KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Learning from Positive and Unlabeled Data

Jessa Bekker

Supervisor:
Prof. dr. Jesse Davis

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Computer Science

December 2018

Learning from Positive and Unlabeled Data

Jessa BEKKER

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Computer Science

December 2018

Acknowledgments

Four years ago, I started my PhD on "Machine Learning Techniques for Physical Therapy". The following years looked completely different from what I expected. Not only did I switch topics (twice!), but doing research was also much more difficult than I anticipated. Overall, I am very glad that I did a PhD and I would absolutely advise my past self to take up the challenge. However,it was certainly a journey with ups and downs. A lot of people are very important to me for enabling the ups and getting me out of the downs. I would like to thank all of them and highlight some of them now.

Jesse played the most important role in my PhD. He sparked my interested in learning from positive and unlabeled data, was always available for guidance and feedback, and would always find the right carrot to make me turn my research into a finished product. He also set a great example in spending energy on things that matter (qualitative research, family and desserts) and not on those that do not (clean desks, typo-free e-mails and mercurial people). In summary, his research skills dovetail perfectly with his personality to make a great advisor.

A big thank you goes out to my jury. Benedicte and Tinne have been following my PhD from the start. I did not exactly make it easy for them, because I changed topics in between all evaluation moments. I am especially impressed with Benedicte, for not only staying in my committee, but also being able to follow and ask insightful questions on topics that are only remotely related to her domain of expertise. But also Tinne impressed me: by the level of detail in which she reviewed my work every time and, additionally, by often being the voice of reason at the faculty council. Straffe Hendrik is of course more than a jury member to me. I always enjoyed our little conversations at the coffee machine or in the hallway. Benoît, you are the real expert on my jury, it is a shame that we did not see the connection earlier. Finally, Adhemar, thank you for chairing my defense.

I would like to thank the Agency for Innovation by Science and Technology in Flanders (IWT), that is now part of the Agency for Innovation and Entrepreneurship (VLAIO), for supporting my research by awarding me with a ‘strategisch basisonderzoek’ PhD fellowship.

One of the main reasons I chose to do a PhD in the Machine Learning group is because I had spotted the group a couple of times on and off campus. They seemed like a fun crowd that I would enjoy being part of. And I was not disappointed! Under Irma’s lead, it was an active and welcoming group. They quickly filled my free time with drinks on the old market, board game nights, trips, and legendary house parties. Thank you Irma, Antoine, Benjamin, Vova, Kurt, Gitte, Sergey&Irina, Wannes, Jan, Jonas, Joris, Bogdan.... I was not the first (initial) outsider to spot the greatness of this group. So did Leen, who also found a way to become a permanent member, granted, she used a different approach :) It was a pleasure for me to have Vova first as a great teacher and later as a great friend. Kurt, you are a big part of the ML spirit: joining every party and always helping where possible. Anton, I really appreciated our 8am coffees. Sergey&Irina you are an awesome combination of absurdness and unlimited kindness, coincidentally this also applies to Masya. Irma, everything is just more fun when you are around.

The group has changed and grown a lot over the years, but it is still a great bunch to be part of! The common coffee breaks, lunches and drinks lie at the heart of the social group that ML is. Tom, Vincent, Robin, Pedro, Nitesh, Mohit, Samuel, Stefano, Sebastijan, Toon, Elia, Jonas, Laurens, Evgeniya, Kilian, Gust, Arcchit, Ondrej, Pieter and Arne, thanks for keeping these habits alive! Vincent and Tom, I immensely enjoyed being your officemate, I think we hit the sweet spot of fun, productiveness and inspiring each other to do better. Vincent, I really enjoy listening to you, both when you explain your research and your personal life ;) Tom, by syncing our lives (training for the same running goals and buying a house/property), we never run out of chatting material. Toon and Sebastijan, we have gone through all milestones together, which does create a special bond. Elia and Stefano, I really had the best time with you at Mardi Gras! Jonas, I love it to have a fellow Harry Potter geek around who irrationally stresses out in a similar fashion to me! Ondrej, keep spreading the word about the awesomeness of Leuven and our group!

