Positive Unlabeled Learning Applications in Music and Healthcare

by

Tom Arjannikov
B. A. & Sc., University of Lethbridge, 2012
M. Sc., University of Lethbridge, 2015

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

Positive Unlabeled Learning Applications in Music and Healthcare

by

Tom Arjannikov
B. A. & Sc., University of Lethbridge, 2012
M. Sc., University of Lethbridge, 2015

Supervisory Committee

Dr. George Tzanetakis., Supervisor
(Department of Computer Science, University of Victoria)

Dr. Kwang Moo Yi, Co-Supervisor
(Department of Computer Science, University of British Columbia)

Dr. Sarah Macoun, Outside Member
(Department of Psychology, University of Victoria)

ABSTRACT

The supervised and semi-supervised machine learning paradigms hinge on the idea that the training data is labeled. The label quality is often brought into question, and problems related to noisy, inaccurate, or missing labels are studied. One of these is an interesting and prevalent problem in the semi-supervised classification area where only some positive labels are known. At the same time, the remaining and often the majority of the available data is unlabeled, i.e., there are no negative examples. Known as Positive-Unlabeled (PU) learning, this problem has been identified with increasing frequency across many disciplines, including but not limited to health science, biology, bioinformatics, geoscience, physics, business, and politics. Also, there are several closely related machine learning problems, such as cost-sensitive learning and mixture proportion estimation.

This dissertation explores the PU learning problem from the perspective of density estimation and proposes a new modular method compatible with the relabeling framework that is common in PU learning literature. This approach is compared with two existing algorithms throughout the manuscript, one from a seminal work by Elkan and Noto and a current state-of-the-art algorithm by Ivanov. Furthermore, this thesis identifies two machine learning application domains that can benefit from PU learning approaches, which were not previously seen that way: predicting length of stay in hospitals and automatic music tagging. Experimental results with multiple synthetic and real-world datasets from different application domains validate the proposed approach.

Accurately predicting the in-hospital length of stay (LOS) at the time of admission can positively impact healthcare metrics, particularly in novel response scenarios such as the Covid-19 pandemic. During the regular steady-state operation, traditional classification algorithms can be used for this purpose to inform planning and resource management. However, when there are sudden changes to the admission and patient statistics, such as during the onset of a pandemic, these approaches break down because reliable training data becomes available only gradually over time. This thesis demonstrates the effectiveness of PU learning approaches in such situations through experiments by simulating the positive-unlabeled scenario using two fully-labeled publicly available LOS datasets.

Music auto-tagging systems are typically trained using tag labels provided by human listeners. In many cases, this labeling is weak, which means that the provided

tags are valid for the associated tracks, but there can be tracks for which a tag would be valid but not present. This situation is analogous to PU learning with the additional complication of being a multi-label scenario. Experimental results on publicly available music datasets with tags representing three different labeling paradigms demonstrate the effectiveness of PU learning techniques in recovering the missing labels and improving auto-tagger performance.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

All machine learning algorithms fall into one of three general categories: supervised, semi-supervised, and unsupervised learning. Historically, the supervised learning area of research was most popular, and more recently, there is an increasingly growing interest in the other two. The supervised and semi-supervised machine learning paradigms hinge on the idea that the training data is labeled. The quality of labels is often brought into question, and problems related to noisy, inaccurate, or missing labels are studied. One of these is an interesting and prevalent problem of binary classification that occurs in the scope of semi-supervised learning known as *Positive-Unlabeled (**PU**) learning* [1–3]. Here, only some positive labels are known, while the remaining and often the majority of available data is unlabeled, i.e., there are no negative examples. Thus, the data can be split into two sets: one that contains only the positive examples denoted as P, and the other, is a mix of positive and negative instances that are unlabeled, denoted as U. Then, the task is to learn a classifier that can distinguish between newly arrived positive and negative instances even though no negative examples are available during training.

Over the years, the PU learning scenario has been identified with increasing frequency in different contexts across many disciplines, including but not limited to health science [4, 5], biology [6, 7], bioinformatics [8–10], geoscience [11], physics [12], and business [13, 14]. In these and many other domains, the PU situation occurs naturally. For example, in health care, some people are diagnosed with a certain illness, and others are not. However, not being diagnosed (as opposed to misdiagnosed) is not the same as not being afflicted by the said illness. Many illnesses often go undiagnosed or misdiagnosed for various reasons. Thus, if we were to learn a model for a particular illness from a general population, we ought not to consider a person

who was not diagnosed with having that illness as a necessarily negative example of that illness. For instance, Chen et al. obtained a unanimous agreement from practitioners working in the field of Alzheimer's Disease that the problem of diagnosing that disease is better formulated as positive-unlabeled rather than the traditional positive-negative learning [4].

A related problem analogous to PU learning arises in multi-label learning, where the data is *weakly labeled* [15, 16], which is also known as the problem of learning with *missing labels* [17]. In multi-label learning [18], as opposed to binary classification, a data instance may belong to multiple categories simultaneously. For instance, a person could be diagnosed with multiple concurrent illnesses (known as comorbidity). This situation is similar to PU because some instances of a particular class concept are not labeled as such. Thereby, when a label is considered independently, we have a scenario identical to PU learning. It can be viewed as a particular case of learning from weakly labeled data where the number of possible labels is one. However, the two problems are also slightly different from each other. For instance, there could be correlations among labels in the multi-label case, which is impossible when there is only one class label. Like PU, weakly labeled data occurs naturally in many domains, such as image classification [19], text categorization [20], and music annotation/tagging [21].

The system performance degrades considerably due to training on data with missing labels in both binary [3] and multi-label [22, 23] learning scenarios. These problems, learning from positive-unlabeled/weakly-labeled data, are especially pervasive when the only reliable source of labels is a human being, which means that obtaining good quality labels is an expensive and time-consuming endeavor. For example, when creating a dataset for training machine learning models capable of identifying cancer in medical images, a medical doctor must examine and label the images before using them for training the models. Often, when labels are more subjective (e.g., emotion recognition tasks), several critics (or annotators) must reach a consensus before a label is accepted into the training dataset.

Although the nature of this problem is not limited to binary and multi-label cases (e.g., multiclass [24, 25]), the scope of the work presented here is focused only on the binary positive-unlabeled and multi-label weakly labeled scenarios.

1.1 Contributions

This dissertation explores the PU learning problem from the density estimation perspective and provides a Density Estimation-based Assymetric Relabeling (DEAR) approach that is compatible with the relabeling framework common in PU learning literature. The advantage of DEAR is its modularity, it is easy to implement using existing and well-known building blocks and maintains the flexibility of using existing algorithms suitable for the situation at hand. The proposed approach is compared with two existing PU learning algorithms throughout the manuscript, one is from a seminal work by Elkan and Noto [26] the other is a more recent approach by Ivanov that was shown to be effective across a wide range of scenarios [27]. The main contribution of this thesis consists of identifying two machine learning application domains that can benefit from PU learning techniques: predicting length of stay in hospitals and automatic music tagging. The latter is most interesting and leads to many possible research directions. Additionally, this thesis provides a framework for exploring many domain specific questions about tags and an initial baseline and evaluation strategies.

1.1.1 Density Estimation-based Asymmetric Relabeling

Within classification, the binary case of PU learning falls directly between fully-supervised one-class and binary classification but with the addition of unlabeled data, which makes it a semi-supervised problem. On the one hand, one-class classifiers can learn from only positive examples, but they do not take advantage of the additional, albeit unlabeled, examples. On the other hand, traditional binary classifiers, such as logistic regression, are unable to learn without negative labels. To overcome this challenge, we can preprocess the positive-unlabeled data and get it as close as possible to the actual positive-negative case; then, any preferred binary classifier can be used as usual. This is a common two-step framework in PU learning literature [3, 28], and the proposed relabeling approach is focused on the first (relabeling) step with the goal to improve the final classification performance after the second step.

The idea behind the proposed approach, initially published in conference proceedings [29], and its improved version presented in Chapter 4 is based on the key observation that the unlabeled set is a mixture of positives and negatives. In contrast, the positive set is pure, which means that the probability density function (PDF) estimated from positive examples will have a higher density in the positive regions of

the feature space than the one estimated from unlabeled examples if the two are compared on even ground. Thus, the approach is to obtain density estimates from the positive and unlabeled examples, independently of each other, and use them to relabel the training data before learning a classifier from it, but only the unlabeled portion of the data, and only those that are likely to be positive (hence asymmetric). Identifying confidently positive examples and removing them from the unlabeled set (by labeling them as positive) increases the concentration/proportion of negative examples within it making it more negative overall. Thereby, after the asymmetric relabeling step, the remaining unlabeled examples can be considered negative with higher confidence than before. Then, a traditional binary classifier can be trained on the relabeled data instead of the original positive-unlabeled.

The experiment results presented throughout this manuscript demonstrate that this method recovers a significant proportion of missing labels in many cases, both with synthetically generated and real-world data. In terms of final classification metrics, DEAR performs overall on par with other PU learning approaches, sometimes better and sometimes worse, depending on the data (which can also be said about the PU algorithms). The time complexity of DEAR depends on the underlying density estimators. However, when using simple histograms, the proposed approach is significantly faster without giving up much efficacy.

1.1.2 Predicting Patient Length of Stay in Hospitals

Accurately predicting the in-hospital length of stay (LOS) at the time of admission can positively impact healthcare metrics, particularly in novel response scenarios such as the Covid-19 pandemic. Machine learning techniques have been used for predicting LOS based on patients' demographic and clinical characteristics. During the regular steady-state operation, traditional supervised-learning classification algorithms can be used for this purpose to inform planning and resource management. However, when there are sudden changes to the admission and patient statistics, such as during the onset of a pandemic or the establishment of a new hospital, these approaches break down because reliable data for training machine learning models becomes available only gradually over time. Chapter 5 simulates this scenario revealing how LOS predictions can be negatively affected during such cold-start transition periods and how PU learning approaches can be leveraged to remedy the situation. The experiments are carried out on two different publicly available LOS datasets, one of which

was collected in response to the Covid-19 pandemic. The other dataset consists of inpatient data collected from the State of New York hospitals in 2017. Parts of the work presented in Chapter 5 have been published in conference proceedings [30, 31].

1.1.3 Treating Weakly Labeled Music Auto-Tagging Data

Music auto-tagging systems are typically trained using tag labels provided by human listeners. In many cases, this labeling is weak, which means that the provided tags are valid for the associated tracks, but there can be tracks for which a tag would be valid but not present. This situation, also known as the problem of missing labels, is analogous to PU learning with the additional complication of being a multi-label scenario. Because of how analogous these situations are, we could use methods from the PU learning literature to treat weakly labeled data before training an auto-tagger on it. Chapter 6 experimentally investigates the effect of weakly labeled data when training music auto-tagging systems on their final performance. Using different publicly available datasets and several popular classifiers, Chapter 6 demonstrates how PU learning techniques can be leveraged to treat the weakly labeled data and improve music auto-tagging systems trained on weakly labeled data.

1.2 Manuscript Layout

Chapter 1 introduced the problem of positive-unlabeled learning and its two applications studied in this dissertation, prediction of patients' length of stay and music auto-tagging, along with a brief overview of the contributions of this work. Chapter 2 positions this thesis in the context of related works both in machine learning literature and its application domains, including healthcare and music. Then, the PU learning problem is formulated in Chapter 3, both in binary and multi-label contexts. That chapter also includes the relevant information about density estimation and classification methodology, PU learning approaches, assumptions, evaluation metrics, and an illustration of the simulation and experimental framework central to this thesis. Chapter 4 presents a modular and relatively lightweight approach for treating weakly labeled training data as a preprocessing step before learning a classifier; it includes experimental results with synthetic and real-world datasets. Chapter 5 identifies the PU learning scenario in healthcare, specifically when predicting in-hospital patient length of stay during cold-start transition periods. Similarly, Chapter 6 identifies the

PU learning scenario in music auto-tagging and extends the binary PU learning methods to weakly labeled multi-label data; experimental results on several music tagging datasets show that this approach is promising. Then, Chapter 7 provides concluding remarks and a short discussion about future directions that the work presented here could take.

Chapter 2

Related Works

This chapter positions the work presented in this dissertation in the context of related works, first in machine learning literature and then in its application domains, including healthcare and music. While this chapter serves as a brief overview of the related research fields, the next chapter will provide more specific (and narrow) methodological connections and the background information necessary for the chapters to follow.

2.1 Within Machine Learning

Depending on the type of data available for modeling, the entire machine learning discipline could be divided into three broad categories: supervised, semi-supervised, and unsupervised. The data to be modeled is fully labeled in supervised learning and completely unlabeled in unsupervised learning, and semi-supervised learning falls between the two [32]. Since the PU data is by definition not fully labeled, PU learning falls in the semi-supervised learning category [33]. In the literature, solutions to PU learning problems often incorporate techniques from any one of the three categories or a combination thereof.

When learning a predictive model from data, the target variable, or the value we want the model to predict, is usually either continuous or discrete.[1] In the first case, we use regression techniques, and in the latter, classification. The classification problem involves identifying which category from a given set of categories a new observation belongs to based on previously observed examples. Here, PU learning

[1] There are also other types of variables, for example circular [34], but they are not relevant here.

spans the gap between binary[2] classification [35] and one-class (positive-only) classification [36, 37]. The latter has strong connections with data description [38] and novelty detection [39] problems. A one-class classifier learns from a set of "positive" examples representing a particular class concept; there are no examples outside the target class. In binary classification, we also have negative examples representing everything that is not the target (positive) class concept. Like the first two, PU learning works with positive examples; however, instead of negative examples or in their absence, we also have an unlabeled mixture of both positive and negative examples. When comparing the three, a one-class classifier can learn in the absence of negative examples but does not perform as well as a PU learning algorithm would when additional unlabeled data is available. If the additional data is fully labeled, then a traditional binary classifier is preferred to the other two. Safe semi-supervised learning [40] takes it a step farther, using labeled positives, negatives, and the additional unlabeled data to further improve the model.

It is worth noting that the performance of one-class approaches can benefit by incorporating active learning techniques [41], and perhaps that strategy could result in a better classifier trained from positive-only data than a positive-unlabeled classifier trained with an abundance of unlabeled data. However, active learning requires an "oracle", and if one is available, then a positive-unlabeled classifier can also benefit from active learning techniques [42]. Furthermore, there are connections to transfer learning, which studies different changes in data distribution from training to deployment. Since PU learning aims to learn on positive-unlabeled training data to predict positive-negative data during model deployment, it fits certain aspects of transfer learning. Transfer learning approaches often borrow semi-supervised techniques, like using unlabeled examples to improve transfer quality [43, 44]. Note that transfer learning and active learning are compatible and can be used in conjunction [45].

Many other machine learning problems are related to PU learning. From supervised learning, these are: learning with missing data [46], learning in the presence of label noise [47, 48], and cost-sensitive learning [49–51]. In semi-supervised learning, problems closely related to PU learning include co-training [52], self-supervised learning [4], multiple-instance learning [53, 54], semi-supervised novelty detection [55], and learning from weakly labeled data [56]. Additionally, safe semi-supervised learning investigates labeling issues in binary classification, where PU learning falls in the

[2]PU learning has also been considered in the multi-class scenario [25], but it is generally formulated as a binary classification problem.

"incomplete supervised learning" category [40, 57]. Also, approaches from clustering [58, 59], and matrix completion [60] have been applied in the PU setting.

Another closely related problem that is slightly different from PU learning is mixture proportion estimation. Here, given a known distribution, the task is to identify the proportion of that distribution in a mixture of distributions [61]. Related to mixture proportion estimation, techniques from class prior estimation [62] can also be leveraged for working with positive-unlabeled data [63].

2.2 Application Domains

The PU learning problem initially identified and formulated in the early 2000s is apparent in many domains. The following are just a few concrete examples. In website classification, one often starts with a set of websites related to the topic of interest, aiming to find similar websites in the otherwise unlabeled set of all remaining websites on the internet [13]. In remote-sensing, a PU learning approach was adopted for the one-class classification of land-cover type from areal photographs [11]. In astronomy, researchers use PU learning to identify pulsar signals in the otherwise unlabeled background noise [12].

Another interesting example of PU learning is presence-only labeling [6], also known as use-availability [7], where researchers are interested in predicting which locations an animal would frequent from a set of all possible and otherwise unlabeled habitat locations. As positive examples, they use historical records of the animal's sightings or other signs. Researchers are often interested in identifying candidate positive instances and would later verify them for authenticity using other methods. Elkan and Noto distinguish this "case-control" scenario from the single training set situation where the dataset is sampled once and then partly labeled, in contrast to two independent draws, "case" and "control" [26].

Similar to other domains, PU situations occur naturally in healthcare, including but not limited to Alzheimer's Disease diagnosis [4], drug activity prediction [5], disease gene identification [8, 9], and prediction of circRNA disease associations [10]. Chapter 5 identifies yet another area where PU approaches can be beneficial: predicting patients' *length of stay* (LOS) at the time of their arrival to a healthcare unit, which can help with care planning and resource management, ultimately resulting in a positive impact on patient outcomes. For instance, it is one of the central topics in clinical pathways [64]. Predicting LOS is often formulated as a binary classification

problem [65] and, more recently, a multi-class one. Here, the algorithms learn from data, such as electronic health records, patient demographics, and clinical characteristics, to predict the LOS as one of two or more discrete classes. For example, Pofahl et al. use an Artificial Neural Network (ANN) algorithm as a binary classifier to predict LOS >7 days in acute patients [66]. Morton et al. compare different methods in the binary classification setting for diabetic patients [67]. Hachesu et al. use Decision Tree (DT), Support Vector Machine (SVM), and ANN algorithms to predict LOS for patients with coronary artery disease as a 3-category classification problem [68]. Harerimana also solves the 3-class problem but for different ranges in LOS for de-identified admission records using deep learning techniques [69]. Recently, there is increasing interest in predicting LOS in hospitals due to the Covid-19 pandemic. For example, Wu et al. perform a multi-variable regression analysis to study the factors influencing LOS among Covid-19 patients [70]. A survey and synthesis of the length of stay datasets and studies for Covid-19 can be found in Rees et al. [71]. Chapter 5 simulates a cold-start scenario, such as during an on-set of a pandemic where drastic changes in patient statistics could adversely affect LOS prediction, and investigates how PU learning approaches can be used to mitigate it.

Chapter 6 identifies another domain that could benefit from PU learning techniques and has not been considered in the literature yet - music auto-tagging. It is usually formulated as a multi-label classification problem where auto-taggers learn, in a supervised fashion, from data whose labels often originate from human annotators. There is a considerable degradation in system performance when a music auto-tagger is trained on weakly labeled data [51, 72]. Many approaches have been proposed in the music information retrieval literature to deal with the missing/weak labels. For example, Lin et al. use music playlists to inform auto-taggers in a multi-task setting [73]. Similarly, Ibrahim et al. use information about music listening context to deal with the problem [74]; meanwhile, Lin and Chen employ cost-sensitive learning techniques [51]. Chapter 6 examines how PU learning approaches can be leveraged to mitigate this problem.

This chapter provided a brief overview of the related research fields. The next chapter covers the methodological background necessary for the chapters to follow, which include a new approach for dealing with PU learning situations and look at its application in healthcare and music.

Chapter 3

Methodology

While the previous chapter gave a more general overview of the relevant research fields, this chapter provides connections to the specific background necessary for the chapters to follow. This includes information about probability distribution estimators, combination rules, well-known traditional classifiers, PU learning approaches, and connections with multi-label learning. Furthermore, this chapter covers the relevant assumptions, evaluation metrics, and the experimental framework including the simulation of PU learning scenarios.

In binary classification, the goal is to learn from the available data a model that can categorize each newly arrived data instance into one of two distinct categories (or classes), usually denoted as *positive* and *negative*. Let $\mathbf{x} \in \mathbf{X}$ be a data instance, which is a set of attribute values $\mathbf{x} = \{x_1, x_2, ..., x_k\}$ that describe the instance in k-dimensional feature space, and $y \in \{0, 1\}$ is a binary label s.t. $y = 1$ indicates that $\mathbf{x}$ is a positive example and $y = 0$ negative. In other words, $\mathbf{x}$ is a vector drawn from either a positive or a negative k-variate distribution, $\mathcal{D}^k_+$ or $\mathcal{D}^k_-$ respectively, s.t. $y = 1$ when $\mathbf{x} \sim \mathcal{D}^k_+$ and $y = 0$ when $\mathbf{x} \sim \mathcal{D}^k_-$. Then, the goal is to learn a binary classifier from the dataset $\mathbf{X}$ and its corresponding labels $\mathbf{Y}$ that assigns the correct class label to a newly arrived instance. Using probabilities, we can formulate it as a discriminator function:

$$d(\mathbf{x}) = \begin{cases} 1, & \text{if } p(y = 1 \,|\, \mathbf{x}) > p(y = 0 \,|\, \mathbf{x}) \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

Note that, when $p(y = 1 \,|\, \mathbf{x}) = p(y = 0 \,|\, \mathbf{x})$ the decision is arbitrary; also, in binary classification $p(y = 1 \,|\, \mathbf{x}) + p(y = 0 \,|\, \mathbf{x}) = 1$.

When the data is fully labeled, we can use Bayes' theorem to compute the conditional probabilities,

$$p(y = 1 \,|\, \mathbf{x}) = \frac{p(\mathbf{x} \,|\, y = 1) \cdot p(y = 1)}{p(\mathbf{x})}$$
$$p(y = 0 \,|\, \mathbf{x}) = \frac{p(\mathbf{x} \,|\, y = 0) \cdot p(y = 0)}{p(\mathbf{x})} \tag{3.2}$$

3.1 Distribution Estimators and Discriminators

The probabilities $p(\mathbf{x} \,|\, y = 1)$ and $p(\mathbf{x} \,|\, y = 0)$ in Equation 3.2 can be estimated directly from the data using one of the readily available distribution estimators from the literature. The ones included in this thesis and outlined below are: the histogram probability mass function, Kernel Density Estimation (KDE) algorithm, and Gaussian probability density function (PDF). Note that, while maintaining the flexibility to do so, there is no need to manually set estimator parameters, such as the number of histogram bins or the KDE kernel bandwidth, because they can also be estimated from the data using well-known methods. Alternatively, an expert practitioner can also set them manually to ensure a proper fit.

3.1.1 Gaussian PDF

One of the most common distributions that appear in natural and man-made processes is the continuous univariate normal or Gaussian, parameterized as:

$$e(x \,|\, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{[-\frac{1}{2}(x-\mu)^2/2\sigma^2]} \tag{3.3}$$

where μ is the mean and σ is the standard deviation, which can be estimated directly from the data. It generalizes to multivariate situations according to the following equation:

$$e(\mathbf{x} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt[d]{2\pi}\sqrt{det(\boldsymbol{\Sigma})}} e^{[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})]} \tag{3.4}$$

where $\boldsymbol{\mu}$ is a vector of means, one for each variable, $\boldsymbol{\Sigma}$ is the covariance matrix computed from the data, and $det(\boldsymbol{\Sigma})$ is the determinant of the covariance matrix.

Both $f(x|\mu, \sigma^2)$ and $f(\mathbf{x}\boldsymbol{\mu}, \boldsymbol{\Sigma})$ are continuous probability density functions (PDF) that can be estimated directly from the data.

3.1.2 Histograms

Another way to obtain an empirical estimate of the distribution of data across a variable is to discretize or segment the variable's domain into a predetermined number of bins and count how many data points fall into each bin. These counts can be normalized by dividing each bin's count by the total number of samples from which the estimate is computed. Because histograms are discrete, when normalized, they serve as the probability mass functions (as opposed to density). This strategy provides an approximation of the underlying distribution from which the data was drawn, but it is not flawless. Besides sampling issues, which affect all estimators, for histograms, the bulk of the error arises from the choice of parameters, namely the number of bins and their width.

Although some optimal fit could be achieved by searching for the correct parameter setting, this search also adds additional time requirements and becomes data-dependent, i.e., different parameter settings are optimal for different sets of data. To deal with these issues and avoid any accidental bias in parameter choice, we can use an automatic method to determine the number of bins and their sizes. Several methods are implemented in the NumPy [75] histogram() library, from which this thesis uses the auto setting that takes the maximum from two estimators: Sturges' [76] (which is R's default):

$$k = \lceil log_2\ n \rceil + 1 \tag{3.5}$$

and the Freedman-Diaconis' [77] one:

$$h = 2\frac{IQR(\mathbf{X})}{n^{1/3}} \tag{3.6}$$

were IQR is the interquartile range computed from the data sample $\mathbf{X}$ with n number of instances, k is number of bins, and h is the width of each bin, s.t. $k = \lceil \frac{max(a_i) - min(a_i)}{h} \rceil$.

One of the main advantages of using histograms is that they can model non-parametric and unknown distributions. However, extending the histogram method from a single variable to the multivariate case results in a combinatorial explosion of computational complexity with respect to the number of variables, thereby making

this method intractable in many real-world applications.

