end, we define the *truncated (scaled) squared loss* $\ell_{\mathrm{TS}}(m)$ as

$$\ell_{\mathrm{TS}}(m) = \begin{cases} \ell_{\mathrm{S}}(m) & 0 < m \leq 1, \\ \ell_{\text{0-1}}(m)/4 & \text{otherwise}, \end{cases} \tag{5.62}$$

so that $\ell_{0\text{-}1}(m) \leq 4\ell_{\mathrm{TS}}(m)$ is much tighter. For $\ell_{\mathrm{TS}}(m)$, $R^{\mathrm{MR}}_{\text{C-PU}}(g)$ and $R^{\mathrm{MR}}_{\text{C-NU}}(g)$ need to be redefined as follows (see Section 2.5 for details):

$$R^{\mathrm{MR}}_{\text{C-PU}}(g) := \theta_{\mathrm{P}}\,\mathrm{E}_{\mathrm{P}}[\widetilde{\ell}_{\mathrm{TS}}(g(\boldsymbol{x}))] + \mathrm{E}_{\mathrm{U}}[\ell_{\mathrm{TS}}(-g(\boldsymbol{x}))], \tag{5.63}$$

$$R^{\mathrm{MR}}_{\text{C-NU}}(g) := \theta_{\mathrm{N}}\,\mathrm{E}_{\mathrm{N}}[\widetilde{\ell}_{\mathrm{TS}}(-g(\boldsymbol{x}))] + \mathrm{E}_{\mathrm{U}}[\ell_{\mathrm{TS}}(g(\boldsymbol{x}))], \tag{5.64}$$

where $\widetilde{\ell}_{\mathrm{TS}}(m) = \ell_{\mathrm{TS}}(m) - \ell_{\mathrm{TS}}(-m)$ is the composite loss of truncated scaled squared loss. The condition $\widetilde{\ell}_{\mathrm{TS}}(m) \neq -m$ means the loss of convexity, but the equivalence is not lost; indeed, we still have

$$R^{\mathrm{MR},\gamma}_{\text{C-PUNU}}(g) = R^{\mathrm{MR},\gamma}_{\text{C-PNPU}}(g) = R^{\mathrm{MR},\gamma}_{\text{C-PNNU}}(g) = R^{\mathrm{MR}}(g).$$

Then, we have the following theorem (its proof is available in Section 5.8.2):

Theorem 14. *Let $\ell_{\mathrm{TS}}(m)$ be the loss for defining the empirical risks (where $R^{\mathrm{MR}}_{\text{C-PU}}(g)$ and $R^{\mathrm{MR}}_{\text{C-NU}}(g)$ are redefined). For any $\delta > 0$, the following inequalities hold separately with probability at least $1-\delta$ for all $g \in \mathcal{G}$:*

$$I^{\mathrm{MR}}(g) \leq 4\widehat{R}^{\mathrm{MR},\gamma}_{\text{C-PUNU}}(g) + c_2^{\mathrm{MR}}(\delta)\cdot\Big(\frac{(1-\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{\gamma\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{5.65}$$

$$I^{\mathrm{MR}}(g) \leq 4\widehat{R}^{\mathrm{MR},\gamma}_{\text{C-PNPU}}(g) + c_2^{\mathrm{MR}}(\delta)\cdot\Big(\frac{\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{(1-\gamma)\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{\gamma}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{5.66}$$

$$I^{\mathrm{MR}}(g) \leq 4\widehat{R}^{\mathrm{MR},\gamma}_{\text{C-PNNU}}(g) + c_2^{\mathrm{MR}}(\delta)\cdot\Big(\frac{(1-\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{\gamma}{\sqrt{n_{\mathrm{U}}}}\Big), \tag{5.67}$$

where $c_2^{\mathrm{MR}}(\delta) = 4C_wC_\phi + \sqrt{2\ln(4/\delta)}$.

Theorem 14 ensures that when $\ell_{\mathrm{TS}}(m)$ is used (for evaluating the empirical risks rather than learning the empirical risk minimizers), $I^{\mathrm{MR}}(g)$ can be bounded from above by four times the empirical risks plus confidence terms in the optimal parametric rate. As $\ell_{\mathrm{TS}}(m) \leq \ell_{\mathrm{S}}(m)$, Theorem 14 is valid (but weaker) if all empirical risks are w.r.t. $\ell_{\mathrm{S}}(m)$.

AUC Case: We then prove the generalization error bounds of our risks for imbalanced binary classification. Similarly to Section 3.5, we assume that a surrogate loss is bounded from above by C_ℓ and satisfying $\ell_{0\text{-}1}(m) \leq \ell(m)$, and denote the Lipschitz constant by L.

Let

$$I^{\mathrm{AUC}}(g) = \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[\ell_{0\text{-}1}(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] \tag{5.68}$$

be the generalization error of g in AUC optimization. For convenience, we define

$$c^{\mathrm{AUC}}(\delta) := 2\sqrt{2}LC_\ell C_{\boldsymbol{w}} C_{\boldsymbol{\phi}} + \frac{3}{2}\sqrt{2\log(2/\delta)}. \tag{5.69}$$

In the following, we prove the generalization error bounds of the semi-supervised AUC optimization method.

For the PNPU-AUC and PNNU-AUC risks, we prove the following generalization error bounds (its proof is also available in Section 5.8.3):

Theorem 15. *For any $\delta > 0$, the following inequalities hold separately with probability at least $1-\delta$ for all $g \in \mathcal{G}$:*

$$\begin{aligned} I^{\mathrm{AUC}}(g) \leq\ & \widehat{R}^{\mathrm{AUC},\gamma}_{\mathrm{PNPU}}(g) + c^{\mathrm{AUC}}(\delta/3) \\ & \times \Big(\frac{1-\gamma}{\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{N}})}} + \frac{\gamma}{\theta_{\mathrm{N}}\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})}} + \frac{\gamma\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}\sqrt{n_{\mathrm{P}}}}\Big), \end{aligned} \tag{5.70}$$

$$\begin{aligned} I^{\mathrm{AUC}}(g) \leq\ & \widehat{R}^{\mathrm{AUC},\gamma}_{\mathrm{PNNU}}(g) + c^{\mathrm{AUC}}(\delta/3) \\ & \times \Big(\frac{1-\gamma}{\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{N}})}} + \frac{\gamma}{\theta_{\mathrm{P}}\sqrt{\min(n_{\mathrm{N}}, n_{\mathrm{U}})}} + \frac{\gamma\theta_{\mathrm{N}}}{\theta_{\mathrm{P}}\sqrt{n_{\mathrm{N}}}}\Big), \end{aligned} \tag{5.71}$$

where $\widehat{R}^{\mathrm{AUC},\gamma}_{\mathrm{PNPU}}$ and $\widehat{R}^{\mathrm{AUC},\gamma}_{\mathrm{PNNU}}$ are unbiased empirical risk estimators corresponding to $R^{\mathrm{AUC},\gamma}_{\mathrm{PNPU}}$ and $R^{\mathrm{AUC},\gamma}_{\mathrm{PNNU}}$, respectively.

Theorem 15 guarantees that $I^{\mathrm{AUC}}(g)$ can be bounded from above by the empirical risk, $\widehat{R}^{\mathrm{AUC}}(g)$, plus the confidence terms of order

$$\mathcal{O}_p\Big(\frac{1}{\sqrt{n_{\mathrm{P}}}} + \frac{1}{\sqrt{n_{\mathrm{N}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big).$$

Again, since n_{P}, n_{N}, and n_{U} can increase independently in our setting, this is the optimal convergence rate without any additional assumptions.

5.5.2 Variance Reduction

The risk estimators presented in Section 5.3 are all unbiased provided that the unbiased risk in PU learning is available with a specific classification measure. In this section, we explore the variance of the risk estimators. We show that unlabeled samples help reduce the variance of our risk estimators compared with its supervised counterpart $\widehat{R}_{\mathrm{PN}}(g)$. For the sake of simplicity, we assume that $n_{\mathrm{U}} \to \infty$, to illustrate the maximum variance reduction that could be achieved.

Misclassification Rate Case: For the sake of simplicity, we define

$$\sigma_{\mathrm{P}}^2(g) := \mathrm{Var}_{\mathrm{P}}[\ell(g(\boldsymbol{x}))], \quad \sigma_{\mathrm{N}}^2(g) := \mathrm{Var}_{\mathrm{N}}[\ell(-g(\boldsymbol{x}))],$$

where $\mathrm{Var}_{\mathrm{P}}$ and $\mathrm{Var}_{\mathrm{N}}$ denote the variance over $p(\boldsymbol{x} \mid y = +1)$ and $p(\boldsymbol{x} \mid y = -1)$. Moreover, denote by

$$\psi_{\mathrm{P}} := \theta_{\mathrm{P}}^2 \sigma_{\mathrm{P}}^2(g)/n_{\mathrm{P}}, \quad \psi_{\mathrm{N}} := \theta_{\mathrm{N}}^2 \sigma_{\mathrm{N}}^2(g)/n_{\mathrm{N}}$$

for short, and let Var be the variance over

$$\begin{aligned} &p(\boldsymbol{x}_1 \mid y = +1) \cdots p(\boldsymbol{x}_{n_{\mathrm{P}}} \mid y = +1) \\ &\times p(\boldsymbol{x}_1 \mid y = -1) \cdots p(\boldsymbol{x}_{n_{\mathrm{N}}} \mid y = -1) \\ &\times p(\boldsymbol{x}_1) \cdots p(\boldsymbol{x}_{n_{\mathrm{U}}}). \end{aligned}$$

We have the following theorems (its proofs are available in Section 5.8.4):

Theorem 16. *Assume $n_{\mathrm{U}} \to \infty$. For any fixed g, let*

$$\gamma_{\text{N-PUNU}}^{\mathrm{MR}} = \operatorname*{argmin}_{\gamma} \mathrm{Var}[\widehat{R}_{\text{N-PUNU}}^{\mathrm{MR},\gamma}(g)] = \frac{\psi_{\mathrm{P}}}{\psi_{\mathrm{P}} + \psi_{\mathrm{N}}}. \tag{5.72}$$

Then, we have $\gamma_{\text{N-PUNU}}^{\mathrm{MR}} \in [0, 1]$. Further,

$$\mathrm{Var}[\widehat{R}_{\text{N-PUNU}}^{\mathrm{MR},\gamma}(g)] < \mathrm{Var}[\widehat{R}_{\mathrm{PN}}^{\mathrm{MR}}(g)] \tag{5.73}$$

for all $\gamma \in (2\gamma_{\text{N-PUNU}}^{\mathrm{MR}} - 1/2, 1/2)$ if $\psi_{\mathrm{P}} < \psi_{\mathrm{N}}$, or for all $\gamma \in (1/2, 2\gamma_{\text{N-PUNU}}^{\mathrm{MR}} - 1/2)$ if $\psi_{\mathrm{P}} > \psi_{\mathrm{N}}$.[3]

Theorem 16 guarantees that the variance is always reduced by $\widehat{R}_{\text{N-PUNU}}^{\mathrm{MR},\gamma}(g)$ if γ is close to $\gamma_{\text{N-PUNU}}^{\mathrm{MR}}$, which is optimal for variance reduction. The interval of such good γ values has the length

$$\min\Big(\frac{|\psi_{\mathrm{P}} - \psi_{\mathrm{N}}|}{\psi_{\mathrm{P}} + \psi_{\mathrm{N}}}, \frac{1}{2}\Big).$$

In particular, if $3\psi_{\mathrm{P}} \leq \psi_{\mathrm{N}}$ or $\psi_{\mathrm{P}} \geq 3\psi_{\mathrm{N}}$, the length is $1/2$.

[3] Being fixed means g is determined before seeing the data for evaluating the empirical risk. For example, if g is trained by some learning method, and the empirical risk is subsequently evaluated on the independent validation/test data, g is regarded as fixed in the evaluation.

Theorem 17. *Assume $n_{\mathrm{U}} \to \infty$. For any fixed g, let*

$$\gamma^{\mathrm{MR}}_{\text{N-PNPU}} = \operatorname*{argmin}_{\gamma} \mathrm{Var}[\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)] = \frac{\psi_{\mathrm{N}} - \psi_{\mathrm{P}}}{\psi_{\mathrm{P}} + \psi_{\mathrm{N}}}, \tag{5.74}$$

$$\gamma^{\mathrm{MR}}_{\text{N-PNNU}} = \operatorname*{argmin}_{\gamma} \mathrm{Var}[\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)] = \frac{\psi_{\mathrm{P}} - \psi_{\mathrm{N}}}{\psi_{\mathrm{P}} + \psi_{\mathrm{N}}}. \tag{5.75}$$

Then, we have $\gamma^{\mathrm{MR}}_{\text{N-PNPU}} \in [0,1]$ if $\psi_{\mathrm{P}} \leq \psi_{\mathrm{N}}$ or $\gamma^{\mathrm{MR}}_{\text{N-PNNU}} \in [0,1]$ if $\psi_{\mathrm{P}} \geq \psi_{\mathrm{N}}$. Additionally,

$$\mathrm{Var}[\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)] < \mathrm{Var}[\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(g)] \tag{5.76}$$

for all $\gamma \in (0, 2\gamma^{\mathrm{MR}}_{\text{N-PNPU}})$ if $\psi_{\mathrm{P}} < \psi_{\mathrm{N}}$, or

$$\mathrm{Var}[\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)] < \mathrm{Var}[\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(g)] \tag{5.77}$$

for all $\gamma \in (0, 2\gamma^{\mathrm{MR}}_{\text{N-PNNU}})$ if $\psi_{\mathrm{P}} > \psi_{\mathrm{N}}$.

Theorem 17 implies that the variance of $\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(g)$ is reduced by either $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)$ if $\psi_{\mathrm{P}} \leq \psi_{\mathrm{N}}$ or $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)$ if $\psi_{\mathrm{P}} \geq \psi_{\mathrm{N}}$, where γ should be close to $\gamma^{\mathrm{MR}}_{\text{N-PNPU}}$ or $\gamma^{\mathrm{MR}}_{\text{N-PNNU}}$. The range of such good γ values is of length

$$\min\Big(\frac{2|\psi_{\mathrm{P}} - \psi_{\mathrm{N}}|}{\psi_{\mathrm{P}} + \psi_{\mathrm{N}}},\ 1\Big).$$

In particular, if $3\psi_{\mathrm{P}} \leq \psi_{\mathrm{N}}$, $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)$ given any $\gamma \in (0,1)$ can reduce the variance, and if $\psi_{\mathrm{P}} \geq 3\psi_{\mathrm{N}}$, $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)$ given any $\gamma \in (0,1)$ can reduce the variance.

As a corollary of Theorems 16 and 17, the minimum variance achievable by $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PUNU}}(g)$, $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)$, and $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)$ at their optimal $\gamma^{\mathrm{MR}}_{\text{N-PUNU}}$, $\gamma^{\mathrm{MR}}_{\text{N-PNPU}}$, and $\gamma^{\mathrm{MR}}_{\text{N-PNNU}}$ is exactly the same, namely, $4\psi_{\mathrm{P}}\psi_{\mathrm{N}}/(\psi_{\mathrm{P}} + \psi_{\mathrm{N}})$. Nevertheless, $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)$ and $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)$ have a much wider range of nice γ values than $\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PUNU}}(g)$.

If we further assume that $\sigma_{\mathrm{P}}(g) = \sigma_{\mathrm{N}}(g)$, the condition in Theorems 16 and 17 as to whether $\psi_{\mathrm{P}} \leq \psi_{\mathrm{N}}$ or $\psi_{\mathrm{P}} \geq \psi_{\mathrm{N}}$ will be independent of g. Also, it will coincide with the condition in *Theorem 7* in Niu et al. (2016) where the minimizers of $\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(g)$, $\widehat{R}^{\mathrm{MR}}_{\mathrm{PU}}(g)$ and $\widehat{R}^{\mathrm{MR}}_{\mathrm{NU}}(g)$ are compared.

A remark is that learning is uninvolved in Theorems 16 and 17, such that $\ell(m)$ can be any loss that satisfies $\ell(m) + \ell(-m) = 1$, and g can be any fixed decision function. For instance, we may adopt $\ell_{\text{0-1}}(m)$ and pick some g resulted from some other learning methods. As a consequence, the variance of $\widehat{I}^{\mathrm{MR}}_{\mathrm{PN}}(g)$ over the validation data can be reduced,

and then the cross-validation should be more stable, given that n_{U} is sufficiently large. Therefore, even without being minimized, our proposed risk estimators are themselves of practical importance.

AUC Case: Similarly to the misclassification rate case, we here investigate if the PNU-AUC risk estimator has smaller variance than its supervised counterpart.

Let us introduce the following variances and covariances:

$$\tau_{\mathrm{PN}}^2(g) = \mathrm{Var}_{\mathrm{PN}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}}))], \tag{5.78}$$

$$\tau_{\mathrm{PP}}^2(g) = \mathrm{Var}_{\mathrm{P\bar{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\bar{\boldsymbol{x}}^{\mathrm{P}}))], \tag{5.79}$$

$$\tau_{\mathrm{NN}}^2(g) = \mathrm{Var}_{\mathrm{N\bar{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\bar{\boldsymbol{x}}^{\mathrm{N}}))], \tag{5.80}$$

$$\tau_{\mathrm{PN,PP}}(g) = \mathrm{Cov}_{\mathrm{PN,P\bar{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}})), \ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\bar{\boldsymbol{x}}^{\mathrm{P}}))], \tag{5.81}$$

$$\tau_{\mathrm{PN,NN}}(g) = \mathrm{Cov}_{\mathrm{PN,N\bar{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}})), \ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\bar{\boldsymbol{x}}^{\mathrm{N}}))], \tag{5.82}$$

$$\tau_{\mathrm{PU,PP}}(g) = \mathrm{Cov}_{\mathrm{PU,P\bar{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}})), \ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\bar{\boldsymbol{x}}^{\mathrm{P}}))], \tag{5.83}$$

$$\tau_{\mathrm{NU,NN}}(g) = \mathrm{Cov}_{\mathrm{NU,N\bar{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}})), \ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\bar{\boldsymbol{x}}^{\mathrm{N}}))], \tag{5.84}$$

where $\mathrm{Var}_{\mathrm{PN}}$, $\mathrm{Var}_{\mathrm{P\bar{P}}}$, and $\mathrm{Var}_{\mathrm{N\bar{N}}}$ are the variances over

$$\begin{aligned}
&p(\boldsymbol{x}^{\mathrm{P}} \mid y = +1)p(\boldsymbol{x}^{\mathrm{N}} \mid y = -1),\\
&p(\boldsymbol{x}^{\mathrm{P}} \mid y = +1)p(\bar{\boldsymbol{x}}^{\mathrm{P}} \mid y = +1),\\
&p(\boldsymbol{x}^{\mathrm{N}} \mid y = -1)p(\bar{\boldsymbol{x}}^{\mathrm{N}} \mid y = -1),
\end{aligned}$$

respectively. $\mathrm{Cov}_{\mathrm{PN,P\bar{P}}}$, $\mathrm{Cov}_{\mathrm{PN,N\bar{N}}}$, $\mathrm{Cov}_{\mathrm{PU,P\bar{P}}}$, and $\mathrm{Cov}_{\mathrm{NU,N\bar{N}}}$ are the covariances over

$$\begin{aligned}
&p(\boldsymbol{x}^{\mathrm{P}} \mid y = +1)p(\boldsymbol{x}^{\mathrm{N}} \mid y = -1)p(\bar{\boldsymbol{x}}^{\mathrm{P}} \mid y = +1),\\
&p(\boldsymbol{x}^{\mathrm{P}} \mid y = +1)p(\boldsymbol{x}^{\mathrm{N}} \mid y = -1)p(\bar{\boldsymbol{x}}^{\mathrm{N}} \mid y = -1),\\
&p(\boldsymbol{x}^{\mathrm{P}} \mid y = +1)p(\boldsymbol{x}^{\mathrm{U}})p(\bar{\boldsymbol{x}}^{\mathrm{P}} \mid y = +1),\\
&p(\boldsymbol{x}^{\mathrm{N}} \mid y = -1)p(\boldsymbol{x}^{\mathrm{U}})p(\bar{\boldsymbol{x}}^{\mathrm{N}} \mid y = -1),
\end{aligned}$$

respectively.