Three honorable mentions are due. Sebastijan, you are an inspiration to me: I don’t know anyone who is as passionate about their research as you. And yet, you understand that not everyone is like you and that this is not out of laziness. I think Nikolina has a great deal to do with this. I love our ice cream dates (even the depressing ones), where I would give Nikolina an update on all the gossip that you forgot to tell her. I know that we will continue this tradition, but now Seb will have to do a bigger effort to remember the gossip!

The third one is Behrouz. I could always count on you to be in the department during the weekend, so I always had great company for coffee to brighten up sad weekend-work days. A profound friendship originated from this with many secret gatherings as a result. Everything is better when it is secret :) Nikolina, Sebastijan and Behrouz, you have been there for me when it was most needed, I hope you know how much that means to me.

I always found it hard to balance several aspects in life: I tend to focus 100% on what seems most urgent at that moment. Doing sports regularly is therefore a challenge. A great motivation to keep going, were the various sport groups that we formed with colleagues. The 'excuses of DTAI', formerly known as 'runners of DTAI', provided me with fun running sessions that always managed to clear my head. Thank you Tom, Kilian, Stella, Pieterjan, Toon, Robin, Arne en Greg! During my research visit in Los Angeles, I gratefully made use of the excellent weather and seven on-campus, open-air swimming pools to start a new hobby. When I came back, I continued this hobby despite of the not-so-excellent weather and slightly less awesome infrastructure. The big improvement here was the company. Sergey, Jonas, Robin, Mohit and Evgeniya, swimming is not really a group-sport, but the dinner and drinks afterwards definitely made it feel like it! Finally, I also really enjoyed being part of the volley group, however, it was a bit unfortunate that I haven't been able to attend for the last 150 sessions or so…

During my research visit in at UCLA, Los Angeles, I made two friends for life: Arthur and Jason. I was lucky to keep seeing them at conferences and I am very sad that now that my PhD is finished, it is unlikely that we will be going to the same conferences. When I lived in LA, I spent all my time with Arthur: at work we shared an office and during the weekend, I would bike down to Culver City to distract him from his work. Jason is the most research-curious person that I have ever met, he is a true inspiration to me.

Although most of my friends are not directly related to my PhD, they are very important to me and undoubtedly endured some (or a lot of) complaining from my side and gave me good general life advise. Menno, thank you for being my friend without an obvious origin, Monday evenings are not the same without you! Hanne, Marie, Jutta, Prianka en Liza, you are my oldest friends. I love it that we always share our suffering while usually having a lot of fun doing primary-school activities that we never feel too old for. Veerle, Kaat&Coco, Anne-Elise&Christophe, Sjana&Shubi, Sanne&Nils en Elke, I immensely enjoy the get-togethers from the babysquat!

Although, Ruben and I have been living by ourselves for some time now, we have some unofficial roommates who from time to time were forced to give exclusively positive feedback on slides and papers. Timon, you will always be our ‘teenage son’. Anna, you are always welcome to stay with us! It is rare for me to be so comfortable with visitors as I am with you. Pieter, please keep coming by, even when it is just to use our faster internet or to train on Ruben’s bike!

My sister Gerlinde is probably the person that I see most of the people that I do not live or work with. I love our culinary activities: cooking, eating ice cream, wine tasting, etc., during which I could always share my PhD stories. After heavy work periods in which I had neglected all my friends, I could always count on her to reintroduce me to the social world. She is also my greatest advertiser: all her patients know about me and my work through her, which has led to valuable connections!

Moeke en Vake, my parents, know that my childhood dream was to become an inventor. They have always supported me to follow this direction and made me believe that I could do it. I think it is extraordinary how they manage to be highly supportive while never being pushy; they always trust me to reach my goals on my own and know that making mistakes and detours are a necessary part of the journey.

I got the inventor-gene from my grandfather and my grandmother is my role model for being dedicated to work while maintaining a fulfilling social life. Oma en opa did not always know exactly what I was doing, but they were always convinced that it was important. I love that I am always invited for lunch on Thursday, and that they always understand when I cannot make it because of my PhD.