3.1.3 Kernel Density Estimation (KDE)

Another popular non-parametric method, which is more sophisticated than histograms, is Kernel Density Estimation (KDE) [78]. Analytically, KDE is an integral of a continuous function that adheres to the axioms of probability [79]. However, when estimated from a sample, it becomes a normalized sum of likelihoods across all kernels, each of which is centered at its corresponding (training) data point $\mathbf{x}_i \in \mathbf{X}$:

$$\hat{e}(\mathbf{x} \,|\, \mathbf{X}) = \frac{1}{nb} \sum_{i=1}^{n} K(\frac{\mathbf{x} - \mathbf{x}_i}{b}), \tag{3.7}$$

where $n = |\mathbf{X}|$ is the number of samples in the dataset, b is the smoothing parameter called *bandwidth*, and K is the kernel s.t. $K(x) = \frac{1}{b}K(\frac{\mathbf{x}}{b})$. KDE is a continuous probability density function that usually results in less error than the Histogram's discrete probability mass function. However, it is much more computationally intensive, especially when estimating data distributed across multiple variables. This thesis makes use of the univariate KDE, and in multivariate cases, uses the feature decomposition and recombination as described in Section 3.1.4. The Statsmodels module [80] for Python has many different kernels and bandwidth estimators; all the KDE results presented in this manuscript are produced using the Gaussian kernel with the bandwidth parameter set to "normal_reference".

3.1.4 Density-based Discriminator

Following Equations 3.1 and 3.2, we can formulate a density-based discriminator from two estimators fitted to the positive and negative distributions, $\hat{e}^+ = p(x \,|\, y = 1)$ and $\hat{e}^- = p(x \,|\, y = 0)$ respectively, and class prior $\alpha = p(y = 1)$:

$$\hat{d}(\mathbf{x}) = \begin{cases} 1, & \text{if } \alpha \cdot \hat{e}^+ > (1 - \alpha) \cdot \hat{e}^- \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

In many situations, the class prior α is directly available to the practitioner, and in cases where class prior is unknown, it can be estimated from the data [62, 81].

When we do not know the underlying distribution type or when the distribution cannot be expressed using a finite number of parameters, non-parametric approaches

become essential. Non-parametric multivariate density estimation methods are particularly useful in machine learning and data mining tasks, yet they have a significant limitation of being computationally intractable. Although efforts have been made to alleviate this issue, it still remains an open problem [82]. To reduce the computational complexity, we can relax the problem by assuming that there are no correlations among features/variables, which is often reasonable. For instance, a person's height and weight are correlated with age in children but not in adult populations. In probability theory, this assumption is known as the conditional independence assumption and, in combination with the Bayes theorem, Equation 3.2, it results in the popular Naïve Bayes classifier. Similarly, we can take the product of the likelihoods from individual estimators independently fitted to each variable:

$$\hat{e}(\mathbf{x}) = \prod_{j=1}^{k} \hat{e}_j(\mathbf{x}) \tag{3.9}$$

For computational convenience and to avoid numerical instability, it is common to take the natural logarithm of the product of likelihoods, which translates to the sum of log-likelihoods:

$$\hat{e}(\mathbf{x}) = \ln \prod_{j=1}^{k} \hat{e}_j(x) = \sum_{j=1}^{k} \ln \hat{e}_j(\mathbf{x}) \tag{3.10}$$

Consequently, the discriminator function becomes:

$$\hat{d}(\mathbf{x}) = \begin{cases} 1, & \text{if } \alpha \sum_{j=1}^{k} \ln \hat{e}_j^{+}(\mathbf{x}) > (1-\alpha) \sum_{j=1}^{k} \ln \hat{e}_j^{-}(\mathbf{x}) \\ 0, & \text{otherwise} \end{cases} \tag{3.11}$$

Because the density estimators produce probabilistic outputs, we can adjust this configuration and express it as a function $\hat{f}$ that outputs a probabilistic likelihood value based on the estimated PDFs, where 0.5 is the decision boundary between two classes:

$$\hat{f}(\mathbf{x}) = \frac{1}{2}\left[1 + \alpha \sum_{j=1}^{k} \ln \hat{e}_j^{+}(\mathbf{x}) - (1-\alpha) \sum_{j=1}^{k} \ln \hat{e}_j^{-}(\mathbf{x})\right] \tag{3.12}$$

3.1.5 Ensemble-based Discriminator

The conditional independence assumption also enables ensemble learning techniques where multiple different learners model the data and the predictions from their in-

dividual perspectives are combined into one decision, which is the final output of the overall system. There are multiple ensemble learning schemata in the literature [83, 84] that could be explored even without making this assumption or by making another. The density estimation-based asymmetric relabeling approach presented in Chapter 4 uses this idea to decompose the multivariate case into k univariate ones. Its initial version [85] builds k univariate density-based discriminators $\hat{d}_j(x_j) \in \{0, 1\}$, one for each variable/attribute, and combines their zero-one outputs using a simple majority voting combination rule:

$$\hat{d}'(\mathbf{x}) = \begin{cases} 1, & \text{if } \frac{1}{k} \sum_{j=1}^{k} \hat{d}_j(x_j) > \theta \\ 0, & \text{otherwise} \end{cases} \tag{3.13}$$

where θ is a user-defined threshold value between 0 and 1, usually set above 0.5 for the majority consensus. Although this works well in practice, there are better methods, such as the slightly more sophisticated yet still computationally simple sum rule (or arithmetic mean rule). Based on the experimental comparison of various classifier combination rules, Kittler et al. found that best performing are those developed under the most restrictive assumptions, which in their study was the sum rule [86].

Using probability density estimators capable of probabilistic output, we can use a combination rule that is more fine-grained than the one in Equation 3.13 by comparing the arithmetic mean of likelihoods among the estimators fitted to one class with that of the other class. However, there is an inherent difference between a weak learner in an ensemble and a density estimator. In Bayesian terms, a weak learner is expected to learn both the probability density distribution $p(\mathbf{x} \mid y)$ and the class prior $p(y)$, but a density estimator is not. Hence, we can include class prior during the comparison step and obtain the following discriminator:

$$\hat{d}(\mathbf{x}) = \begin{cases} 1, & \text{if } \frac{\alpha}{k} \sum_{j=1}^{k} \hat{e}_j^{+}(x_j) > \frac{1-\alpha}{k} \sum_{j=1}^{k} \hat{e}_j^{-}(x_j) \\ 0, & \text{otherwise} \end{cases} \tag{3.14}$$

Similarly to the density-based discriminator in previous section, so long as the underlying estimators produce probabilistic outputs, we can express Equation 3.14 as a function $\hat{f}$ that produces a probabilistic output with 0.5 as the decision boundary

between two classes:

$$\hat{f}(\mathbf{x}) = \frac{1}{2}\left[1 + \frac{\alpha}{k}\sum_{j=1}^{k}\hat{e}_j^{+}(x_j) - \frac{1-\alpha}{k}\sum_{j=1}^{k}\hat{e}_j^{-}(x_j)\right] \tag{3.15}$$

3.1.6 Weighted Probabilistic Output

The formulation in Equation 3.15 has the additional benefit that, when auxiliary information about features is available, each attribute can have a different level of impact on the final decision based on how informative or important it is. We can incorporate this information into Equation 3.15 as weights w_j:

$$\begin{aligned}\hat{f}(\mathbf{x}) = \frac{1}{2}\left[1 + \frac{\alpha}{k}\sum_{j=1}^{k} w_j \cdot \hat{e}_j^{+}(x_j) - \frac{1-\alpha}{k}\sum_{j=1}^{k} w_j \cdot \hat{e}_j^{-}(x_j)\right], \\ \text{s.t. } w_j \in [0,1] \text{ and } \sum_{j=1}^{k} w_j = 1.\end{aligned} \tag{3.16}$$

Note that setting any weight to 0 excludes that variable/attribute from the model. The same weighting strategy can be applied to the density-based discriminator in Equation 3.12:

$$\begin{aligned}\hat{f}(\mathbf{x}) = \frac{1}{2}\left[1 + \frac{\alpha}{k}\sum_{j=1}^{k} w_j \cdot \ln \hat{e}_j^{+}(x_j) - \frac{1-\alpha}{k}\sum_{j=1}^{k} w_j \cdot \ln \hat{e}_j^{-}(x_j)\right], \\ \text{s.t. } w_j \in [0,1] \text{ and } \sum_{j=1}^{k} w_j = 1.\end{aligned} \tag{3.17}$$

3.2 Traditional Binary Classifiers

Binary classification is one of the most explored topics in machine learning and data mining. A variety of classifiers have been proposed with different characteristics related to various aspects of performance, such as: what type of data they are best suited for, time and space complexity both at training and prediction stages, guarantees on convergence, whether they are distributable, etc. In this thesis, binary classifiers are used as building blocks for more complex machine learning architectures. As they are relatively well known, this section provides a summary of each. More details can be found in any textbook about machine learning [87] or data mining

[88].

3.2.1 Naïve Bayes

The Naïve Bayes (NB) classifier is based on a probabilistic formulation of the classification problem as described in the previous section. The Naive assumption refers to the assumed conditional independence of the features given the class. This assumption decomposes the challenging problem of estimating a multi-variate density function over the features to the simpler problems of estimating univariate distributions over each feature for each class [88].

3.2.2 Logistic Regression

Logistic Regression (LR) is an algorithm that fits a linear function (i.e., a linear combination of weights on the input variables) via regression by minimizing the loss on the soft output of a logistic (or sigmoid) function. It is a general statistical model that can be used for classification. The independent random variables (predictors/attributes) can be either binary or continuous, each having an independent weight parameter. Their linear combination encodes the log-odds that the input corresponds to the output of 1, usually meaning the positive class; exponentiating the output of LR recovers the odds. The target (output) variable y can be either binary $y \in \{0, 1\}$ or continuous $y \in [0, 1]$. In the continuous case, the closer the value is to either 0 or 1, the stronger is the association between the input values and the corresponding class label reflecting the certainty or confidence of the instance belonging to that class. Logistic regression has been widely adopted across different application domains and has been especially popular in biostatistics and healthcare.

3.2.3 The k-Nearest Neighbor

The k-Nearest Neighbor (kNN) is a non-parametric approach useful in both regression and classification problems [89, 90]. This method works by memorizing the training data, and then, given a new input instance, it returns the k training instances closest to that instance. In classification, the majority class among the k instances determines the class of the input instance. The way kNN measures closeness or distance is a critical component of the algorithm. Some distance metrics, such as the Euclidean distance, can be adversely affected by the difference in scale between attributes, and in

practice, attribute normalization/standardization is often recommended. Moreover, choosing the appropriate distance metric for the data can be beneficial, and methods from a related area of research called distance metric learning [91] are shown to be useful [92].

3.2.4 Decision Tree

The Decision Tree (DT) algorithm is a popular binary classification approach originally introduced for discrete attributes but has since been extended to work with continuous ones [93, 94]. It learns a tree-like structure from the data, splitting the training dataset into disjoint subsets at each node based on the single attribute that yields the highest class-wise information gain (or reduction in entropy). This binary partitioning is a classic divide-and-conquer approach that leads to fast performance times. The dataset need not be memorized as the attribute-based split is recorded as a conditional statement that summarizes the decision at each node. For example, when classifying humans into children and adults categories, at one of the nodes, there may be a decision like "if height is more than 5 foot, then it is an adult child, otherwise continue to next node (lower tree level)." During the classification step, the attributes of a new instance in conjunction with the rules at each node dictate which branches to follow, starting at the root and reaching the final classification decision at a leaf node. Thus, a set of humanly interpretable rules can be derived, indicating which attribute values lead to which class and leading an intuitive human understanding of the resulting classification decision. In addition to speed, one of the main advantages of using DTs is their interpretability. Since their inception, DT classifiers have been extended to handle multi-class and multi-label problems, among others, and their classification performance can improve through ensemble learning techniques, e.g., decision forests [95].

3.2.5 Support Vector Machines

Support Vector Machines (SVM) are binary classifiers that map the input vector of attributes onto a high dimensional space (using a kernel trick), where they derive a hyperplane separating the two classes based on the training data. Moreover, SVM algorithms choose the hyperplane in such a way as to maximize the margin separating the two classes. Different kernels have been proposed in the literature; the experimental results presented in this manuscript use the Gaussian kernel. SVMs require

to memorize some (possibly all) of the training data points as support vectors that describe the class-separating hyperplane. During the classification step, SVMs map a new data instance onto the same space and assigns it a label based on which side of this hyperplane it falls. SVMs can be quite time-consuming to train, but their predictions are relatively fast. They have been shown effective in high-dimensional feature spaces and can be extended to produce probabilistic outputs and to handle classification problems beyond binary, including but not limited to: one-class [96], multi-class [97], and multi-output and multi-task [98] classification.

3.2.6 Artificial Neural Network

Artificial Neural Network (ANN) is another popular approach often used in classification tasks. It is inspired by biological neural networks like the human brain. ANN consists of multiple interconnected artificial neurons usually arranged in layers with information flowing into the input layer through the hidden layers (one at a time) and finally out of the output layer. Each neuron is a computational unit approximating a linear function; it can accept one or more inputs and propagate their weighted sum to the next layer based on its activation function. Sets of these neurons are often connected in multiple layers, and together they can represent a non-linear function. ANNs with multiple hidden layers are referred to as deep learning networks. The input consists of attributes describing the phenomenon in question, for example, a digital image of hand-written digits [99]. The information propagates through the network, thus simulating a synapse in a biological brain, and the output layer activations are used to classify, in the example above, the images into one of 10 discrete digit classes. During training, the error is back-propagated through the hidden layers to update the weights of the hidden neurons [99]. This training stage is usually computationally heavy and time-consuming. Once the network is trained, its predictions are relatively quick.

3.3 PU Learning

PU-learning problems are a form of semi-supervised learning in which there are reliable positive labels but no reliable negative labels. So the data consists of positive samples and unlabeled samples (which potentially can be either positive or negative). Various approaches have been proposed to deal with PU-learning problems; the ones

relevant in the context of this thesis are outlined below.

3.3.1 Relabeling Approach

The central theme of this dissertation and a frequent PU learning approach in the literature is the two-step approach, with an optional third step [10, 100–103]:

Step 1 - Identify the unlabeled instances that are likely to be negative (optionally, positive ones also).

Step 2 - Learn a classifier (or several) from the instances identified in the first step.

Step 3 - If applicable, select the best classifier from the second step.

The approach detailed in Chapter 4 follows the same recipe with the focus on the first step, which consists of categorizing all of the unlabeled instances into likely positive and likely negative ones.

3.3.2 EN Algorithm

In the context of PU learning, neither $p(y = 1)$ nor $p(y = 0)$ are directly available. To remedy this, in their earlier seminal paper, Elkan and Noto [26] added a binary random variable to this formulation, $s \in \{0, 1\}$, which corresponds to the fact that either an instance is labeled, $s = 1$, or not, $s = 0$. Then, each data instance is as a 3-tuple $\langle \mathbf{x}, y, s \rangle$ drawn from a fixed unknown distribution $p(\mathbf{x}, y, s)$. This formulation gives a probabilistic account for the labeling mechanism, and under certain assumptions, it allows the class prior, $p(y = 1)$, to be estimated from the non-traditional positive and unlabeled data.

Operating under the SCAR assumption (Eq. 3.19), Elkan and Noto prove that $p(y = 1|x) = p(s = 1|x)/c$ [26]. Since $s = 1$ corresponds to the positive and $s = 0$ to the unlabeled data, the probability $p(s = 1|x)$ can be estimated directly from the PU data using a traditional binary classifier. If that classifier is well-calibrated, its outputs can be corrected using the constant c to obtain the requisite $p(y = 1|x)$. Thus the problem becomes estimating the constant c, and Elkan and Noto propose three different estimators for it [26]. The EN algorithm results presented in this dissertation use the one they believe is the best choice (estimator 1).

3.3.3 DEDPUL Algorithm

A much more recent approach, DEDPUL, was shown to outperform current state-of-the-art (SOTA) PU learning methods on several publicly available datasets [27]. It is based on the idea of transforming the features corresponding to positive and unlabeled instances into prediction probabilities using a well-calibrated binary classifier trained on PU data to estimate $p(s = 1|x)$. Then, the minimum ratio between the probability distributions of the positive and unlabeled probabilistic predictions estimates the posterior prediction probabilities, which, in turn, can be used to re-train the classifier. The two steps are repeated iteratively until convergence. An artificial neural network is used as the underlying classification algorithm. Because of the iterative nature of the approach, this algorithm is significantly slower than the EN algorithm.

3.3.4 Class Prior

Class prior can be estimated and incorporated into PU learning methods in three ways: preprocessing, postprocessing, and method modification [3]. Preprocessing creates a new dataset from the positive-unlabeled either by re-balancing methods, empirical risk minimization methods, or methods that incorporate the label probabilities. Postprocessing makes use of the idea that the probability of an example being labeled is directly proportional to the probability of it being positive, then predictions from a binary classifier learned from positive-unlabeled data can be corrected after they are made. As the name implies, method modification adjusts how the algorithm learns to account for positive-unlabeled training data rather than positive-negative.

3.4 Multi-Label Learning with Weak Labels

Multi-label implies that each data point can be a positive example from different class concepts simultaneously. Weakly labeled means that not all class concepts were considered when labeling each data instance. Formally, in multi-label learning, each data instance $\mathbf{x} \in \mathbf{X}$ is accompanied by a set of binary labels/tags $\mathbf{y} = \{y_1, y_2, ..., y_T\}$ that describe the instance in T-dimensional semantic space, where each $y_t \in \{0, 1\}$ is a binary value s.t. $y_t = 1$ indicates that the corresponding tag is valid for $\mathbf{x}$ and $y_t = 0$ otherwise.

When data is weakly labeled, the problem becomes very similar to the PU learning one. When $y_t = 0$, we don't know whether it is a truly negative example of that class

concept or an unlabeled positive one. This situation is identical to the positive-unlabeled case when each class concept is considered individually by itself. The goal in multi-label classification is to learn from the available data a model that can assign the correct set of multiple tags to each newly arrived data instance. That is to return a vector $\mathbf{y}$ with each y_t set correctly to 0 or 1. There is an implicit assumption that the number of tags is fixed at T; therefore, the annotation vocabulary must be finite, which is not always the case [21]. However, those situations fall outside the scope of this dissertation.

Binary relevance transforms a multi-label problem with T labels into T binary classification problems, one per label, by making an assumption that there are no correlations among labels [104]. Then, the goal becomes to find a discriminator function for each tag t that maps the corresponding binary tag to a new data instance:

$$\underset{t\in\{1..T\}}{d_t(\mathbf{x})} = \begin{cases} 1, & \text{if } p(y_t = 1 \,|\, \mathbf{x}) > p(y_t = 0 \,|\, \mathbf{x}) \\ 0, & \text{otherwise} \end{cases} \tag{3.18}$$

However, correlations among the different labels often carry additional information that is useful for classification, and several approaches have been proposed in the multi-label domain to incorporate label correlations into the binary relevance transformation. Nonetheless, there is much room for improvement by taking into account nuances like imbalanced distribution of labels and different labeling importance.

In the context of this dissertation, the approach presented in Chapter 4 can be easily extended via binary relevance to deal with multi-label data, and Chapter 6 explores this idea through experiments in music auto-tagging.

3.5 Assumptions

When talking about the labeling mechanism, Elkan and Noto make an important distinction between their *single training set* scenario [26] and the *case-control* one described by Ward et al. [6]. In the single training set scenario, the tuples $\langle \mathbf{x}, y, s \rangle$ are drawn identically and independently (i.i.d.) from $p(\mathbf{x}, y, s)$, and only $\langle \mathbf{x}, s \rangle$ is recorded. In the case-control scenario, two independent sets are drawn: case and control. From the first set, all x such that $s = 1$ are recorded as presence cases, and the rest are discarded; from the second set, all $\mathbf{x}$ are recorded as background sample. Most PU methods can handle both scenarios, however, the derivation and interpretation

differ [3]. The work presented here follows the single training set scenario.

In their survey of the PU learning field, Bekker and Davis identify ten different assumptions that could be put into three general categories [3]. Only those relevant in the context of this thesis are outlined in this section (below).

3.5.1 Assumptions About the Labeling Mechanism

We know that each labeled example is a positive one, thus $p(y = 1|s = 1) = 1$. However, there are two possible explanations for an unlabeled one. Either it is a negative example, or it is a positive one that was not selected for labeling. Because we do not have access to that information, to make progress, we must make some assumptions about the unlabeled examples.

Selected Completely at Random (SCAR) Assumption

Before anything else, it is important to consider the labeling mechanism; that is, how did we obtain a subset of positively-labeled instances from the otherwise unlabeled data? The work presented in this dissertation makes the most common assumption from the literature known as the *Selected Completely at Random* (SCAR) assumption. Under SCAR, the labeled positive examples are chosen completely at random (or i.i.d.) from all positive examples, which means the probability of a positive example being labeled is independent of $\mathbf{x}$ [3]. This assumption is equivalent to the "missing completely at random" that Elkan and Noto [26] stated formally as:

$$p(s = 1|\mathbf{x}, y = 1) = p(s = 1|y = 1). \tag{3.19}$$

The relabeling approach detailed in Chapter 4 and the aforementioned EN and DED-PUL algorithms, all operate under this assumption.

3.5.2 Assumptions About the Data

When designing PU learning algorithms, it is useful to make some assumptions about the class distribution in the data. In practice, it is not possible to confirm if these assumptions are true for a particular dataset. However, in many cases, it has been shown that algorithms designed based on them result in improved classification results in PU learning scenarios.

Negativity Assumption

A simple approach to dealing with a PU scenario is to assume that all unlabeled examples are negative. Known as the *negativity assumption*, it is popular in PU learning because it enables the use of traditional binary classifiers with PU data [3]. However, it can result in poor classification performance when there are many unlabeled instances. An analogous assumption is common when designing music auto-tagging systems, and similar observations have been made in the literature about considerable degradation in auto-tagger performance if trained on weakly labeled data [51, 72]. Section 3.7 illustrates what happens to the performance of traditional binary classifiers trained on PU data under this assumption. All of the results obtained under this assumption are denoted NEG throughout this dissertation.

Separability Assumption

Under the *separability assumption*, there exists a threshold τ and a function $g(\mathbf{x})$ that separates the feature space into positive and negative sub-spaces, such that for all positive examples, $g(\mathbf{x}) > \tau$, and for all negative ones, $g(\mathbf{x}) < \tau$. Thus, if the positive class distribution can be estimated accurately from the given positive examples, the problem of learning a binary classifier becomes a search for the correct τ.

The relabeling approach proposed in Chapter 4 operates in a similar regime, assuming that a function $\hat{g}'(\mathbf{x})$ can be empirically estimated from positive and unlabeled examples for which the threshold τ exists that separates the function's domain into positive and negative subdomains.

Smoothness Assumption

Formally, the *smoothness assumption* states that given two instances $\mathbf{x}_1$ and $\mathbf{x}_2$, if they are similar, then so are the probabilities $p(y = 1|\mathbf{x}_1)$ and $p(y = 1|\mathbf{x}_2)$. This assumption enables the use of distance/similarity measures, such as Mahalanobis distance [105], to identify those instances that are far away from the labeled positives as reliable negative examples. There is an analogous assumption in the multi-label classification called *sample-level smoothness*. It states that if two samples are similar, then so are their predicted label vectors [5, 23].

Irreducibility

Known as the *irreducibility assumption* in PU learning, it was introduced by Scott et al. as mutual irreducibility [106]. Formally stated, the assumption is: given two distributions, $\mathcal{D}_1$ (positive) and $\mathcal{D}_2$ (negative), $\mathcal{D}_1$ cannot be a mixture containing $\mathcal{D}_2$ and vice versa. This assumption is required for identifying class prior [63] and for mixture proportion estimation [61], which are problems with similar but not identical characteristics to PU learning [63]. It is also related to proper novelty distribution [55] and max-canonical form [107],

3.6 Evaluation Metrics

In binary classification, the usual approach for comparing the performance of different classifiers is to report a summary measure computed from a confusion matrix (Table 3.1) which consists of four values: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

Table 3.1: Confusion Matrix

	actual +	actual −
predicted +	TP	FP
predicted −	FN	TN

In the literature, the most frequently reported and easily understood figure of merit is *accuracy*, $A \in [0, 1]$:

$$A = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.20}$$

Unfortunately, accuracy is not very informative in class imbalance situations, which can affect the evaluation of classification systems. For this reason, *balanced accuracy* was proposed, $BA \in [0.5, 1]$, which helps discern whether a classifier is overly biased towards the majority class:

$$BA = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) \tag{3.21}$$

Figure 3.1 illustrates the difference between traditional and balanced accuracy scores using a synthetically generated dataset. As expected, when there's an equal

number of positive and negative instances, the two scores are identical (at the 0.5 mark on the x-axis). However, we can see how the Accuracy score is overly optimistic at the extremes where the classifier becomes biased towards the majority class at the cost of miss-classifying instances belonging to the minority class. Meanwhile, balanced accuracy indicates a decrease in the classifier's performance when such bias occurs. A detailed discussion of how classification accuracy and balanced accuracy should be used can be found in Brodersen et al. [108].