Then, we have the following theorem (its proof is available in Section 5.8.5):

Theorem 18. *Assume $n_{\mathrm{U}} \to \infty$. For any fixed g, the minimizers of the variance of the empirical PNPU-AUC and PNNU-AUC risks are respectively obtained by*

$$\gamma_{\mathrm{PNPU}}^{\mathrm{AUC}} = \operatorname*{argmin}_{\gamma} \mathrm{Var}[\widehat{R}_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g)] = \frac{\overline{\psi}_{\mathrm{PN}} - \overline{\psi}_{\mathrm{PP}}/2}{\overline{\psi}_{\mathrm{PN}} + \overline{\psi}_{\mathrm{PU}} - \overline{\psi}_{\mathrm{PP}}}, \tag{5.85}$$

$$\gamma_{\mathrm{PNNU}}^{\mathrm{AUC}} = \operatorname*{argmin}_{\gamma} \mathrm{Var}[\widehat{R}_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}(g)] = \frac{\overline{\psi}_{\mathrm{PN}} - \overline{\psi}_{\mathrm{NN}}/2}{\overline{\psi}_{\mathrm{PN}} + \overline{\psi}_{\mathrm{NU}} - \overline{\psi}_{\mathrm{NN}}}, \tag{5.86}$$

where

$$\overline{\psi}_{\mathrm{PN}} = \frac{1}{n_{\mathrm{P}} n_{\mathrm{N}}} \tau_{\mathrm{PN}}^2(g), \tag{5.87}$$

$$\overline{\psi}_{\mathrm{PU}} = \frac{\theta_{\mathrm{P}}^2}{\theta_{\mathrm{N}}^2 {n_{\mathrm{P}}}^2} \tau_{\mathrm{PP}}^2(g) - \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}^2 n_{\mathrm{P}}} \tau_{\mathrm{PU,PP}}(g), \tag{5.88}$$

$$\overline{\psi}_{\mathrm{PP}} = \frac{1}{\theta_{\mathrm{N}} n_{\mathrm{P}}} \tau_{\mathrm{PN,PU}}(g) - \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}} n_{\mathrm{P}}} \tau_{\mathrm{PN,PP}}(g), \tag{5.89}$$

$$\overline{\psi}_{\mathrm{NU}} = \frac{\theta_{\mathrm{N}}^2}{\theta_{\mathrm{P}}^2 {n_{\mathrm{N}}}^2} \tau_{\mathrm{NN}}^2(g) - \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}}^2 n_{\mathrm{N}}} \tau_{\mathrm{NU,NN}}(g), \tag{5.90}$$

$$\overline{\psi}_{\mathrm{NN}} = \frac{1}{\theta_{\mathrm{P}} n_{\mathrm{N}}} \tau_{\mathrm{PN,NU}}(g) - \frac{\theta_{\mathrm{N}}}{\theta_{\mathrm{P}} n_{\mathrm{N}}} \tau_{\mathrm{PN,NN}}(g). \tag{5.91}$$

Additionally, we have

$$\mathrm{Var}[\widehat{R}_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g)] < \mathrm{Var}[\widehat{R}_{\mathrm{PN}}^{\mathrm{AUC}}(g)] \tag{5.92}$$

for any $\gamma \in (0, 2\gamma_{\mathrm{PNPU}}^{\mathrm{AUC}})$ if $\overline{\psi}_{\mathrm{PN}} + \overline{\psi}_{\mathrm{PU}} > \overline{\psi}_{\mathrm{PP}}$ and $2\overline{\psi}_{\mathrm{PN}} > \overline{\psi}_{\mathrm{PP}}$. Similarly, we have

$$\mathrm{Var}[\widehat{R}_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}(g)] < \mathrm{Var}[\widehat{R}_{\mathrm{PN}}^{\mathrm{AUC}}(g)] \tag{5.93}$$

for any $\gamma \in (0, 2\gamma_{\mathrm{PNNU}}^{\mathrm{AUC}})$ if $\overline{\psi}_{\mathrm{PN}} + \overline{\psi}_{\mathrm{NU}} > \overline{\psi}_{\mathrm{NN}}$ and $2\overline{\psi}_{\mathrm{PN}} > \overline{\psi}_{\mathrm{NN}}$.

This theorem means that, if γ is chosen appropriately, our proposed risk estimators, $\widehat{R}_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}$ and $\widehat{R}_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}$, have smaller variance than the standard supervised risk estimator $\widehat{R}_{\mathrm{PN}}^{\mathrm{AUC}}$. A practical consequence of Theorem 18 is that when we conduct cross-validation for hyperparameter selection, we may use the PNU-AUC risk estimators $\widehat{R}_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}$ and $\widehat{R}_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}$ instead of the standard supervised risk estimator $\widehat{R}_{\mathrm{PN}}^{\mathrm{AUC}}$ since they are more stable (see Section 5.4.4 for details).

5.6 Experiments on Balanced Classification

In this section, we first numerically analyze our semi-supervised learning approach with the misclassification rate and then compare the methods against existing methods. All experiments were carried out using a PC equipped with two 2.60GHz Intel® Xeon® E5-2640 v3 CPUs.

5.6.1 Experimental Analyses

Here, we numerically analyze the behavior of the PNU classification method.

Common Setup: As a classifier, we use the Gaussian kernel model:

$$g(\boldsymbol{x}) = \sum_{i=1}^{n} w_i \exp\Big(-\frac{\|\boldsymbol{x}-\boldsymbol{x}_i\|^2}{2\sigma^2}\Big),$$

where $n = n_{\mathrm{P}} + n_{\mathrm{N}}$, $\{w_i\}_{i=1}^n$ are the parameters, $\{\boldsymbol{x}_i\}_{i=1}^n = \mathcal{X}_{\mathrm{P}} \cup \mathcal{X}_{\mathrm{N}}$, and $\sigma > 0$ is the Gaussian bandwidth. The bandwidth candidates are

$$\mathrm{median}(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{i,j=1}^n) \times \Big\{\frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, \frac{3}{2}, 2\Big\}.$$

We denote the classifier trained by minimizing the empirical PN risk by $\widehat{g}_{\mathrm{PN}}$. The number of labeled samples for training is 20, where the class-prior was 0.5. Note that the class-prior of test data was the same as that of unlabeled data.

Effect of Different Loss Functions: We first compare the performance of our semi-supervised classification methods with different loss functions. Specifically, we compare the PNU classification method with the ramp loss (RL), squared loss (SL), logistic loss (LL), and double hinge loss (DH) and the PUNU classification method with the squared loss and logistic loss.

Table 5.1 summarizes the average and standard error of the misclassification rates over 30 trials, showing that the clear difference was not observed between loss functions for both the PNU and PUNU classification methods. As discussed in Section 5.3.3, the PNU classification method tends to outperform the PUNU classification method. This tendency will be confirmed by the experiments on more datasets in Section 5.6.2.

Figure 5.3 plots the computation time, showing that the PNU and PUNU classification methods with the squared loss were the fastest. The methods with the logistic loss were

TABLE 5.1: Average and standard error of the misclassification rates of each method over 30 trials for benchmark datasets. The boldface denotes the best and comparable methods in terms of the average misclassification rate according to the t-test at the significance level 5%.

Dataset	θ_{p}	$\widehat{\theta}_{\mathrm{p}}$	$n_{\mathrm{L}} = 50, n_{\mathrm{U}} = 300$					
			PNU(RL)	PNU(DH)	PNU(SL)	PNU(LL)	PUNU(SL)	PUNU(LL)
Phoneme	0.5	0.58 (0.02)	**28.4 (1.0)**	**27.7 (0.7)**	**27.9 (0.9)**	**28.2 (0.8)**	**28.7 (0.9)**	**28.1 (0.8)**
cod-rna	0.5	0.63 (0.02)	**16.4 (2.1)**	**16.9 (1.9)**	**16.5 (2.0)**	**17.4 (1.7)**	**20.6 (1.7)**	23.3 (1.8)
Waveform	0.5	0.52 (0.01)	14.9 (0.6)	**13.7 (0.5)**	**14.3 (0.5)**	**13.3 (0.4)**	15.8 (0.6)	**13.6 (0.5)**
Spambase	0.5	0.62 (0.02)	**24.6 (1.3)**	**23.1 (1.1)**	**24.7 (1.4)**	**24.5 (1.4)**	**25.3 (1.1)**	**24.4 (1.0)**
phishing	0.5	0.58 (0.01)	**13.3 (0.6)**	**13.7 (0.7)**	**13.3 (0.7)**	**14.9 (0.8)**	15.8 (0.7)	17.4 (0.7)
Coil2	0.5	0.62 (0.02)	**27.3 (1.4)**	**26.7 (1.4)**	**27.4 (1.3)**	**27.6 (1.4)**	**27.4 (1.5)**	**28.4 (1.4)**

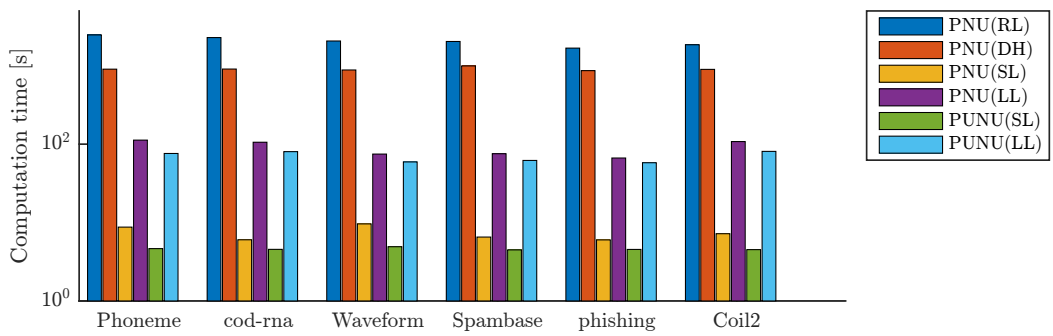


FIGURE 5.3: The average computation time over 30 trials for benchmark datasets.

slower than those with the squared loss. The ramp loss and double-hinge loss methods were much slower than other methods.

In summary, from the viewpoint of performance and computational efficiency, the squared loss was shown to be advantageous and thus we use it in the subsequent experiments.

Variance Reduction in Practice: Next, we numerically investigate how many unlabeled samples are sufficient in practice so that the variance of the empirical PNU risk is smaller than that of the PN risk: $\mathrm{Var}[\widehat{R}^{\mathrm{MR},\eta}_{\mathrm{PNU}}(g)] < \mathrm{Var}[\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(g)]$ given a fixed classifier g.

As the fixed classifier, we used the classifier $\widehat{g}_{\mathrm{PN}}$, where the hyperparameters were determined by five-fold cross-validation. To compute the variance of the empirical PN and PNU risks, $\mathrm{Var}[\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(\widehat{g}_{\mathrm{PN}})]$ and $\mathrm{Var}[\widehat{R}^{\mathrm{MR},\eta}_{\mathrm{PNU}}(\widehat{g}_{\mathrm{PN}})]$, we repeatedly drew additional $n^{\mathrm{V}}_{\mathrm{P}} = 10$ positive, $n^{\mathrm{V}}_{\mathrm{N}} = 10$ negative, and $n^{\mathrm{V}}_{\mathrm{U}}$ unlabeled samples from the rest of the dataset. The

additional samples were also used for estimating $\widehat{\sigma}_{\mathrm{P}}(\widehat{g}_{\mathrm{PN}})$ and $\widehat{\sigma}_{\mathrm{N}}(\widehat{g}_{\mathrm{PN}})$ to compute η, i.e., γ as in Section 5.4.4.

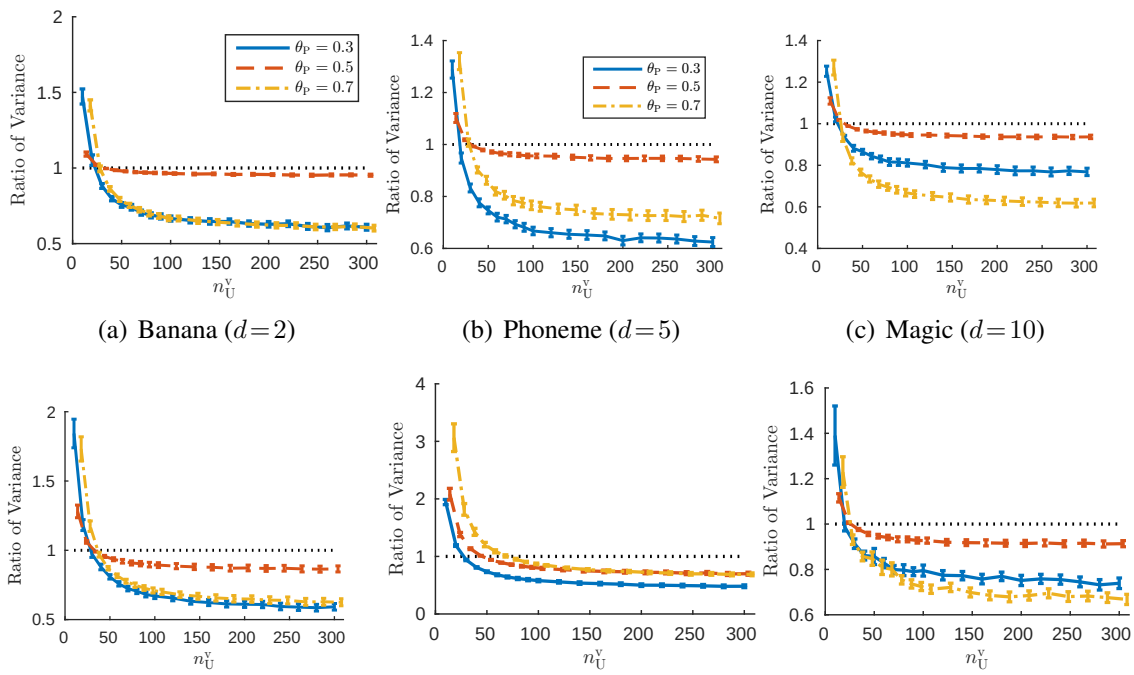
Figure 5.4 shows the ratio between the variance of the empirical PNU risk and that of the PN risk, $\mathrm{Var}[\widehat{R}^{\mathrm{MR},\eta}_{\mathrm{PNU}}(\widehat{g}_{\mathrm{PN}})]/\,\mathrm{Var}[\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(\widehat{g}_{\mathrm{PN}})]$, where the number of unlabeled samples for validation $n^{\mathrm{V}}_{\mathrm{U}}$ increases from 10 to 300. With a rather small number of unlabeled samples, we see that the ratio becomes less than 1, meaning that the variance of the empirical PNU risk becomes smaller than that of the PN risk. Although the variance reduction is proved given an infinite number of unlabeled samples, in practice, a finite number of samples is sufficient for reducing the variance.

Compared to cases when $\theta_{\mathrm{P}} = 0.3$ and 0.7, the effect of variance reduction is small when $\theta_{\mathrm{P}} = 0.5$ because $\gamma^{\mathrm{MR}}_{\text{N-PNPU}} \approx \gamma^{\mathrm{MR}}_{\text{N-PNNU}} \approx 0$ if we assume $\sigma_{\mathrm{P}}(g) \approx \sigma_{\mathrm{N}}(g)$ when $n_{\mathrm{P}} \approx n_{\mathrm{N}}$ and $\theta_{\mathrm{P}} = 0.5$ (i.e., $\psi_{\mathrm{P}} \approx \psi_{\mathrm{N}}$. See Theorem 17). That is, the PNU risk is dominated by the PN risk, implying that $\mathrm{Var}[\widehat{R}^{\mathrm{MR},\eta}_{\mathrm{PNU}}(g)] \approx \mathrm{Var}[\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(g)]$. It is worth noting taht the class-prior is not the only factor for variance reduction; for example, the variance reduction will be large if $\theta_{\mathrm{P}} = 0.5$, $n_{\mathrm{P}} \gg n_{\mathrm{N}}$, and $\sigma_{\mathrm{P}}(g) \approx \sigma_{\mathrm{N}}(g)$ due to $\gamma^{\mathrm{MR}}_{\text{N-PNPU}} \not\approx 0$ (because $\psi_{\mathrm{P}} \ll \psi_{\mathrm{N}}$).

PNU Risk in Validation: As discussed in Section 5.5.2, the empirical PNU risk will be a reliable validation score since its variance is smaller than the empirical PN risk, Here, we show that the empirical PNU risk is a promising alternative to a validation score.

To focus on the effect of the risk function for validation only, we fix a training method. More specifically, we prepared two classifiers trained by the same risk, e.g., the empirical PN risk. We then tune the classifiers with the empirical PN and PNU risks denoted by $\widehat{g}^{\mathrm{PN}}_{\mathrm{PN}}$ and $\widehat{g}^{\mathrm{PNU}}_{\mathrm{PN}}$, respectively. For validation, we draw additional $n^{\mathrm{V}}_{\mathrm{P}} = 10$ positive, $n^{\mathrm{V}}_{\mathrm{N}} = 10$ negative, and $n^{\mathrm{V}}_{\mathrm{U}}$ unlabeled samples from the rest of the dataset.