Last but definitely not least: Ruben. Thank you for always providing a nice home for me to return to. I think it is pretty amazing that you never lose patience with me, even when I’m in full-on stress mode. You are always there for me to remind me about what really matters.

Abstract

The goal of binary classification is to train a model that can distinguish between examples belonging to one of two classes: positive and negative. Traditional algorithms for training such a model assume access to labeled examples, where the label is the desired output of the model: positive or negative. However, in practice the labels might be hard to obtain for various reasons: the need for expert knowledge, costly tests, or just the vast number of examples. Therefore, it is desirable to have algorithms that can learn when only a fraction of the examples are labeled. In some situations, only a subset of the positive examples are labeled, but none of the negative examples. For example, the task of predicting if a patient has a disease, can use patients that were diagnosed with that disease as positive examples, however, being undiagnosed does not imply not having the disease. Or, for movie recommendation, watched movies are positive examples of good movies, but unwatched movies can also be good. The field that aims to design algorithms to learn from this kind of data is called *learning from positive and unlabeled data* or PU learning in short. To enable learning in this harder setting, assumptions need to be made either about the distribution of the classes, or the labeling mechanism that dictates which examples get labeled, or both.

This dissertation studies the commonly made assumptions in PU learning with two objectives: 1) to exploit them for improving PU algorithms, and 2) to propose more realistic assumptions that still enable learning. The assumption of interest for this dissertation is the common *selected completely at random* (SCAR) assumption, which assumes that the set of labeled examples is a uniform subset of the positive examples. In this case, learning is possible if the class prior, that is the ratio of positive examples in the data, is known. Therefore, estimating the class prior is a crucial subtask in PU learning.

This dissertation has four main contributions. First, it provides a comprehensive survey of the field. Second, it proposes a novel class prior estimation algorithm that is equally accurate as the state-of-the-art methods while being an order of magnitude faster. Third, it investigates whether the SCAR assumption can also be leveraged when learning from relational PU data and shows how the propositional techniques can be modified to this end. Fourth, it proposes a more realistic assumption: *selected at random* (SAR), which assumes that the probability for a positive example to be selected to be labeled can depend on its attributes. This is a big step forward for PU learning, as it enables addressing numerous new real problems. For this setting, it shows how a known labeling mechanism can be taken into account by the learning algorithm, it investigates which additional assumptions are necessary to estimate the labeling mechanism from the data, and it proposes a practical algorithm to learn in this setting when the labeling mechanism is unknown.

Beknopte samenvatting

Het doel van de binaire classificatietaak is om een model te trainen dat het onderscheid kan maken tussen twee klassen: positief en negatief. Traditionele algoritmen om zulke modellen te trainen veronderstellen toegang te hebben tot gelabelde voorbeelden, waar het label de gewenste uitvoer van het model is: positief of negatief. In de praktijk is het echter soms moeilijk om zulke labels te verkrijgen, vanwege verschillende redenen: de nood aan expertenkennis, kostelijke testen, of gewoon de enorme hoeveelheid aan voorbeelden. Daarom is het wenselijk om over algoritmen te beschikken die kunnen leren wanneer slechts een fractie van de voorbeelden gelabeld is. In sommige situaties is slechts een deel van de positieve voorbeelden gelabeld en geen enkele van de negatieve voorbeelden. Wanneer we bijvoorbeeld trachten om patiënten te identificeren die een ziekte hebben, kunnen patiënten die de diagnose van die ziekte gehad hebben, gebruikt worden als positieve voorbeelden. Het is echter niet zo dat het ontbreken van een diagnose impliceert dat de patiënt de ziekte niet heeft. Het aanbevelen van films is een ander voorbeeld van deze situatie. Films die iemand bekeken heeft in het verleden zijn films waar die persoon in geïnteresseerd is, maar onbekeken films kunnen ook interessant zijn. Het domein dat als doel heeft om algoritmen te ontwerpen voor dit type data heet *leren van positieve en ongelabelde voorbeelden* of PU learning in het kort. Om het mogelijk te maken in deze moeilijkere setting te leren, moeten er veronderstellingen gemaakt worden over ofwel de verdeling van de data, ofwel het mechanisme waarmee positieve voorbeelden geselecteerd werden om te labelen, ofwel over beide.