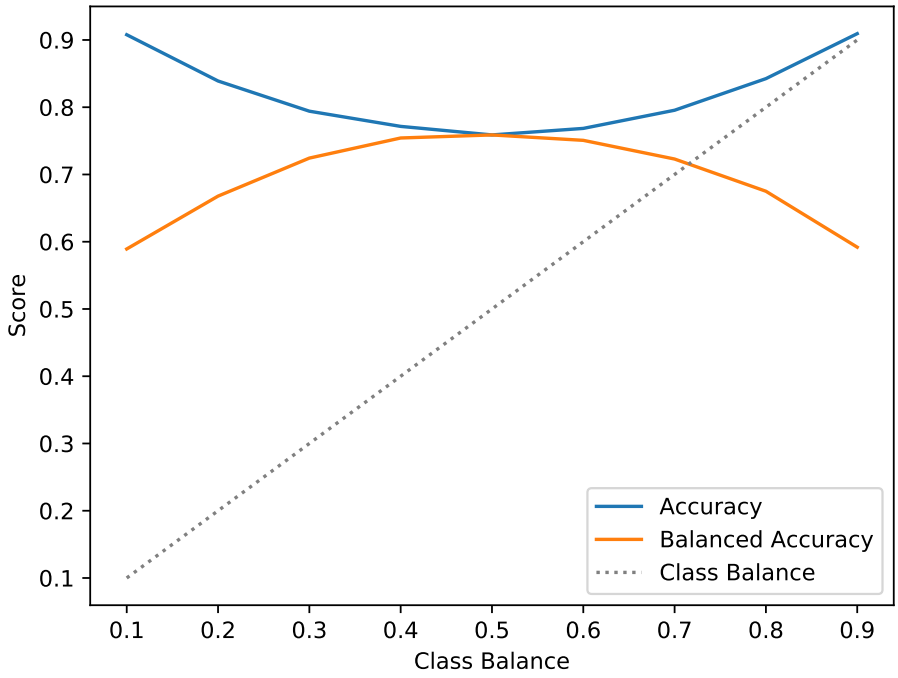


Figure 3.1: This figure compares traditional and balanced accuracy scores of the Naïve Bayes classifier in different positive-negative class balance settings using a dataset of 10,000 instances. The dataset is made up of samples drawn from two unit-variance Gaussian distributions with positives drawn from the one centered at 1 and negatives from the one centered at 2. There is an overlap between the two distributions' supports, guaranteeing some Bayesian error.

In addition to accuracy, information retrieval and machine learning practitioners often use Precision (P) and Recall (R), which are defined as:

$$P = \frac{TP}{TP + FP} \tag{3.22}$$

$$R = \frac{TP}{TP + FN} \tag{3.23}$$

The two are frequently combined into what is called the F_1-score (commonly referred to as simply F-score) and defined as the harmonic mean of precision and recall:

$$F_1 = 2\frac{P * R}{P + R} \tag{3.24}$$

Another way to evaluate a classifier's performance is to examine the tradeoff between sensitivity and specificity of its outputs. This tradeoff can be visualized by plotting a curve known as the Receiver Operating Characteristic (ROC) curve, which compares the cumulative distribution of true positives (y-axis) versus the cumulative distribution of false positives (x-axis). In other words, it characterizes what happens to the False Positive Rate (FPR) as the True Positive Rate (TPR) increases and vice versa.

$$TPR = sensitivity = P = \frac{TP}{TP + FN} \tag{3.25}$$

$$FPR = 1 - \text{specificity} = \frac{FP}{FP + FN} \tag{3.26}$$

When a classifier can produce probabilistic output, we can define a threshold discriminating between the positive and negative domains in the probability space. When this threshold is set to one of the two extremes (either 0 or 1), it forces all instances into one class (negative or positive, respectively). As we move this threshold from one end to the other, we can observe a change in either TPR, FPR, or both. When plotted on a graph with the y-axis being TPR and the x-axis being FRP, it results in the ROC curve. This curve is often summarized by reporting the area under it, denoted as AUCROC. It is possible to compute the ROC curve from discrete {0,1} classifier outputs; however, it is less accurate than the one obtained from the continuous-valued outputs.

When computing any of the evaluation metrics above, it is common to apply k-fold cross-validation (usually $k = 10$) and report the average of the metric across folds.

3.7 Simulating the Positive-Unlabeled Scenario

To evaluate the performance of PU learning approaches, one would require a positive-unlabeled dataset to train a classifier and then a new set of fully labeled instances to evaluate that classifier. Since most publicly available datasets for classification

are fully labeled, we can simulate the positive-unlabeled scenario. Starting with fully labeled data, we can un-label all negative instances and a predetermined portion of the positive ones, which results in a desired positive-unlabeled dataset. In all of the results presented in this dissertation, the positives chosen for unlabeling are selected completely at random to satisfy the SCAR assumption (i.e., the individual choices are independently and identically distributed). Since the original labels are available during the evaluation step, we can compare them with the classifier predictions (or the relabeling performance in the first step).

As an illustration, let us consider a simple class-balanced example with two Gaussian distributions, one representing positive class and the other one negative. Because much of the real-world data is not perfectly separable, let's choose the distributions' parameters such that there is some overlap between them, as depicted in Figure 3.2(a). When we train a classifier using examples drawn randomly (i.i.d.) from these two distributions, such that $p(y = 1) = 0.5$, it should attain close to the best-case performance obtainable from the *original* ground-truth labels. This result is denoted as ORG throughout this dissertation. Then, we can simulate the PU scenario under the SCAR assumption and train a classifier on the PU data as prescribed by the *negativity* assumption, which should yield a good baseline, denoted throughout as NEG. Let's see what happens to the classification boundary of a classifier (e.g., Naive Bayes) trained on the positive-unlabeled data, depicted by the dashed green line in Figure Figure 3.2. As the unlabeled proportion of positives increases, the discrimination boundary eventually passes a certain threshold, beyond which the classifier stops distinguishing between the positive and negative instances, classifying everything as the majority class.

Training on positive-unlabeled data, as opposed to positive-negative, has characteristically different effects on the different classifiers, as shown in Figure 3.3. For example, DT stands out from the rest, having a relatively linear response in its performance metrics as a function of the proportion of unlabeled positives.

3.7.1 Results of the Relabeling Approach

To illustrate how PU learning solutions can remedy the situation, we can adopt the two-step relabeling approach described earlier. In the first step, we can use a PU learning algorithm to learn from PU data and classify the unlabeled instances into

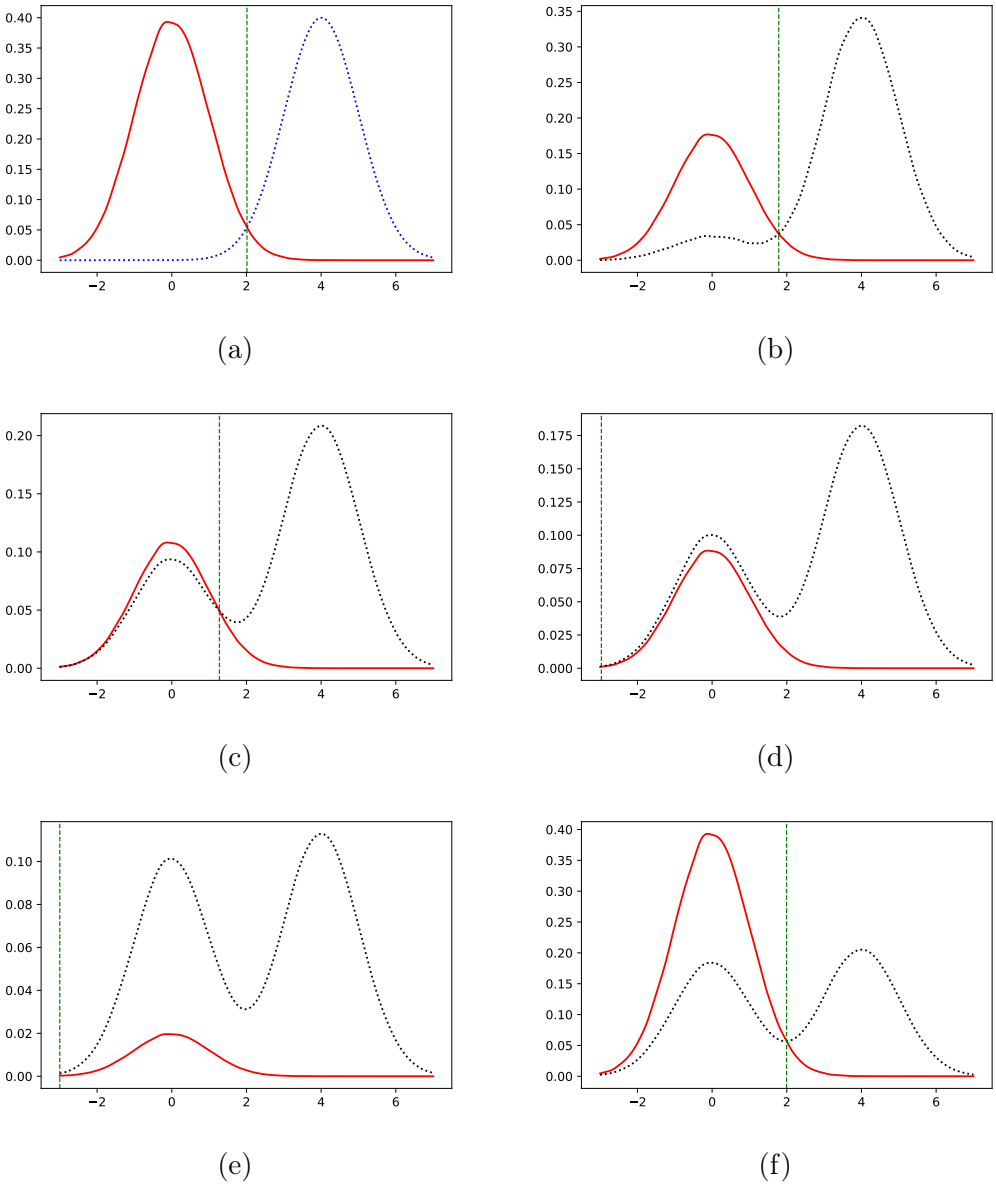
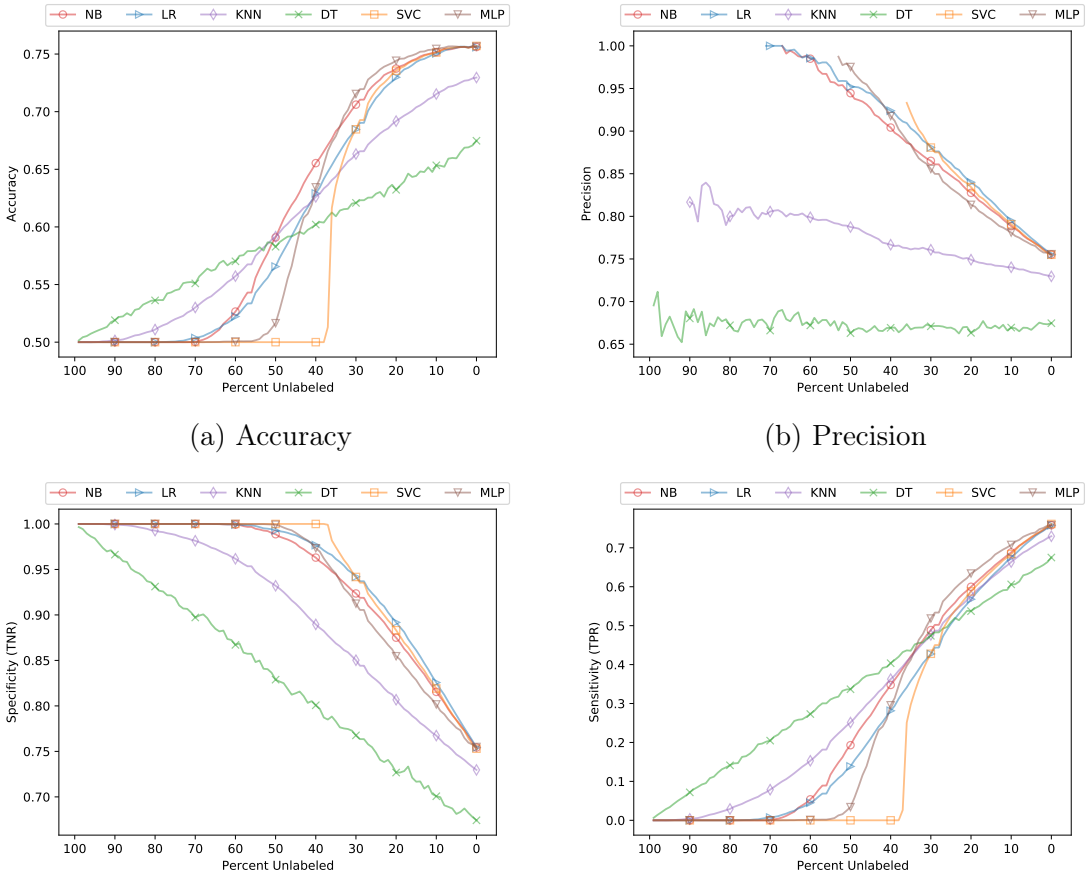


Figure 3.2: An illustration of what happens to Bayes decision boundary (green dashed line) under the naive negativity assumption (a-e) as the percentage of unlabeled positives increases, and (f) is how the approach proposed in Chapter 4 sees it. (a) The ground truth: two identical Gaussian distributions of unit variance, positive one centered at 0 and negative at 4, with a small overlap around the value 2. (b) 10% of positives and all negatives are unlabeled (grey) while 90% of positives remain labeled (red). (c) 45% of positives are unlabeled. (d) 55% of positives are unlabeled. (e) 90% of positives are unlabeled. (f) The way DEAR approach (Chapter 4) "sees" the positive and unlabeled distributions when 90% of positives are unlabeled.



(a) Accuracy

(b) Precision

(c) Specificity (True Negative Rate)

(d) Sensitivity (True Positive Rate)

Figure 3.3: The NEG classification performance of the different classifiers under the *negativity* assumption at different percentages of positive instances being unlabeled with class-balanced data consisting of 10,000 instances drawn from two univariate Gaussian whose means are two standard deviations away from each other. The ORG case, where all the data retains original (correct) labels, is at the far right point of each plot, where the value on the x-axis is 0. Note that the y-axis limits are different between these four plots.

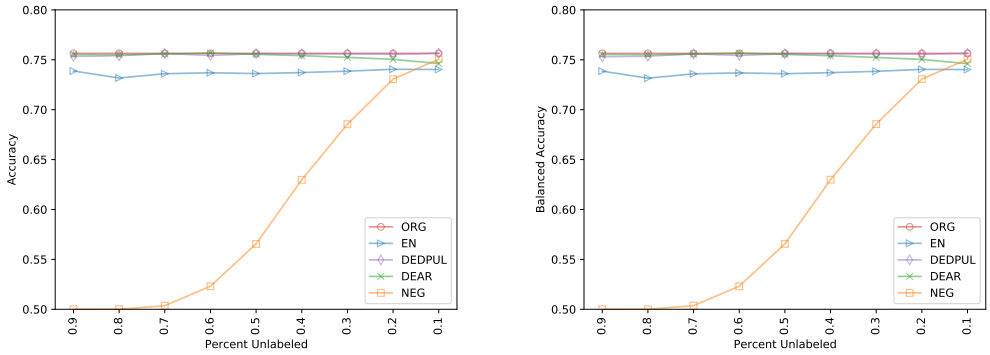
positive and negative. Then, during the second step, we can train a traditional classifier, e.g., logistic regression, on the relabeled data and see if it performs better than it would under the negativity assumption (without relabeling).

To demonstrate, let's consider another synthetic dataset that is relatively easy to classify but not trivial or completely separable. Additionally, we can demonstrate the difference between traditional accuracy and balanced accuracy by forcing some class imbalance on the original ground-truth dataset (before the simulation). In the example illustrated in Figure 3.4, we can observe how the classification accuracy (left

side) and balanced accuracy (right side) of a traditional classifier are affected with respect to the different proportions of unlabeled positives. The ORG performance is obtained using the original positive and negative labels for training without any unlabeling. The NEG accuracy is the classification accuracy obtained by considering all the unlabeled samples as negative (the negativity assumption). As is evident from the figure, the classifier performance is not particularly affected for small amounts of unlabeling (between 0% and 10%). However, as the amount of unlabeling increases beyond 30% the classifier performance is significantly affected and it is completely degraded beyond 70%, where the classifier performs no better than random choice. As evident from the figure, PU learning approaches are successful at mitigating the effect of unlabeling and improving classification accuracy over the naive NEG approach. It is also evident from Figure 3.4 that classification accuracy can be misleading for situations with class imbalance. Moreover, we can see that DEDPUL is more sensitive to class imbalance than the EN algorithm. Additionally, it is evident from Figure 3.4 that class imbalance has it's own effects on both the classification performance and on PU learning methods as the only difference between the three rows of plots is class-balance of the underlying data.

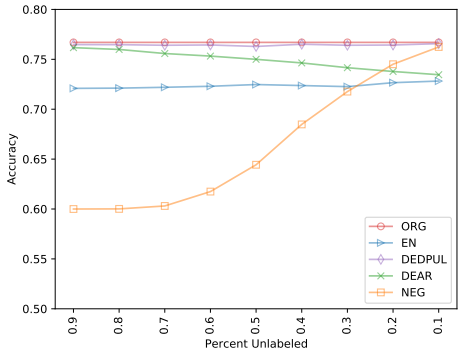
3.8 Conclusion

This chapter provided a formulation of the PU learning problem both in binary and multi-label contexts. Additionally, it included the relevant information about density estimation and classification methodology, PU learning approaches, assumptions, evaluation metrics, and an illustration of the simulation and experimental framework central to this thesis. Building on this information, Chapter 4 presents a modular and relatively lightweight approach for relabeling the unlabeled subset of positive-unlabeled data as a data preprocessing step that is taken before learning a classifier. This approach is compared with two existing algorithms from the PU literature using synthetic and real-world datasets and then in two novel scenarios that have not been previously identified as PU learning, as described in Chapters 5 and 6.

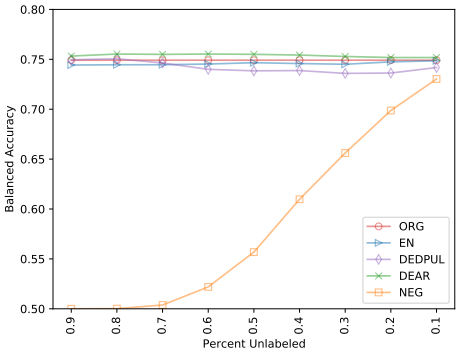


(a) Traditional Accuracy

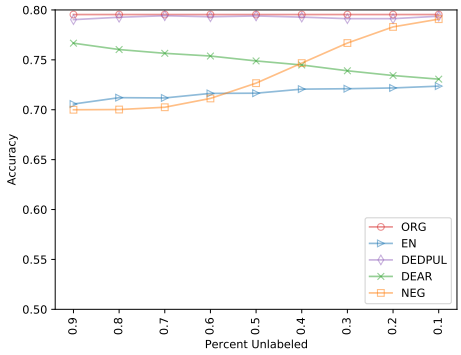
(b) Balanced Accuracy



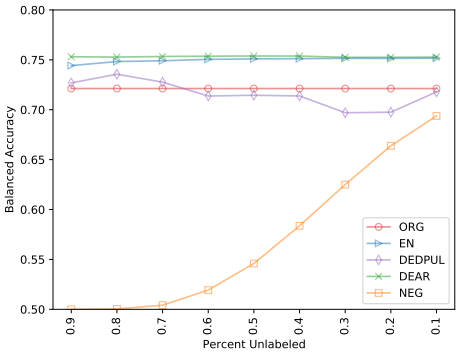
(c) Traditional Accuracy



(d) Balanced Accuracy



(e) Traditional Accuracy



(f) Balanced Accuracy

Figure 3.4: Comparison between traditional accuracy and balanced accuracy evaluation of a logistic regression model trained on positive-unlabeled data at different percentages of positive instances being unlabeled during the model training stage. The dataset in this figure was synthetically generated from two identical two-dimensional Gaussian distributions whose means are one standard deviation away from each other; the class balance is 50% positive and 50% negative instances in (a) and (b), 60% negative in (c) and (d) and 70% negative in (e) and (f). Between the three rows of figures the only parameter change is data class-balance, which clearly has it's own effects on classification performance and on PU methods.

Chapter 4

Density Estimation-based Asymmetric Relabeling (DEAR)

When working with positive-unlabeled data, using traditional supervised learning approaches can be challenging. On the one hand, traditional binary classifiers, such as logistic regression, cannot learn without negative labels. On the other hand, one-class classifiers can learn only from positive examples, but they don't take advantage of the additional, albeit unlabeled, examples. To overcome this challenge, we can preprocess the positive-unlabeled data and get it as close as possible to the correct positive-negative case; then, any preferred binary classifier could be used as usual. This two-step approach, outlined in Chapter 3 Section 3.3.1, is popular in the literature. In their survey, Bekker and Davis identify fourteen papers between 2003 and 2018 that utilize this approach [3]; however, their list is not exhaustive. This chapter proposes a Density Estimation-based Asymmetric Relabeling (DEAR) approach that fits this relabeling framework and is intended for preprocessing PU data. It uses probability distribution estimators, such as histograms, to recover additional likely-positive examples from the unlabeled data. Then, the remaining unlabeled data could be considered negative. If the goal is only to label the previously unlabeled portion of the data, then we can stop early. Otherwise, the newly labeled training dataset can be used to learn a traditional binary classifier as usual. There is also a third possibility; given a new previously unseen instance, we can assign it a positive or negative label based on the probabilistic confidence value obtained via density difference or [109] density ratio [110] between the estimated positive and unlabeled densities. However, this third case was not investigated in this thesis.

4.1 Proposed Approach

Within the two-step relabeling framework, the procedure described in this chapter focuses on the first step with the goal of improving the overall classification performance in the second step while keeping two ideas in mind: we want to use any classifier in the second step (after relabeling), and we want to avoid the added complication of adapting the training to deal with weighted instances or that of biased learning. The proposed approach also has the advantage of choosing the most suitable probability density or probability mass estimator based on the data at hand. Another advantage is that it is easy to implement using existing and well-known building blocks that are part of any statistics or machine learning practitioner's inventory.

Knowing that the unlabeled subset of the training data is a mixture of positives and negatives and the positive set is pure, we should expect that the probability density function (PDF) estimated from positive examples will have a higher density in the positive regions of the feature space than the one estimated from unlabeled examples if the two are compared on even ground. Even if there are very few labeled positive examples, as illustrated in Figure 3.2(e), the proposed approach would "see" the positive and unlabeled PDFs as depicted in Figure 3.2(f), thereby identifying many of the unlabeled examples that are likely positive. However, without knowing the class prior, we are incurring an error, which is one of the shortcomings of the proposed approach.

In the first step of the two-step framework, the proposed approach is to obtain density estimates from the positive and unlabeled examples, independently of each other, and use them to relabel the training data before learning a classifier from it, but only the unlabeled portion of the data, and only those that are likely to be positive (hence asymmetric). Identifying confidently positive examples and removing them from the unlabeled set (by labeling them as positive) increases the concentration/proportion of negative examples within it making it more negative overall. Thereby, after the asymmetric relabeling step, the remaining unlabeled examples can be considered negative with higher confidence than before. Then, in the second step of the two-step framework, a traditional binary classifier can be trained on the newly derived dataset of pseudo-positives and pseudo-negatives instead of the original positive and unlabeled one. A classifier trained on the relabeled data is expected to perform better than the same classifier trained on the PU data.

The proposed approach operates under two assumptions, which are detailed in

Chapter 3:

a) the SCAR assumption about the labeling mechanism (see Section 3.5.1),

b) and the separability assumption about the data (see Section 3.5.2).

Given that the dataset is large enough and that we chose the correct estimator, assuming a) means that the PDF estimated from the given positive examples will closely match the probability density of the underlying positive class. Assuming b) (there is a function $g(\mathbf{x})$ and a threshold τ that separates the function's domain into positive and negative subdomains) allows us to approximate that function empirically from the positive and unlabeled examples in the absence of negatives.

The experiment results presented throughout this dissertation demonstrate that this method recovers a significant proportion of missing labels in many cases, both with synthetically generated and real-world data. In terms of final classification metrics, DEAR performs overall on par with other PU learning approaches, sometimes better and sometimes worse, depending on the data (which can also be said about the PU algorithms). The time complexity of DEAR depends on the underlying density estimators. However, when using simple histograms, the proposed approach is significantly faster without giving up much efficacy.