Figure 5.5 shows the ratio between the misclassification rate of $\widehat{g}^{\mathrm{PNU}}_{\mathrm{PN}}$ and that of $\widehat{g}^{\mathrm{PN}}_{\mathrm{PN}}$: $R^{\mathrm{MR}}(\widehat{g}^{\mathrm{PNU}}_{\mathrm{PN}})/R^{\mathrm{MR}}(\widehat{g}^{\mathrm{PN}}_{\mathrm{PN}})$. With a rather small number of unlabeled samples, the ratio becomes less than 1, i.e., $\widehat{g}^{\mathrm{PNU}}_{\mathrm{PN}}$ achieves better performance than $\widehat{g}^{\mathrm{PN}}_{\mathrm{PN}}$. In particular, when $\theta_{\mathrm{P}} = 0.3$ and 0.7, $\widehat{g}^{\mathrm{PNU}}_{\mathrm{PN}}$ improved substantially; the large variance reduction tends to give the large improvement (cf. Figure 5.4). This result shows that the use of the empirical PNU risk for validation improved the classification performance given a rather small number of unlabeled samples.



(a) Banana ($d=2$) (b) Phoneme ($d=5$) (c) Magic ($d=10$)

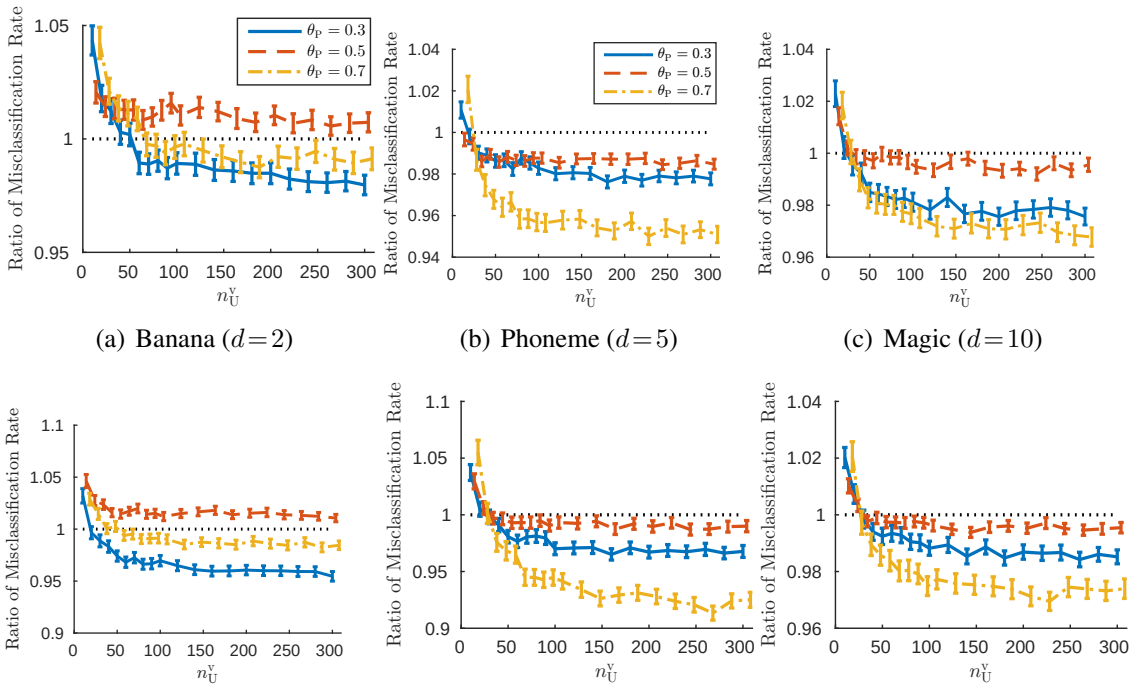
(d) Image ($d=18$) (e) Waveform ($d=21$) (f) Splice ($d=60$)

FIGURE 5.4: Average and standard error of the ratio between the variance of empirical PNU risk and that of PN risk, $\mathrm{Var}[\widehat{R}^{\mathrm{MR},\eta}_{\mathrm{PNU}}(\widehat{g}_{\mathrm{PN}})]/\mathrm{Var}[\widehat{R}^{\mathrm{MR}}_{\mathrm{PN}}(\widehat{g}_{\mathrm{PN}})]$, as a function of the number of unlabeled samples over 100 trials. Although the variance reduction is proved for an infinite number of samples, it can be observed with a finite number of samples.

5.6.2 Comparison with Existing Methods

In this subsection, we numerically compare our methods against existing semi-supervised classification methods.

Common Setup: We compare our methods against five conventional semi-supervised classification methods: the *entropy regularization* (ER) (Grandvalet and Bengio, 2004), the *Laplacian support vector machine* (LapSVM) (Belkin et al., 2006; Melacci and Belkin, 2011), the *squared-loss mutual information regularization* (SMIR) (Niu et al., 2013), the *weakly labeled support vector machine* (WellSVM) (Li et al., 2013), and the *safe semi-supervised support vector machine* (S4VM) (Li and Zhou, 2015). See Section 2.6 for the details of these methods.



(a) Banana ($d=2$) (b) Phoneme ($d=5$) (c) Magic ($d=10$)

(d) Image ($d=18$) (e) Waveform ($d=21$) (f) Splice ($d=60$)

FIGURE 5.5: Average and standard error of the ratio between the misclassification rates of $\widehat{g}_{\mathrm{PN}}^{\mathrm{PNU}}$ and $\widehat{g}_{\mathrm{PN}}^{\mathrm{PN}}$ as a function of unlabeled samples over 1000 trials. In many cases, the ratio becomes less than 1, implying that the PNU risk is a promising alternative to the standard PN risk in validation if unlabeled data are available.

DataSets: We used sixteen benchmark datasets taken from the *UCI Machine Learning Repository* (Lichman, 2013), the *Semi-Supervised Learning book* (Chapelle et al., 2006), the *LIBSVM* (Chang and Lin, 2011), the *ELENA Project*,[4] and the paper by Chapelle and Zien (2005).[5] Each feature was scaled to $[0, 1]$. Similarly to the setting in Section 5.6.1, we used the Gaussian kernel model for all methods. The training data is $\{\boldsymbol{x}_i\}_{i=1}^n = \mathcal{X}_{\mathrm{P}} \cup \mathcal{X}_{\mathrm{N}} \cup \mathcal{X}_{\mathrm{U}}$, where $n = n_{\mathrm{P}} + n_{\mathrm{N}} + n_{\mathrm{U}}$. We selected all hyper-parameters with validation samples of size 20 ($n_{\mathrm{P}}^{\mathrm{V}} = n_{\mathrm{N}}^{\mathrm{V}} = 10$). For training, we drew n_{L} labeled and $n_{\mathrm{U}} = 300$ unlabeled samples. The class-prior of labeled data was set at 0.7 and that of unlabeled samples was set at $\theta_{\mathrm{P}} = 0.5$ that were assumed to be known. In practice, the class-prior, θ_{P}, can be estimated by methods proposed, e.g., by Blanchard et al. (2010), Ramaswamy et al. (2016), or Kawakubo et al. (2016).

[4] https://www.elen.ucl.ac.be/neural-nets/Research/Projects/ELENA/elena.htm

[5] http://olivier.chapelle.cc/lds/

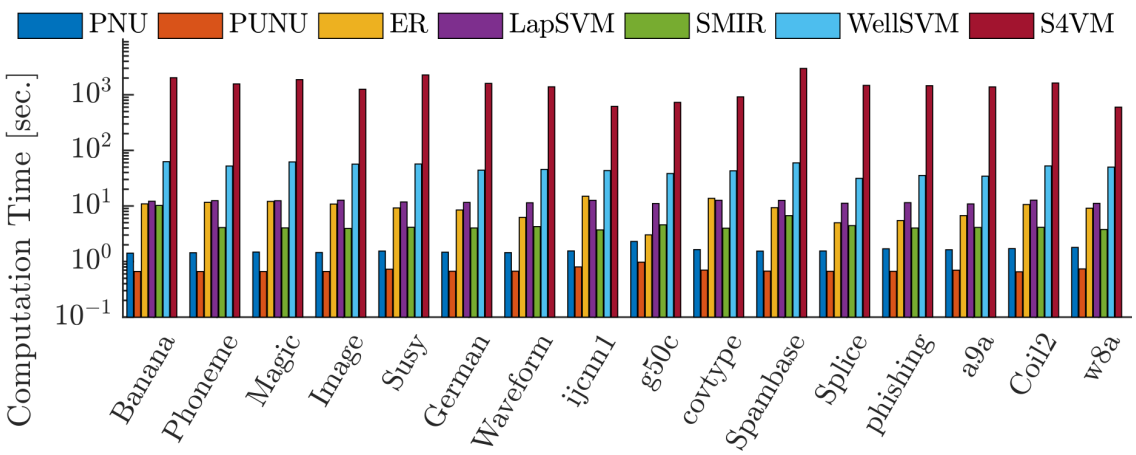


FIGURE 5.6: Average computation time over 50 trials for benchmark datasets when $n_{\mathrm{L}} = 50$.

Table 5.2 lists the average with standard error of the misclassification rates over 50 trials and the number of best/comparable performances of each method in the bottom row. The superior performance of the PNU classification method over the PUNU classification method agrees well with the discussion in Section 5.3.3. For the g50c dataset, which well satisfies the low-density separation principle, WellSVM achieved the best performance. However, in the Banana dataset, where the two classes are highly overlapped, the performance of WellSVM was worse than the other methods. On the other hand, the PNU classification method achieved consistently better performance than or comparable performance to the other methods and its performance did not degenerate considerably across datasets. These results show that the idea of using PU classification in semi-supervised classification is promising.

Figure 5.6 plots the computation time, showing that our method was computationally efficient compared with the other methods.

5.6.3 Image Classification:

Next, we apply the PNU classification method to image classification. We used the *Places* 205 *dataset* (Zhou et al., 2014), which contains 2.5 million images in 205 scene classes. As a feature vector, we used a 4096-dimensional feature vector extracted from each image by *AlexNet* (Krizhevsky et al., 2012) under the framework of *Caffe*,[6] which is available on the project website.[7] We chose two scenes with similar names to construct binary

[6] http://caffe.berkeleyvision.org/
[7] http://places.csail.mit.edu/

TABLE 5.2: Average and standard error of the misclassification rates of each method over 50 trials for benchmark datasets. Boldface numbers denote the best and comparable methods in terms of average misclassifications rate according to a t-test at a significance level of 5%. The bottom row gives the number of best/comparable cases of each method.

Dataset	n_{L}	PNU	PUNU	ER	LapSVM	SMIR	WellSVM	S4VM
Banana	10	**30.1 (1.0)**	**32.1 (1.1)**	35.8 (1.0)	36.9 (1.0)	37.7 (1.1)	41.8 (0.6)	45.3 (1.0)
$d = 2$	50	**19.0 (0.6)**	26.4 (1.2)	20.6 (0.7)	21.3 (0.7)	21.1 (1.0)	42.6 (0.5)	38.7 (0.9)
Phoneme	10	32.5 (0.8)	33.5 (1.0)	33.4 (1.2)	36.5 (1.5)	36.4 (1.2)	**28.4 (0.6)**	33.7 (1.4)
$d = 5$	50	28.1 (0.5)	32.8 (0.9)	27.8 (0.6)	27.0 (0.8)	28.6 (1.0)	26.8 (0.4)	**25.1 (0.2)**
Magic	10	**31.7 (0.8)**	34.1 (0.9)	34.2 (1.1)	37.9 (1.3)	36.0 (1.2)	**30.1 (0.8)**	33.3 (0.9)
$d = 10$	50	**29.9 (0.8)**	33.4 (0.9)	30.9 (0.5)	31.0 (0.9)	**30.8 (0.9)**	**28.8 (0.8)**	**29.2 (0.4)**
Image	10	**29.8 (0.9)**	**31.7 (0.8)**	33.7 (1.1)	36.6 (1.2)	36.7 (1.2)	34.7 (1.1)	35.9 (1.0)
$d = 18$	50	**20.7 (0.8)**	26.6 (1.1)	**20.8 (0.8)**	**20.3 (1.0)**	**20.9 (0.9)**	27.2 (1.0)	23.2 (0.7)
Susy	10	**44.6 (0.6)**	**45.0 (0.6)**	47.7 (0.4)	48.2 (0.4)	**45.1 (0.7)**	48.0 (0.3)	46.8 (0.3)
$d = 18$	50	**38.9 (0.6)**	41.5 (0.6)	**37.9 (0.7)**	43.1 (0.6)	43.9 (0.8)	43.8 (0.7)	42.1 (0.4)
German	10	**40.8 (0.9)**	**42.4 (0.7)**	43.6 (0.9)	45.9 (0.7)	46.2 (0.8)	**42.4 (0.8)**	**42.0 (0.7)**
$d = 20$	50	**36.2 (0.8)**	39.0 (0.8)	38.9 (0.6)	40.6 (0.6)	38.4 (1.1)	38.5 (1.0)	**34.9 (0.5)**
Waveform	10	**17.4 (0.6)**	**18.0 (0.9)**	18.5 (0.6)	24.9 (1.4)	**18.0 (1.0)**	**16.7 (0.6)**	20.8 (0.8)
$d = 21$	50	16.3 (0.6)	23.7 (1.2)	**14.2 (0.4)**	18.1 (0.8)	**15.4 (0.6)**	15.5 (0.5)	15.3 (0.3)
ijcnn1	10	43.6 (0.6)	**40.3 (1.0)**	49.7 (0.1)	49.2 (0.3)	44.0 (1.0)	45.9 (0.7)	49.3 (0.8)
$d = 22$	50	**34.5 (0.8)**	37.1 (0.9)	**35.5 (0.8)**	**33.4 (1.1)**	49.4 (0.3)	46.2 (0.8)	48.6 (0.4)
g50c	10	11.4 (0.6)	12.5 (0.6)	23.3 (2.3)	39.8 (1.6)	21.9 (1.3)	**6.6 (0.4)**	27.0 (1.4)
$d = 50$	50	12.5 (1.1)	10.1 (0.6)	8.7 (0.4)	22.5 (1.5)	10.6 (0.6)	**7.4 (0.4)**	12.1 (0.5)
covtype	10	**46.2 (0.4)**	**46.0 (0.4)**	**46.0 (0.5)**	47.1 (0.5)	47.9 (0.5)	**46.9 (0.6)**	**46.4 (0.4)**
$d = 54$	50	**41.3 (0.5)**	42.3 (0.5)	**41.0 (0.4)**	**41.5 (0.5)**	46.2 (0.8)	43.6 (0.6)	**40.8 (0.4)**
Spambase	10	27.2 (0.9)	28.1 (1.1)	31.8 (1.4)	39.7 (1.4)	30.9 (1.3)	**23.8 (0.8)**	36.1 (1.5)
$d = 57$	50	23.4 (1.0)	26.6 (1.0)	22.1 (0.7)	28.5 (1.3)	20.9 (0.5)	**19.1 (0.4)**	24.5 (0.9)
Splice	10	**38.3 (0.8)**	**39.3 (0.8)**	43.9 (0.8)	47.9 (0.5)	41.6 (0.7)	42.0 (1.0)	42.4 (0.6)
$d = 60$	50	**30.6 (0.8)**	34.7 (0.9)	**30.9 (0.8)**	38.8 (1.0)	**30.6 (0.9)**	40.9 (0.8)	35.9 (0.7)
phishing	10	**24.2 (1.2)**	**25.8 (1.0)**	**27.3 (1.6)**	37.2 (1.6)	27.6 (1.6)	27.5 (1.4)	31.7 (1.3)
$d = 68$	50	**15.8 (0.6)**	18.3 (0.8)	**15.4 (0.5)**	21.1 (1.3)	**14.7 (0.8)**	17.2 (0.7)	16.7 (0.8)
a9a	10	**31.4 (0.9)**	**31.3 (1.0)**	34.3 (1.2)	41.0 (1.1)	37.3 (1.3)	**33.1 (1.2)**	34.3 (1.2)
$d = 83$	50	27.9 (0.6)	29.9 (0.8)	28.6 (0.7)	33.3 (1.0)	**26.9 (0.7)**	28.9 (0.8)	**26.2 (0.4)**
Coil2	10	**38.7 (0.8)**	**40.1 (0.8)**	42.8 (0.7)	43.9 (0.8)	43.2 (0.8)	**39.1 (0.9)**	44.0 (0.8)
$d = 241$	50	**23.2 (0.6)**	30.5 (0.9)	**23.6 (0.9)**	**22.8 (0.9)**	25.1 (0.9)	**22.6 (0.8)**	25.4 (0.8)
w8a	10	**35.9 (0.9)**	**33.6 (1.0)**	41.6 (1.0)	46.6 (0.8)	39.4 (0.9)	42.1 (0.8)	43.0 (0.8)
$d = 300$	50	**28.1 (0.7)**	**27.6 (0.6)**	**27.0 (0.9)**	38.7 (0.8)	**28.0 (0.9)**	33.7 (0.8)	35.2 (1.0)
#Best/Comp.		23	13	11	4	9	13	7

classification tasks (see the description of datasets in Table 5.3). We drew 100 labeled and n_{U} unlabeled samples from each task; the class-prior of labeled and unlabeled data were respectively set at 0.5 and $\theta_{\mathrm{P}} = m_{\mathrm{P}}/(m_{\mathrm{P}} + m_{\mathrm{N}})$, where m_{P} and m_{N} respectively denote the number of total samples in positive and negative scenes. We used a linear classifier $g(\boldsymbol{x}) = \boldsymbol{w}^{\top}\boldsymbol{x} + w_0$, where $\boldsymbol{w}$ is the weight vector and w_0 is the offset (in the SMIR, the linear kernel model is used; see Section 2.6 for details).

TABLE 5.3: The description of the dataset used in the image classification experiment.

Dataset	Data sources	#Samples
Arts	Art Gallery	$(m_{\mathrm{P}}=15000)$
	vs.	
	Art Studio	$(m_{\mathrm{N}}=15000)$
Deserts	Desert Sand	$(m_{\mathrm{P}}=15000)$
	vs.	
	Desert Vegetation	$(m_{\mathrm{N}}=5556)$
Fields	Field Wild	$(m_{\mathrm{P}} = 15000)$
	vs.	
	Field Cultivated	$(m_{\mathrm{N}} = 8117)$
Stadiums	Stadium Baseball	$(m_{\mathrm{P}}=15000)$
	vs.	
	Stadium Football	$(m_{\mathrm{N}}=15000)$
Platforms	Subway Station	$(m_{\mathrm{P}} = 5597)$
	vs.	
	Train Station	$(m_{\mathrm{N}}=15000)$
Temples	Temple East Asia	$(m_{\mathrm{P}}=8691)$
	vs.	
	Temple South Asia	$(m_{\mathrm{N}}=7178)$

The hyper-parameters in the PNU classification method were selected by five-fold cross-validation. The class-prior $p(y = +1) = \theta_{\mathrm{P}}$ was estimated using the method based on energy distance minimization (Kawakubo et al., 2016).