Dit proefschrift bestudeert de veelgemaakte veronderstellingen door PU learning met twee doelen: 1) deze veronderstellingen te gebruiken om betere PU learning algoritmen te ontwerpen, en 2) om realistischere veronderstellingen voor te stellen waarmee het nog steeds mogelijk is om te leren. De belangrijkste veronderstelling voor dit proefschrift is de veelgemaakte *Selected Completely At Random* (SCAR) veronderstelling die er vanuit gaat dat de gelabelde voorbeelden volledig willekeurig geselecteerd werden van de verzameling positieve voorbeelden. In dit geval is het mogelijk om goede modellen te leren wanneer

de a-priori positieve kans gekend is, dit is ratio positieve voorbeelden in de data. Het schatten van die kans is daarom een cruciale deeltaak van PU learning.

Dit proefschrift heeft vier hoofdbijdragen. Ten eerste geeft het een uitgebreid overzicht van het domein. Ten tweede stelt het een nieuwe methode voor om de a-priori positieve kans te schatten. Deze methode is even accuraat als de beste technieken die tot nu toe gekend zijn, maar is een ordegrootte sneller. Ten derde onderzoekt het of de SCAR veronderstelling ook bruikbaar is voor het leren van relationele PU data en toont het hoe bestaande propositionele technieken hiervoor omgevormd kunnen worden. Tenslotte stelt het een nieuwe, realistischere veronderstelling voor: *Selected At Random* (SAR), die er rekening mee houdt dat de kans voor een positief voorbeeld om geselecteerd te worden kan afhangen van de waarden van zijn attributen. Dit is een grote stap voor PU learning want hiermee is het in staat om tal van nieuwe problemen aan te pakken. Voor deze setting wordt er getoond hoe een gekend labelingsmechanisme in rekening gebracht kan worden tijdens het leren, wordt er onderzocht welke bijkomende veronderstellingen er nodig zijn om het labelingsmechanisme af te leiden uit de data en wordt er een praktisch algoritme voorgesteld dat in staat is een model te leren wanneer het labelingsmechanisme ongekend is.

List of Abbreviations

MAE Mean Absolute Error. 94

MSE Mean Square Error. 94, 108, 110, 112, 116

NU Negative and Unlabeled. 56, 61, 63, 65, 67

PU Positive and Unlabeled. v, vi, viii, 2–5, 47, 48, 52, 54–56, 60, 62, 64, 66, 68, 70, 73, 74, 76–81, 89, 90, 92–94, 101, 102, 106, 108, 113, 117–121

PU learning Learning from positive and unlabeled data. v–viii, 2, 3, 5, 7, 70, 73, 87, 90, 91, 101, 105, 117–121

SAR Selected At Random. vi, viii, 4–6, 90, 92, 99, 101, 106, 108, 111–115, 117, 119, 121

SCAR Selected Completely At Random. v–viii, 2–5, 47, 49, 69, 70, 74, 76, 89–92, 101, 105, 108–112, 114, 115, 117–119, 121

SNAR Selected Not At Random. 90, 92

List of Symbols

x	The vector of attributes of an example
$\mathbf{x}$	A set of vectors of attributes of examples
y	Indicator variable for an example to be positive
$\mathbf{y}$	A set of indicator variables for examples to be positive
s	Indicator variable for an example to be labeled
$\mathbf{s}$	A set of indicator variables for examples to be labeled
α	Class prior $\alpha = \Pr(y = 1)$
c	Label frequency $c = \Pr(s = 1\|y = 1)$
e	Propensity score function $e(x) = \Pr(s = 1\|y = 1, x)$
$f(x)$	Probability density function of the instance space (true population)
$f_+(x)$	Probability density function of the positive instance space
$f_-(x)$	Probability density function of the negative instance space
$f_l(x)$	probability density function of the labeled instance space
$f_u(x)$	Probability density function of the unlabeled instance space
P	Number of positive examples
N	Number of negative examples
L	Number of labeled examples
U	Number of unlabeled examples
T	Total number of examples
$\hat{\bullet}$	An estimate for $\bullet$.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

It wasn't a dark and stormy night. It should have been, but that's the weather for you. For every mad scientist who's had a convenient thunderstorm just on the night his Great Work is finished and lying on the slab, there have been dozens who've sat around aimlessly under the peaceful stars while Igor clocks up the overtime.