4.1.1 Relabeling Algorithm

The proposed relabeling algorithm consists of two components: a probability density estimator suitable for the data at hand and a way of making a positive-negative decision about the unlabeled samples based on the probability density estimated independently from the positive and unlabeled samples. The relabeling algorithm, listed in Figure 4.1, makes this decision using either one of the probabilistic formulations stated in Equations 3.12 and 3.16. As for the probability density estimator, we can choose from many available in the literature so long as it provides a probabilistic likelihood for a given instance, reflecting whether it belongs to the estimated distribution or not. Thus, the approach requires that the estimator outputs a value $0 \leq e(\mathbf{x}) \leq 1$; and, given two instances drawn from two different distributions $\mathbf{x}_1 \sim \mathcal{D}_1$ and $\mathbf{x}_2 \sim \mathcal{D}_2$, s.t. $\mathcal{D}_1 \neq \mathcal{D}_2$, if the estimator is fit to $\mathcal{D}_1$ then it should output $\hat{e}_{\mathcal{D}_1}(\mathbf{x}_1) > \hat{e}_{\mathcal{D}_1}(\mathbf{x}_2)$ reflecting that $p(\mathcal{D}_1 \,|\, \mathbf{x}_1) > p(\mathcal{D}_1 \,|\, \mathbf{x}_2)$. Note that, correctly estimating an unknown distribution is an open problem and comes with it's own complication. The proposed approach assumes that the chosen estimator can fit the estimated distribution well.

Input: Training Data $(\mathbf{X}, \mathbf{Y}^{PU})$
Output: Vector $\mathbf{Y}^{new}$
1: Split $\mathbf{X}$ into $\mathbf{X}^{+}$ and $\mathbf{X}^{?}$, s.t. $\mathbf{X}^{+} \cap \mathbf{X}^{?} = \emptyset$
2: Estimate positive probability density: $\hat{e}^{+} \sim \mathbf{X}^{+}$
3: Estimate unlabeled probability density: $\hat{e}^{?} \sim \mathbf{X}^{?}$
4: Create a copy of the given labels: $\mathbf{Y}^{new} \leftarrow \mathbf{Y}^{PU}$
5: **for** $(\mathbf{x}, y)$ from $(\mathbf{X}, \mathbf{Y}^{new})$ **do**
6: **if** $y = ?$ **then**
7: **if** $\hat{f}(\mathbf{x}, \hat{e}^{+}, \hat{e}^{?}) > \tau$, **then**
8: $y \leftarrow 1$
9: **end if**
10: **end if**
11: **end for**

Figure 4.1: An algorithm that (possibly) identifies some unlabeled instances as positive. The threshold τ is a user-set value between 0 and 1 (usually 0.5), and $y = ?$ indicates an unlabeled data instance.

As listed in Figure 4.1, the input for the preprocessing subroutine consists of positive and negative examples, which are all unlabeled except for some of the positives labeled as such. Then, the two abovementioned components are implemented as follows. First (lines 2 and 3), two separate estimators of probability density, probability mass, or another one-class representation are learned independently: $\hat{e}^{+} = p(\mathbf{x} \mid y = 1)$ from positive examples and $\hat{e}^{?} = p(\mathbf{x} \mid y = ?)$ from the unlabeled ones. Second (lines 4 through 11), only the unlabeled examples are considered for relabeling based on the likelihoods provided by the discriminator resulting from combining the two estimators.

4.2 Experiments

As outlined in Section 3.7, to evaluate the efficacy of the proposed approach, the experiments simulate the PU learning setting by starting with the case where every instance is unlabeled. Then, some proportion of positive instances is selected randomly from all positive instances without replacement and labeled as positives satisfying the SCAR assumption about the data. The remainder of positive and negative instances are left unlabeled. Since we know the original (correct) labels, we can see how well the proposed approach works both during the relabeling step and in the final classification step of the two-step approach.

A sweep across different proportions (between 0% and 99%) of positives being left unlabeled reveals the effects of PU scenario on different classifiers (see Figure 3.3). The ORG classification performance is established when the number of unlabeled instances is 0%. First, for each percentage of unlabeling, the NEG result is obtained by training a classifier on the positive-unlabeled dataset under the negativity assumption (assuming all unlabeled instances are negative). Then the proposed approach is used to relabel the unlabeled set, and a new classifier is trained and evaluated. If the proposed approach has merit, we expect a classifier trained on the relabeled data to perform better than NEG.

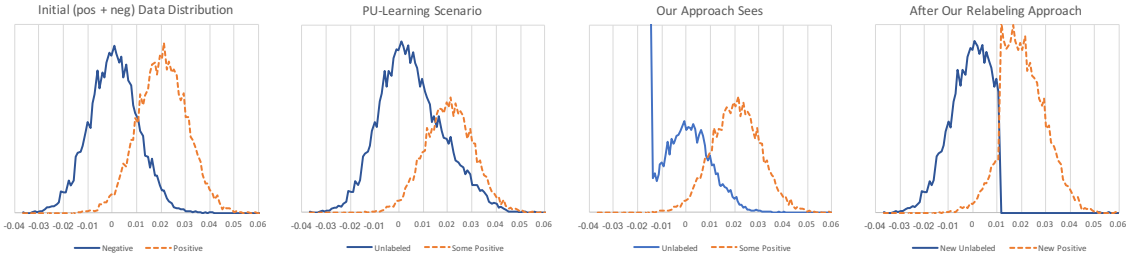
Because the data in the experiments is class-balanced, the main measure of performance at the second step of the two-step framework is classification accuracy. Additionally, several other indicators can be helpful: the count of true positives and the count of true negatives in the training data before and after running the relabeling algorithm, and the number of "unlabeled" instances relabeled by the proposed approach.

All of the experiments are performed using 10-fold cross-validation. During each fold, 90% of the data is used in the experiment, and 10% is reserved for testing the classifier during the second step of the two-step framework. Then, the NEG and ORG results are compared to the proposed approach. The experiments are implemented in Python 3 using the scikit-learn version 0.24.1 [111] machine learning library. All of the classifiers and estimators are used with default parameters, which usually yield good results.

4.3 Results With Synthetic Data

The first set of experiments uses synthetically generated data to get a sense of how the proposed approach works. The experiments are carried out in a very controlled manner with all the necessary assumptions satisfied.

Let's begin with two univariate normal (Gaussian) distributions as the underlying distributions for positive and negative classes, as described in Section 3.7. One at a time, we can vary the distribution parameters and the proportion of positives being unlabeled. Figure 4.2 shows, for one combination of these parameters, how the distribution is affected by the unlabeling of positive samples and how the proposed relabeling approach remedies the situation. The tail end of the unlabeled distribution that was "dominated" by the positive PDF became part of the positive distribution



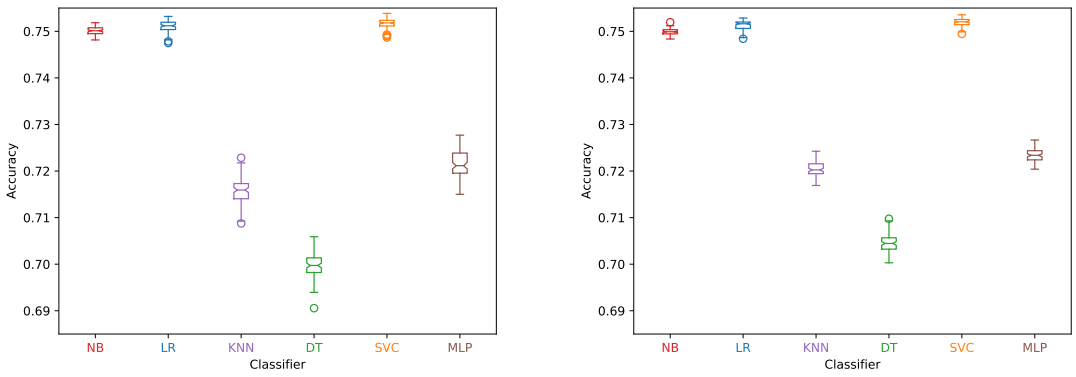
(a) (b) (c) (d)

Figure 4.2: (a) Histograms of two univariate normal distributions correctly labeled as positive (red) and negative (blue) instances. (b) The same distributions after simulating the PU situation with 30% of (unlabeled) positives labeled as negative, the way it looks under the naive *negativity* assumption. (c) The PDFs as the proposed approach "sees" them. (d) Histograms of the relabeled data, after applying the proposed relabeling method.
The final accuracy of SVM classifier trained on this data is 0.842 on the left (original), 0.740 in the middle (positive-unlabeled), and 0.838 on the right (relabeled).

in its entirety, both positive and negative instances. This relabeling shifts the classification boundary closer to the correct location at the cost of some negative examples being labeled as positive.

Even though the distribution of labels resulting from the proposed approach is not identical to the original case, the final classification accuracy is improved considerably after preprocessing the PU training data. In many cases, similar to the experiment depicted by Figure 4.2, the final classification accuracy is restored almost completely (noted in the caption).

Figure 4.3 illustrates how different classifiers perform on the data that was relabeled using a histogram on the left and a Gaussian PDF estimator on the right. Each boxplot represents 100 simulations with the same parameter settings except one; the only change was the random seed value for the experiment components involving random selection. Among the top three best performing classifiers, we could see that LR is slightly better than NB in terms of classification accuracy. However, it is difficult to tell from the figure whether SVC performs better or worse than LR. We can perform the student t-test to get an idea of how similar or different these distributions are. Table 4.1 presents the results of the standard t-test of the pair-wise comparisons between all six classifiers; however, the variances among the distributions are slightly different. Table 4.2 shows the results of Welch's adaptation of the standard t-test to handle different variances between the two distributions under consideration. Even



(a) Histogram PMF (b) Gaussian PDF

Figure 4.3: The dataset for this figure was synthetically generated from two identical two-dimensional Gaussian distributions whose means are one standard deviation away from each other; the class balance is 50% positive and 50% negative instances. Then, 50% of positives were unlabeled (along with negatives) to simulate the PU scenario. This PU data was then relabeled using the proposed approach with Histogram estimators on the left and a Gaussian PDF estimate on the right. The experiment was performed 100 times with different random seeds for all the randomized components (data generator, 10-fold cross-validation with shuffle, selecting positives for labeling (SCAR assumption).

though the p-values are slightly different between the two tables, it is clear that the difference between all classifiers is statistically significant, but to a much lesser degree between SVC and LR (highlighted in bold).

Table 4.1: Equal variance t-test, DEAR+HISTOGRAM, Accuracy.

	clf / clf	NB	LR	KNN	DT	SVC	MLP
t-test statistic	NB	0.00e+00	-6.27e+00	1.23e+02	1.78e+02	-1.05e+01	1.03e+02
	LR		0.00e+00	1.23e+02	1.76e+02	**-3.57e+00**	1.03e+02
	KNN			0.00e+00	4.24e+01	-1.26e+02	-1.52e+01
	DT				0.00e+00	-1.79e+02	-5.75e+01
	SVC					0.00e+00	1.06e+02
	MLP						0.00e+00
p-value	NB	1.00e+00	2.25e-09	4.02e-189	2.81e-220	6.45e-21	1.12e-173
	LR		1.00e+00	1.03e-188	2.97e-219	**4.41e-04**	1.05e-173
	KNN			1.00e+00	2.50e-101	6.83e-191	3.46e-35
	DT				1.00e+00	5.25e-221	1.75e-125
	SVC					1.00e+00	4.04e-176
	MLP						1.00e+00

Table 4.2: Welch's (unqual variance) t-test, DEAR+HISTOGRAM, Accuracy.

	clf \ clf	NB	LR	KNN	DT	SVC	MLP
t-test statistic	NB	0.00e+00	-6.27e+00	1.23e+02	1.78e+02	-1.05e+01	1.03e+02
	LR		0.00e+00	1.23e+02	1.76e+02	**-3.57e+00**	1.03e+02
	KNN			0.00e+00	4.24e+01	-1.26e+02	-1.52e+01
	DT				0.00e+00	-1.79e+02	-5.75e+01
	SVC					0.00e+00	1.06e+02
	MLP						0.00e+00
p-value	NB	1.00e+00	2.47e-09	4.67e-129	2.21e-147	8.32e-21	1.61e-119
	LR		1.00e+00	1.47e-139	3.93e-159	**4.41e-04**	2.76e-129
	KNN			1.00e+00	2.63e-101	1.72e-137	3.46e-35
	DT				1.00e+00	3.13e-156	1.86e-125
	SVC					1.00e+00	1.05e-127
	MLP						1.00e+00

Another question arising from Figure 4.3 is whether the histogram (left) or the Gaussian PDF estimator (right) performs better and with which classifier. Student t-test results presented in Tables 4.3 and 4.4 reveal that the latter performs best with all but the Naïve Bayes classifier.

Table 4.3: Equal variance t-test, DEAR, Accuracy, Histogram vs. Gaussian

Histogram \ Gaussian	NB	LR	KNN	DT	SVC	MLP
statistic	2.40e+00	-1.15e+00	-1.53e+01	-1.45e+01	-2.52e+00	-6.56e+00
p-value	1.73e-02	2.52e-01	2.12e-35	5.44e-33	1.25e-02	4.70e-10

Table 4.4: Welch's (unqual variance) t-test, DEAR, Accuracy, Histogram vs. Gaussian

Histogram \ Gaussian	NB	LR	KNN	DT	SVC	MLP
statistic	2.40e+00	-1.15e+00	-1.53e+01	-1.45e+01	-2.52e+00	-6.56e+00
p-value	1.74e-02	2.52e-01	1.01e-32	4.79e-32	1.25e-02	8.46e-10

Figure 4.4 illustrates a more extreme case with bi-variate distributions where very few positive examples are labeled. Moreover, the positive and negative distributions have a considerable overlap and different variance from each other. The proposed approach recovers a large portion of positive examples. However, the bottom right plot of Figure 4.4 clearly illustrates the drawback of using a univariate estimator as opposed to one that could handle multiple dimensions in the feature space.

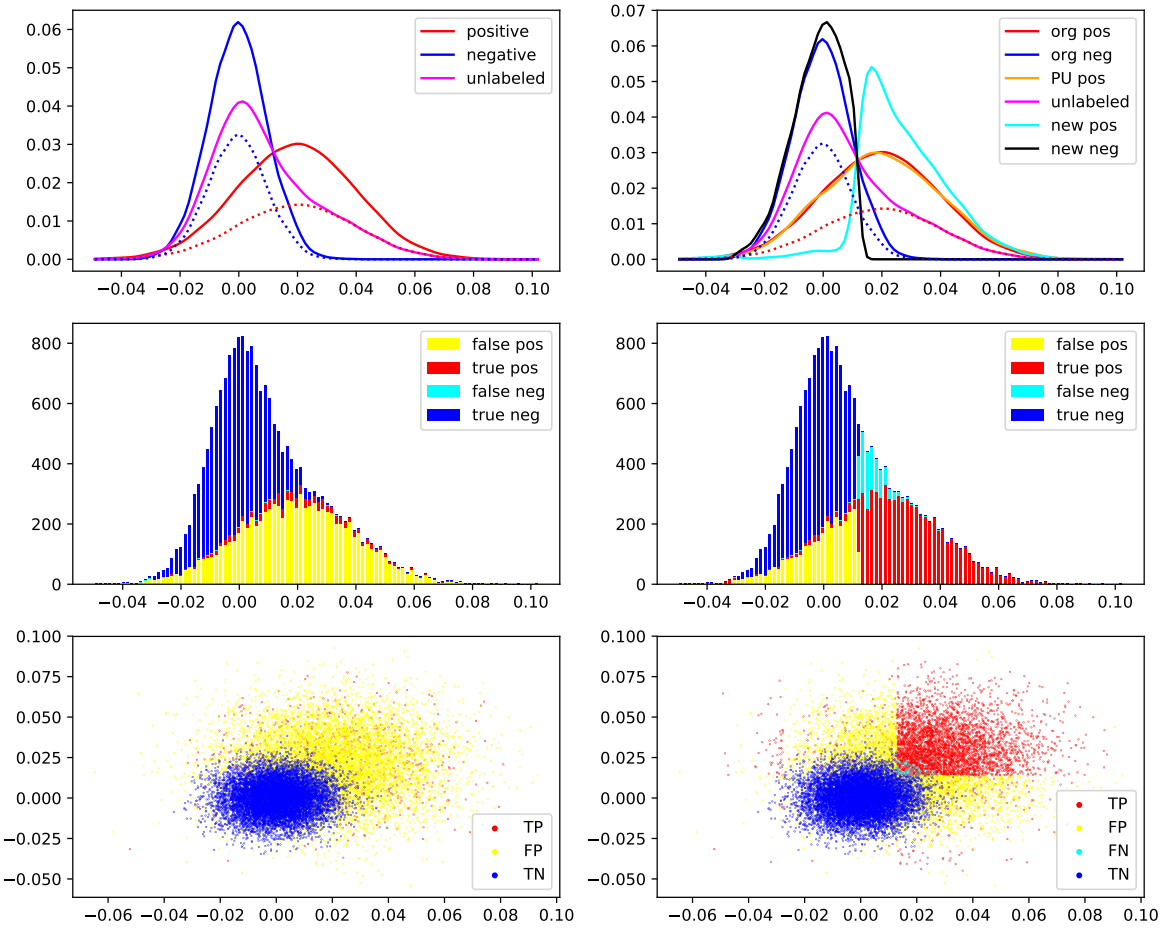


Figure 4.4: This is an example of a more severe PU situation with a considerable overlap between the positive and unlabeled distributions and very few labeled positive examples to learn from; additionally, the true positive and negative PDFs have different variances. The dotted lines in the two upper images depict the positive (red) and negative (blue) components of the unlabeled PDF. The two middle images are stacked bar plots with colored areas representing the relative proportions of TP, TN, FP, and FN. The two lower plots show each instance's position in the two-dimensional space of the underlying variables. The three plots on the left side illustrate the data before relabeling (few TP, no FN) and the right side - after (many more TP at the cost of some FN). This figure illustates how the initial version of the proposed approach with the basic histograms and majority voting [85] can identify a significant proportion of positive instances from the unlabeled ones.

Experimental results indicate that the performance of different classifiers was degraded in response to both class prior imbalance and the proportion of unlabeled positives; the higher the imbalance/proportion, the worse the performance. Class imbalance reflected negatively in both steps of the two-step framework and negatively affected all three configurations, ORG, NEG, and the classifier performance after relabeling. However, in most situations, the relabeling treatment resulted in better performance than one would obtain under the negativity assumption (NEG), often almost completely recovering the final classification performance. It's also worth noting that when Bayes error rate is high, and the amount of unlabeled positive is very small or zero, the final classification performance is actually worse than with the original (untreated) data. Also, as expected, the overall system performance increases when the number of explanatory variables is increased. Using a Laplace distribution in place of a Gaussian distribution yielded very similar results.

4.4 Results With Real-Word Data

To test the applicability of the proposed approach in real-world situations, we can use some of the many publicly available datasets that represent real problems that practitioners in the field aim to solve. The ones selected for the experiments presented here are summarized in Table 4.5 and described below. Four of these reflect the single-label paradigm and were obtained from the UCI Machine Learning Repository[1]. The other two datasets are from the multi-label paradigm and were obtained from the MULAN multi-label dataset collection[2] [112]. In the multi-label case, each label representing a class concept is treated as an independent binary classification problem, and only those labels were selected for experiments that are represented by at least $10,000$ positive and $10,000$ negative instances. To control for adverse effects of class imbalance, all of the datasets were randomly sub-sampled with equal (balanced) class representation.

4.4.1 Data Description

HEPMASS [113] is a dataset from the area of high energy physics, it is a subset of data introduced by Baldi et al. in their search for exotic particles. The original

[1] `https://archive.ics.uci.edu/ml/index.php`

[2] `http://mulan.sourceforge.net/datasets-mlc.html`

Table 4.5: The real-world datasets in our experiments. Final column indicates the randomly sampled and class balanced sub-set of the original data from which the results are obtained.

Dataset Label	Number of Features	Positive Instances in Data	Negative Instances in Data	Instances in Experiments, Class-balanced
HEPMASS	28	1,750,000	1,750,000	20,000
HTRU-2	8	1,639	16,259	3,278
SKIN_SEG	3	50,859	194,198	20,000
SUSY	18	2,500,000	2,500,000	20,000
MEDIAMILL_33	120	27,938	15,969	20,000
MEDIAMILL_65	120	15,080	28,827	20,000
MEDIAMILL_66	120	15,690	28,217	20,000
NUS_WIDE_1	128	20,404	141,385	20,000
NUS_WIDE_8	128	10,579	151,210	20,000
NUS_WIDE_13	128	32,418	129,371	20,000

dataset is composed of several subsets, the results reported here are from the one that closely resembles binary classification, where the signal particle has a $mass = 1000$. Note that particle mass is not included in the attributes. The other subsets have multiple values for a signal particle, which is analogous to the multi-class problem and not directly suitable for PU approaches.

HTRU-2 [114] resulted from the High Time Resolution Survey conducted to collect pulsar candidates, which are rare types of Neutron star whose radio emissions are detectable on Earth [12]. The dataset contains 1,639 real pulsar examples among the spurious examples caused by RFI/noise. Each instance is described by 8 statistical characteristics drawn from each candidate observation.

SKIN_SEG [115] was originally used in the image segmentation task specifically to train an algorithm for identifying the precise regions of an image containing human skin [116] and for adaptive digital makeup [117]. Each instance in this dataset is described by three attributes corresponding to the amount of red, green, and blue components of an RGB value describing a color.

SUSY [118] was introduced by Baldi et al. as part of their search for exotic particles in the area of high energy physics. Their work is in the area where researchers study the basic constituents of matter. Because of the statistical nature of this work, machine learning is proven to be useful in the field. This dataset consists of two types

of instances, ones where a signal particle is present and the others a background noise. It is the benchmark dataset for the supersymmetry particle.

NUS_ WIDE [119] is a collection of images annotated with labels from 81 concepts. The experiments use the second version, which is distributed with cVLAD features [120]. This is a multi-label dataset; hence we transform it into multiple binary classification datasets via binary relevance decomposition. Labels 1, 8, and 13 are selected for experiments and denoted by the number following the dataset name (e.g. NUS_WIDE_1).

MEDIAMILL [121] dataset was originally introduced to evaluate performance of generic video indexing using 101 semantic concepts and to gain insight into intermediate steps that affect the performance of multimedia analysis methods. Similar to NUS_WIDE, this multi-label dataset is decomposed via binary relevance, and only labels 33, 65, and 66 were selected, which is reflected in their corresponding names (e.g. MEDIAMILL_33).

To maintain class balance and have a reasonable sample size, $10,000$ instances of each class were randomly chosen for experiments from each dataset except HTRU-2. There are only $1,639$ positive instances in HTRU-2, which set the limit for the sample size.

4.4.2 Experiment Results

It should be noted that the results reported here are very optimistic because the selected data is class-balanced. When the class priors are $p(y = 1) = p(y = 0) = 0.5$, Equations 3.1 and 3.2 simplify to:

$$f(x) = \begin{cases} 1, & \text{if } p(x|y = 1) > p(x|y = 0) \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

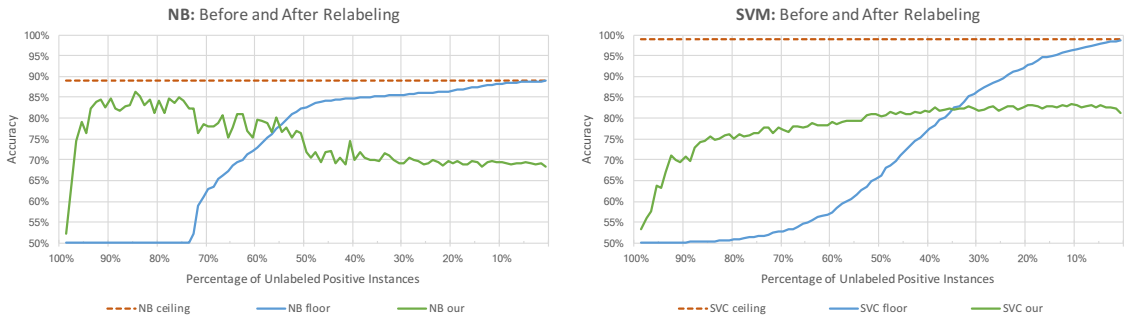
This means that likelihood alone is enough to make good predictions. It also means that after relabeling, if there is roughly the same number of pseudo-positive as compared to pseudo-negative instances, then the class prior is automatically known to the traditional binary classifier. This class-balanced situation is a special case, and in situations with class imbalance, the system performance becomes increasingly worse as the class prior deviation from 0.5 increases.

The classification accuracy reported in Table 4.6 constitutes the upper bound

(ORG) performance for the experiments. The expectation is that if we run our algorithm on a perfectly labeled dataset, we would inevitably degrade the situation if any of the true negative instances become relabeled as positive. Thus, we should not hope to achieve a better performance than ORG, although sometimes it happens in practice. The results in Table 4.6 suggest that SUSY is the hardest dataset to model as four out of five classifiers achieve their worst performance with it as compared to all other datasets.

Table 4.6: The upper boundaries of classification accuracy (ORG) for the different classifiers with the real-world datasets.