Table 5.4 lists the average with standard error of the misclassification rates over 30 trials, where methods taking more than 2 hours were omitted and indicated as *N/A*. The results show that the PNU classification method was most effective. The average computation times are shown in Figure 5.7, revealing again that the PNU classification method was the fastest method.

TABLE 5.4: Average and standard error of misclassification rates over 30 trials for the Places 205 dataset. Boldface numbers denote the best and comparable methods in terms of the average misclassification rate according to a t-test at a significance level of 5%.

Dataset	n_{U}	θ_{P}	$\widehat{\theta}_{\mathrm{P}}$	PNU	ER	LapSVM	SMIR	WellSVM
Arts	1000	0.50	0.49 (0.01)	**27.4 (1.3)**	**26.6 (0.5)**	**26.1 (0.7)**	40.1 (3.9)	**27.5 (0.5)**
	5000	0.50	0.50 (0.01)	**24.8 (0.6)**	26.1 (0.5)	26.1 (0.4)	30.1 (1.6)	N/A
	10000	0.50	0.52 (0.01)	**25.6 (0.7)**	**25.4 (0.5)**	**25.5 (0.6)**	N/A	N/A
Deserts	1000	0.73	0.67 (0.01)	**13.0 (0.5)**	15.3 (0.6)	16.7 (0.8)	17.2 (0.8)	18.2 (0.7)
	5000	0.73	0.67 (0.01)	**13.4 (0.4)**	**13.3 (0.5)**	16.6 (0.6)	24.4 (0.6)	N/A
	10000	0.73	0.68 (0.01)	**13.3 (0.5)**	**13.7 (0.6)**	16.8 (0.8)	N/A	N/A
Fields	1000	0.65	0.57 (0.01)	**22.4 (1.0)**	26.2 (1.0)	26.6 (1.3)	28.2 (1.1)	26.6 (0.8)
	5000	0.65	0.57 (0.01)	**20.6 (0.5)**	22.6 (0.6)	24.7 (0.8)	29.6 (1.2)	N/A
	10000	0.65	0.57 (0.01)	**21.6 (0.6)**	**22.5 (0.6)**	25.0 (0.9)	N/A	N/A
Stadiums	1000	0.50	0.50 (0.01)	**11.4 (0.4)**	**11.5 (0.5)**	12.5 (0.5)	**17.4 (3.6)**	**11.7 (0.4)**
	5000	0.50	0.50 (0.01)	**11.0 (0.5)**	**10.9 (0.3)**	**11.1 (0.3)**	13.4 (0.7)	N/A
	10000	0.50	0.51 (0.00)	**10.7 (0.3)**	**10.9 (0.3)**	**11.2 (0.2)**	N/A	N/A
Platforms	1000	0.27	0.33 (0.01)	**21.8 (0.5)**	23.9 (0.6)	24.1 (0.5)	30.1 (2.3)	26.2 (0.8)
	5000	0.27	0.34 (0.01)	**23.3 (0.8)**	**24.4 (0.7)**	**24.9 (0.7)**	26.6 (0.3)	N/A
	10000	0.27	0.34 (0.01)	**21.4 (0.5)**	24.3 (0.6)	24.8 (0.5)	N/A	N/A
Temples	1000	0.55	0.51 (0.01)	**43.9 (0.7)**	**43.9 (0.6)**	**43.4 (0.6)**	50.7 (1.6)	**44.3 (0.5)**
	5000	0.55	0.54 (0.01)	**43.4 (0.9)**	**43.0 (0.6)**	**43.1 (1.0)**	**43.6 (0.7)**	N/A
	10000	0.55	0.50 (0.01)	**45.2 (0.8)**	**44.4 (0.8)**	**44.2 (0.7)**	N/A	N/A

5.7 Experiments on Imbalanced Classification

In this section, we numerically investigate the behavior of the PNU-AUC optimization method and evaluate their performance on various datasets.

As the classifier, we used the linear-in-parameter model. In all experiments except text classification tasks, we used the Gaussian kernel basis function expressed as $\phi_\ell(\boldsymbol{x}) = \exp(-\|\boldsymbol{x}-\boldsymbol{x}_\ell\|^2/(2\sigma^2))$, where $\sigma > 0$ is the Gaussian bandwidth, $\{\boldsymbol{x}_\ell\}_{\ell=1}^b$ are the samples randomly selected from training samples $\{\boldsymbol{x}_i\}_{i=1}^n$ and n is the number of training samples. In text classification tasks, we used the linear kernel basis function: $\phi_\ell(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{x}_\ell$. The number of basis functions was set at $b = \min(n, 200)$. The candidates of the Gaussian bandwidth were $\mathrm{median}(\{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\}_{i,j=1}^n) \times \{1/8,\ 1/4,\ 1/2,\ 1,\ 2\}$ and those of the regularization parameter were $\{10^{-3},\ 10^{-2},\ 10^{-1},\ 10^0,\ 10^1\}$. All hyper-parameters were determined by five-fold cross-validation. As the loss function, we used the squared loss function $\ell_{\mathrm{S}}(m) = (1-m)^2$.

We used 15 benchmark datasets from the *IDA Benchmark Repository* (Rätsch et al.,

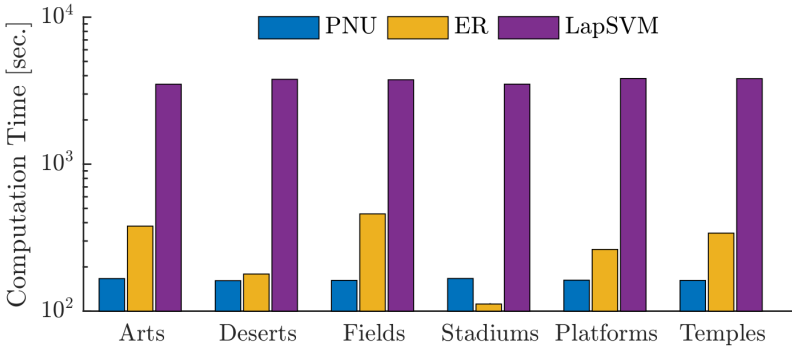


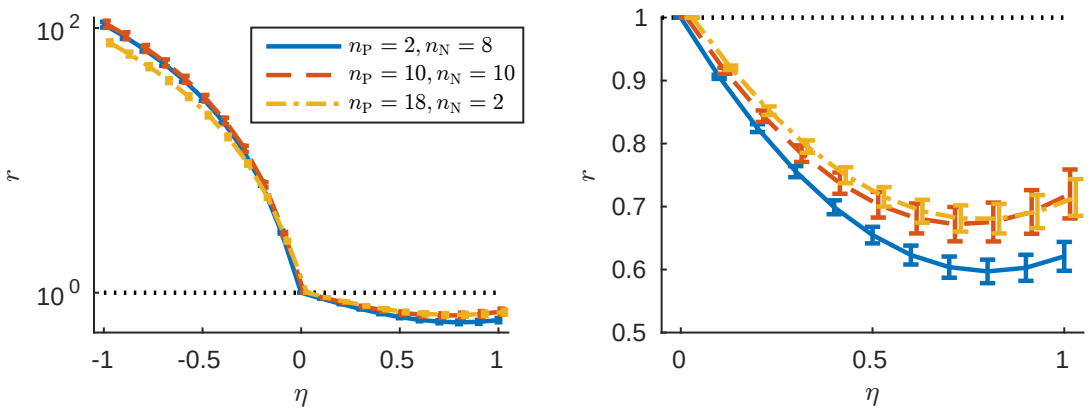
FIGURE 5.7: Average computation time over 30 trials for the Places 205 dataset when $n_{\mathrm{U}} = 10000$.

2001), the *Semi-Supervised Learning* book (Chapelle et al., 2006), the *LIBSVM* (Chang and Lin, 2011), and the *UCI Machine Learning Repository* (Lichman, 2013). Furthermore, in text classification tasks, we used the *Reuters Corpus Volume I* dataset (Lewis et al., 2004), the *Amazon Review* dataset (Dredze et al., 2008), and the 20 *Newsgroups* dataset (Lang, 1995). Since these text classification datasets are originally for multi-class classification, by dividing the whole classes into two group or choosing the two classes randomly, we constructed binary classification tasks: the rcv1, amazon2, and news20 datasets. The rcv1 and news20 datasets are available at the website of LIBSVM (Chang and Lin, 2011), and the amazon2 is designed by ourselves, which consists of the product reviews of books and music from the *Amazon7* dataset (Blondel et al., 2013).

5.7.1 Experimental Analysis

Range of Variance Reduction: Firstly, we numerically confirm the effect of variance reduction. We compare the variance of the empirical PNU-AUC risk against the variance of the empirical PN-AUC risk, $\mathrm{Var}[\widehat{R}_{\mathrm{PNU}}^{\mathrm{AUC},\eta}(g)]$ vs. $\mathrm{Var}[\widehat{R}_{\mathrm{PN}}^{\mathrm{AUC}}(g)]$, under a fixed classifier g.

As the fixed classifier, we used the minimizer of the empirical PN-AUC risk, denoted by $\widehat{g}_{\mathrm{PN}}$. The number of positive and negative samples for training varied as $(n_{\mathrm{P}}, n_{\mathrm{N}}) = (2, 8)$, $(10, 10)$, and $(18, 2)$. We then computed the variance of the empirical PN-AUC and PNU-AUC risks with additional 10 positive, 10 negative, and 300 unlabeled samples. As the dataset, we used the Banana dataset (Rätsch et al., 2001). In this experiment, the class-prior was set at $\theta_{\mathrm{P}} = 0.1$ and assumed to be known.



(a) $-1 \leq \eta \leq 1$ (b) $\eta > 0$

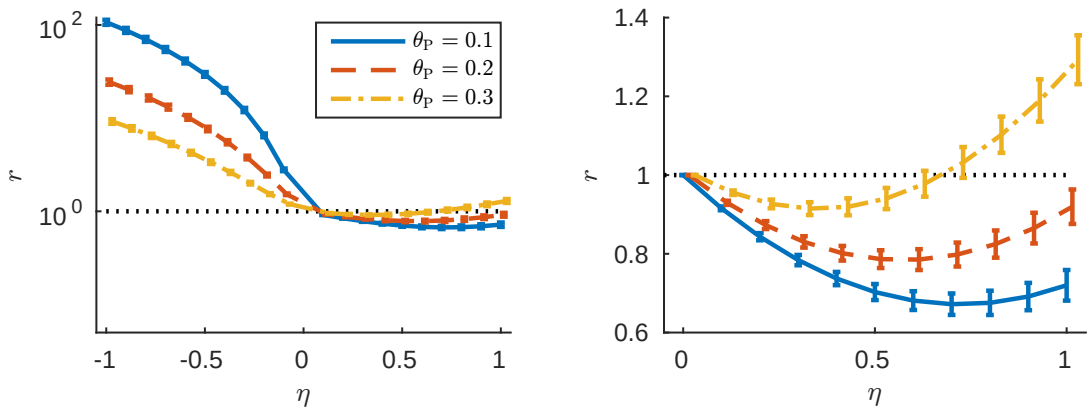
FIGURE 5.8: Average with standard error of the ratio between the variance of the empirical PNU risk and that of the PN risk, $r = \mathrm{Var}[\widehat{R}_{\mathrm{PNU}}^{\mathrm{AUC},\eta}(\widehat{g}_{\mathrm{PN}})]/\mathrm{Var}[\widehat{R}_{\mathrm{PN}}^{\mathrm{AUC}}(\widehat{g}_{\mathrm{PN}})]$, as a function of the combination parameter η over 100 trials on the Banana dataset. The class-prior is $\theta_{\mathrm{P}} = 0.1$ and the number of positive and negative samples varies as $(n_{\mathrm{P}}, n_{\mathrm{N}}) = (2, 8)$, $(10, 10)$, and $(18, 2)$. Left: values of r as a function of η. Right: values for $\eta > 0$ are magnified.

Figure 5.8 plots the value of the variance of the empirical PNU-AUC risk divided by that of the PN-AUC risk,

$$r := \frac{\mathrm{Var}[\widehat{R}_{\mathrm{PNU}}^{\mathrm{AUC},\eta}(\widehat{g}_{\mathrm{PN}})]}{\mathrm{Var}[\widehat{R}_{\mathrm{PN}}^{\mathrm{AUC}}(\widehat{g}_{\mathrm{PN}})]}, \tag{5.94}$$

as a function of the combination parameter η under different numbers of positive and negative samples. The results show that $r < 1$ can be achieved by an appropriate choice of η, meaning that the variance of the empirical PNU-AUC risk can be smaller than that of the PN-AUC risk.

We then investigate how the class-prior affects the variance reduction. In this experiment, the number of positive and negative samples for $\widehat{g}_{\mathrm{PN}}$ were set at $n_{\mathrm{P}} = 10$ and $n_{\mathrm{N}} = 10$, respectively. Figure 5.9 showed the values of r as a function of the combination parameter η under different class-priors. In particular, the variance can be reduced for $\eta > 0$ when the class-prior, θ_{P}, is 0.1 and 0.2. On the other hand, the range of the value of η that yields variance reduction becomes smaller when the class-prior is 0.3. However, this may not be that problematic in practice, because AUC optimization is effective when



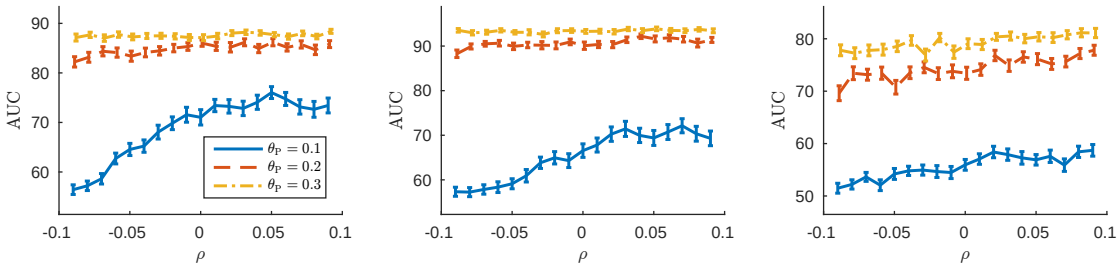
(a) $-1 \leq \eta \leq 1$ (b) $\eta > 0$

FIGURE 5.9: Average with standard error of the ratio between the variance of the PNU-AUC risk and that of the PN-AUC risk, $r = \mathrm{Var}[\widehat{R}_{\mathrm{PNU}}^{\mathrm{AUC},\eta}(\widehat{g}_{\mathrm{PN}})]/\mathrm{Var}[\widehat{R}_{\mathrm{PN}}^{\mathrm{AUC}}(\widehat{g}_{\mathrm{PN}})]$, as a function of the combination parameter η over 100 trials on the Banana dataset. A class-prior varies as $\theta_{\mathrm{P}} = 0.1$, 0.2, and 0.3. Left: values of r as a function of η. Right: values for $\eta > 0$ are magnified. When $\theta_{\mathrm{P}} = 0.1$, 0.2, the variance of the empirical PNU-AUC risk is smaller than that of the PN risk for $\eta > 0$.

two classes are highly imbalanced, i.e., the class-prior is far from 0.5; when the class-prior is close to 0.5, we may simply use the standard misclassification rate minimization approach.

Sensitivity Analysis: Secondly, we investigate the effect of the estimation accuracy of the class-prior for the PNU-AUC optimization method. Specifically, we used the class-prior $\widehat{\theta}_{\mathrm{P}} = \theta_{\mathrm{P}} + \rho$, which is the addition of the true class-prior θ_{P} and noise $\rho \in \{-0.09, -0.08, \ldots, 0.09\}$, as the estimated class-prior for the PNU-AUC optimization method. Under the different values of the class-prior $\theta_{\mathrm{P}} = 0.1$, 0.2, and 0.3, we trained a classifier with samples of size $n_{\mathrm{P}} = \theta_{\mathrm{P}} \cdot 50$, $n_{\mathrm{N}} = \theta_{\mathrm{N}} \cdot 50$, and $n_{\mathrm{U}} = 1000$.

Figure 5.10 summarizes the average with standard error of the AUC as a function of the noise, showing that when $\theta_{\mathrm{P}} = 0.2$ and 0.3, the performance of the PNU-AUC optimization method is stable even when the estimated class-prior has some noise. In contrast, when $\theta_{\mathrm{P}} = 0.1$, the performance largely decreases as the noise is close to $\rho = -0.09$. Since the true class-prior is small, it is sensitive to the negative bias. In particular, when $\rho = -0.09$, the ratio between the estimated and true class-priors is larger than other values. For instance, when $\rho = -0.09$ and $\theta_{\mathrm{P}} = 0.2$, $\theta_{\mathrm{P}}/\widehat{\theta}_{\mathrm{P}} \approx 1.8$, but when $\rho = -0.09$



(a) Banana ($d = 2$) (b) cod-rna ($d = 8$) (c) w8a ($d = 300$)

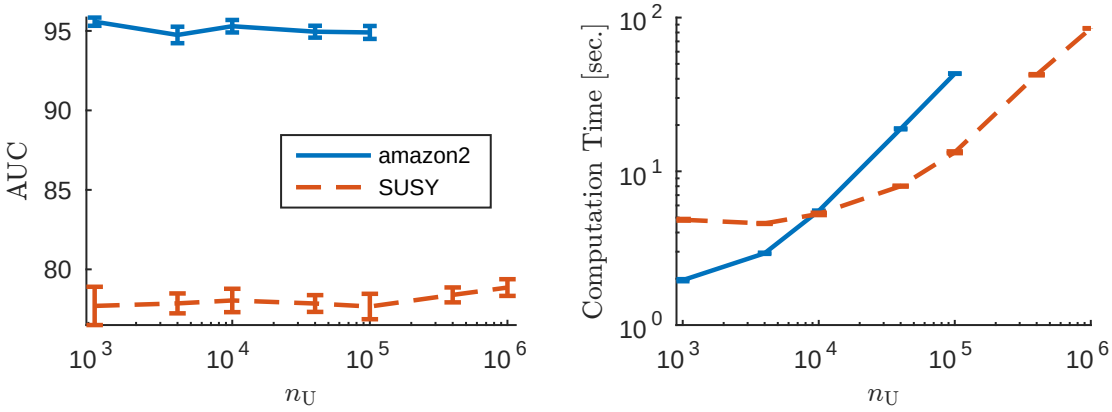
FIGURE 5.10: Average with standard error of the AUC as a function of the noise ρ over 100 trials. The PNU-AUC optimization method used the noisy class-prior $\widehat{\theta}_{\mathrm{P}} = \theta_{\mathrm{P}} + \rho$ in training. The plots show that when $\theta_{\mathrm{P}} = 0.2$ and 0.3, the performance of the PNU-AUC optimization method is stable even when the estimated class-prior has some noise. However, when $\theta_{\mathrm{P}} = 0.1$, as the noise is close to $\rho = -0.09$, the performance largely decreases. Since the true class-prior is small, it is sensitive to the negative bias.

and $\theta_{\mathrm{P}} = 0.1$, $\theta_{\mathrm{P}}/\widehat{\theta}_{\mathrm{P}} \approx 10$. In contrast, the positive bias does not heavily affect the performance even when $\theta_{\mathrm{P}} = 0.1$.