Neil Gaiman & Terry Pratchett, Good Omens

One goal of machine learning is to learn a predictive model from data. In binary classification, the task is to predict for unseen examples to which of two classes they belong, usually called 'positive' and 'negative'. When training a binary classifier, the data the machine learning algorithm has access to, consists of examples that are labeled as positive or negative.

In reality, it can be difficult, or even impossible, to obtain a sufficiently large, representative, labeled dataset. It can be challenging for several reasons: Expert knowledge might be required, when for example the task is to predict if people have diabetes. Additional costly tests, like MRI scans, could be needed to know the class with certainty. Additionally, a lot of data might be needed, either because of the problem's complexity or the need to retrain the model on a regular basis to keep up with reality, as would be the case for news article recommendation. The need for large amounts of data amplifies the previous problems.

In some scenarios, it is considerably easier to obtain labeled examples from one class than the other, in fact, sometimes, it may even be impossible to obtain labeled examples for one of the classes. For example, to predict whether people have a disease, patients that were diagnosed with it are examples of the disease class. However, people who were not diagnosed may still have the disease [19]. Another example scenario is recommendation systems that recommend products of interest. Examples of interesting products are those that a user bought before. However, other reasons than disinterest exist for not buying a product [120]. These kind of datasets, that contain labeled examples from one class and unlabeled for both classes, are called *Positive and Unlabeled (PU)* data [65, 30]. By convention, the class for which labels are available is considered the positive class.

Learning from PU data (PU learning) is much harder than learning from fully labeled datasets. Therefore, additional assumptions are necessary to enable learning. The most common assumptions are 1) separability, where the positive and negative examples are expected not to overlap [61, 71], and 2) *Selected Completely At Random (SCAR)*, where the set of labeled examples is a uniformly random subset of the set of positive examples [61, 30, 82, 94, 48].

One way to learn from SCAR PU data is to modify traditional learning methods to work for PU data by incorporating the class prior, which is the prior probability of the positive class [22, 12, 30, 44, 25]. The class prior can be known from domain knowledge or estimated from a smaller fully-labeled dataset. However, when the class prior is unknown, it can be estimated directly from the PU data [30, 6, 99, 24, 94, 48].

Despite the tremendous progress made on analyzing the PU learning setting, the existing literature is highly focused and we identified three important gaps in the current literature.

First, the current literature focuses on small datasets. However, in many PU learning scenarios, the datasets are likely to be large. For example, disease prediction can use electronic health data [19] and automatically constructed knowledge bases like YAGO and the Google Knowledge Graph consists of millions of facts [78, 107]. Current accurate approaches for class prior estimation in PU data are not scalable [94, 48]. Hence, there is a need for accurate and scalable class prior estimation methods.

Second, the current literature focuses on propositional data. However, PU data also naturally occurs for typical problems that have relational data. For example, knowledge base completion is the problem that aims to automatically complete a knowledge base, which consists of known relations between objects [78, 107]. Here, the positive and unlabeled examples are the relations that respectively appear or do not appear in the knowledge base. Despite the common presence of relational PU data, the relational and propositional PU learning fields have developed mostly independently. As a result, it is not yet known if important insights, like the effectiveness of the SCAR assumption, can be exploited for relational problems.

Third, the current literature focuses mostly on simplistic assumptions like class separability and SCAR labels [61, 71, 30, 82, 94, 48]. Unfortunately, these assumptions are often violated in real-world datasets. For example, a patient's medical record will only contain a diagnosis if she visits a doctor, which will be influenced by factors such as the severity of the symptoms and her socio-economic status. Within the context of PU learning, there has been little (or no) work that focuses on coping with biases in the observed positive labels.