Dataset	SVM	kNN	DT	NB	ANN
HTRU-2	0.7727	0.9245	0.9040	0.9071	0.9267
SKIN_SEG	0.9904	0.9985	0.9976	0.8892	0.9981
SUSY	0.7920	0.7399	0.7044	0.7189	0.7915
HEPMASS	0.9099	0.8852	0.8611	0.8909	0.9015



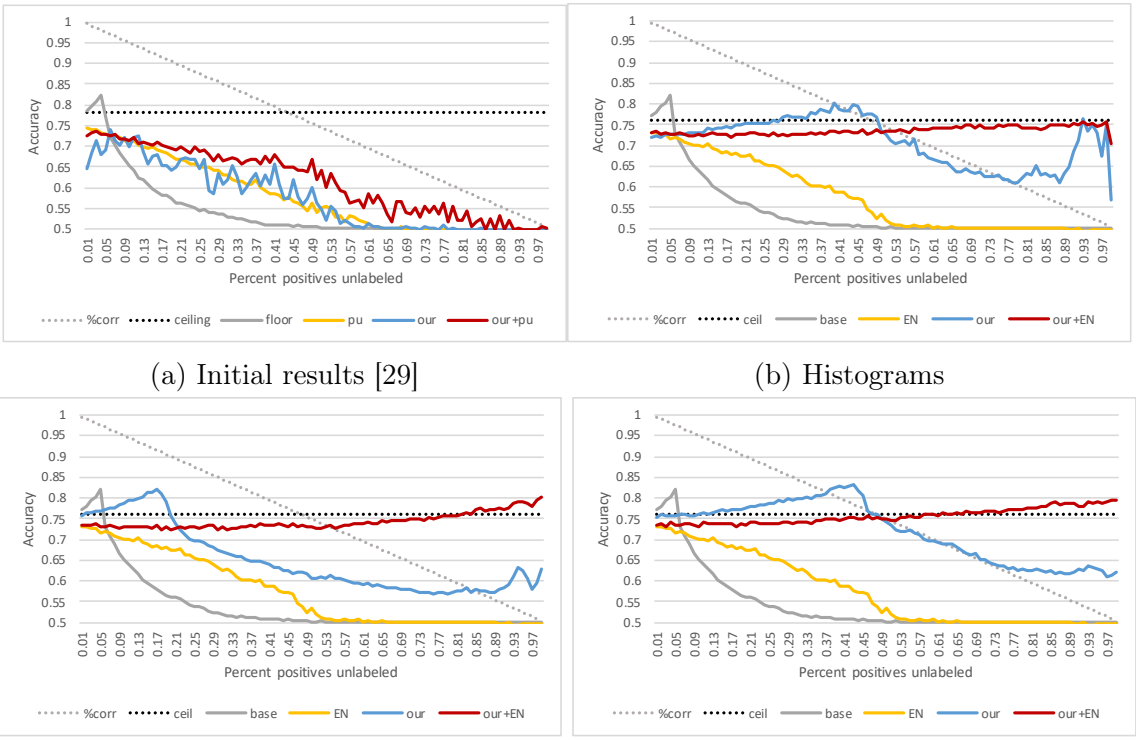
(a) Naïve Bayes classifier (b) SVM classifier

Figure 4.5: Classifier accuracy with SKIN_SEG dataset before (blue) and after (green) treating the PU training data using the proposed relabeling approach (with Histograms).

From the real-world datasets described in the previous section, SKIN_SEG resulted in the highest overall accuracy in all experiment configurations. Furthermore, from the non-parametric probability density/mass estimators, KDE showed the best overall performance but was also much slower than the histogram estimator, which was faster without giving up much accuracy.

The HTRU-2 dataset was challenging in the experiments with the initial version of the proposed approach [85]. Changing the estimator combination strategy from a

simple majority vote (3.13) to the one based on probabilities (3.15) resulted in a considerable improvement in classification accuracy. This can be seen in Figure 4.6, which also shows that non-parametric probability density/mass estimators outperform the univariate Gaussian one. These observations suggest that some of the informative features in the HTRU-2 dataset do not follow a Gaussian-like distribution.



(a) Initial results [29]

(b) Histograms

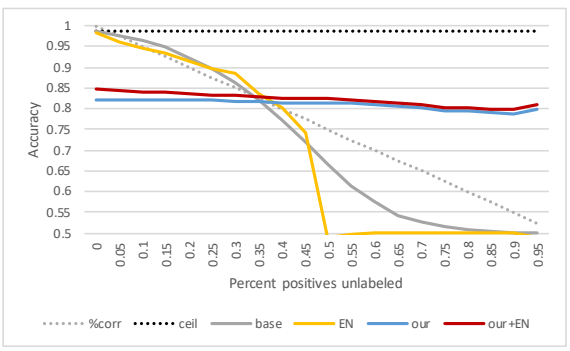
(c) Univariate Gaussian

(d) KDE

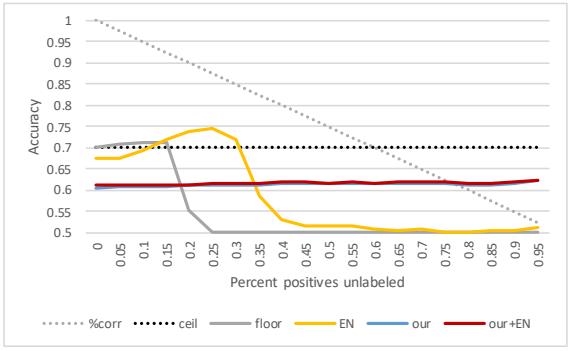
Figure 4.6: Comparing different density estimation and feature combination rules using SVM with class-balanced HTRU-2 dataset at different proportions of unlabeled positives.

Figure 4.7 shows the KDE performance for relabeling different real-world datasets. It compares the proposed approach (DEAR) to EN and shows what happens when both are used together, DEAR followed by EN.

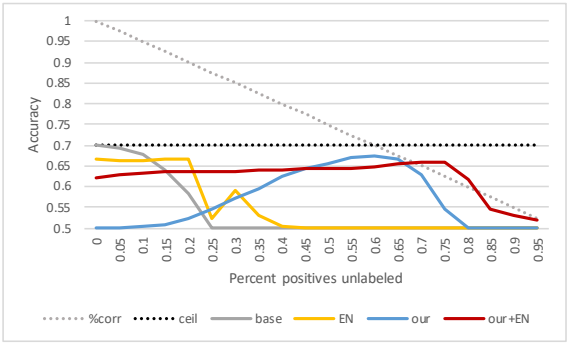
In all of the experiments, when most of the positive labels are present (very few unlabeled positives), both DEAR and EN result in accuracy that is lower than the NEG performance. However, as the number of unlabeled positives increases, at some point, both approaches surpass NEG. In the experiments in this chapter, EN estimator uses SVM to estimate c (see Equation 3.3.2), which makes the results comparable to the SVM classifier on its own (Figures 4.6 and 4.7).



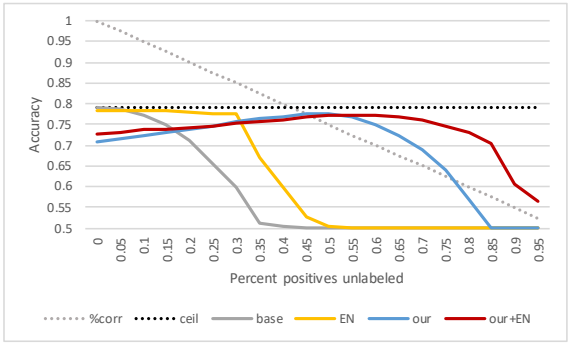
(a) SKIN_SEG dataset



(b) MEDIAMILL-33 dataset



(c) NUS_WIDE_8 dataset



(d) NUS_WIDE_13 dataset

Figure 4.7: Accuracy using KDE for relabeling.

Experimental results suggest that using EN with the data that was relabeled by DEAR usually works best. In some occasions, the relabeling step causes more harm than gain, especially when a small proportion of positives is unlabeled. This can be seen, for example, in Figure 4.7b where fewer than 35% of positives are unlabeled.

4.5 Conclusion

This chapter presented a density-based asymmetric relabeling approach for treating the positive-unlabeled data as a data preprocessing step ahead of training a classifier, where we start with a relatively few labeled examples and use them to identify which of the unlabeled instances are likely positive. The experiment results demonstrate that, in many cases, this method recovers a significant proportion of missing labels, both with synthetically generated and real-world data. The results show that this approach is especially helpful when the proportion of unlabeled positives is large, which often is the case where researchers are interested in the few (rare) examples of some phenomena hidden within an abundant amount of mundane (uninteresting) samples, such as in the presence-only scenario.

The proposed approach has the following advantages. First, it allows the use of any traditional binary classifier and the use of domain-specific evaluation metrics. Thus, if practitioners in a given area find a specific methodology most effective, they can go ahead and use their favorite tools within this framework. Second, it allows a choice of density estimator suitable for the features. All of the estimators tested in this chapter are of generative type. Generative models are especially useful because they don't memorize the data and constitute a compact representation of the underlying probabilities space. Third, it allows the added flexibility of giving different weights to different attributes when the final labeling decision is made. This also allows the use of other attribute-based assumptions about the labeling mechanism. Finally, this setup is modular and allows for choosing the best combination of components for the task at hand.

It's worth noting that the proposed method requires only one pass across the data, and subsequent passes do not identify any additional unlabeled positive instances. However, it could be possible to adapt it for use with an expectation-maximization algorithm.

4.5.1 Future Directions

There are several directions for future work involving the proposed approach. Domain experts almost always have a good idea about the proportion of positive and negative examples in their data. Different scenarios with either user-specified or estimated class priors and their influence on the proposed approach could be investigated. If the class prior can be provided with some reliability, the proposed approach can be used directly as a classifier without requiring a re-training step and an additional classifier.

The main underlying assumption behind the design of the proposed approach is SCAR. We believe that the proposed approach can be adjusted to take advantage of the SAR or the probabilistic gap assumption instead. These assumptions are less restrictive and would provide more theoretical support about why and how the method works.

During the decomposition of the features, DEAR makes the assumption that they are independent and therefore uses density estimators for each feature independently. This is similar to the "naive" assumption in Naïve Bayes classification and allows for a simple and modular implementation. In many real-world scenarios, this assumption does not hold, and some information can be lost. Another similar situation occurs during data transformation, in which label correlations can also occur. It is possible to design a scheme that would take into account these correlations and incorporate it somehow in the relabeling process.

We believe that we have compared the proposed approach with sufficient alternatives and over several datasets to demonstrate its potential. In the other chapters of this thesis, we show how it can be used and how it compares with alternatives in more real-world scenarios from healthcare and music information retrieval. A more thorough comparison with other approaches and with more datasets could be conducted in order to provide more information about which methods are better and under what conditions; besides DEAR, this is a desired contribution in the PU learning field [3].

Chapter 5

Predicting Patient Length of Stay in Hospitals

Like in many other domains, PU situations occur naturally in healthcare; and the presence of unlabeled examples in the training data can significantly decrease the performance of a predictive model if they are treated as negative samples without applying specific PU learning strategies. This observation is also supported by expert opinion; for instance, Chen et al. obtained a unanimous agreement from practitioners working in the field of Alzheimer's Disease that diagnosing it is better formulated as PU rather than the traditional binary classification ML problem [4]. PU situations are prevalent across many healthcare areas, including but not limited to disease gene identification [8, 9], drug activity prediction [5], Alzheimer's Disease diagnosis [4], and prediction of circRNA disease associations [10]. This chapter looks at one such area where PU situations occur but have not yet been identified as such.

Accurately predicting patients' *length of stay* (LOS) at the time of their arrival to a healthcare unit can help with care planning and resource management, ultimately resulting in a positive impact on patient outcomes. For instance, it is one of the central topics in clinical pathways [64]. With the advent of Electronic Health Records and the gradual shift towards sharing and centralizing the storage of patient records in electronic format among independent healthcare units, like hospitals, pharmacies, etc., each patient's healthcare record becomes more and more complete and informative. This trend also improves the applicability of machine learning approaches in healthcare settings. LOS is relatively easy to predict for some routine procedures, such as dental prophylaxis, but not for others, such as trauma or emergency general surgery,

where ML approaches can be beneficial [122]. This problem is often formulated as binary classification [65] and, more recently, as a multi-class one [69], where the algorithms learn from data, such as electronic health records, patient demographics, and clinical characteristics, to predict the LOS as one of two or more discrete number of classes. Recently, there is an increasing interest in predicting LOS in hospitals due to the Covid-19 pandemic. For example, Wu et al. perform a multi-variable regression analysis to study the factors influencing LOS among Covid-19 patients [70]. A survey and synthesis of the length of stay datasets and studies for Covid-19 can be found in Rees et al. [71].

During regular hospital operations, and when changes in the patient characteristics and statistics are gradual, it is possible to collect reliable data over a certain period for training ML models. However, in some situations, the changes in the data statistics are so significant that they negatively affect the models' predictive performance. Examples of such scenarios include the onset of a pandemic when historical data is not informative, when a hospital first begins operating and has no historical data available, or when a new treatment drastically affects the LOS. During such cold-start transition periods, reliable training examples become available gradually over time. First, we start with all of the data being unlabeled, then move across positive-unlabeled scenarios with varying class distributions, and eventually end up a stable state with mainly positive and negative, but also some unlabeled samples. This chapter investigates how the predictive performance of a machine learning system could be negatively affected during such cold-start transition periods and how to leverage PU learning techniques to remedy this degradation.

To motivate the approach and the experiments described in this chapter, let us consider an imaginary but realistic example where we want to train a classifier to predict whether a patient will remain in the hospital for less than 30 days (*short stay*) or more than 30 days (*long stay*), but we don't have any patient data yet (e.g., a new hospital just opened). Thus, at the beginning of the first day, we do not have any reliable examples of neither short nor long stays, and until at least one patient is discharged, all of our data is unlabeled. During the first 30 days of the hospital operation, as time passes and patients complete their hospital stay, we can start collecting reliable examples for the short stay category. However, for the newly arrived patients and those who have not yet been discharged during that period, we do not know in which category they will end up; thus, we consider their data as unlabeled. Until a patient crosses the 30-day mark while still in the hospital, we have

no reliable examples of long stays. Suppose our hospital can accommodate patients only for the short term. In that case, it can be useful to know early on who fits this category and who needs to be transferred to another unit. This information can be particularly important during novel response scenarios such as the Covid-19 pandemic, where healthcare resources become suddenly overwhelmed with a surge of patients whose statistical characteristics are not yet captured by the currently deployed classification system.

5.1 Proposed Approach

When time is of the essence, in such cold-start transition periods, one obvious solution is to start training the new system right of way, even with the lack of any long stay examples. A naive approach would assume that all of the unlabeled samples are negative and train the new classifier as soon as there are enough short stay examples. This idea is known as the negativity assumption, detailed in Chapter 3 Subsection 3.5.2. It is the weakest assumption in PU literature that usually results in the worst-case classification performance, although often better than random. All of the results obtained under this assumption are denoted NEG throughout this thesis. As demonstrated in the literature, a better solution is to estimate the likely positive and negative instances from all of the unlabeled samples, label them accordingly, and train the new classifier on the relabeled data. This approach, detailed in Chapter 3 Subsection 3.3.1, is common in PU learning literature because it enables the use of a traditional binary classifier or any other that might be specifically designed for the data at hand or otherwise preferred by the practitioner.

5.2 Data

To verify the efficacy of the proposed approach, ideally, we could monitor hospital records on a daily basis, starting at an onset of a cold-start situation, and use the proposed approach to classify all of the patients and compare the classification results with the actual discharge records at a later day. However, this is not always practically feasible, with the biggest barrier being the restriction on access to such data for many good reasons, for instance, to protect patient privacy. As machine learning becomes increasingly useful in healthcare settings, so increases the amount of effort directed to alleviate these barriers by collecting, anonymizing, and making publicly available

the different kinds of real-world data that effectively describe healthcare issues. We use two such datasets (described below) that are characteristic of the LOS scenario. Because the datasets are fully labeled, we can use them to simulate PU situations that arise during cold-start transition periods.

5.2.1 Janatahack Dataset

The dataset denoted as Janatahack was collected in response to the alarms raised by healthcare management across the world due to the recent Covid-19 pandemic causing undue pressures on our healthcare systems. This dataset is publicly available and was released as part of the "Janatahack: Healthcare Analytics II" challenge [123]. It consists of 318,438 patient records described by 18 attributes. However, two of the attributes, Case ID and Patient ID, are not relevant for classification tasks, which leaves one target and 15 descriptor variables for modeling.

Some of the attributes are categorical/nominal; therefore, we use one-hot encoding to convert each of them into multiple boolean attributes, which are more suitable for most classification algorithms. One of the categorical attributes is the target variable (LOS), where each category corresponds to a 10-day range, with the last one being greater than 100 days. See Table 5.1 for counts of instances per LOS category.

Since we are interested in short-term versus long-term, we group the 11 categories into two: less than 30 days and greater than 30 days. This threshold is arbitrarily in the context of this chapter, but it could be chosen with specific considerations in mind since the meaning of short-term and long-term changes depending on the situation. For instance, nursing homes have a much longer acceptable LOS than an intensive care unit.

5.2.2 NYDSOH Dataset

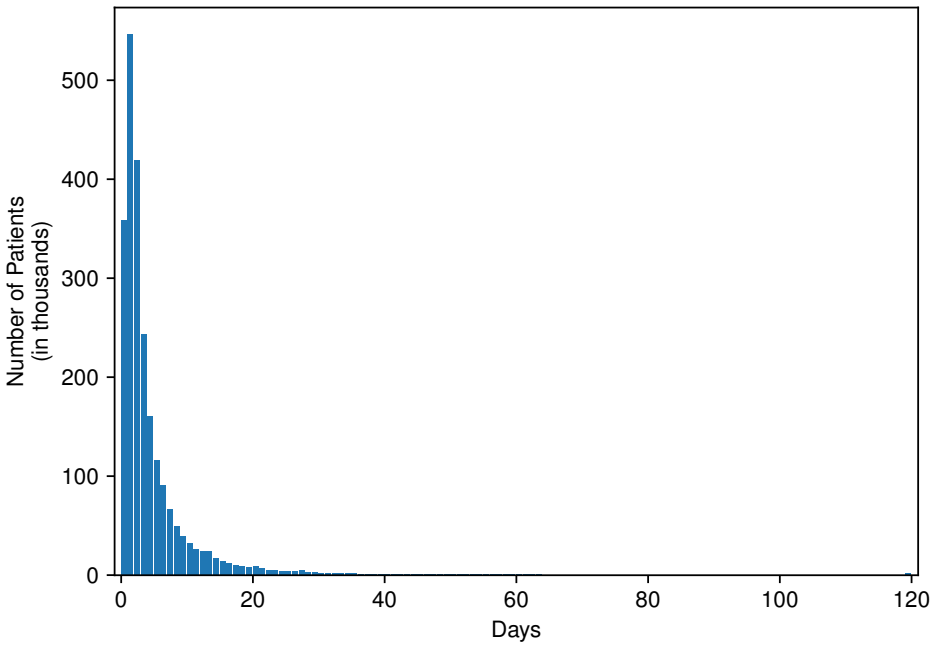
The dataset denoted as NYDSOH-2017 is publicly available through the Statewide Planning and Research Cooperative System (SPARCS). It contains de-identified discharge-level details on patient characteristics, diagnoses, treatments, services, and charges for 2,343,569 patients admitted in 2017 to the State of New York Hospitals [124]. There are a total of 33 attributes describing each patient in the dataset. After removing financial and geographic information about each visit and field descriptions, which are not useful for this task, there are 15 attributes left. All of the remaining attributes are categorical except one, birth weight, which is expressed as an integer

with zero representing those patients who are not newly born during the corresponding visit. As part of data preprocessing, the categorical attributes are converted into multiple boolean ones via the one-hot encoding technique, which results in a total of 876 boolean attributes for each patient. Because individually identifiable information such as the admission and discharge date and time of day is not present in the dataset, the LOS is provided as the total number of patient days calculated as $(DischargeDate - AdmissionDate) + 1$. Also, all stays greater than 120 days are binned together with the label "120 +" and comprise 0.1% of the data. The LOS distribution, depicted in Figure 5.1a, follows a power-law-like distribution with more than 73% of the patient discharges within the first 5 days of stay. Figure 5.1b depicts the same distribution but with the y-axis following the logarithmic scale, which reveals the power-law-like character of the distribution. For further information about the power-law distribution and its characteristics, see Clauset et al. [125]. Table 5.2 shows how the LOS attribute maps to the binary class labels for the experiments described in this chapter.

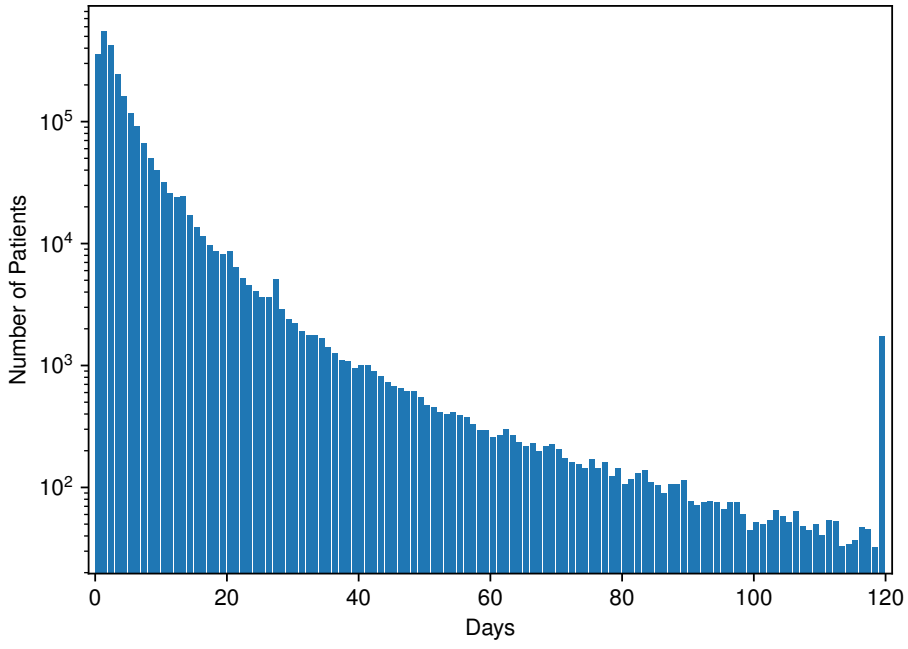
5.3 Cold-start Simulation

To simulate the PU learning scenario that arises during cold-start transition periods, we start with a fully labeled training set with labels corresponding to the binary LOS splits shown in Tables 5.1 and 5.2. Then, we randomly choose some of the positive instances to form the positive set P while the remaining positive and negative training instances form the unlabeled set U, thereby creating a PU dataset that resembles the real LOS scenario as described above. We can then use PU learning algorithms to estimate the latent class boundary and provide labels for the unlabeled portion of the data and then train a traditional binary classifier on the relabeled data. We use this classifier to label the testing set that represents the newly arriving patients and compare the results with the ground truth, which is available to use because the original data is fully labeled. Because patients could be arriving at different rates, the percentage of unlabeled cases fluctuates in reality. Thus, we sweep across different percentages of unlabeled positives and evaluate the effectiveness of the proposed approach to predicting LOS in the corresponding cold-start scenarios.

When the percentage of unlabeled positives is zero, we obtain the optimal classifier performance without using the relabeling approach since the data is fully labeled. We denote this as the *ORG* performance in the results below. Opposite of ORG is the



(a) LOS distribution in NYSDOH-2017 dataset.



(b) LOS distribution in NYSDOH-2017 dataset. Here the y-axis is in logarithmic scale to demonstrate power-law-like character of the distribution.

Table 5.1: Number of instances in each class in the Janatahack dataset.

Label	0-10	11-20	21-30	31-40	41-50	51-60	61-70	71-80	81-90	91-100	>100
Original	23,604	78,139	87,491	55,159	11,743	35,018	2,744	10,254	4,838	2,765	6,683
Binary	189,234 (59.4%)			129,204 (40.6%)							

Table 5.2: Number of instances in each class in the binarized NYSDOH-2017 dataset.

Label	≤1	2	3	4	5	6	7	8	9	>9
Original	357,944	546,125	419,322	243,112	159,892	115,697	90,902	66,316	49,743	39,424
≤1 vs. >1	(15.3%)	1,985,625 (84.7%)								
≤2 vs. >2	(38.6%)	904,069	1,439,500 (61.4%)							
≤3 vs. >3	(56.5%)	1,323,391	1,020,178 (43.5%)							
≤4 vs. >4	(66.8%)	1,566,503	777,066 (33.2%)							
≤5 vs. >5	(73.7%)	1,726,395	617,174 (26.3%)							

worst-case scenario, where the data is left untreated by the relabeling approach, and the classifier learns directly from positive-unlabeled data as if it was positive-negative. This result is expected under the negativity assumption detailed in Chapter 3 Section 3.5.2. We denote this case as *NEG* in the results.

5.4 Experiments

The overall machine learning pipeline is set up as follows. First, the dataset is randomly subsampled (without replacement) down to 100,000 instances while maintaining a specified class balance. This is done to reduce the required computation time without sacrificing the experiment integrity. Then the data is split into training and testing subsets via the 10-fold cross-validation procedure. At each fold, one-tenth of the data is used for testing and the remainder for training. The final performance metrics (e.g., accuracy and balanced accuracy) are averaged across all ten folds.