Scalability: Here, we report the scalability of the PNU-AUC optimization method. Specifically, we evaluated the AUC and computation time while increasing the number of unlabeled samples. We picked two large datasets: the SUSY and amazon2 datasets. The number of positive and negative samples were set at $n_{\mathrm{P}} = 40$ and $n_{\mathrm{N}} = 160$, respectively.

Figure 5.11 summarizes the average with standard error of the AUC and computation time as a function of the number of unlabeled samples. The AUC on the SUSY dataset slightly increased at $n_{\mathrm{U}} = 1{,}000{,}000$, but the improvement on the amazon2 dataset was not noticeable or the performance decreased slightly. In this experiment, the increase of the size of unlabeled data did not improve the performance of the classifier significantly, but it did not affect adversely, i.e., it did not cause significant performance degeneration.

The result of computation time shows that the PNU-AUC optimization method can handle approximately 1,000,000 samples within reasonable computation time in this experiment. The longer computation time on the SUSY dataset before $n_{\mathrm{U}} = 10{,}000$ is because we need to choose one additional hyperparameter, i.e., the bandwidth of the Gaussian kernel basis function, compared with the linear kernel basis function. However, the effect gradually decreases; after $n_{\mathrm{U}} = 10{,}000$, the matrix multiplication of the high dimensional matrix on the amazon2 dataset ($d = 262{,}144$) requires more computation time



(a) AUC (b) Computation time

FIGURE 5.11: Average with standard error of the AUC as a function of the number of unlabeled data n_{U} over 20 trials.

than the SUSY dataset ($d = 18$).

5.7.2 Comparison with Existing Methods

In this subsection, we report the classification performance of the PNU-AUC optimization method.

Here, we compare the PNU-AUC optimization method against existing AUC optimization methods: the semi-supervised rankboost (SSRankboost) (Amini et al., 2008),[8] the semi-supervised AUC-optimized logistic sigmoid (sAUC-LS) (Fujino and Ueda, 2016),[9] and the optimum AUC with a generative model (OptAG) (Fujino and Ueda, 2016).

The classifier was trained with samples of size $n_{\mathrm{P}} = \theta_{\mathrm{P}} \cdot n_{\mathrm{L}}$, $n_{\mathrm{N}} = n_{\mathrm{L}} - n_{\mathrm{P}}$, and $n_{\mathrm{U}} = 1000$, where n_{L} is the number of labeled samples. For the PNU-AUC optimization method, the squared loss function was used and the candidates of the combination parameter η were $\{-0.9, -0.8, \ldots, 0.9\}$. For the class-prior estimation, we used the energy distance minimization method (Kawakubo et al., 2016). The results of the estimated class-prior are summarized in Table 5.5.

For SSRankboost, the discount factor and the number of neighbors were chosen from $\{10^{-3},\ 10^{-2},\ 10^{-1}\}$ and $\{2,\ 3,\ \ldots,\ 7\}$, respectively. For sAUC-LS and OptAG, the regularization parameter for the entropy regularizer was chosen from $\{1,\ 10\}$. Furthermore,

[8] We used the code available at http://ama.liglab.fr/~amini/SSRankBoost/

[9] This method is equivalent to OptAG without a generative model, which only employs a discriminative model with the entropy minimization principle. To eliminate the adverse effect of the wrongly chosen generative model, we added this method for comparison.

TABLE 5.5: Average and standard error of the estimated class-prior over 50 trials on benchmark datasets in semi-supervised learning setting.

Dataset	n_{L}	$\theta_{\mathrm{P}} = 0.1$	$\theta_{\mathrm{P}} = 0.2$
Banana	50	0.12 (0.01)	0.21 (0.02)
($d = 2$)	100	0.10 (0.01)	0.20 (0.01)
skin_nonskin	50	0.11 (0.01)	0.21 (0.01)
($d = 3$)	100	0.10 (0.01)	0.20 (0.01)
cod-rna	50	0.12 (0.01)	0.22 (0.01)
($d = 8$)	100	0.12 (0.01)	0.21 (0.01)
Magic	50	0.09 (0.01)	0.17 (0.01)
($d = 10$)	100	0.07 (0.01)	0.20 (0.01)
Image	50	0.12 (0.01)	0.22 (0.01)
($d = 18$)	100	0.11 (0.01)	0.20 (0.01)
SUSY	50	0.10 (0.01)	0.20 (0.01)
($d = 18$)	100	0.10 (0.01)	0.19 (0.01)
Ringnorm	50	0.06 (0.00)	0.15 (0.00)
($d = 20$)	100	0.07 (0.00)	0.17 (0.00)
Twonorm	50	0.10 (0.00)	0.20 (0.00)
($d = 20$)	100	0.10 (0.00)	0.20 (0.00)
Waveform	50	0.11 (0.01)	0.20 (0.01)
($d = 21$)	100	0.09 (0.01)	0.19 (0.01)
covtype	50	0.09 (0.01)	0.20 (0.01)
($d = 54$)	100	0.09 (0.01)	0.18 (0.01)
phishing	50	0.10 (0.00)	0.20 (0.00)
($d = 68$)	100	0.10 (0.00)	0.20 (0.00)
a9a	50	0.10 (0.01)	0.20 (0.01)
($d = 83$)	100	0.10 (0.00)	0.21 (0.01)
mushrooms	50	0.10 (0.00)	0.20 (0.00)
($d = 112$)	100	0.10 (0.00)	0.20 (0.00)
USPS	50	0.10 (0.00)	0.20 (0.01)
($d = 241$)	100	0.09 (0.01)	0.19 (0.01)
w8a	50	0.10 (0.00)	0.19 (0.00)
($d = 300$)	100	0.09 (0.00)	0.20 (0.01)

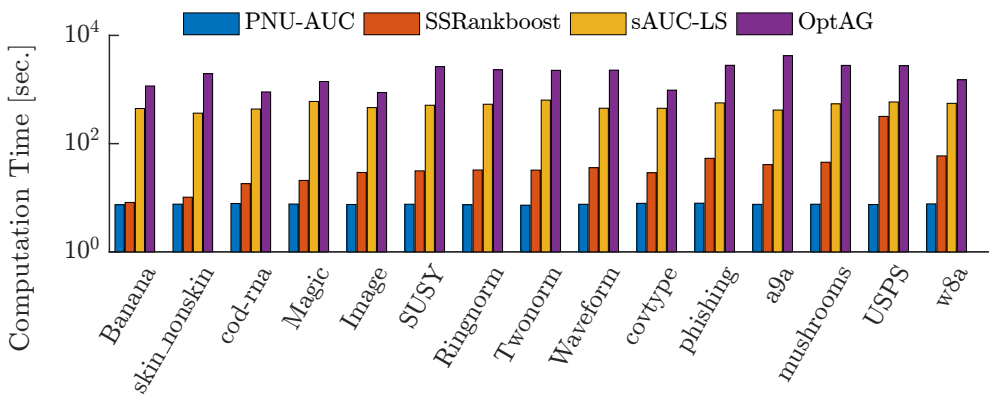


FIGURE 5.12: Average computation time of each method on benchmark datasets when $n_{\mathrm{L}} = 100$ and $\theta_{\mathrm{P}} = 0.1$ over 50 trials.

as the generative model of OptAG, we adapted the Gaussian distribution for the data distribution and the Gaussian and Gamma distributions for the prior of the data distribution.

Table 5.6 lists the average with standard error of the AUC over 50 trials, showing that the PNU-AUC optimization method achieves better performance than or comparable performance to the existing methods on many datasets. Figure 5.12 summarizes the average computation time over 50 trials. The computation time of the PNU-AUC optimization method includes both the class-prior estimation and the empirical risk minimization. The results show that even though the method involves the class-prior estimation, the computation time is relatively faster than SSRankboost and much faster than sAUC-LS and OptAG. The reason for longer computation time of sAUC-LS and OptAG is that their implementation is based on the logistic loss in which the number of operations for loss evaluation is $\mathcal{O}(n_{\mathrm{P}}n_{\mathrm{N}} + n_{\mathrm{P}}n_{\mathrm{U}} + n_{\mathrm{N}}n_{\mathrm{U}})$, unlike the PNU-AUC optimization method with the squared loss in which the number of operations for loss evaluation is $\mathcal{O}(n_{\mathrm{P}}+n_{\mathrm{N}}+n_{\mathrm{U}})$ (cf. the discussion about the computational complexity in Section 3.4).

5.7.3 Text Classification

Finally, we report the performance of the PNU-AUC optimization method on text classification tasks.

A classifier was trained with samples of size $n_{\mathrm{P}} = 20$, $n_{\mathrm{N}} = 80$, and $n_{\mathrm{U}} = 10{,}000$. The true class-prior was set at $\theta_{\mathrm{P}} = 0.2$ and estimated by the method based on energy distance minimization (Kawakubo et al., 2016). For the generative model of OptAG, we employed naive Bayes (NB) multinomial models and a Dirichlet prior for the prior distribution of the NB model as described in Fujino and Ueda (2016).

TABLE 5.6: Average and standard error of the AUC over 50 trials on benchmark datasets. The boldface denotes the best and comparable methods in terms of the average AUC according to the t-test at the significance level 5%. The bottom row shows the number of best/comparable cases of each method. SSRboost is an abbreviation for SSRankboost.

Dataset	n_{L}	$\theta_{\mathrm{P}} = 0.1$				$\theta_{\mathrm{P}} = 0.2$			
		PNU-AUC	SSRboost	sAUC-LS	OptAG	PNU-AUC	SSRboost	sAUC-LS	OptAG
Banana	50	**84.6 (1.3)**	61.6 (1.0)	**82.0 (1.3)**	**84.0 (1.3)**	**86.7 (0.9)**	66.1 (0.9)	83.3 (1.5)	**86.9 (0.7)**
($d = 2$)	100	**88.4 (0.7)**	66.2 (0.7)	84.0 (1.2)	**88.7 (0.5)**	**91.2 (0.4)**	69.1 (0.5)	88.5 (0.6)	89.9 (0.6)
skin_nonskin	50	**96.5 (0.9)**	92.2 (0.8)	96.4 (0.6)	**97.8 (0.6)**	**98.1 (0.6)**	93.7 (0.6)	**98.8 (0.2)**	**99.1 (0.2)**
($d = 3$)	100	98.8 (0.3)	94.5 (0.4)	99.0 (0.1)	**99.5 (0.0)**	**99.1 (0.3)**	95.6 (0.2)	**99.2 (0.3)**	**99.3 (0.2)**
cod-rna	50	**86.7 (1.4)**	79.4 (1.0)	61.6 (1.5)	63.3 (1.4)	**91.2 (1.0)**	86.8 (0.6)	66.0 (1.5)	68.7 (1.3)
($d = 8$)	100	**93.4 (0.6)**	88.7 (0.5)	67.3 (1.5)	67.7 (1.3)	**95.6 (0.5)**	91.7 (0.3)	74.1 (1.5)	75.9 (0.9)
Magic	50	**77.0 (1.1)**	74.8 (0.8)	73.7 (1.6)	**77.4 (0.7)**	**77.2 (1.3)**	**78.2 (0.5)**	76.5 (0.7)	75.4 (1.1)
($d = 10$)	100	**79.5 (0.5)**	**79.3 (0.6)**	74.5 (1.6)	77.1 (1.0)	**81.5 (0.4)**	**81.2 (0.4)**	75.8 (1.0)	78.3 (0.5)
Image	50	**81.6 (1.7)**	70.0 (1.1)	76.5 (1.6)	**80.7 (1.6)**	**86.1 (0.8)**	78.8 (0.6)	81.5 (1.0)	82.9 (1.1)
($d = 18$)	100	**88.3 (0.6)**	80.9 (0.7)	83.4 (1.2)	85.2 (1.0)	**92.1 (0.4)**	87.1 (0.5)	86.5 (0.6)	87.8 (0.4)
SUSY	50	63.0 (1.1)	**67.7 (1.3)**	56.2 (0.9)	56.2 (1.0)	64.7 (0.8)	**70.9 (0.8)**	55.6 (0.9)	57.7 (0.7)
($d = 18$)	100	65.4 (1.1)	**73.4 (0.5)**	59.0 (0.9)	59.0 (0.8)	69.4 (1.0)	**76.2 (0.4)**	58.9 (0.7)	58.3 (0.6)
Ringnorm	50	**98.3 (0.4)**	79.9 (0.7)	**98.7 (0.5)**	**99.0 (0.6)**	98.4 (0.4)	86.6 (0.4)	**99.5 (0.3)**	**99.1 (0.4)**
($d = 20$)	100	98.8 (0.3)	88.0 (0.4)	**99.8 (0.0)**	**99.8 (0.0)**	**99.4 (0.2)**	90.8 (0.3)	**99.5 (0.4)**	**99.6 (0.2)**
Twonorm	50	**96.9 (0.6)**	90.3 (0.4)	94.4 (1.1)	**96.1 (0.3)**	**97.5 (0.5)**	93.2 (0.3)	**97.7 (0.2)**	**97.4 (0.2)**
($d = 20$)	100	**98.6 (0.1)**	94.7 (0.2)	96.6 (0.2)	96.5 (0.2)	**99.0 (0.1)**	96.8 (0.1)	98.3 (0.1)	98.0 (0.2)
Waveform	50	**86.7 (1.3)**	**88.9 (0.4)**	85.5 (1.5)	85.8 (0.8)	**92.0 (0.6)**	**91.3 (0.2)**	88.6 (0.8)	89.4 (0.6)
($d = 21$)	100	**92.8 (0.4)**	91.8 (0.2)	87.8 (1.1)	87.7 (1.2)	**94.7 (0.2)**	93.1 (0.1)	90.1 (0.4)	89.7 (0.5)
covtype	50	57.8 (1.3)	**63.1 (1.0)**	55.7 (0.9)	58.9 (1.1)	60.3 (1.0)	**65.6 (0.8)**	56.4 (0.9)	57.8 (0.9)
($d = 54$)	100	60.7 (1.1)	**66.7 (0.8)**	59.1 (0.9)	60.3 (0.9)	64.2 (0.6)	**70.6 (0.6)**	57.7 (0.9)	60.6 (0.7)
phishing	50	89.8 (1.1)	**91.9 (0.6)**	71.1 (1.6)	74.0 (0.9)	91.8 (0.7)	**94.2 (0.3)**	74.7 (1.5)	77.1 (1.2)
($d = 68$)	100	**94.6 (0.2)**	**94.8 (0.2)**	76.3 (1.3)	76.9 (1.2)	94.9 (0.4)	**96.3 (0.1)**	80.5 (1.3)	82.0 (1.0)
a9a	50	69.4 (1.7)	**75.1 (0.9)**	**75.3 (1.4)**	**73.8 (1.4)**	**78.2 (1.0)**	**79.3 (0.5)**	**78.3 (1.1)**	**79.2 (0.7)**
($d = 83$)	100	77.9 (1.1)	**80.1 (0.5)**	**80.3 (0.5)**	**79.5 (0.8)**	82.0 (0.5)	**83.2 (0.3)**	81.5 (0.5)	81.5 (0.7)
mushrooms	50	96.0 (0.7)	96.0 (0.6)	96.6 (0.5)	**97.8 (0.3)**	96.4 (0.7)	**98.0 (0.2)**	**97.2 (0.5)**	**97.6 (0.8)**
($d = 112$)	100	**98.1 (0.5)**	98.2 (0.1)	97.9 (0.3)	**98.9 (0.2)**	**98.7 (0.2)**	**98.7 (0.1)**	98.2 (0.4)	**99.1 (0.4)**
USPS	50	**85.6 (1.0)**	73.9 (0.9)	79.5 (1.4)	82.7 (0.8)	**88.2 (0.7)**	80.5 (0.7)	81.2 (0.9)	85.1 (1.2)
($d = 241$)	100	**90.3 (0.5)**	79.4 (0.7)	84.2 (1.0)	86.0 (1.0)	**93.7 (0.5)**	85.2 (0.5)	83.3 (0.9)	89.7 (0.3)
w8a	50	**69.6 (1.1)**	**70.9 (0.9)**	52.5 (0.9)	52.9 (1.1)	**79.2 (0.9)**	75.4 (0.9)	52.6 (0.9)	55.2 (0.9)
($d = 300$)	100	**79.9 (1.1)**	77.1 (0.7)	53.1 (1.0)	53.9 (1.1)	**85.6 (0.6)**	81.0 (0.5)	55.7 (0.8)	56.2 (0.9)
#Best/Comp.		20	11	5	13	21	13	7	9

TABLE 5.7: Average with standard error of the AUC over 20 trials on the text classification datasets. The boldface denotes the best and comparable methods in terms of the average AUC according to the t-test at the significance level 5%.

Dataset	d	$\widehat{\theta}_{\mathrm{P}}$	PNU-AUC	SSRankboost	sAUC-LS	OptAG
rcv1	47236	0.21 (0.00)	**91.1 (0.7)**	76.6 (0.7)	79.6 (2.3)	79.4 (2.0)
amazon2	262144	0.21 (0.01)	**87.2 (1.3)**	68.7 (1.8)	55.9 (0.4)	56.8 (0.5)
news20	1355191	0.20 (0.00)	**79.8 (1.1)**	74.2 (2.0)	66.4 (1.6)	71.5 (1.2)

Table 5.7 lists the average with standard error of the AUC over 20 trials, showing that the PNU-AUC optimization method outperforms the existing methods. Figure 5.13 summarizes the average computation time of each method. These results show that the PNU-AUC optimization method achieves better performance with short computation time.

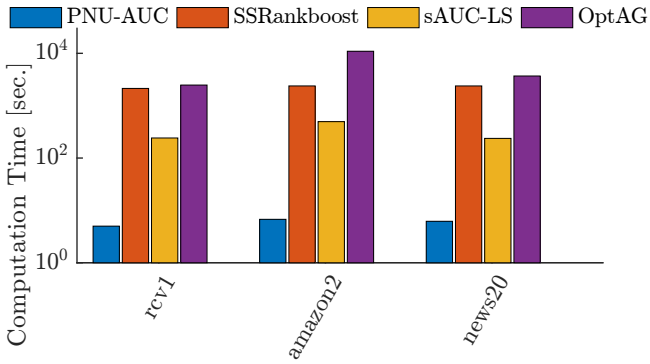


FIGURE 5.13: Average computation time of each method on the text classification datasets.

5.8 Proofs of Theorems

In this section, we give the proofs of Theorems in Section 5.5.

5.8.1 Proof of Theorem 13

To simplify the notation, let $R_{\mathrm{P}}(g)$, $R_{\mathrm{N}}(g)$, $R_{\mathrm{U,P}}(g)$, and $R_{\mathrm{U,N}}(g)$ be the risks of classifier g under loss ℓ:

$$R_{\mathrm{P}}(g) := \mathrm{E}_{\mathrm{P}}[\ell(g(\boldsymbol{x}))], \tag{5.95}$$

$$R_{\mathrm{N}}(g) := \mathrm{E}_{\mathrm{N}}[\ell(-g(\boldsymbol{x}))], \tag{5.96}$$

$$R_{\mathrm{U,P}}(g) := \mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}))], \tag{5.97}$$

$$R_{\mathrm{U,N}}(g) := \mathrm{E}_{\mathrm{U}}[\ell(-g(\boldsymbol{x}))], \tag{5.98}$$

Recall that

$$\begin{aligned} R^{\mathrm{MR},\gamma}_{\text{N-PUNU}}(g) &= (1-\gamma)R^{\mathrm{MR}}_{\text{N-PU}}(g) + \gamma R^{\mathrm{MR}}_{\text{N-NU}}(g) \\ &= (2-2\gamma)\theta_{\mathrm{P}}R_{\mathrm{P}}(g) + 2\gamma\theta_{\mathrm{N}}R_{\mathrm{N}}(g) \\ &\quad + (1-\gamma)R_{\mathrm{U,N}}(g) + \gamma R_{\mathrm{U,P}}(g) + \mathrm{Const}, \end{aligned} \tag{5.99}$$

$$\begin{aligned} R^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g) &= (1-\gamma)R^{\mathrm{MR}}_{\mathrm{PN}}(g) + \gamma R^{\mathrm{MR}}_{\text{N-PU}}(g) \\ &= (1+\gamma)\theta_{\mathrm{P}}R_{\mathrm{P}}(g) + (1-\gamma)\theta_{\mathrm{N}}R_{\mathrm{N}}(g) + \gamma R_{\mathrm{U,N}}(g) + \mathrm{Const}, \end{aligned} \tag{5.100}$$

$$\begin{aligned} R^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g) &= (1-\gamma)R^{\mathrm{MR}}_{\mathrm{PN}}(g) + \gamma R^{\mathrm{MR}}_{\text{N-NU}}(g) \\ &= (1-\gamma)\theta_{\mathrm{P}}R_{\mathrm{P}}(g) + (1+\gamma)\theta_{\mathrm{N}}R_{\mathrm{N}}(g) + \gamma R_{\mathrm{U,P}}(g) + \mathrm{Const}. \end{aligned} \tag{5.101}$$

Let $\widehat{R}_{\mathrm{P}}(g)$, $\widehat{R}_{\mathrm{N}}(g)$, $\widehat{R}_{\mathrm{U,P}}(g)$ and $\widehat{R}_{\mathrm{U,N}}(g)$ be the empirical risks. In order to prove Theorem 13, the following concentration lemma is needed:

Lemma 19. *For any $\delta > 0$, we have these uniform deviation bounds with probability at least $1 - \delta/3$:*

$$\sup_{g\in\mathcal{G}}(R_{\mathrm{P}}(g) - \widehat{R}_{\mathrm{P}}(g)) \le \frac{C_w C_\phi}{\sqrt{n_{\mathrm{P}}}} + \sqrt{\frac{\ln(3/\delta)}{2n_{\mathrm{P}}}}, \tag{5.102}$$

$$\sup_{g\in\mathcal{G}}(R_{\mathrm{N}}(g) - \widehat{R}_{\mathrm{N}}(g)) \le \frac{C_w C_\phi}{\sqrt{n_{\mathrm{N}}}} + \sqrt{\frac{\ln(3/\delta)}{2n_{\mathrm{N}}}}, \tag{5.103}$$

$$\sup_{g\in\mathcal{G}}(R_{\mathrm{U,P}}(g) - \widehat{R}_{\mathrm{U,P}}(g)) \le \frac{C_w C_\phi}{\sqrt{n_{\mathrm{U}}}} + \sqrt{\frac{\ln(3/\delta)}{2n_{\mathrm{U}}}}, \tag{5.104}$$

$$\sup_{g\in\mathcal{G}}(R_{\mathrm{U,N}}(g) - \widehat{R}_{\mathrm{U,N}}(g)) \le \frac{C_w C_\phi}{\sqrt{n_{\mathrm{U}}}} + \sqrt{\frac{\ln(3/\delta)}{2n_{\mathrm{U}}}}. \tag{5.105}$$

All inequalities in Lemma 19 are from the basic *uniform deviation bound* using the Rademacher complexity (Mohri et al., 2012), *Talagrand's contraction lemma* (Ledoux and Talagrand, 1991), as well as the fact that the Lipschitz constant of ℓ_{R} is $1/2$. For these reasons, the detailed proof of Lemma 19 is omitted.

Consider $R^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)$. It is clear that

$$\begin{aligned}\sup_{g\in\mathcal{G}}(R^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g) - \widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)) \leq{}& (1+\gamma)\theta_{\mathrm{P}} \sup_{g\in\mathcal{G}}(R_{\mathrm{P}}(g) - \widehat{R}_{\mathrm{P}}(g)) \\ &+ (1-\gamma)\theta_{\mathrm{N}} \sup_{g\in\mathcal{G}}(R_{\mathrm{N}}(g) - \widehat{R}_{\mathrm{N}}(g)) \\ &+ \gamma \sup_{g\in\mathcal{G}}(R_{\mathrm{U,N}}(g) - \widehat{R}_{\mathrm{U,N}}(g)). \end{aligned} \tag{5.106}$$

Therefore, by applying Lemma 19, for any $\delta > 0$, it holds with probability at least $1-\delta$ that

$$\sup_{g\in\mathcal{G}}(R^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g) - \widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)) \leq \frac{1}{2}c_1^{\mathrm{MR}}(\delta)\Big(\frac{(1+\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{(1-\gamma)\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{\gamma}{\sqrt{n_{\mathrm{U}}}}\Big). \tag{5.107}$$

Since $I^{\mathrm{MR}}(g) \leq 2R^{\mathrm{MR},\gamma}_{\text{N-PNPU}}$, with the same probability,

$$\sup_{g\in\mathcal{G}}(I(g) - 2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)) \leq c_1^{\mathrm{MR}}(\delta)\Big(\frac{(1+\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{(1-\gamma)\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{\gamma}{\sqrt{n_{\mathrm{U}}}}\Big). \tag{5.108}$$

Similarly,

$$\sup_{g\in\mathcal{G}}(I^{\mathrm{MR}}(g) - 2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)) \leq c_1^{\mathrm{MR}}(\delta)\Big(\frac{(1-\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{(1+\gamma)\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{\gamma}{\sqrt{n_{\mathrm{U}}}}\Big) \tag{5.109}$$

with probability at least $1-\delta$.

Finally, $R^{\mathrm{MR},\gamma}_{\text{N-PUNU}}(g)$ is slightly more involved, for that there are both $R_{\mathrm{U,P}}(g)$ and $R_{\mathrm{U,N}}(g)$. From $\ell_{\mathrm{R}}(m) + \ell_{\mathrm{R}}(-m) = 1$, we can know $R_{\mathrm{U,P}}(g) + R_{\mathrm{U,N}}(g) = 1$ and then

$$(1-\gamma)R_{\mathrm{U,N}}(g) + \gamma R_{\mathrm{U,P}}(g) = \begin{cases}(2\gamma-1)R_{\mathrm{U,P}}(g) + \mathrm{Const} & \gamma \geq 1/2, \\ (1-2\gamma)R_{\mathrm{U,N}}(g) + \mathrm{Const} & \gamma < 1/2.\end{cases} \tag{5.110}$$

As a result,

$$\sup_{g\in\mathcal{G}}(I^{\mathrm{MR}}(g) - 2\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PUNU}}(g)) \leq c_1^{\mathrm{MR}}(\delta)\Big(\frac{(2-2\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{2\gamma}{\sqrt{n_{\mathrm{N}}}} + \frac{|2\gamma-1|}{\sqrt{n_{\mathrm{U}}}}\Big) \tag{5.111}$$

with probability at least $1-\delta$.

5.8.2 Proof of Theorem 14

In fact,

$$\ell_{\mathrm{TS}}(m) = \begin{cases} 1/4 & m \le 0, \\ (m-1)^2/4 & 0 < m \le 1, \\ 0 & m > 1, \end{cases} \tag{5.112}$$

and after plugging this $\ell_{\mathrm{TS}}(m)$ into $\widetilde{\ell}_{\mathrm{TS}}(m)$,

$$\begin{aligned} \widetilde{\ell}_{\mathrm{TS}}(m) &= \ell_{\mathrm{TS}}(m) - \ell_{\mathrm{TS}}(-m) \\ &= \begin{cases} 1/4 & m \le -1, \\ 1/4 - (m+1)^2/4 & -1 < m \le 0, \\ (m-1)^2/4 - 1/4 & 0 < m \le 1, \\ -1/4 & m > 1. \end{cases} \end{aligned} \tag{5.113}$$

It is easy to see that $\ell_{\mathrm{TS}}(m)$ and $\widetilde{\ell}_{\mathrm{TS}}(m)$ are Lipschitz continuous with the same Lipschitz constant $1/2$.

Next, recall that

$$\begin{aligned} R^{\mathrm{MR},\gamma}_{\text{C-PUNU}}(g) &= (1-\gamma)R^{\mathrm{MR}}_{\text{C-PU}}(g) + \gamma R^{\mathrm{MR}}_{\text{C-NU}}(g) \\ &= (1-\gamma)\theta_{\mathrm{P}}R'_{\mathrm{P}}(g) + \gamma\theta_{\mathrm{N}}R'_{\mathrm{N}}(g) + (1-\gamma)R_{\mathrm{U,N}}(g) + \gamma R_{\mathrm{U,P}}(g), \end{aligned} \tag{5.114}$$

$$\begin{aligned} R^{\mathrm{MR},\gamma}_{\text{C-PNPU}}(g) &= (1-\gamma)R^{\mathrm{MR}}_{\mathrm{PN}}(g) + \gamma R^{\mathrm{MR}}_{\text{C-PU}}(g) \\ &= (1-\gamma)\theta_{\mathrm{P}}R_{\mathrm{P}}(g) + (1-\gamma)\theta_{\mathrm{N}}R_{\mathrm{N}}(g) + \gamma\theta_{\mathrm{P}}R'_{\mathrm{P}}(g) + \gamma R_{\mathrm{U,N}}(g), \end{aligned} \tag{5.115}$$

$$\begin{aligned} R^{\mathrm{MR},\gamma}_{\text{C-PNNU}}(g) &= (1-\gamma)R^{\mathrm{MR}}_{\mathrm{PN}}(g) + \gamma R^{\mathrm{MR}}_{\text{C-NU}}(g) \\ &= (1-\gamma)\theta_{\mathrm{P}}R_{\mathrm{P}}(g) + (1-\gamma)\theta_{\mathrm{N}}R_{\mathrm{N}}(g) + \gamma\theta_{\mathrm{N}}R'_{\mathrm{N}}(g) + \gamma R_{\mathrm{U,P}}(g). \end{aligned} \tag{5.116}$$

Let $\widehat{R}_{\mathrm{P}}(g)$, $\widehat{R}_{\mathrm{N}}(g)$, $\widehat{R}_{\mathrm{U,P}}(g)$, $\widehat{R}_{\mathrm{U,N}}(g)$, $\widehat{R}'_{\mathrm{P}}(g)$ and $\widehat{R}'_{\mathrm{N}}(g)$ be the empirical risks. Again, the following concentration lemma is needed:

Lemma 20. *For any $\delta > 0$, we have these uniform deviation bounds with probability at least $1 - \delta/4$:*

$$\sup_{g\in\mathcal{G}}(R_{\mathrm{P}}(g) - \widehat{R}_{\mathrm{P}}(g)) \leq \frac{C_w C_\phi}{\sqrt{n_{\mathrm{P}}}} + \sqrt{\frac{\ln(4/\delta)}{32 n_{\mathrm{P}}}}, \tag{5.117}$$

$$\sup_{g\in\mathcal{G}}(R_{\mathrm{N}}(g) - \widehat{R}_{\mathrm{N}}(g)) \leq \frac{C_w C_\phi}{\sqrt{n_{\mathrm{N}}}} + \sqrt{\frac{\ln(4/\delta)}{32 n_{\mathrm{N}}}}, \tag{5.118}$$

$$\sup_{g\in\mathcal{G}}(R_{\mathrm{U,P}}(g) - \widehat{R}_{\mathrm{U,P}}(g)) \leq \frac{C_w C_\phi}{\sqrt{n_{\mathrm{U}}}} + \sqrt{\frac{\ln(4/\delta)}{32 n_{\mathrm{U}}}}, \tag{5.119}$$

$$\sup_{g\in\mathcal{G}}(R_{\mathrm{U,N}}(g) - \widehat{R}_{\mathrm{U,N}}(g)) \leq \frac{C_w C_\phi}{\sqrt{n_{\mathrm{U}}}} + \sqrt{\frac{\ln(4/\delta)}{32 n_{\mathrm{U}}}}, \tag{5.120}$$

$$\sup_{g\in\mathcal{G}}(R'_{\mathrm{P}}(g) - \widehat{R}'_{\mathrm{P}}(g)) \leq \frac{C_w C_\phi}{\sqrt{n_{\mathrm{P}}}} + \sqrt{\frac{\ln(4/\delta)}{8 n_{\mathrm{P}}}}, \tag{5.121}$$

$$\sup_{g\in\mathcal{G}}(R'_{\mathrm{N}}(g) - \widehat{R}'_{\mathrm{N}}(g)) \leq \frac{C_w C_\phi}{\sqrt{n_{\mathrm{N}}}} + \sqrt{\frac{\ln(4/\delta)}{8 n_{\mathrm{N}}}}. \tag{5.122}$$

The detailed proof of Lemma 20 is omitted for the same reason as Lemma 19. The difference is due to that $0 \leq \ell_{\mathrm{TS}}(m) \leq 1/4$ and $-1/4 \leq \widetilde{\ell}_{\mathrm{TS}}(m) \leq 1/4$ whereas $0 \leq \ell_{\mathrm{R}}(m) \leq 1$ just like $0 \leq \ell_{\text{0-1}}(m) \leq 1$. For convenience, we will relax $1/32$ to $1/8$ in the square root for $R_{\mathrm{P}}(g)$, $R_{\mathrm{N}}(g)$, $R_{\mathrm{U,P}}(g)$, $R_{\mathrm{U,N}}(g)$.

Consider $R^{\mathrm{MR},\gamma}_{\text{C-PUNU}}(g)$. By applying Lemma 20, for any $\delta > 0$, it holds with probability at least $1 - \delta$ that

$$\sup_{g\in\mathcal{G}}(R^{\mathrm{MR},\gamma}_{\text{C-PUNU}}(g) - \widehat{R}^{\mathrm{MR},\gamma}_{\text{C-PUNU}}(g)) \leq \frac{1}{4} c_2^{\mathrm{MR}}(\delta)\Big(\frac{(1-\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{\gamma\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big). \tag{5.123}$$

Since $I(g) \leq 4R^{\mathrm{MR},\gamma}_{\text{C-PUNU}}$, with the same probability,

$$\sup_{g\in\mathcal{G}}(I^{\mathrm{MR}}(g) - 4\widehat{R}^{\mathrm{MR},\gamma}_{\text{C-PUNU}}(g)) \leq c_2^{\mathrm{MR}}(\delta)\Big(\frac{(1-\gamma)\theta_{\mathrm{P}}}{\sqrt{n_{\mathrm{P}}}} + \frac{\gamma\theta_{\mathrm{N}}}{\sqrt{n_{\mathrm{N}}}} + \frac{1}{\sqrt{n_{\mathrm{U}}}}\Big). \tag{5.124}$$

The other two generalization error bounds can be proven similarly.

5.8.3 Proof of Theorem 15

Next, we prove the generalization error bounds of the PNPU-AUC and PNNU-AUC risks in Theorem 15. The proof techniques are similar to Theorem 5 for PU-AUC optimization in Chapter 3. We first prove the following risk bounds:

Lemma 21. *For any $\delta > 0$, the following inequalities hold separately with probability at least $1-\delta$ for all $g \in \mathcal{G}$:*

$$R_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g) - \widehat{R}_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g) \leq c^{\mathrm{AUC}}(\delta/3)\Big(\frac{1-\gamma}{\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{N}})}} + \frac{\gamma}{\theta_{\mathrm{N}}\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})}} + \frac{\theta_{\mathrm{P}}\gamma}{\theta_{\mathrm{N}}\sqrt{n_{\mathrm{P}}}}\Big), \quad (5.125)$$

$$R_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}(g) - \widehat{R}_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}(g) \leq c^{\mathrm{AUC}}(\delta/3)\Big(\frac{1-\gamma}{\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{N}})}} + \frac{\gamma}{\theta_{\mathrm{P}}\sqrt{\min(n_{\mathrm{N}}, n_{\mathrm{U}})}} + \frac{\theta_{\mathrm{N}}\gamma}{\theta_{\mathrm{P}}\sqrt{n_{\mathrm{N}}}}\Big). \quad (5.126)$$

Proof. Recall the PNPU-AUC and PNNU-AUC risks:

$$R_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g) := (1-\gamma)R_{\mathrm{PN}}^{\mathrm{AUC}}(g) + \gamma R_{\mathrm{PU}}^{\mathrm{AUC}}(g), \quad (5.127)$$

$$R_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}(g) := (1-\gamma)R_{\mathrm{PN}}^{\mathrm{AUC}}(g) + \gamma R_{\mathrm{NU}}^{\mathrm{AUC}}(g). \quad (5.128)$$