During the first step of the relabeling framework, we compare the relabeling strategy presented in Chapter 4, denoted as DEAR, with two PU learning algorithms from the literature. The one denoted EN is from the earlier seminal work by Elkan and Noto [26], and the other, denoted as DEDPUL, is the more recent approach by Ivanov that was shown to be effective across a wide range of scenarios [27]. It is worth noting that, due to the nature of the data (detailed in the following section), the Gaussian PDF estimator was not a suitable choice, and the results were obtained using the histogram probability mass estimators.

During the second step of the relabeling framework, we utilize one of four common classifiers: decision tree (DT), logistic regression (LR), K-Nearest Neighbor (KNN), and Naïve Bayes (NB) [87]. These are well-understood classifiers with different properties and characteristics. The main goal of the experiments is to investigate the detrimental effect of unlabeling on a classifier's performance and the ability of PU learning approaches to mitigate this effect in the context of predicting LOS during cold-start transition periods. Therefore, no hyperparameter tuning is performed, and each classifier is trained using the default recommended settings as implemented in the Python machine learning library scikit-learn version 0.24.1 [111].

5.5 Results

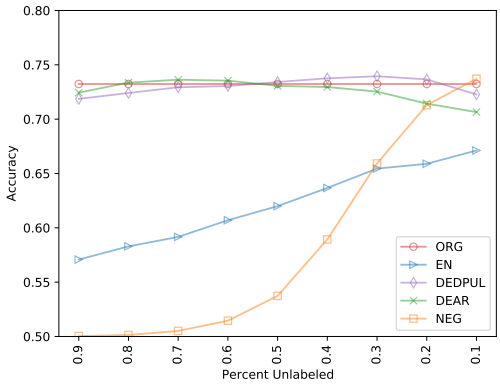
5.5.1 Janatahack Dataset Experiments

Figure 5.2 shows how the classification accuracy is affected for the five configurations (NEG, DEAR, EN, DEDPUL, and ORG) for different amounts of unlabeling and using four different classifiers (Logistic Regression, Decision Tree, Naïve Bayes, and K-nearest neighbor). Similar to the synthetic example in Chapter 3, it can be observed that the gap between NEG and ORG increases as the amount of unlabeling increases. This gap is what the two PU learning techniques try to reduce. For the ORG configuration, the best classifier is Logistic Regression, and in most cases, this is true for most other configurations and amounts of unlabeling. Overall, DEDPUL performs significantly better for PU learning than EN for all classifiers except for the Naïve Bayes case, in which they both perform similarly. We speculate that this is the case because the EN method is based on a probabilistic formulation which is similar to the underlying probabilistic principles behind Naïve Bayes. DEDPUL is able to practically recover the original ORG performance even at extreme amounts of unlabeling. The proposed DEAR approach outperforms EN in most cases, coming close to and sometimes beating DEDPUL performance. DEAR obtains the best performance on the class-balanced data and in cases with large proportions of unlabeled positives.

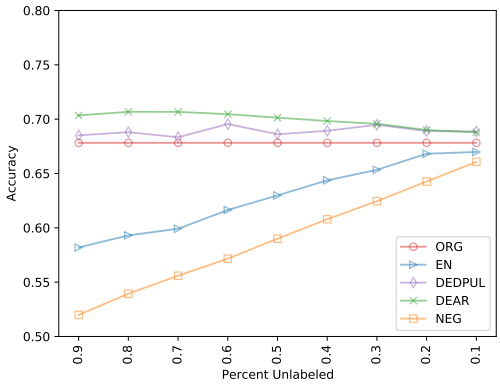
Table 5.3: Janatahack Accuracy with (60/40 class proportion)

% unlabeled	classifier	NEG	EN	DEDPUL	DEAR	ORG
0.5	LR	62.80	63.09	76.86	75.72	77.27
	DT	64.94	62.34	72.31	72.07	69.78
	NB	62.60	61.65	64.06	62.11	63.41
	KNN	63.65	58.11	70.76	70.54	69.09
0.8	LR	60.15	56.91	75.82	75.38	77.27
	DT	62.10	58.33	72.87	73.36	69.78
	NB	62.42	62.07	64.10	61.27	63.41
	KNN	60.35	52.25	70.76	70.53	69.09

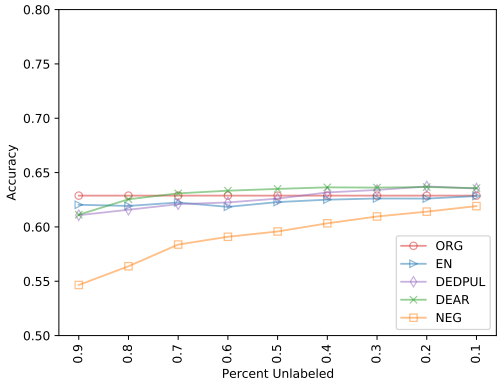
Table 5.3 focuses on two specific amounts of unlabeling (0.5 and 0.8) and shows how the different configurations affect the classification accuracy for the four different classifiers. As can be seen, for these amounts of unlabeling, the EN algorithm is not



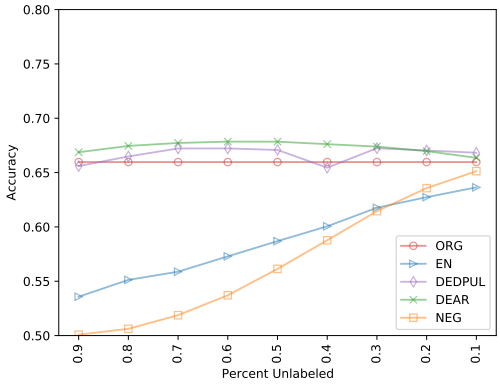
(a) Logistic Regression (LR)



(b) Decision Tree (DT)

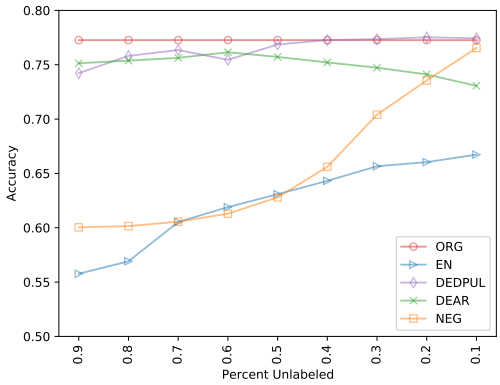


(c) Naïve Bayes (NB)

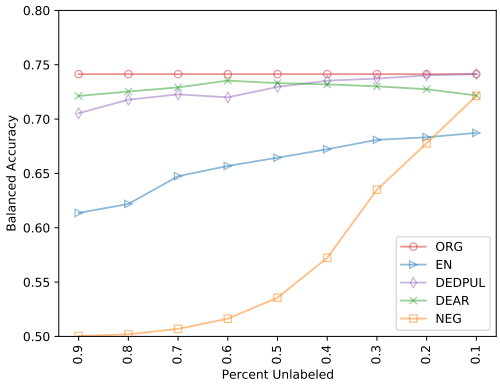


(d) K-nearest neighbor (KNN)

Figure 5.2: Accuracy with class-balanced Janatahack dataset.



(a) Traditional accuracy



(b) Balanced accuracy

Figure 5.3: Accuracy with Janatahack dataset, 60-40 class balance.

Table 5.4: Janatahack accuracy with balanced classes.

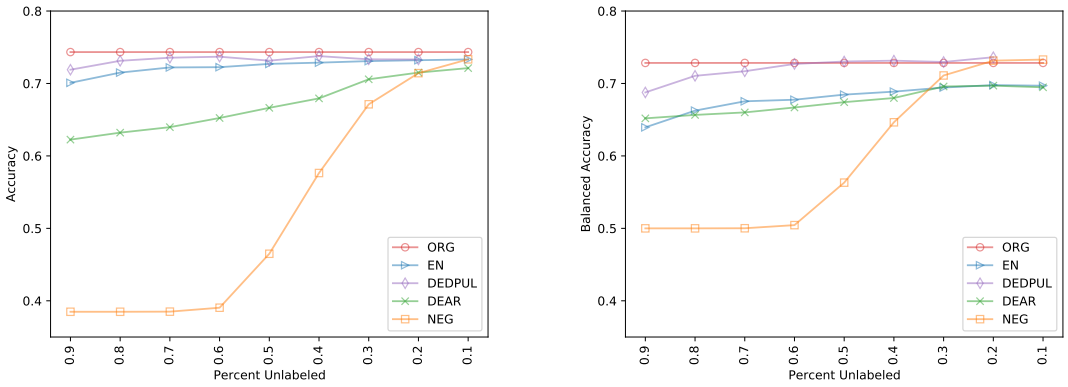
% unlabeled	classifier	NEG	EN	DEDPUL	DEAR	ORG
0.5	LR	53.74	61.99	73.42	73.06	73.24
	DT	59.00	62.98	68.61	70.13	67.82
	NB	59.58	62.28	62.60	63.50	62.88
	KNN	56.13	58.69	67.08	67.84	65.97
0.8	LR	50.14	58.29	72.41	73.36	73.24
	DT	53.94	59.30	68.80	70.68	67.82
	NB	56.38	61.93	61.58	62.54	62.88
	KNN	50.62	55.12	66.48	67.45	65.97

as effective as DEDPUL, which achieved classification accuracy similar to the ORG. The proposed DEAR approach also performed better than EN, achieving similar results to DEDPUL but slightly worse. The interpretation of classification accuracy could be affected by the prior class balance. Table 5.4 shows similar results for the case in which the two classes have an equal number of instances. Here, the proposed DEAR approach attained slightly higher accuracy than DEDPUL. Because the data is class-balanced, the classification accuracy score is the same as the balanced accuracy. Finally, Table 5.5 shows the detailed results for all different amounts of unlabeling and balanced class prior for the Logistic Regression, which is the best performing classifier for this particular configuration. It can be seen that the EN algorithm is able to improve the accuracy partially, but again, it is not as effective as DEDPUL or DEAR.

Table 5.5: Accuracy of logistic regression with class-balanced Janatahack dataset.

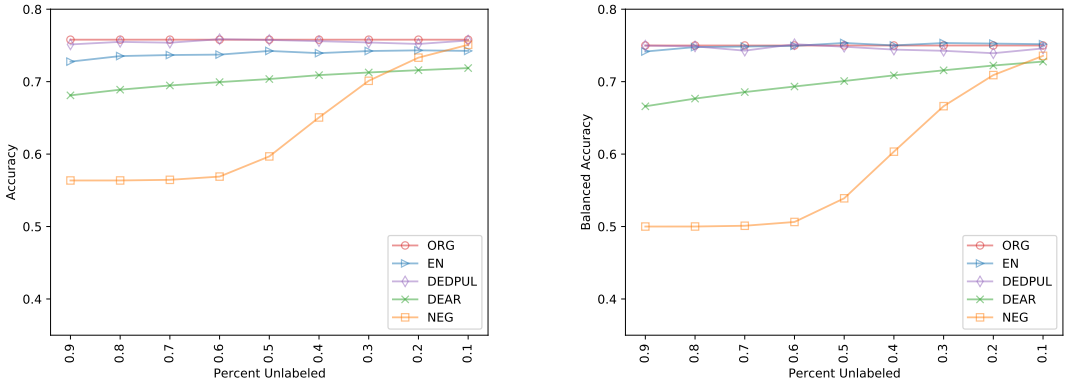
% unlabeled	NEG	EN	DEDPUL	DEAR	ORG
0.1	73.72	67.12	72.28	70.65	73.24
0.2	71.29	65.89	73.66	71.43	73.24
0.3	65.92	65.45	73.95	72.54	73.24
0.4	58.93	63.67	73.76	72.96	73.24
0.5	53.74	61.99	73.42	73.06	73.24
0.6	51.45	60.70	73.04	73.54	73.24
0.7	50.51	59.16	72.94	73.63	73.24
0.8	50.14	58.29	72.41	73.36	73.24
0.9	50.04	57.08	71.86	72.44	73.24

5.5.2 NYSDOH Dataset Experiments



(a) Traditional Accuracy (b) Balanced Accuracy

Figure 5.4: Logistic Regression with NYSDOH $\leq$2 vs. >2 dataset.



(a) Traditional Accuracy. (b) Balanced Accuracy

Figure 5.5: Logistic Regression with NYSDOH $\leq$3 vs. >3 dataset.

Figures 5.4 and 5.5 show how the classification performance of logistic regression is affected for different amounts of unlabeling for the four configurations (NEG, EN, DEDPUL, ORG) and two different LOS splits. Similar to the Janatahack dataset, the gap between NEG and ORG increases as the amount of unlabeling increases. As can be seen from the figures, using a PU learning approach recovers the classification performance to a large extent and brings it close to the ORG performance. Overall, the DEDPUL algorithm performs better than the EN algorithm, but it is more computationally expensive. Tables 5.6 and 5.7 show the confusion matrices for the same two splits and, similarly, we can see that the degradation in classification accuracy is recovered. Looking at the results at the confusion matrix level of detail

Table 5.6: DEDPUL with NYSDOH ≤2 vs. >2 where 50% positives are unlabeled.

actual / pred.	ORG		NEG		RELABELED	
	P	N	P	N	P	N
P	48,816	12,967	8,409	402	47,083	11,982
N	12,699	25,518	53,106	38,083	14,432	26,503
accuracy	0.743		0.465		0.736	
bal. acc.	0.728		0.563		0.727	

Table 5.7: DEDPUL with NYSDOH ≤3 vs. >3 where 20% positives are unlabeled.

actual / pred.	ORG		NEG		RELABELED	
	P	N	P	N	P	N
P	29,979	10,543	22,702	5,745	28,354	9,230
N	13,663	45,815	20,940	50,613	15,288	47,128
accuracy	0.758		0.733		0.755	
bal. acc.	0.750		0.709		0.742	

also reveals that the amount of unlabeled positives has a specific effect on classifiers trained in both situations: on the relabeled data (treated) and with the negativity assumption (untreated). In both cases, the classifier exhibits a bias towards the negative class, recovering more true negatives than in the ORG case, albeit at the cost of true positives.

5.6 Conclusions and Future Work

The experiments detailed in this chapter and their results demonstrate how the performance of a classification system trained to predict patients' LOS at a hospital can be negatively affected during cold-start transition periods. The drop in performance can be significant, especially when a large percentage of the samples is unlabeled. This degradation can be mitigated via PU learning methods bringing classification performance close to what can be obtained during stable state periods when fully labeled data is available.

In many cases, experiment results show that the original (ORG) classification performance can be fully recovered through the use of PU methods. The results indicate that in most cases, the DEDPUL algorithm performs the best. Between the other two, EN works better for the NYSDOH dataset and DEAR for the Janatakack

dataset. As for time requirements, the EN algorithm is significantly faster to train, but DEDPUL usually performs better. While keeping in mind that the speed of DEAR depends on the underlying density estimator, the experiment results presented in this chapter were obtained using the simple uni-variate histogram probability mass estimator (under the feature-independence assumption). As a result, DEAR was significantly faster than the other two approaches.

There are many future work directions. An obvious one is to investigate more PU learning algorithms and datasets that represent the LOS problem. A more dynamic simulation of incoming and discharging patients could be conducted to explore a combination of PU learning and online learning techniques. This chapter mainly focused on investigating the effects of different amounts of unlabeling during the cold-start transition period. However, there is an interesting interplay between prior class balance and unlabeling proportions. We have informally observed that the effectiveness of PU learning techniques reduces significantly when the original data exhibits significant prior class imbalance. Adapting existing PU learning approaches to this scenario would be another direction for future work.

Chapter 6

Treating Weakly Labeled Music Auto-Tagging Data

Music tagging, also known as semantic annotation, is a process of assigning labels or tags to music pieces that provide meaningful descriptions and help categorize and search music collections. The ever-increasing amount and growth rate of digital music content is supported by the developments in automated music recommendation and retrieval systems. Most such systems nowadays are data-driven, which means they rely on the quality of the underlying data, such as semantic tags.

Arguably, human listeners are the best and, in some cases, the only source of semantic tags, e.g., genre or mood. There are many ways of acquiring tags from human listeners. Turnbull et al. discuss their strengths and weaknesses and identify five distinct categories [21]. The different methods vary in their scalability and the quality of tags they produce. The most common include mining web pages and social networking sites. For instance, Last.fm, a social networking website, has a mechanism by which users can add free-text annotations to music pieces and, in turn, use these annotations for search and retrieval. However, these approaches are also the source of many issues, one of which is the focus of this chapter, specifically the problem of weakly labeled data. One of the more reliable methods of procuring good quality annotations is by surveying music experts using a set of well-defined questions, a controlled vocabulary of annotations, and some level of inter-annotator agreement. For example, Pandora Radio, a music streaming service, employs full-time music experts whose primary role is to listen to music and provide annotations used by Pandora's recommendation and playlist generation algorithms. However, due to the

high cost and poor scalability of such methods, automated approaches are in demand, and many auto-taggers have been proposed in the research literature.

The goal of music auto-tagging is to create a system that can accurately assign a tag label to each newly arrived music piece only if the tag is a valid semantic descriptor for it. When dealing with only one tag, it is equivalent to binary classification because either the tag describes the music piece or it does not; both cannot be simultaneously true. However, when working with multiple tags, more than one tag can be simultaneously valid for the given piece of music. For example, a happy and upbeat pop song could be tagged simultaneously as "happy," "upbeat," and "pop," but not "downbeat" or "sad." This situation corresponds directly to the multi-label classification problem [23] in machine learning.

The auto-tagging problem is typically cast as a multi-label classification problem. Earlier auto-tagging systems assume no correlations among tags and often decompose the multi-label problem with n tags into n binary classification problems via the binary relevance approach. This chapter follows the same schema, and a separate classifier is trained for each tag. As is evident from auto-tagging, and more generally, from multi-label classification literature, capturing label correlations and incorporating them into the system can yield better results. However, the central topic of this chapter is the problem of missing labels, which could compound the difficulty of adding label correlations into the system. Thus, it is left to future work, and the focus here is on improving the individual binary classifiers as the first step.

Auto-tagging systems are usually trained using tag labels provided by human listeners. In many cases, this labeling is weak, which means that the provided tags are valid for the associated tracks, but there can be tracks for which a tag would be valid but not present. When considering weakly labeled data from the perspective of a binary classifier within the binary relevance decomposition, it can be thought of as a particular case of semi-supervised learning where only some of the positive instances are labeled while the unlabeled data consists of both positive and negative instances. Then, the task is to learn a classification model that can distinguish between newly arrived positive and negative instances even though no labeled negative examples are available during training, which is known as the PU learning problem.

There is a considerable degradation in system performance if trained on data with missing labels in all three cases: music auto-tagging [51, 72], multi-label classification [23], and PU learning [3]. Because of how analogous these situations are, it seems reasonable to try and improve the overall system's performance by treating

the weakly labeled music tags with PU learning approaches before using them for training an auto-tagger.

6.1 Proposed Approach

When training an auto-tagger on weakly labeled data, we can preprocess the data using PU methods to obtain a better performing system than we would otherwise. Following the schema of early music auto-taggers, we can decompose a multi-label problem with T labels into T binary classification problems, one per label, by making the binary relevance assumption that there is no correlation between labels. In the case of weakly labeled multi-label data, when $y_t = 0$ for a particular $t \in T$, it could mean that the corresponding tag is not valid for the given data instance, or it could mean that the tag is valid but was not labeled as such. Thus, applying binary relevance yields T independent positive-unlabeled problems that can be solved using approaches from the PU learning literature. There is an implicit assumption that the number of tags is fixed at T; therefore, the annotation vocabulary must be finite, which is not always the case, but it is usual in music annotation [21].

The proposed approach follows the two-step relabeling framework described in Chapter 3 Section 3.3.1 independently for each binary classifier within the binary relevance framework. In the first step, we categorize the unlabeled instances into likely positive and likely negative ones via a PU learning algorithm. In the second step, we train a traditional binary classifier on the newly labeled data. We omit the optional third step but provide experimental results for different binary classification algorithms. After applying this treatment for all tags, we obtain our auto-tagger consisting of T independent binary classifiers, one per tag.

6.2 Data

This chapter includes the results from three publicly available music datasets summarized in Table 6.1 and detailed below: GTZAN [126], MTT [127, 128], and C10K [129]. Two of the datasets are distributed with 30-second audio clips from which MFCC features are computed via the librosa 0.8.0 library (default settings) [130]. For C10K dataset, we use the pre-computed MFCC features that are distributed with it. Music tags used for the experiments come from three different sources: survey, game, and auto-tagging. The different sources of music tags vary in their scalability and the

quality of tags they produce; for a detailed comparison and discussion, see Turnbull et al. [21]. Their "comparison includes a discussion of both scalability (financial cost, human involvement, and computational resources) and quality (the cold start problem & popularity bias, strong vs. weak labeling, vocabulary structure & size, and annotation accuracy)." [21]

The GTZAN dataset [126] is one of the usual baselines for a music retrieval system being relatively small in size and having a balanced sample from ten different western music genres. Since there are no tags provided with the dataset, they are computed from the audio features using a pre-trained auto-tagger called musicnn [131, 132]. There are several musicnn configurations provided, the experiments presented here use the system trained on the MTT dataset. This auto-tagger is capable of outputting top-n tags with n ranging from 1 to 50. The experiments were carried out with different values for n and the results presented here are from the top-10 configuration, denoted as GTZAN-10.

The MTT dataset [127] was created as part of research on the evaluation of audio tagging algorithms. Law et al. collected its annotations using an online game called "TagATune." The dataset contains only those tags that are associated with a clip only if it was generated independently by more than two players, and each tag is associated with at least 50 clips. Because this dataset is distributed with audio, we can obtain an additional set of tags from the musicnn auto-tagger. As with the abovementioned GTZAN dataset, the results include the top-10 configuration allowing us to compare the two datasets on equal footing. Additionally, we can compare the effects of weak labeling between different tags sources for the same audio.

The C10K dataset [129] contains three weakly labeled sets of tags that were harvested from Pandora's website. The experiments presented here use the acoustic subset consisting of 354 tags with the expectation that they are better represented in the audio features as opposed to genres, which are culturally derived.

The choice of features can affect the classifier performance, and thus for the datasets with audio, the features are extracted in an identical way. Tags can also add experimental complications; for example, some tags are easier to predict than others. One of the solutions proposed in the literature is to remove the tags from the training data that cannot be predicted beyond some threshold [133, 134]. Thus, two more datasets are created: MTT-O-50 and C10K-O-50, by selecting only the top 50 most popular tags in terms of support (i.e., number of tracks for which the tag is valid).

Table 6.1: Music datasets and their statistics. Grey background indicates that training tags were generated by an auto-tagger, white - by humans.

Dataset Notation	Type of Tags	Num. of Tracks	Num. of Features	Num. of Tags	Tags Per Track	Tracks Per Tag	Audio Lengths
GTZAN-10	musicnn	1,000	40	49	10	0.200	30s
MTT-10	musicnn	25,851	40	50	10	0.200	30s
MTT-O	original			188	3.46	0.018	
C10K-O	original	10,061	26	354	8.74	0.025	none

6.2.1 Simulating Missing Labels in Data

To simulate the weakly labeled scenario, we start with fully labeled training data. Then, for each tag, we randomly choose a pre-determined proportion of samples where $y_t = 1$ and set their y_t to 0, thereby unlabeling them. This selection method satisfies the SCAR assumption discussed in Section 3.5.1 and synthetically creates a weakly labeled version of a multi-label dataset where each individual tag corresponds to a PU learning problem suitable for PU learning algorithms. When the percentage of unlabeled positives is zero and the dataset is left in its original state, we denote the auto-tagger performance as original ORG. When the unlabeling procedure modifies the dataset, and the classifier learns under the negativity assumption discussed in Section 3.5.2, its results are denoted as NEG, and we would expect the auto-tagger to perform worse than in the ORG case. To evaluate the effect of missing labels in the data and their treatment with PU learning, we can try different amounts of unlabeling and compare the auto-tagger performance with the data treated by our proposed approach versus the NEG auto-tagger and expect an improvement on the latter.

6.3 Experiments

The effectiveness of the proposed approach is evaluated using logistic regression (LR) [87]. The main goal of the experiments is to understand the detrimental effect of missing labels on auto-tagging performance and the ability of PU learning approaches to mitigate this effect. Therefore, no hyperparameter tuning is performed, and each classifier is trained using the default recommended settings. The scikit-learn-0.24.1 Python machine learning library is used [111]. In the case of the DEAR approach,

the Gaussian distribution estimator is a suitable choice for the statistical summaries of the MFCCs.

Each dataset is split into training and testing subsets via the 5-fold cross-validation procedure. At each fold, 20% of the dataset instances are selected randomly (i.i.d) without replacement for testing the classifiers, and the remaining 80% of the data is used for training. The final figure of merit (e.g., AUCROC) is summarized across all folds using the arithmetic mean.

In music auto-tagging, a common approach for evaluating the model's performance is to report for each experiment configuration the mean area under the receiver operating characteristic curve (AUCROC). First, the AUC is computed for each tag category, in our case for each classifier in the binary relevance decomposition, then their arithmetic mean is computed across all tags respectively. The AUCROC can be computed for each tag using either the discrete (1 or 0) classifier outputs or the continuous probabilistic ones (if supported by the classifier).

F1-score and recall add further detail to the analysis in this chapter. They are computed for both the auto-tagger output during the testing stage and the relabeled training data.

6.4 Results

First, let's examine the effects of weakly labeled training data in music auto-tagging using the methodology described in the previous sections. The first question we wanted to investigate was how weak labeling affects the effectiveness of auto-taggers. The weak labeling process can be artificially simulated as described in Section 6.2.1. Ideally, this simulation should be applied to strongly labeled data in which the presence or absence of a tag is verified. However, existing datasets with tags are known to have weak labels. For example, Law et al. [128] run a post-hoc experiment where the "results show that when humans evaluate the retrieved list, the mean average precision of all methods are significantly higher than if we use the ground truth tags as the judge." using the MTT dataset.

The tags that are automatically predicted using a state-of-the-art auto-tagger based on deep neural networks (musicnn) can serve as a good proxy for strong labels [131, 132]. Because these tags are predicted consistently across all tracks, we know that there is no inherent weak labeling. In addition, because the selected tags are predicted with higher confidence, we also know that these tags can be predicted

Table 6.2: AUCROC of LR (above) and DT (below)

Logistic Regression						
Dataset	% unlabeled	MOD	EN	DEDPUL	DEAR	ORG
GTZAN-10	0.3	0.651	0.659	0.654	0.749	0.716
	0.5	0.594	0.632	0.627	0.741	0.716
	0.7	0.541	0.588	0.588	0.715	0.716
MTT-10	0.3	0.635	0.812	0.757	0.761	0.727
	0.5	0.560	0.811	0.764	0.757	0.727
	0.7	0.510	0.809	0.768	0.751	0.727
Decision Tree						
Dataset	% unlabeled	MOD	EN	DEDPUL	DEAR	ORG
GTZAN-10	0.3	0.635	0.632	0.629	0.719	0.684
	0.5	0.595	0.597	0.591	0.703	0.684
	0.7	0.551	0.557	0.566	0.685	0.684
MTT-10	0.3	0.688	0.810	0.752	0.758	0.771
	0.5	0.635	0.804	0.758	0.752	0.771
	0.7	0.581	0.796	0.757	0.746	0.771

reliably from audio features. These reasons give us more confidence that what we observe when simulating weak labeling and how PU learning can address the problem is similar to what would happen if we had "true" strong labels. The simulated weak label experiments also serve as a way to establish terms and vocabulary around this problem. The next subsection investigates how PU learning can address auto-tagging of original human-provided tags, where there's probably some proportion of missing labels to begin with.

Table 6.2 shows how the AUCROC of a logistic regression and decision tree classifiers is affected for different configurations and amounts of unlabeling. The AUCROC in this table is computed using the discrete (0 or 1) classifier outputs. The ORG score is obtained using the original positive and negative labels for training without any unlabeling. The NEG score is obtained by considering all the unlabeled samples as negative (the negativity assumption). It can be observed that the difference/gap between NEG and ORG increases with the amount of unlabeling. This gap is caused by the simulated weak labeling, and it is what the two PU learning techniques try to reduce. For the ORG configuration, the best auto-tagger uses Logistic Regression (LR), and in most cases, this is true for other configurations and amounts of unla-

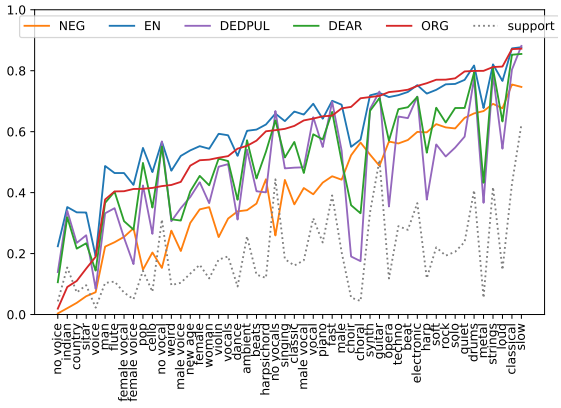
beling. The three PU learning approaches (EN, DEDPUL, DEAR) are successful at mitigating the effect of weak labeling and improving auto-tagger performance, bringing it closer to the ORG configuration. The DEAR approach obtains best AUCROC scores with the GTZAN-10 dataset and EN with MTT-10.

Figure 6.1 shows a more detailed view of these experiments for MTT, showing the changes in the F1-score for each tag. The decrease caused by the simulated weak labeling can be observed as the gap between the NEG (orange) and ORG (red) curves. All PU learning algorithms are able to improve the F1 reducing the gap. Two different views of identical results are provided: one sorted by tag support and one sorted by the ORG F1-score. One can observe that there is a correlation between higher support and better predictability, but it is not strict.

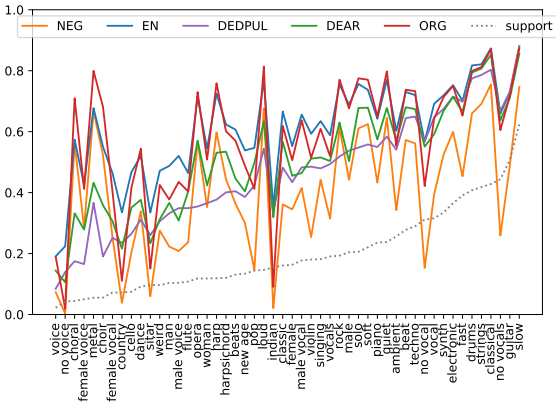
Figure 6.2 provides a more detailed view of the unlabeling followed by the proposed PU learning treatment and then auto-tagging. The two top plots show how relabeling the samples of the training set to positive and negative affects recall and F1-score for individual tags (relabeled training data). After this relabeling process, a new auto-tagger is trained and used to predict the test set. Figure 6.2 shows the corresponding recall and F1-score in the bottom row.

Table 6.3: AUCROC of LR and DT auto-taggers with different datasets at 70% of positives unlabeled.

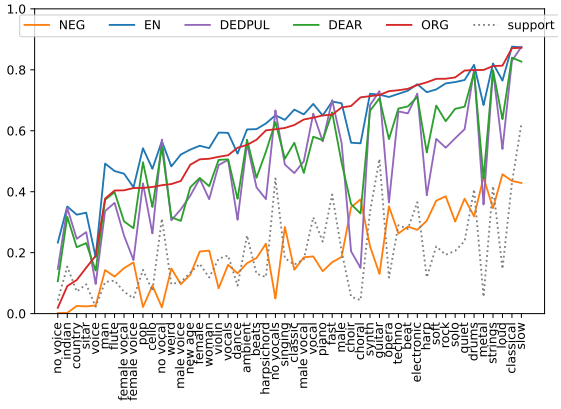
Dataset	Clfassifier	NEG	EN	DEDPUL	DEAR	ORG
GTZAN-10	LR	0.541	0.588	0.588	0.715	0.716
	DT	0.551	0.557	0.566	0.685	0.684
MTT-10	LR	0.510	0.809	0.768	0.751	0.727
	DT	0.581	0.796	0.757	0.746	0.771
MTT-O	LR	0.501	0.576	0.575	0.692	0.514
	DT	0.511	0.570	0.566	0.691	0.542
C10K-O	LR	0.501	0.528	0.533	0.666	0.510
	DT	0.509	0.525	0.523	0.665	0.531
MTT-O-50	LR	0.501	0.750	0.720	0.767	0.546
	DT	0.527	0.722	0.697	0.701	0.595
C10K-O-50	LR	0.501	0.650	0.646	0.711	0.534
	DT	0.519	0.612	0.610	0.643	0.563



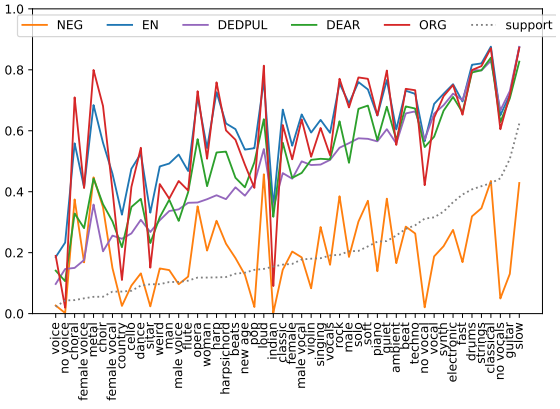
(a) PU=0.3



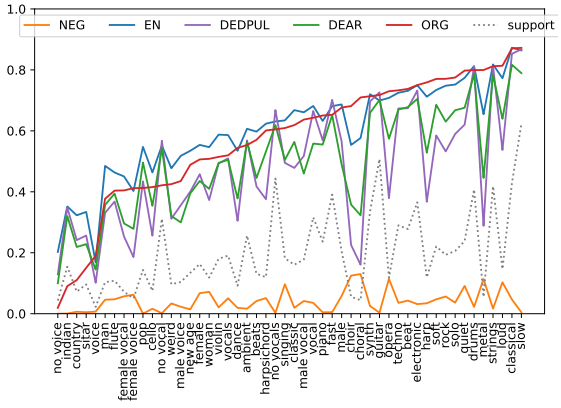
(b) PU=0.3



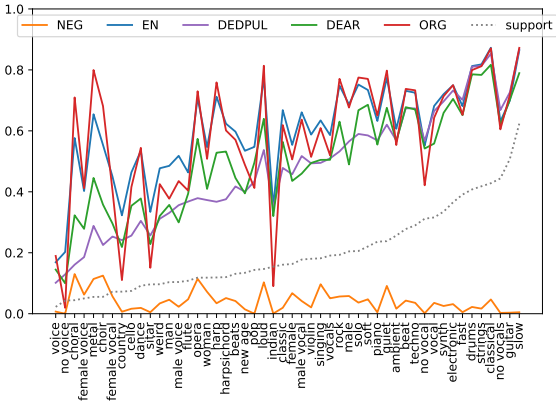
(c) PU=0.5



(d) PU=0.5

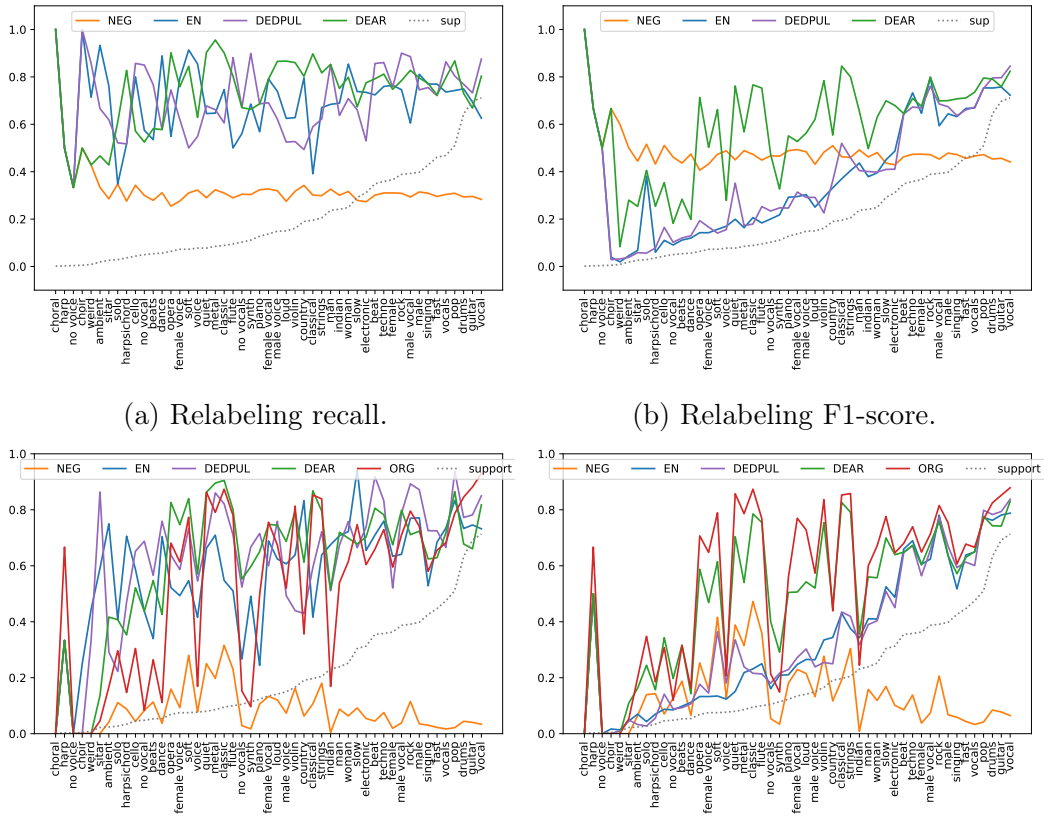


(e) PU=0.7



(f) PU=0.7

Figure 6.1: LR with MTT-10 dataset at different levels of unlabeling sorted by f1-score (left) and support (right).



(a) Relabeling recall.

(b) Relabeling F1-score.

(c) Auto-tagger recall.

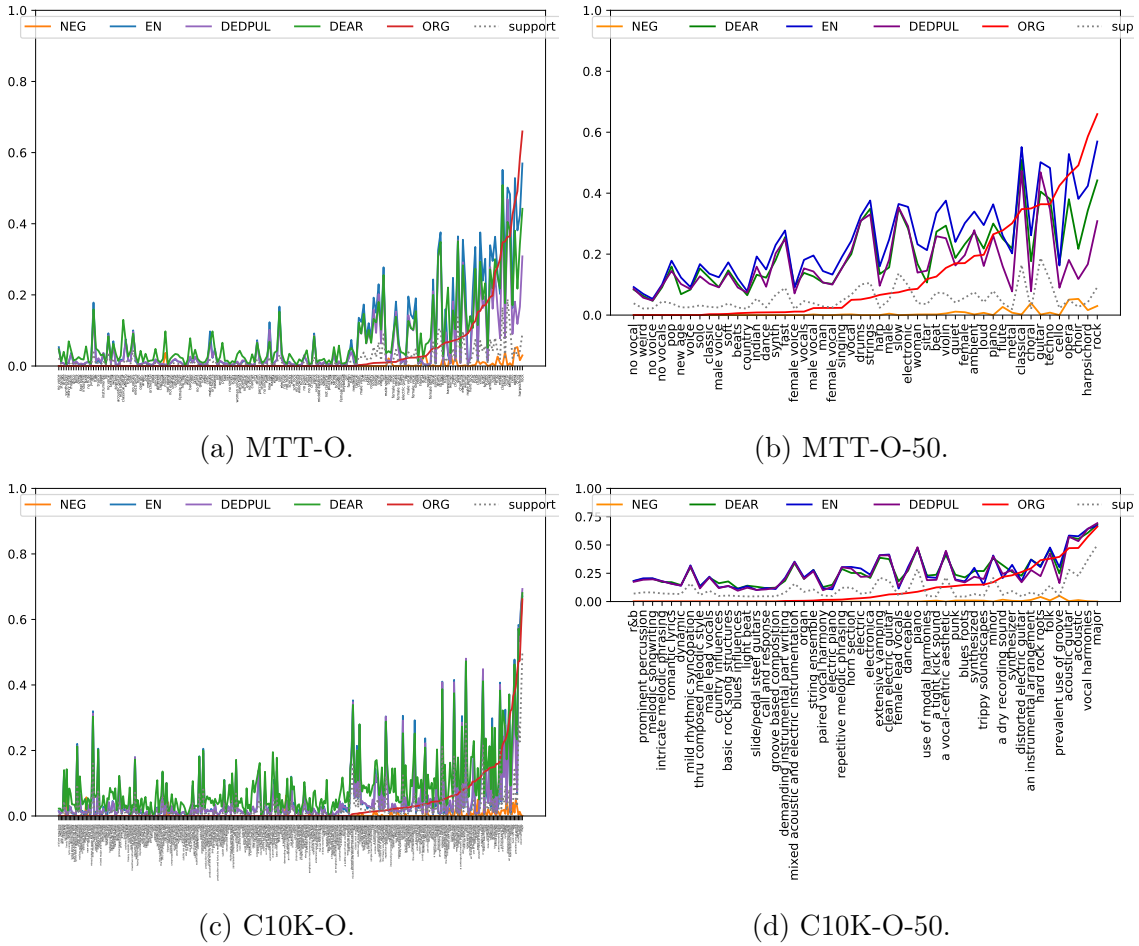
(d) Auto-tagger F1-score.

Figure 6.2: Recall and f1-score with GTZAN-10 from the relabeled training data (above, sorted by support) and the final LR auto-tagger (below, sorted by ORG F1)

6.4.1 Original Tags

Figure 6.3 shows the experiment results with the original tags for the MTT and C10K datasets (as opposed to musicnn tags). The one on the left is a complete set of labels, and on the right consists of the top-50 most populous ones. As can be seen in Table 6.3 the auto-tagger learned from the original labels after they are treated by PU learning methods performs much better than the one trained on the original data, one of which (C10K) we know to be weakly labeled [129]. This result can also be seen in Figures 6.3b and 6.3d, especially at the left side of the plot. Note that in Table 6.3 the AUCROC is computed on the binary classification outputs. This is more indicative of how well the PU-learning approach is able to recover the predicted labels without taking into account the probabilistic output.

Based on the observations with simulated weak labeling, we can expect that applying PU learning to a tagging problem with weak labels can lead to improved results.



(a) MTT-O.

(b) MTT-O-50.

(c) C10K-O.

(d) C10K-O-50.

Figure 6.3: LR auto-tagger F1 with MTT-O and C10K-O and 70% unlabeling, sorted by the ORG F1, complete (left), top-50 tags (right)

Table 6.4 shows the results of applying the two PU learning approaches to datasets directly (w/o unlabeling). As can be seen, for the GTZAN-10 dataset, which contains "proxy" strong labels, the PU learning actually reduces AUCROC as there is no weak labeling. In contrast, for the other four datasets in which original labels are used, the use of PU learning results in improved AUCROC. Note that the AUCROC in these experiments is computed on the binary classifier outputs. These experiments are performed using the LR base classifier. As expected, the scores and associated improvement are higher when the top 50 tags are used. These experiments suggest that there is inherent weak labeling in the original tags for these datasets and that the proposed PU learning approach is effective at improving auto-tagger performance.

Table 6.4: AUCROC of LR without any unlabeling

Dataset	ORG	EN	DEDPUL
GTZAN-10	0.715	0.686	0.673
MTT-O	0.514	0.655	0.639
C10K-O	0.510	0.589	0.590
MTT-O-50	0.546	0.762	0.724
C10K-O-50	0.534	0.690	0.672

6.5 Discussion and Future Work

This chapter investigated how using weakly labeled data leads to reduced auto-tagging performance on three datasets with different characteristics. Using a simulation of unlabeling and experimentally investigating different PU learning approaches, we can observe that the deterioration of auto-tagging performance can be significant, especially when there are large amounts of unlabeling. This deterioration can be mitigated using PU learning methods. In many cases, experiment results show that the original (ORG) auto-tagger performance can be fully recovered. The results indicate that, in most cases, the EN algorithm is significantly faster to train and often performs better than the DEDPUL algorithm. The DEAR performance is somewhere between the other two, sometimes even better than both (e.g., GTZAN). This conclusion is, to a large extent, independent of classifier and amount of unlabeling.

It is important to note that the SCAR assumption is probably not the best for music tags. For example, a listener might assume that a track that already has been tagged as heavy metal does not need to be tagged as electric guitar even though such a tag would be valid. There are also situations in which weak labeling might arise indirectly, such as synonyms like female voice and female vocals, or antonyms like fast and slow. In this case, the unlabeled samples for each synonym will frequently be labeled with the other synonym, and the antonym's negated version could be missing. Therefore, it is worth investigating the different situations whereby a tag would be missing from a track and incorporate the appropriate strategies into approaches that aim to work with weakly labeled data.

We used a basic binary relevance auto-tagger [135] while focusing on the effects and treatment of weakly labeled training data. We expect that more sophisticated systems such as the ones that use self-attention convolutional neural networks and scattering transform [136] would benefit as well. It would be interesting to see the

proposed approach in the context of methods that prioritize few relevant tags rather than many irrelevant ones, promote hard to predict infrequent but rewarding tags, work with long tails, or deal with synonyms [51, 137].

In the broader context of multi-label learning, there are multiple ways of incorporating a binary classifier into a multi-label system, including those that take advantage of label correlations, such as label ranking, classifier chains, and label power set [18]. However, their performance is negatively affected by missing labels, and there is ongoing research on the topic [23]. For instance, Bi and Kwok propose incorporating label correlations in the presence of weak labels and show that binary relevance outperforms classifier chains [138]. More recently, Teisseyre investigated how the classifier chains algorithm, designed specifically for multi-label learning, is negatively affected when there are missing labels in the training data [139].

The use of traditional figures of merit such as AUCROC and F1-score has been shown to not always be reflective of how the system would perform in the real world [140]. A more reliable way to evaluate the effect of PU learning when training using weak labels in auto-tagging could be a post-hoc analysis of the results by human listeners [128]. However, this could require significant human labor, especially when multiple datasets, algorithms, and configurations need to be considered.

In addition to the above, there are several directions this work can take. For instance, it would be beneficial to investigate more sophisticated simulations of weak labeling, such as unlabeling all tags of a particular track or taking into account tag correlations. The use of PU learning for training from weak labels can also be integrated with more complex auto-taggers that are based on deep learning. Evaluating this approach with additional datasets as well as other modalities such as video and images would also be interesting.

Chapter 7

Conclusion and Future Directions

The focus of this dissertation has been the problem of Positive-Unlabeled (PU) learning. A modular and flexible approach for treating the non-traditional positive-unlabeled data prior to learning a traditional binary (positive-negative) classifier is proposed. The approach is based on a two-step relabeling technique common in PU learning literature. Different method of density estimation can be used in the first step followed by different classifiers for the second step. This approach is compared throughout the thesis with two existing algorithms from the literature, one of which (DEDPUL) was recently shown to attain the current state-of-the-art performance [27].

Additionally, two application scenarios in which ideas from PU-learning can be leveraged to solve important problems are investigated. In the healthcare setting, when predicting the length of stay (LOS) in a hospital using machine learning approaches, PU-learning can be used during cold-start and transition periods where the probability distributions and densities of the problem change over time. During these periods, some of the data is reliably labeled but some needs to be considered as unlabeled. Through a variety of experiments with different datasets, classifiers, and PU-learning strategies we show that taking advantage of PU-learning approaches, including the one proposed in this thesis, can lead to significant improvements in the effectiveness of predicting length of stay as compared to a more traditional approach of treating the unlabeled samples as negative samples during transition periods.

The problem of missing labels (weakly labeled data) arises naturally in auto-tagging scenarios where multimedia items such as music tracks are annotated by humans with multiple labels/tags. For various reasons there are many cases in which a tag that would be considered relevant for a particular track is not identified as such. If each tag is considered individually this weak labeling can be modeled as a

PU-learning process. A common machine learning method for training auto-tagging systems is binary relevance decomposition where each tag is considered separately as a binary classification problem. Using experiments with datasets for music auto-tagging we show that unlabeling can deteriorate the effectiveness of ML-based auto-tagging systems and that taking into account the potential unlabeling and using PU-learning approach to reduce its effect can lead to significantly better results. Our experiments also show that even when no synthetic unlabeling is performed, PU-learning can lead to improved effectiveness suggesting that existing music auto-tagging datasets are weakly labeled.

7.1 Future Directions

In the previous chapters, the contributions of this dissertation and directions for future work for each one of them have been described. We summarize these directions for future work in this section.

7.1.1 Density Estimation Asymmetric Relabeling

The proposed approach is modular and allows different density estimation and classification algorithms to be used. More extensive experiments with different approaches to density estimation, PU learning methods, and classification algorithms can be performed to provide guidelines for what works best. It is likely that this decision depends on the specifics of the problem. Another important direction for future work would be to incorporate additional information, such as class priors, feature correlations, and label correlations, so that they can be taken into account during the relabeling process of the algorithm.

7.1.2 Predicting length of stay

Experimental results presented in Chapter 5 confirm that PU approaches can be useful in mitigating, to some degree, the negative effects of sudden statistically significant changes in data used for predicting patients' length of stay (LOS) at a healthcare unit. However, the simulation can be improved in many ways to resemble more closely real life scenarios in the following ways. The arrival and discharge of patients in the hospital can be modeled more accurately using stochastic modeling informed by actual

hospital records. This would correspond to a more dynamic online learning situation in which the amount of unlabeling (and associated PU-learning) would fluctuate over time. It would be an interesting challenge to investigate how the proposed DEAR approach (as well as other PU-learning approaches) could be adapted to this online scenario.

7.1.3 Music auto-tagging

In music auto-tagging, it is possible that the weak labeling process is not performed independently for each tag. For example, correlated tags might affect whether unlabeling takes place or not. As an extreme example in the case of synonyms (female voice, female singer) if they are treated as separate tags the unlabeled samples for one of the synonyms will correspond to the labeled samples of the other. It is also possible that the unlabeleing is correlated at the track level rather than the tag level. For example a new track might not have enough human tag annotations. It would be interesting to investigate these more complicated unlabeling scenarios through simulation and devise approaches better suited to address them.