Based on Theorem 6 in Chapter 3, for any $\delta > 0$, we have these uniform deviation bounds with probability at least $1 - \delta/3$:

$$\sup_{g \in \mathcal{G}} \Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] - \frac{1}{n_{\mathrm{P}} n_{\mathrm{N}}} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{j=1}^{n_{\mathrm{N}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_j^{\mathrm{N}})) \Big) \leq \frac{c^{\mathrm{AUC}}(\delta/3)}{\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{N}})}}, \tag{5.129}$$

$$\sup_{g \in \mathcal{G}} \Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] - \frac{1}{n_{\mathrm{P}} n_{\mathrm{U}}} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{k=1}^{n_{\mathrm{U}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\boldsymbol{x}_k^{\mathrm{U}})) \Big) \leq \frac{c^{\mathrm{AUC}}(\delta/3)}{\sqrt{\min(n_{\mathrm{P}}, n_{\mathrm{U}})}}, \tag{5.130}$$

$$\sup_{g \in \mathcal{G}} \Big(\mathrm{E}_{\mathrm{U}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{U}}) - g(\boldsymbol{x}^{\mathrm{N}}))]] - \frac{1}{n_{\mathrm{N}} n_{\mathrm{U}}} \sum_{k=1}^{n_{\mathrm{U}}} \sum_{j=1}^{n_{\mathrm{N}}} \ell(g(\boldsymbol{x}_k^{\mathrm{U}}) - g(\boldsymbol{x}_j^{\mathrm{N}})) \Big) \leq \frac{c^{\mathrm{AUC}}(\delta/3)}{\sqrt{\min(n_{\mathrm{N}}, n_{\mathrm{U}})}}, \tag{5.131}$$

$$\sup_{g \in \mathcal{G}} \Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] - \frac{1}{{n_{\mathrm{P}}}^2} \sum_{i=1}^{n_{\mathrm{P}}} \sum_{i'=1}^{n_{\mathrm{P}}} \ell(g(\boldsymbol{x}_i^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}_{i'}^{\mathrm{P}})) \Big) \leq \frac{c^{\mathrm{AUC}}(\delta/3)}{\sqrt{n_{\mathrm{P}}}}, \tag{5.132}$$

$$\sup_{g \in \mathcal{G}} \Big(\mathrm{E}_{\mathrm{N}}[\mathrm{E}_{\overline{\mathrm{N}}}[\ell(g(\boldsymbol{x}^{\mathrm{N}}) - g(\overline{\boldsymbol{x}}^{\mathrm{N}}))]] - \frac{1}{{n_{\mathrm{N}}}^2} \sum_{j=1}^{n_{\mathrm{N}}} \sum_{j'=1}^{n_{\mathrm{N}}} \ell(g(\boldsymbol{x}_j^{\mathrm{N}}) - g(\overline{\boldsymbol{x}}_{j'}^{\mathrm{N}})) \Big) \leq \frac{c^{\mathrm{AUC}}(\delta/3)}{\sqrt{n_{\mathrm{N}}}}. \tag{5.133}$$

Combining three bounds from the above, for any $\delta > 0$, with probability $1-\delta$, we have

$$
\begin{aligned}
&\sup_{g\in\mathcal{G}}\Big(R_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g) - \widehat{R}_{\mathrm{PNPU}}^{\mathrm{AUC},\gamma}(g)\Big)\\
&\leq (1-\gamma)\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{N}}[\ell(g(\boldsymbol{x}^{\mathrm{P}})-g(\boldsymbol{x}^{\mathrm{N}}))]] - \frac{1}{n_{\mathrm{P}}n_{\mathrm{N}}}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{j=1}^{n_{\mathrm{N}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}})-g(\boldsymbol{x}_j^{\mathrm{N}}))\Big)\\
&\quad+\frac{\gamma}{\theta_{\mathrm{N}}}\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}})-g(\boldsymbol{x}^{\mathrm{U}}))]] - \frac{1}{n_{\mathrm{P}}n_{\mathrm{U}}}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}})-g(\boldsymbol{x}_k^{\mathrm{U}}))\Big)\\
&\quad+\frac{\gamma\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}}\sup_{g\in\mathcal{G}}\Big(\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}})-g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] - \frac{1}{{n_{\mathrm{P}}}^2}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'=1}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}_i^{\mathrm{P}})-g(\overline{\boldsymbol{x}}_{i'}^{\mathrm{P}}))\Big)\\
&\leq c^{\mathrm{AUC}}(\delta/3)\Big(\frac{1-\gamma}{\sqrt{\min(n_{\mathrm{P}},n_{\mathrm{N}})}}+\frac{\gamma}{\theta_{\mathrm{N}}\sqrt{\min(n_{\mathrm{P}},n_{\mathrm{U}})}}+\frac{\gamma\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}\sqrt{n_{\mathrm{P}}}}\Big).
\end{aligned}
\tag{5.134}
$$

This concludes the risk bounds of the PNPU-AUC risk.

Similarly, we prove the risk bounds of the PNNU-AUC risk. □

Again, $I^{\mathrm{AUC}}(g) \leq R^{\mathrm{AUC}}(g)$ holds in our setting. This leads to Theorem 15.

5.8.4 Proofs of Theorems 16 and 17

Note that g is independent of the data for evaluating $\widehat{R}_{\text{N-PUNU}}^{\mathrm{MR},\gamma}(g)$, since it is fixed in the evaluation. Thus, $\mathrm{Var}_{\mathrm{P}}[\widehat{R}_{\mathrm{P}}(g)] = \sigma_{\mathrm{P}}^2(g)/n_{\mathrm{P}}$ and $\mathrm{Var}_{\mathrm{N}}[\widehat{R}_{\mathrm{N}}(g)] = \sigma_{\mathrm{N}}^2(g)/n_{\mathrm{N}}$, where $\widehat{R}_{\mathrm{P}}$ and $\widehat{R}_{\mathrm{N}}$ are the sample averages of $R_{\mathrm{P}}(g) = \mathrm{E}_{\mathrm{P}}[\ell(g(\boldsymbol{x}))]$ and $R_{\mathrm{N}}(g) = \mathrm{E}_{\mathrm{N}}[\ell(-g(\boldsymbol{x}))]$, respectively. When $n_{\mathrm{U}} \to \infty$,

$$
\begin{aligned}
\mathrm{Var}[\widehat{R}_{\text{N-PUNU}}^{\mathrm{MR},\gamma}(g)] &= 4(1-\gamma)^2\theta_{\mathrm{P}}^2\,\mathrm{Var}_{\mathrm{P}}[\widehat{R}_{\mathrm{P}}(g)] + 4\gamma^2\theta_{\mathrm{N}}^2\,\mathrm{Var}_{\mathrm{N}}[\widehat{R}_{\mathrm{N}}(g)]\\
&= 4(1-\gamma)^2\psi_{\mathrm{P}} + 4\gamma^2\psi_{\mathrm{N}}\\
&= 4(\psi_{\mathrm{P}}+\psi_{\mathrm{N}})\gamma^2 - 8\psi_{\mathrm{P}}\gamma + 4\psi_{\mathrm{P}},
\end{aligned}
\tag{5.135}
$$

and it is obvious that $\gamma_{\text{N-PUNU}}^{\mathrm{MR}} \in [0,1]$. All other claims in Theorem 16 follow from that $\mathrm{Var}[\widehat{R}_{\text{N-PUNU}}^{\mathrm{MR},\gamma}(g)]$ is quadratic in γ, that $\mathrm{Var}[\widehat{R}_{\text{N-PUNU}}^{\mathrm{MR},\gamma}(g)] = \mathrm{Var}[\widehat{R}_{\mathrm{PN}}^{\mathrm{MR}}(g)]$ at $\gamma = 1/2$, and that $\gamma_{\text{N-PUNU}}^{\mathrm{MR}} < 1/2$ if $\psi_{\mathrm{P}} < \psi_{\mathrm{N}}$ or $\gamma_{\text{N-PUNU}}^{\mathrm{MR}} > 1/2$ if $\psi_{\mathrm{P}} > \psi_{\mathrm{N}}$.

Likewise, when $n_{\mathrm{U}} \to \infty$,

$$\mathrm{Var}[\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNPU}}(g)] = (1+\gamma)^2\psi_{\mathrm{P}} + (1-\gamma)^2\psi_{\mathrm{N}}, \tag{5.136}$$

$$\mathrm{Var}[\widehat{R}^{\mathrm{MR},\gamma}_{\text{N-PNNU}}(g)] = (1-\gamma)^2\psi_{\mathrm{P}} + (1+\gamma)^2\psi_{\mathrm{N}}, \tag{5.137}$$

and $\gamma^{\mathrm{MR}}_{\text{N-PNPU}} \geq 0$ if $\psi_{\mathrm{P}} \leq \psi_{\mathrm{N}}$ or $\gamma^{\mathrm{MR}}_{\text{N-PNNU}} \geq 0$ if $\psi_{\mathrm{P}} \geq \psi_{\mathrm{N}}$. The rest of proof of Theorem 17 is analogous to that of Theorem 16.

5.8.5 Proof of Theorem 18

Proof. The empirical PNPU-AUC risk can be expressed as

$$\begin{aligned}
\widehat{R}^{\mathrm{AUC},\gamma}_{\mathrm{PNPU}}(g) &= (1-\gamma)\widehat{R}^{\mathrm{AUC}}_{\mathrm{PN}}(g) + \gamma\widehat{R}^{\mathrm{AUC}}_{\mathrm{PU}}(g) \\
&= \frac{1-\gamma}{n_{\mathrm{P}}n_{\mathrm{N}}}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{j=1}^{n_{\mathrm{N}}}\ell(g(\boldsymbol{x}^{\mathrm{P}}_i) - g(\boldsymbol{x}^{\mathrm{N}}_j)) \\
&\quad + \frac{\gamma}{\theta_{\mathrm{N}}n_{\mathrm{P}}n_{\mathrm{U}}}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{k=1}^{n_{\mathrm{U}}}\ell(g(\boldsymbol{x}^{\mathrm{P}}_i) - g(\boldsymbol{x}^{\mathrm{U}}_k))]] \\
&\quad - \frac{\gamma\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}{n_{\mathrm{P}}}^2}\sum_{i=1}^{n_{\mathrm{P}}}\sum_{i'=1}^{n_{\mathrm{P}}}\ell(g(\boldsymbol{x}^{\mathrm{P}}_i) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}_{i'}))]].
\end{aligned} \tag{5.138}$$

Assume $n_{\mathrm{U}} \to \infty$, we obtain

$$\begin{aligned}
\mathrm{Var}[\widehat{R}^{\mathrm{AUC},\gamma}_{\mathrm{PNPU}}(g)] &= \frac{(1-\gamma)^2}{n_{\mathrm{P}}n_{\mathrm{N}}}\tau^2_{\mathrm{PN}}(g) + \frac{\gamma^2\theta^2_{\mathrm{P}}}{{n_{\mathrm{P}}}^2\theta^2_{\mathrm{N}}}\tau^2_{\mathrm{PP}}(g) + \frac{(1-\gamma)\gamma}{\theta_{\mathrm{N}}n_{\mathrm{P}}}\tau_{\mathrm{PN,PU}}(g) \\
&\quad - \frac{\gamma^2\theta_{\mathrm{P}}}{\theta^2_{\mathrm{N}}n_{\mathrm{P}}}\tau_{\mathrm{PU,PP}}(g) - \frac{(1-\gamma)\gamma\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}n_{\mathrm{P}}}\tau_{\mathrm{PN,PP}}(g) \\
&= (1-\gamma)^2\overline{\psi}_{\mathrm{PN}} + \gamma^2\overline{\psi}_{\mathrm{PU}} + (1-\gamma)\gamma\overline{\psi}_{\mathrm{PP}},
\end{aligned} \tag{5.139}$$

where the terms divided by n_{U} are disappeared. Setting the derivative with respect to γ at zero, we obtain the minimizer in Eq. (5.85).

For the empirical PNNU-AUC risk, when $n_{\mathrm{U}} \to \infty$, we obtain

$$\begin{aligned}\mathrm{Var}[\widehat{R}_{\mathrm{PNNU}}^{\mathrm{AUC},\gamma}(g)] &= \frac{(1-\gamma)^2}{n_{\mathrm{P}} n_{\mathrm{N}}} \tau_{\mathrm{PN}}^2(g) + \frac{\gamma^2 \theta_{\mathrm{N}}^2}{{n_{\mathrm{N}}}^2 \theta_{\mathrm{P}}^2} \tau_{\mathrm{NN}}^2(g) + \frac{(1-\gamma)\gamma}{\theta_{\mathrm{P}} n_{\mathrm{N}}} \tau_{\mathrm{PN,NU}}(g) \\ &\quad - \frac{\gamma^2 \theta_{\mathrm{N}}}{\theta_{\mathrm{P}}^2 n_{\mathrm{N}}} \tau_{\mathrm{NU,NN}}(g) - \frac{(1-\gamma)\gamma\theta_{\mathrm{N}}}{\theta_{\mathrm{P}} n_{\mathrm{N}}} \tau_{\mathrm{PN,NN}}(g) \\ &= (1-\gamma)^2 \overline{\psi}_{\mathrm{PN}} + \gamma^2 \overline{\psi}_{\mathrm{NU}} + (1-\gamma)\gamma \overline{\psi}_{\mathrm{NN}}. \end{aligned} \tag{5.140}$$

Setting the derivative with respect to γ at zero, we obtain the minimizer in Eq. (5.86). □

Chapter 6

Conclusions and Future Work

In this chapter, we present conclusions and discuss several future perspectives of this research.

6.1 Conclusions

This dissertation was devoted to expanding the possibility of semi-supervised learning.

In Chapter 3, we considered imbalanced classification from positive-unlabeled (PU) learning scenario. The problem was solved by directly maximizing AUC with respect to a classifier. We presented the risk for AUC maximization in PU learning, revealing that the existing AUC maximization method from PU data is biased. The generalization error bound was proved without assuming distributional assumptions on the data. Experimental results reported the effectiveness of our method.

In Chapter 4, we developed statistical dependency analysis tools based on squared-loss mutual information (SMI) from PU data. We proposed an SMI estimation method from only PU data and proved its optimal convergence to true SMI. The SMI estimator was applied to dimension reduction and independence test. In particular, our dimension reduction can be executed without conducting a class-prior estimation in advance, which is a significant advantage in practice. We experimentally confirmed the usefulness of the SMI-based statistical dependency analysis methods.

In Chapter 5, we explored a semi-supervised learning approach based on positive-unlabeled learning. The approach allows us to leverage unlabeled data without conventional distributional assumptions such as the cluster assumption. We provided two realizations of the risks in semi-supervised learning; the misclassification rate and AUC as the classification measures. We elucidated properties of the risks and showed the generalization error bounds and the variance reduction of the empirical risks. Finally, extensive experimental results confirmed the effectiveness of our approach.

6.2 Future Work

Finally, we discuss future directions of this dissertation.

6.2.1 Semi-Supervised SMI Estimation

The first future direction is to extend the SMI estimation from PU data to SMI estimation from PNU data.

A simple way is to combine the PN-SMI risk with the PU-SMI or NU-SMI risks, where the NU-SMI risk can be defined as:

$$R_{\mathrm{NU}}^{\mathrm{SMI}}(g) = \frac{1}{2}\int \Big(g(\boldsymbol{x}) - \frac{p(\boldsymbol{x} \mid y=-1)}{p(\boldsymbol{x})}\Big)^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} - C_{\mathrm{NU}}, \tag{6.1}$$

where $C_{\mathrm{NU}} := \frac{1}{2}\int \frac{p^2(\boldsymbol{x}|y=-1)}{p(\boldsymbol{x})}\mathrm{d}\boldsymbol{x}$. Along with the line of PNU learning, we minimize the PNU-SMI risk defined as

$$R_{\mathrm{PNU}}^{\mathrm{SMI},\eta}(g) = \begin{cases} (1-\eta)R_{\mathrm{PN}}^{\mathrm{SMI}}(g) + \eta R_{\mathrm{PU}}^{\mathrm{SMI}}(g) & (\eta \geq 0), \\ (1+\eta)R_{\mathrm{PN}}^{\mathrm{SMI}}(g) - \eta R_{\mathrm{NU}}^{\mathrm{SMI}}(g) & (\eta < 0), \end{cases} \tag{6.2}$$

where $-1 \leq \eta \leq 1$. Then, with the learned model $g^* = \underset{g}{\mathrm{argmin}}\ R_{\mathrm{PNU}}^{\mathrm{SMI},\eta}$, we can compute SMI by plugging the model into the following definition of SMI:

$$\mathrm{SMI} = \begin{cases} (1-\eta)\mathrm{PN\text{-}SMI} + \eta\mathrm{PU\text{-}SMI} & (\eta \geq 0), \\ (1+\eta)\mathrm{PN\text{-}SMI} - \eta\mathrm{NU\text{-}SMI} & (\eta < 0), \end{cases} \tag{6.3}$$

where $\mathrm{PN\text{-}SMI}$ is the expression of SMI in Eq. (4.3) and $\mathrm{NU\text{-}SMI}$ is defined as

$$\mathrm{NU\text{-}SMI} := \frac{\theta_{\mathrm{N}}}{2\theta_{\mathrm{P}}}\int \Big(\frac{p(\boldsymbol{x} \mid y=-1)}{p(\boldsymbol{x})} - 1\Big)^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}. \tag{6.4}$$

Alternatively, for SMI estimation, we can simply use unlabeled data even under the supervised learning setting unlike classification tasks. Recall the PN-SMI risk in Eq. (2.97):

$$R_{\mathrm{PN}}^{\mathrm{SMI}}(q) = \frac{1}{2}\sum_{y=\pm 1}\int q^2(\boldsymbol{x}, y)p(\boldsymbol{x})p(y)\mathrm{d}\boldsymbol{x} - \sum_{y=\pm 1}\int q(\boldsymbol{x}, y)p(\boldsymbol{x}, y)\mathrm{d}\boldsymbol{x}. \tag{6.5}$$

To approximate the PN-SMI risk, the original SMI estimation method uses the obtained labeled sample for computing sample averages regarding $p(\boldsymbol{x}, y)$, $p(\boldsymbol{x})$, and $p(y)$. However, for computing sample averages regarding $p(\boldsymbol{x})$ instead of marginalized labeled samples, we can use additional unlabeled samples,[1] meaning that this procedure is SMI estimation from labeled and unlabeled data.

Since both two approaches do not assume any strong distributional assumption for utilizing unlabeled data for SMI estimation, our future work includes comparing two approaches from both empirical and theoretical viewpoints.

6.2.2 AUC Optimization from PU Data without Class-Priors

Recall that the PU-AUC risk is expressed as

$$R_{\mathrm{PU}}^{\mathrm{AUC}}(g) = \frac{1}{\theta_{\mathrm{N}}} \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]] - \frac{\theta_{\mathrm{P}}}{\theta_{\mathrm{N}}} \mathrm{E}_{\mathrm{P}} \mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]. \tag{6.6}$$

The key observation is that the second term becomes constant with loss functions satisfying

$$\ell(m) + \ell(-m) = C, \tag{6.7}$$

where C is a constant. Since $\boldsymbol{x}^{\mathrm{P}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x} \mid y = +1)$ and $\overline{\boldsymbol{x}}^{\mathrm{P}} \overset{\text{i.i.d.}}{\sim} p(\boldsymbol{x} \mid y = +1)$ are interchangeable, we have

$$\begin{aligned}
&\mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] \\
&= \frac{1}{2} \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] + \frac{1}{2} \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\overline{\boldsymbol{x}}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{P}}))]] \\
&= \frac{1}{2} \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}}))]] + \frac{1}{2} \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(-(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}})))]] \\
&= \frac{1}{2} \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\overline{\mathrm{P}}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}})) + \ell(-(g(\boldsymbol{x}^{\mathrm{P}}) - g(\overline{\boldsymbol{x}}^{\mathrm{P}})))]] \\
&= \frac{C}{2}.
\end{aligned} \tag{6.8}$$

[1] This is because $p(\boldsymbol{x}) = \sum_y p(\boldsymbol{x}, y)$.