Most modern music auto-tagging systems are based on deep-learning apporaches and provide predictions for all tags simultaneously. These systems have been shown to outperform more traditional ML approaches such as the ones investigated in this paper. Incorporating PU-learning approaches to such architectures is not trivial as the feature extraction and classification stages are to some extent combined into the network. It might be possible to use truncated layers and transfer learning to address this issue.

Bibliography

[1] B. Zhang and W. Zuo, "Learning from positive and unlabeled examples: A survey," in *International Symposiums on Information Processing*, pp. 650–654, IEEE, 2008.

[2] K. Jaskie and A. Spanias, "Positive and unlabeled learning algorithms and applications: A survey," in *International Conference on Information, Intelligence, Systems and Applications*, pp. 1–8, IEEE, 2019.

[3] J. Bekker and J. Davis, "Learning from positive and unlabeled data: A survey," *Machine Learning*, vol. 109, no. 4, pp. 719–760, 2020.

[4] X. Chen, W. Chen, T. Chen, Y. Yuan, C. Gong, K. Chen, and Z. Wang, "Self-PU: Self boosted and calibrated positive-unlabeled training," in *International Conference on Machine Learning*, pp. 1510–1519, PMLR, 2020.

[5] B. Wu, Z. Liu, S. Wang, B.-G. Hu, and Q. Ji, "Multi-label learning with missing labels," in *International Conference on Pattern Recognition*, pp. 1964–1968, IEEE, 2014.

[6] G. Ward, T. Hastie, S. Barry, J. Elith, and J. R. Leathwick, "Presence-only data and the EM algorithm," *Biometrics*, vol. 65, no. 2, pp. 554–563, 2009.

[7] K. A. Keating and S. Cherry, "Use and interpretation of logistic regression in habitat-selection studies," *The Journal of Wildlife Management*, vol. 68, no. 4, pp. 774–789, 2004.

[8] P. Yang, X.-L. Li, J.-P. Mei, C.-K. Kwoh, and S.-K. Ng, "Positive-unlabeled learning for disease gene identification," *Bioinformatics*, vol. 28, no. 20, pp. 2640–2647, 2012.

[9] P. Yang, X. Li, H.-N. Chua, C.-K. Kwoh, and S.-K. Ng, "Ensemble positive unlabeled learning for disease gene identification," *PloS one*, vol. 9, no. 5, p. e97079, 2014.

[10] X. Zeng, Y. Zhong, W. Lin, and Q. Zou, "Predicting disease-associated circular RNAs using deep forests combined with positive-unlabeled learning methods," *Briefings in bioinformatics*, vol. 21, no. 4, pp. 1425–1436, 2020.

[11] W. Li, Q. Guo, and C. Elkan, "A positive and unlabeled learning algorithm for one-class classification of remote-sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 2, pp. 717–725, 2011.

[12] R. Lyon, B. Stappers, S. Cooper, J. Brooke, and J. Knowles, "Fifty years of pulsar candidate selection: from simple filters to a new principled real-time classification approach," *Monthly Notices of the Royal Astronomical Society*, vol. 459, no. 1, pp. 1104–1123, 2016.

[13] H. Yu, J. Han, and K.-C. Chang, "PEBL: Web page classification without negative examples," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 1, pp. 70–81, 2004.

[14] Y. Ren, D. Ji, and H. Zhang, "Positive unlabeled learning for deceptive reviews detection," in *Conference on empirical methods in natural language processing*, pp. 488–498, 2014.

[15] Y.-Y. Sun, Y. Zhang, and Z.-H. Zhou, "Multi-label learning with weak label," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 24, pp. 593–598, July 2010.

[16] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, pp. 44–53, 08 2017.

[17] H.-F. Yu, P. Jain, P. Kar, and I. Dhillon, "Large-scale multi-label learning with missing labels," in *International conference on machine learning*, pp. 593–601, PMLR, 2014.

[18] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE transactions on knowledge and data engineering*, vol. 26, no. 8, pp. 1819–1837, 2014.

[19] A. Bergamo and L. Torresani, "Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach," *Advances in neural information processing systems*, vol. 23, pp. 181–189, 2010.

[20] Y. Meng, J. Shen, C. Zhang, and J. Han, "Weakly-supervised neural text classification," in *ACM International Conference on Information and Knowledge Management*, pp. 983–992, 2018.

[21] D. Turnbull, L. Barrington, and G. R. Lanckriet, "Five approaches to collecting tags for music," in *International Society for Music Information Retrieval Conference*, pp. 225–230, ISMIR, 2008.

[22] T. Wei, L.-Z. Guo, Y.-F. Li, and W. Gao, "Learning safe multi-label prediction for weakly labeled data," *Machine Learning*, vol. 107, no. 4, pp. 703–725, 2018.

[23] W. Liu, X. Shen, H. Wang, and I. W. Tsang, "The emerging trends of multi-label learning," *arXiv preprint arXiv:2011.11197*, 2020.

[24] Y. Xu, C. Xu, C. Xu, and D. Tao, "Multi-positive and unlabeled learning," in *International Joint Conference on Artificial Intelligence*, pp. 3182–3188, 2017.

[25] S. Shu, Z. Lin, Y. Yan, and L. Li, "Learning from multi-class positive and unlabeled data," in *IEEE International Conference on Data Mining*, pp. 1256–1261, IEEE, 2020.

[26] C. Elkan and K. Noto, "Learning classifiers from only positive and unlabeled data," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 213–220, ACM, 2008.

[27] D. Ivanov, "DEDPUL: Difference-of-estimated-densities-based positive-unlabeled learning," in *IEEE International Conference on Machine Learning and Applications*, pp. 782–790, IEEE, 2020.

[28] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu, "Building text classifiers using positive and unlabeled examples," in *IEEE International Conference on Data Mining*, pp. 179–186, IEEE, 2003.

[29] T. Arjannikov and G. Tzanetakis, "Histogram-based asymmetric relabeling for learning from only positive and unlabeled data," in *IEEE International Conference on Machine Learning and Applications*, pp. 1065–1070, 2017.

[30] T. Arjannikov and G. Tzanetakis, "An empirical investigation of PU learning for predicting length of stay," in *IEEE International Conference on Healthcare Informatics*, IEEE, 2021.

[31] T. Arjannikov and G. Tzanetakis, "Cold-start hospital length of stay prediction using positive-unlabeled learning," in *IEEE EMBS International Conference on Biomedical & Health Informatics*, IEEE, 2021.

[32] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised Learning*. Adaptive computation and machine learning, MIT Press, 2010.

[33] X. Zhu, "Semi-supervised learning literature survey," *Computer Science, University of Wisconsin-Madison*, vol. 2, no. 3, p. 4, 2006.

[34] S. R. Jammalamadaka and A. Sengupta, *Topics in circular statistics*, vol. 5. world scientific, 2001.

[35] S. Boucheron, O. Bousquet, and G. Lugosi, "Theory of classification: A survey of some recent advances," *ESAIM: probability and statistics*, vol. 9, pp. 323–375, 2005.

[36] M. M. Moya and D. R. Hush, "Network constraints and multi-objective optimization for one-class classification," *Neural Networks*, vol. 9, no. 3, pp. 463–474, 1996.

[37] S. S. Khan and M. G. Madden, "A survey of recent trends in one class classification," in *Irish conference on artificial intelligence and cognitive science*, pp. 188–197, Springer, 2009.

[38] D. M. Tax and R. P. Duin, "Support vector data description," *Machine learning*, vol. 54, no. 1, pp. 45–66, 2004.

[39] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Advances in neural information processing systems*, pp. 582–588, 2000.

[40] Y.-F. Li and D.-M. Liang, "Safe semi-supervised learning: a brief introduction," *Frontiers of Computer Science*, vol. 13, no. 4, pp. 669–676, 2019.

[41] V. Barnabé-Lortie, C. Bellinger, and N. Japkowicz, "Active learning for one-class classification," in *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*, pp. 390–395, IEEE, 2015.

[42] A. Ghasemi, H. R. Rabiee, M. Fadaee, M. T. Manzuri, and M. H. Rohban, "Active learning from positive and unlabeled data," in *2011 IEEE 11th International Conference on Data Mining Workshops*, pp. 244–250, IEEE, 2011.

[43] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[44] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[45] X. Tang, B. Du, J. Huang, Z. Wang, and L. Zhang, "On combining active and transfer learning for medical data classification," *IET Computer Vision*, vol. 13, no. 2, pp. 194–205, 2019.

[46] R. J. Little and D. B. Rubin, *Statistical analysis with missing data*, vol. 793. John Wiley & Sons, 2019.

[47] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari, "Learning with noisy labels," in *Advances in neural information processing systems*, pp. 1196–1204, 2013.

[48] B. Frénay and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 5, pp. 845–869, 2014.

[49] C. Elkan, "The foundations of cost-sensitive learning," in *International joint conference on artificial intelligence*, vol. 17, pp. 973–978, Lawrence Erlbaum Associates Ltd, 2001.

[50] M. C. du Plessis, G. Niu, and M. Sugiyama, "Analysis of learning from positive and unlabeled data," in *Advances in neural information processing systems*, pp. 703–711, 2014.

[51] Y.-H. Lin and H. H. Chen, "Tag propagation and cost-sensitive learning for music auto-tagging," *IEEE Transactions on Multimedia*, vol. 23, pp. 1605–1616, 2021.

[52] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Annual conference on Computational learning theory*, pp. 92–100, ACM, 1998.

[53] Z.-H. Zhou and J.-M. Xu, "On the relation between multi-instance learning and semi-supervised learning," in *International conference on Machine learning*, pp. 1167–1174, ACM, 2007.

[54] Y. Li, D. M. Tax, R. P. Duin, and M. Loog, "The link between multiple-instance learning and learning from only positive and unlabelled examples," in *International Workshop on Multiple Classifier Systems*, pp. 157–166, Springer, 2013.

[55] G. Blanchard, G. Lee, and C. Scott, "Semi-supervised novelty detection," *Journal of Machine Learning Research*, vol. 11, no. Nov, pp. 2973–3009, 2010.

[56] J. Hernández-González, I. Inza, and J. A. Lozano, "Weak supervision and other non-standard classification problems: a taxonomy," *Pattern Recognition Letters*, vol. 69, pp. 49–55, 2016.

[57] Y.-F. Li, L.-Z. Guo, and Z.-H. Zhou, "Towards safe weakly supervised learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 334–346, 2019.

[58] L. Liu and T. Peng, "Clustering-based method for positive and unlabeled text categorization enhanced by improved tfidf.," *J. Inf. Sci. Eng.*, vol. 30, no. 5, pp. 1463–1481, 2014.

[59] M. N. Nguyen, X.-L. Li, and S.-K. Ng, "Positive unlabeled learning for time series classification," in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[60] C.-J. Hsieh, N. Natarajan, and I. Dhillon, "Pu learning for matrix completion," in *International conference on machine learning*, pp. 2445–2453, PMLR, 2015.

[61] Y. Yao, T. Liu, B. Han, M. Gong, G. Niu, M. Sugiyama, and D. Tao, "Towards mixture proportion estimation without irreducibility," *arXiv preprint arXiv:2002.03673*, 2020.

[62] M. C. du Plessis, G. Niu, and M. Sugiyama, "Class-prior estimation for learning from positive and unlabeled data," in *Asian Conference on Machine Learning*, pp. 221–236, PMLR, 2016.

[63] S. Jain, M. White, and P. Radivojac, "Estimating the class prior and posterior from noisy positives and unlabeled data," in *Advances in Neural Information Processing Systems*, pp. 2693–2701, 2016.

[64] T. Rotter, L. Kinsman, E. L. James, A. Machotta, H. Gothe, J. Willis, P. Snow, and J. Kugler, "Clinical pathways: effects on professional practice, patient outcomes, length of stay and hospital costs," *Cochrane Database of Systematic Reviews*, no. 3, 2010.

[65] A. Awad, M. Bader-El-Den, and J. McNicholas, "Patient length of stay and mortality prediction: a survey," *Health services management research*, vol. 30, no. 2, pp. 105–120, 2017.

[66] W. E. Pofahl, S. M. Walczak, E. Rhone, and S. D. Izenberg, "Use of an artificial neural network to predict length of stay in acute pancreatitis," *The American Surgeon*, vol. 64, no. 9, p. 868, 1998.

[67] A. Morton, E. Marzban, G. Giannoulis, A. Patel, R. Aparasu, and I. A. Kakadiaris, "A comparison of supervised machine learning techniques for predicting short-term in-hospital length of stay among diabetic patients," in *IEEE International Conference on Machine Learning and Applications*, pp. 428–431, IEEE, 2014.

[68] P. R. Hachesu, M. Ahmadi, S. Alizadeh, and F. Sadoughi, "Use of data mining techniques to determine and predict length of stay of cardiac patients," *Healthcare informatics research*, vol. 19, no. 2, p. 121, 2013.

[69] G. Harerimana, J. W. Kim, and B. Jang, "A deep attention model to forecast the length of stay and the in-hospital mortality right on admission from ICD codes and demographic data," *Journal of Biomedical Informatics*, vol. 118, p. 103778, 2021.

[70] S. Wu, L. Xue, H. Legido-Quigley, M. Khan, H. Wu, X. Peng, X. Li, and P. Li, "Understanding factors influencing the length of hospital stay among non-severe COVID-19 patients: A retrospective cohort study in a fangcang shelter hospital," *Plos one*, vol. 15, no. 10, p. e0240959, 2020.

[71] E. M. Rees, E. S. Nightingale, Y. Jafari, N. R. Waterlow, S. Clifford, C. A. Pearson, T. Jombart, S. R. Procter, G. M. Knight, C. W. Group, *et al.*, "COVID-19 length of hospital stay: a systematic review and data synthesis," *BMC medicine*, vol. 18, no. 1, pp. 1–22, 2020.

[72] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green, "Automatic generation of social tags for music recommendation," *Advances in neural information processing systems*, vol. 20, pp. 385–392, 2007.

[73] Y.-H. Lin, C.-H. Chung, and H. H. Chen, "Playlist-based tag propagation for improving music auto-tagging," in *European Signal Processing Conference*, pp. 2270–2274, IEEE, 2018.

[74] K. M. Ibrahim, J. Royo-Letelier, E. V. Epure, G. Peeters, and G. Richard, "Audio-based auto-tagging with contextual tags for music," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 16–20, IEEE, 2020.

[75] T. E. Oliphant, *A guide to NumPy*, vol. 1. Trelgol Publishing USA, 2006.

[76] H. A. Sturges, "The choice of a class interval," *Journal of the american statistical association*, vol. 21, no. 153, pp. 65–66, 1926.

[77] D. Freedman and P. Diaconis, "On the histogram as a density estimator: L_2 theory," *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, vol. 57, no. 4, pp. 453–476, 1981.

[78] M. P. Wand and M. C. Jones, *Kernel smoothing*. CRC press, 1994.

[79] A. N. Kolmogorov and A. T. Bharucha-Reid, *Foundations of the theory of probability: Second English Edition*. Courier Dover Publications, 2018.

[80] S. Seabold and J. Perktold, "Statsmodels: Econometric and statistical modeling with Python," in *Python in Science Conference*, vol. 57, pp. 92–96, Austin, TX, 2010.

[81] M. Saerens, P. Latinne, and C. Decaestecker, "Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure," *Neural computation*, vol. 14, no. 1, pp. 21–41, 2002.

[82] A. G. Gray and A. W. Moore, "Nonparametric density estimation: Toward computational tractability," in *SIAM International Conference on Data Mining*, pp. 203–211, SIAM, 2003.

[83] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.

[84] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, vol. 14, no. 2, pp. 241–258, 2020.

[85] T. Arjannikov and G. Tzanetakis, "Histogram-based asymmetric relabeling for learning from only positive and unlabeled data," in *IEEE International Conference on Machine Learning and Applications*, pp. 1065–1070, IEEE, 2017.

[86] J. Kittler, M. Hatef, R. P. Duin, and J. Matas, "On combining classifiers," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[87] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[88] J. Han, M. Kamber, and J. Pei, "Data mining concepts and techniques third edition," *The Morgan Kaufmann Series in Data Management Systems*, vol. 5, no. 4, pp. 83–124, 2011.

[89] E. Fix and J. L. Hodges, "Discriminatory analysis. nonparametric discrimination: Consistency properties," *International Statistical Review/Revue Internationale de Statistique*, vol. 57, no. 3, pp. 238–247, 1989.

[90] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[91] P. Moutafis, M. Leng, and I. A. Kakadiaris, "An overview and empirical comparison of distance metric learning methods," *IEEE transactions on cybernetics*, vol. 47, no. 3, pp. 612–625, 2016.

[92] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification.," *Journal of machine learning research*, vol. 10, no. 2, 2009.

[93] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[94] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Routledge, 2017.

[95] L. Rokach, "Decision forest: Twenty years of research," *Information Fusion*, vol. 27, pp. 111–125, 2016.

[96] Y. Chen, X. S. Zhou, and T. S. Huang, "One-class svm for learning in image retrieval," in *International Conference on Image Processing (Cat. No. 01CH37205)*, vol. 1, pp. 34–37, IEEE, 2001.

[97] J. Weston and C. Watkins, "Multi-class support vector machines," tech. rep., CSD-TR-98-04., Depanment of Computer Science, Royal Holloway, University of London, Egham, TW20 OEX, UK, 1998.

[98] F. Cai and V. Cherkassky, "Svm+ regression and multi-task learning," in *2009 International Joint Conference on Neural Networks*, pp. 418–424, IEEE, 2009.

[99] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[100] H. Yu, "Single-class classification with mapping convergence," *Machine Learning*, vol. 61, no. 1, pp. 49–69, 2005.

[101] X.-L. Li and B. Liu, "Learning from positive and unlabeled examples with different data distributions," in *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pp. 218–229, Springer, 2005.

[102] C. Wang, C. Ding, R. F. Meraz, and S. R. Holbrook, "PSoL: a positive sample only learning algorithm for finding non-coding RNA genes," *Bioinformatics*, vol. 22, no. 21, pp. 2590–2596, 2006.

[103] F. He, T. Liu, G. I. Webb, and D. Tao, "Instance-dependent PU learning by bayesian optimal relabeling," *arXiv preprint arXiv:1808.02180*, 2018.

[104] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, and X. Geng, "Binary relevance for multi-label learning: an overview," *Frontiers of Computer Science*, vol. 12, no. 2, pp. 191–202, 2018.

[105] T. Ke, H. Lv, M. Sun, and L. Zhang, "A biased least squares support vector machine based on Mahalanobis distance for PU learning," *Physica A: Statistical Mechanics and its Applications*, vol. 509, pp. 422–438, 2018.

[106] C. Scott, G. Blanchard, and G. Handy, "Classification with asymmetric label noise: Consistency and maximal denoising," in *Conference On Learning Theory*, pp. 489–511, 2013.

[107] S. Jain, M. White, M. W. Trosset, and P. Radivojac, "Nonparametric semi-supervised learning of class proportions," *arXiv preprint arXiv:1601.01944*, 2016.

[108] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, "The balanced accuracy and its posterior distribution," in *International Conference on Pattern Recognition*, pp. 3121–3124, 2010.

[109] M. Sugiyama, T. Kanamori, T. Suzuki, M. C. du Plessis, S. Liu, and I. Takeuchi, "Density-difference estimation," *Neural Computation*, vol. 25, no. 10, pp. 2734–2775, 2013.

[110] M. Sugiyama, T. Suzuki, and T. Kanamori, "Density-ratio matching under the Bregman divergence: a unified framework of density-ratio estimation," *Annals of the Institute of Statistical Mathematics*, vol. 64, no. 5, pp. 1009–1044, 2012.

[111] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[112] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas, "MULAN: A java library for multi-label learning," *Journal of Machine Learning Research*, vol. 12, pp. 2411–2414, 2011.

[113] P. Baldi, K. Cranmer, T. Faucett, P. Sadowski, and D. Whiteson, "Parameterized neural networks for high-energy physics," *The European Physical Journal C*, vol. 76, p. 235, 2016.

[114] R. Lyon, "HTRU2." "http://archive.ics.uci.edu/ml", 2017. UCI Machine Learning Repository, DOI: 10.6084/m9.figshare.3080389.v1.

[115] R. B. Bhatt and A. Dhall, "Skin segmentation dataset." "http://archive.ics.uci.edu/ml", 2012. UCI Machine Learning Repository.

[116] R. B. Bhatt, G. Sharma, A. Dhall, and S. Chaudhury, "Efficient skin region segmentation using low complexity fuzzy decision tree model," in *Annual IEEE India Conference*, pp. 1–4, IEEE, 2009.

[117] A. Dhall, G. Sharma, R. Bhatt, and G. Khan, "Adaptive digital makeup," *Advances in Visual Computing*, pp. 728–736, 2009.

[118] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning," *Nature communications*, vol. 5, p. 4308, 2014.

[119] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-WIDE: a real-world web image database from National University of Singapore," in *ACM international conference on image and video retrieval*, p. 48, ACM, 2009.

[120] E. Spyromitros-Xioufis, S. Papadopoulos, I. Y. Kompatsiaris, G. Tsoumakas, and I. Vlahavas, "A comprehensive study over vlad and product quantization in large-scale image retrieval," *IEEE Transactions on Multimedia*, vol. 16, no. 6, pp. 1713–1728, 2014.

[121] C. G. Snoek, M. Worring, J. C. Van Gemert, J.-M. Geusebroek, and A. W. Smeulders, "The challenge problem for automated detection of 101 semantic concepts in multimedia," in *ACM international conference on Multimedia*, pp. 421–430, ACM, 2006.

[122] B. Stocker, H. K. Weiss, N. Weingarten, K. Engelhardt, M. Engoren, and J. Posluszny, "Predicting length of stay for trauma and emergency general surgery patients," *The American Journal of Surgery*, vol. 220, no. 3, pp. 757–764, 2020.

[123] "Data was obtained from Janatahack: Healthcare Analytics II." https://datahack.analyticsvidhya.com/contest/janatahack-healthcare-analytics-ii/. Accessed: 2021-02-23.

[124] "Hospital inpatient discharges (SPARCS De-Identified): 2017." https://health.data.ny.gov/dataset/Hospital-Inpatient-Discharges-SPARCS-De-Identified/22g3-z7e7. Accessed: 2021-03-11.

[125] A. Clauset, C. R. Shalizi, and M. E. Newman, "Power-law distributions in empirical data," *SIAM review*, vol. 51, no. 4, pp. 661–703, 2009.

[126] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.

[127] E. Law, K. West, M. I. Mandel, M. Bay, and J. S. Downie, "Evaluation of algorithms using games: The case of music tagging," in *International Society for Music Information Retrieval Conference*, pp. 387–392, ISMIR, 2009.

[128] E. Law, B. Settles, and T. Mitchell, "Learning to tag using noisy labels," in *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pp. 1–29, 2010.

[129] D. Tingle, Y. Kim, and D. Turnbull, "Exploring automatic music annotation with acoustically-objective tags," in *International conference on Multimedia information retrieval*, pp. 55–62, ACM, 2010.

[130] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in Python," in *Python in science conference*, vol. 8, pp. 18–25, 2015.

[131] J. Pons, O. Nieto, M. Prockup, E. M. Schmidt, A. F. Ehmann, and X. Serra, "End-to-end learning for music audio tagging at scale," in *International Society for Music Information Retrieval Conference*, pp. 637–644, ISMIR, 2018.

[132] J. Pons and X. Serra, "musicnn: pre-trained convolutional neural networks for music audio tagging," in *Late-breaking/demo session in 2019 International Society for Music Information Retrieval Conference*, ISMIR, 2019.

[133] D. A. Torres, D. Turnbull, L. Barrington, and G. R. Lanckriet, "Identifying words that are musically meaningful," in *International Society for Music Information Retrieval Conference*, vol. 7, pp. 405–410, 2007.

[134] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere, "Autotagger: A model for predicting social tags from acoustic features on large music databases," *Journal of New Music Research*, vol. 37, no. 2, pp. 115–135, 2008.

[135] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE transactions on multimedia*, vol. 13, no. 2, pp. 303–319, 2010.

[136] G. Song, Z. Wang, F. Han, S. Ding, and X. Gu, "Music auto-tagging using scattering transform and convolutional neural network with self-attention," *Applied Soft Computing*, vol. 96, p. 106702, 2020.

[137] H. Jain, Y. Prabhu, and M. Varma, "Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 935–944, 2016.

[138] W. Bi and J. Kwok, "Multilabel classification with label correlations and missing labels," in *AAAI Conference on Artificial Intelligence*, vol. 28, pp. 1680–1686, 2014.

[139] P. Teisseyre, "Classifier chains for positive unlabelled multi-label learning," *Knowledge-Based Systems*, vol. 213, p. 106709, 2021.

[140] B. L. Sturm, "Classification accuracy is not enough," *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 371–406, 2013.