In the risk minimization, we can ignore the constant and the scaling parameter $1/\theta_{\mathrm{N}}$. Thus, we can show the following relation:

$$\begin{aligned} g^* &= \underset{g}{\operatorname{argmin}}\ R_{\mathrm{PU}}^{\mathrm{AUC}}(g) \\ &= \underset{g}{\operatorname{argmin}}\ \mathrm{E}_{\mathrm{P}}[\mathrm{E}_{\mathrm{U}}[\ell(g(\boldsymbol{x}^{\mathrm{P}}) - g(\boldsymbol{x}^{\mathrm{U}}))]]. \end{aligned} \tag{6.9}$$

The advantage of Eq. (6.9) is that a classifier can be obtained without resorting to the class-prior estimation, meaning that the empirical risk minimization is not affected by the accuracy of the estimated class-prior. This idea would be an undoubtedly interesting future direction of the practical PU learning method for imbalanced classification.

6.2.3 Multi-Class Extension of Semi-Supervised AUC Optimization

Next, we discuss a multi-class extension of semi-supervised AUC optimization. The idea is based on the AUC optimization method from PU data without a class-prior estimation discussed in the previous section.

Let

$$\mathcal{Y} := \{1, \ldots, c\} \tag{6.10}$$

be the domain of class labels, where c is the number of classes. Furthermore, let us define classifiers for each class as

$$G := \{g_y \mid g_y \colon \mathbb{R}^d \to \mathbb{R},\ \sum_{y'=1}^{c} g_{y'} = 1,\ \ y = 1, \ldots, c\}. \tag{6.11}$$

Let us define the risk in the multi-class version of supervised AUC optimization as

$$R_{\mathrm{MC}}^{\mathrm{AUC}}(G) := \frac{1}{|\mathcal{Y}|} \sum_{y} \sum_{y' \neq y} \theta_{y'} \int \ell(g_y(\boldsymbol{x}) - g_y(\boldsymbol{x}'))p(\boldsymbol{x} \mid y)p(\boldsymbol{x}' \mid y')\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}'. \tag{6.12}$$

Then, the risk in semi-supervised multi-class AUC optimization is given by

$$R_{\mathrm{SSL}}^{\mathrm{AUC}}(G) := \frac{1}{|\mathcal{Y}|} \sum_{y} \int \ell(g_y(\boldsymbol{x}) - g_y(\boldsymbol{x}'))p(\boldsymbol{x} \mid y)p(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}'. \tag{6.13}$$

Similarly to the calculation in the binary case shown in the previous section, with the loss functions satisfying $\ell(m) + \ell(-m) = C$, we can reformulate $R_{\mathrm{SSL}}^{\mathrm{AUC}}$ as

$$\begin{aligned} R_{\mathrm{SSL}}^{\mathrm{AUC}}(G) &= \frac{C}{2|\mathcal{Y}|} + \frac{1}{|\mathcal{Y}|} \sum_{y} \sum_{y' \neq y} \theta_{y'} \int \ell(g_y(\boldsymbol{x}) - g_y(\boldsymbol{x}'))p(\boldsymbol{x} \mid y)p(\boldsymbol{x}' \mid y')\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}' \\ &= \frac{C}{2|\mathcal{Y}|} + R_{\mathrm{MC}}^{\mathrm{AUC}}(G). \end{aligned} \tag{6.14}$$

Again, the first term is constant; the minimization of $R_{\mathrm{SSL}}^{\mathrm{AUC}}$ corresponds to the minimization of the second term, i.e., the risk in supervised multi-class AUC optimization $R_{\mathrm{MC}}^{\mathrm{AUC}}$. Therefore, we can obtain the classifiers from labeled and unlabeled data. Investigating theoretical and empirical properties of this multi-class semi-supervised AUC optimization risk is also an important future direction.

6.2.4 Essential Mechanism of Extracting Information from Data

The final future direction is rather abstract, but it would be an important research question. The question is revealing a mechanism of extracting information from unlabeled data.

To achieve better performance from limited labeled data by utilizing a large amount of unlabeled data, we need a carefully-designed algorithm to extract information hidden in the data. As discussed in this dissertation several times, the conventional assumption such as the low-density separation principle does not always work well. For overcoming the issue, this dissertation was devoted to the method utilizing unlabeled data without such a distributional assumption. However, we are still on the way of practical semi-supervised learning methods and feel that we are extracting an only small portion of information from the data. Note that achieving $1/\sqrt{n_U}$ is optimal in terms of statistical estimation. Towards making machine learning methods more practical, we need general and yet powerful semi-supervised learning methods.

To pursue this research question, there is no clear direction yet, but several recent advances in machine learning might become a clue and shed light in a part of our ultimate goal. There are some interesting studies, e.g., influence functions for analyzing black-box models (Koh and Liang, 2017) such as a deep neural network model, and iterative machine teaching for obtaining a better model more quickly (Liu et al., 2017). The key of these studies is to analyze the loss function and utilize the property of the loss for their purposes. In this dissertation, our focus was the form of risk functions. To go one step further for more practical semi-supervised learning, analysis of loss functions would

be a current possible approach for revealing informative data points in semi-supervised learning scenario and indicating a way of extracting more information from unlabeled data.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from https://www.tensorflow.org/.

Ali, S. M. and Slivey, S. D. (1966). A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society. Series B*, 28:131–152.

Amini, M. R., Truong, T. V., and Goutte, C. (2008). A boosting algorithm for learning bipartite ranking functions with partially labeled data. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 99–106.

Bartlett, P. L., Jordan, M. I., and McAuliffe, J. D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156.

Basu, A., Harris, I. R., Hjort, N. L., and Jones, M. C. (1998). Robust and efficient estimation by minimising a density power divergence. *Biometrika*, 85(3):549–559.

Belkin, M., Niyogi, P., and Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434.

Blanchard, G., Lee, G., and Scott, C. (2010). Semi-supervised novelty detection. *Journal of Machine Learning Research*, 11:2973–3009.

Blondel, M., Seki, K., and Uehara, K. (2013). Block coordinate descent algorithms for large-scale sparse multiclass classification. *Machine Learning*, 93(1):31–52.

Bonnans, J. F. and Cominetti, R. (1996). Perturbed optimization in Banach spaces I: A general theory based on a weak directional constraint qualificationn; II: A theory based on a strong directional qualification condition; III: Semiinfinite optimization. *SIAM Journal on Control and Optimization*, 34(4):1151–1171, 1172–1189, and 1555–1567.

Bonnans, J. F. and Shapiro, A. (1998). Optimization problems with perturbations: A guided tour. *SIAM Review*, 40(2):228–264.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, New York, NY, USA.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chapelle, O., Schölkopf, B., and Zien, A., editors (2006). *Semi-Supervised Learning*. MIT Press.

Chapelle, O., Sindhwani, V., and Keerthi, S. S. (2008). Optimization techniques for semi-supervised support vector machines. *Journal of Machine Learning Research*, 9:203–233.

Chapelle, O., Weston, J., and Schôlkopf, B. (2002). Cluster kernels for semi-supervised learning. In *Advances in Neural Information Processing Systems 15*, pages 585–592.

Chapelle, O. and Zien, A. (2005). Semi-supervised classification by low density separation. In *AISTATS*, pages 57–64.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*.

Collobert, R., Sinz, F., Weston, J., and Bottou, L. (2006). Trading convexity for scalability. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 201–208.

Cortes, C. and Mohri, M. (2003). AUC optimization vs. error rate minimization. In *Advances in Neural Information Processing Systems 16*, pages 313–320.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory*. John Wiley & Sons, Inc., Hoboken, NJ, USA, 2nd edition.

Cozman, F. G., Cohen, I., and Cirelo, M. C. (2003). Semi-supervised learning of mixture models. In *Proceedings of the 20th International Conference on Machine Learning*, pages 99–106.

Csizár, I. (1967). Information-type measures of difference of probability dhdistributions and indirect observation. *Studia Scientiarum Mathematicarum Hungarica*, 2:229–318.

Dredze, M., Crammer, K., and Pereira, F. (2008). Confidence-weighted linear classification. In *Proceedings of the 25th International Conference on Machine learning*, pages 264–271.

du Plessis, M. C., Niu, G., and Sugiyama, M. (2014). Analysis of learning from positive and unlabeled data. In *Advances in Neural Information Processing Systems 27*, pages 703–711.

du Plessis, M. C., Niu, G., and Sugiyama, M. (2015). Convex formulation for learning from positive and unlabeled data. In *Proceedings of 32nd International Conference on Machine Learning*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1386–1394.

du Plessis, M. C., Niu, G., and Sugiyama, M. (2017). Class-prior estimation for learning from positive and unlabeled data. *Machine Learning*, 106(4):463–492.

du Plessis, M. C. and Sugiyama, M. (2014). Class prior estimation from positive and unlabeled data. *IEICE Transactions on Information and Systems*, E97-D(5):1358–1362.

Efron, B. and Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC Press.

Elkan, C. and Noto, K. (2008). Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 213–220.

Fujino, A. and Ueda, N. (2016). A semi-supervised AUC optimization method with generative models. In *IEEE 16th International Conference on Data Mining*, pages 883–888.

Gao, W., Wang, L., Jin, R., Zhu, S., and Zhou, Z.-H. (2016). One-pass AUC optimization. *Artificial Intelligence*, 236(C):1–29.

Gao, W. and Zhou, Z.-H. (2015). On the consistency of AUC pairwise optimization. In *International Joint Conference on Artificial Intelligence*, pages 939–945.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Grandvalet, Y. and Bengio, Y. (2004). Semi-supervised learning by entropy minimization. In *Advances in Neural Information Processing Systems 17*, pages 529–536.

Grandvalet, Y. and Bengio, Y. (2006). Entropy regularization. In Chapelle, O., Schôlkopf, B., and Zien, A., editors, *Semi-Supervised Learning*. MIT Press, Cambridge Massachusetts.

Hanley, J. A. and McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer-Verlag, New York, 2 edition.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Herschtal, A. and Raskutti, B. (2004). Optimising area under the ROC curve using gradient descent. In *Proceedings of the 21st International Conference on Machine Learning*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456.

Jain, S., White, M., and Radivojac, P. (2016). Estimating the class prior and posterior from noisy positives and unlabeled data. In *Advances in Neural Information Processing Systems 29*.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM International Cconference on Multimedia*, pages 675–678. ACM.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142.

Kanamori, T., Hido, S., and Sugiyama, M. (2009). A least-squares approach to direct importance estimation. *Journal of Machine Learning Research*, 10:1391–1445.

Kanamori, T., Suzuki, T., and Sugiyama, M. (2012). Statistical analysis of kernel-based least-squares density-ratio estimation. *Machine Learning*, 86(3):335–367.

Kawakubo, H., du Plessis, M. C., and Sugiyama, M. (2016). Computationally efficient class-prior estimation under class balance change using energy distance. *IEICE Transactions on Information and Systems*, E99-D(1):176–186.

Kelley, Jr, J. E. (1960). The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712.

Keziou, A. (2003). Dual representation of φ-divergences and applications. *Comptes Rendus Mathématique*, 336(10):857–862.

Koh, P. W. and Liang, P. (2017). Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1885–1894.

Kotlowski, W., Dembczynski, K. J., and Huellermeier, E. (2011). Bipartite ranking through minimization of univariate loss. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1113–1120.

Krause, A., Perona, P., and Gomes, R. G. (2010). Discriminative clustering by regularized information maximization. In *Advances in Neural Information Processing Systems*, pages 775–783.

Krijthe, J. H. and Loog, M. (2017). Robust semi-supervised least squares classification by implicit constraints. *Pattern Recognition*, 63:115–126.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105.

Lang, K. (1995). NewsWeeder: Learning to filter netnews. In *In Proceedings of the 12th International Machine Learning Conference*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Ledoux, M. and Talagrand, M. (1991). *Probability in Banach Spaces: Isoperimetry and Processes*. Springer.

Lee, W. S. and Liu, B. (2003). Learning with positive and unlabeled examples using weighted logistic regression. In *Proceedings of the 20th International Conference on Machine Learning*, pages 448–455.

Letouzey, F., Denis, F., and Gilleron, R. (2000). Learning from positive and unlabeled examples. In *Proceedings of the 11th International Conference on Algorithmic Learning Theory*, pages 71–85, Berlin, Heidelberg. Springer Berlin Heidelberg.

Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004). RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397.

Li, K.-C. (1991). Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327.

Li, X.-L. and Liu, B. (2005). Learning from positive and unlabeled examples with different data distributions. In *Proceedings of 16th European Conference on Machine Learning*, pages 218–229.

Li, Y.-F., Tsang, I. W., Kwok, J. T., and Zhou, Z.-H. (2013). Convex and scalable weakly labeled SVMs. *Journal of Machine Learning Research*, 14(1):2151–2188.

Li, Y.-F. and Zhou, Z.-H. (2015). Towards making unlabeled data never hurt. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1):175–188.

Lichman, M. (2013). UCI machine learning repository.

Linsker, R. (1988). Self-organization in a perceptual network. *Computer*, 21(3):105–117.

Liu, B., Dai, Y., Li, X., Lee, W. S., and Yu, P. S. (2003). Building text classifiers using positive and unlabeled examples. In *Third IEEE International Conference on Data Mining*, pages 179–186.

Liu, W., Dai, B., Humayun, A., Tay, C., Yu, C., Smith, L. B., Rehg, J. M., and Song, L. (2017). Iterative machine teaching. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 2149–2158.

Loog, M. (2016). Contrastive pessimistic likelihood estimation for semi-supervised classification. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 38(3):462–475.

Melacci, S. and Belkin, M. (2011). Laplacian support vector machines trained in the primal. *Journal of Machine Learning Research*, 12:1149–1184.

Mendelson, S. (2008). Lower bounds for the empirical minimization algorithm. *IEEE Transactions on Information Theory*, 54(8):3797–3803.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2012). *Foundations of Machine Learning*. MIT Press.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Nguyen, X. L., Wainwright, M. J., and Jordan, M. I. (2007). Nonparametric estimation of the likelihood ratio and divergence functionals. In *IEEE International Symposium on Information Theory*, pages 2016–2020.

Niu, G., du Plessis, M. C., Sakai, T., Ma, Y., and Sugiyama, M. (2016). Theoretical comparisons of positive-unlabeled learning against positive-negative learning. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 1199–1207.

Niu, G., Jitkrittum, W., Dai, B., Hachiya, H., and Sugiyama, M. (2013). Squared-loss mutual information regularization: A novel information-theoretic approach to semi-supervised learning. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pages 10–18.

Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer.

Pearson, K. (1990). On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arise from random sampling. *Philosophical Magazine Series 5*, 50(302):157–175.

Quiñonero Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. (2009). *When Training and Test Sets Are Different: Characterizing Learning Transfer*, pages 3–28. MIT Press.

Rakhlin, A., Shamir, O., and Sridharan, K. (2012). Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on Machine Learning*, pages 449–456.

Ramaswamy, H. G., Scott, C., and Tewari, A. (2016). Mixture proportion estimation via kernel embedding of distributions. In *Proceedings of the 33rd International Conference on Machine Learning*.

Rätsch, G., Onoda, T., and Müller, K.-R. (2001). Soft margins for adaboost. *Machine learning*, 42(3):287–320.

Rigollet, P. (2007). Generalization error bounds in semi-supervised classification under the cluster assumption. *Journal of Machine Learning Research*, 8:1369–1392.

Sakai, T. and Sugiyama, M. (2014). Computationally efficient estimation of squared-loss mutual information with multiplicative kernel models. *IEICE Transactions on Information and Systems*, E97-D(4):968–971.

Sakai, T., Sugiyama, M., Kitagawa, K., and Suzuki, K. (2015). Registration of infrared transmission images using squared-loss mutual information. *Precision Engineering*, 39:187–193.

Schölkopf, B. and Smola, A. J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Sechidis, K., Calvo, B., and Brown, G. (2014). Statistical hypothesis testing in positive unlabelled data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 66–81.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA.

Shannon, C. (1948). A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423.

Singh, S. (2000). *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*. Anchor.

Sokolovska, N., Cappé, O., and Yvon, F. (2008). The asymptotics of semi-supervised learning in discriminative probabilistic models. In *Proceedings of the 25th International Conference on Machine Learning*, pages 984–991.

Sugiyama, M. (2013). Machine learning with squared-loss mutual information. *Entropy*, 15:80–112.

Sugiyama, M., Niu, G., Yamada, M., Kimura, M., and Hachiya, H. (2014). Information-maximization clustering based on squared-loss mutual information. *Neural Computation*, 26(1):84–131.

Sugiyama, M. and Suzuki, T. (2011). Least-squares independence test. *IEICE Transactions on Information and Systems*, E94-D:1333–1336.

Sundararajan, S., Priyanka, G., and Selvaraj, Keerthi, S. S. (2011). A pairwise ranking based approach to learning with positive and unlabeled examples. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 663–672.

Suzuki, T. and Sugiyama, M. (2013). Sufficient dimension reduction via squared-loss mutual information. *Neural Computation*, 25(3):725–758.

Suzuki, T., Sugiyama, M., Kanamori, T., and Sese, J. (2009). Mutual information estimation reveals global associations between stimuli and biological processes. *BMC Bioinformatics*, 10:S52:1–12.

Székely, G. J. and Rizzo, M. L. (2013). Energy statistics: A class of statistics based on distances. *Journal of Statistical Planning and Inference*, 143(8):1249–1272.

Tokui, S., Oono, K., Hido, S., and Clayton, J. (2015). Chainer: A next-generation open source framework for deep learning. In *Proceedings of Workshop on Machine Learning Systems in The Twenty-ninth Annual Conference on Neural Information Processing Systems*.

Torkkola, K. (2003). Feature extraction by non-parametric mutual information maximization. *Journal of Machine Learning Research*, 3:1415–1438.

Torralba, A., Fergus, R., and Freeman, W. T. (2008). 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970.

Usunier, N., Amini, M., and Patrick, G. (2006). Generalization error bounds for classifiers trained with interdependent data. In Weiss, Y., Schölkopf, P. B., and Platt, J. C., editors, *Advances in Neural Information Processing Systems 18*, pages 1369–1376.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA.

Vapnik, V. N. (1998). *Statistical Learning Theory*. John Wiley & Sons.

Yamada, M., Sigal, L., Raptis, M., Toyoda, M., Chang, Y., and Sugiyama, M. (2015). Cross-domain matching with squared-loss mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1764–1776.

Ying, Y., Wen, L., and Lyu, S. (2016). Stochastic online AUC maximization. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 451–459.

Yuille, A. L. and Rangarajan, A. (2002). The concave-convex procedure (CCCP). In *Advances in Neural Information Processing Systems 14*, pages 1033–1040.

Zhao, P., Jin, R., Yang, T., and Hoi, S. C. (2011). Online AUC maximization. In *Proceedings of the 28th International Conference on Machine Learning*, pages 233–240.

Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., and Oliva, A. (2014). Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems 27*, pages 487–495.