1.1 Dissertation Statement

This dissertation pushes the boundaries of learning from positive and unlabeled data (PU learning) by addressing the aforementioned gaps. Concretely, this is approached by studying the current assumptions made in this field. We aim to answer the following questions:

1. Can the SCAR assumption be exploited to enable scalable class prior estimation in PU data?
2. How can insights and methods based on the SCAR assumption be leveraged for relational PU data?
3. Do more realistic assumptions than the SCAR assumption exist that still enable learning from PU data?

1.2 Contributions

This dissertation addresses its statement through four main contributions.

1.2.1 Contribution 1: Literature Survey

Learning from positive and unlabeled data has been gaining attention in the last two decades. However, there is currently no good overview of the field and it seems to be developing almost independently in different communities. Therefore, presenting the accumulated knowledge of this domain in a structured way is the first contribution of this dissertation.

1.2.2 Contribution 2: Scalable Class Prior Estimation in Positive and Unlabeled Data

We present a novel algorithm for estimating the class prior in PU data under the established *Selected Completely At Random (SCAR)* assumption. The SCAR assumption means that the labeled examples are a uniform random subset of the positive examples. Estimating the class prior is an important step for learning a classifier in this setting.

The key insight behind our algorithm TIcE (Tree Induction for c Estimation), is that bounds on the class prior can be derived from subsets of the PU data. We provide a theoretical study of these bounds and show how to use them to estimate the class prior through an efficient decision tree algorithm.

An extensive empirical evaluation on eleven real-world datasets shows that TIcE's estimates are equivalently accurate as those of the state-of-the-art methods while being an order of magnitude faster.

The python source code of the proposed algorithm TIcE is available on `https://dtai.cs.kuleuven.be/software/tice`.

1.2.3 Contribution 3: Learning from Positive and Unlabeled Relational Data under the Selected Completely At Random Assumption

We investigate whether the SCAR assumption can be utilized when learning from relational data in a similar fashion. To this end, we propose two methods for incorporating the class prior in relational classifier learning. Furthermore, we also show how to modify our method TIcE to TIcER, to operate in the relational domain.

An extensive evaluation on four common relational dataset shows that TIcER performs equally well as established methods for relational data when the data is easily separable into its classes and for more complex tasks, TIcER outperforms them.

1.2.4 Contribution 4: Beyond the Selected Completely At Random Assumption

Often, the SCAR assumption is too restrictive for real problems, therefore we propose the more realistic *Selected At Random* (SAR) assumption. This assumption states that the labeling mechanism is not necessary uniformly random, but, that it can depend on the attribute values of the example. This new assumption is a big step for PU learning as it enables addressing a whole new range of problems where PU data naturally occurs, but the SCAR assumption does not hold. For example, under the SCAR assumption, every person with a disease is equally likely to be diagnosed, while in reality, people with more obvious symptoms will get diagnosed more often.

We propose a method for incorporating a known labeling mechanism when training a classifier and analyze it in an empirical-risk-based framework. Furthermore, an analysis is conducted to see which additional assumptions are necessary to enable learning with an unknown labeling mechanism. Based on this, a practical algorithm is proposed.

An extensive evaluation on eight real-world datasets show that for SAR PU data, our approaches result in improved performance over making the SCAR assumption.

The python source code of this work is available on `https://dtai.cs.kuleuven.be/software/sar`.

1.3 Structure of the Dissertation

The structure of this dissertation is as follows.

Chapter 2 provides the necessary background on learning from positive and unlabeled data, relational data and causal inference. We submitted a survey paper based on the section about positive and unlabeled data to the Machine Learning Journal.

Chapter 3 introduces a scalable method for learning from positive and unlabeled data under the established *Selected Completely At Random* (SCAR) assumption. This chapter is based on the following publication:

Bekker, J., and Davis, J. Estimating the Class Prior in Positive and Unlabeled Data through Decision Tree Induction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018, New Orleans, Louisiana, United States; 2-7 February 2018)* (2018a), pp. 2712–2719.

Chapter 4 shows that propositional insights and methods for learning from positive and unlabeled data can also be used in the relational domain. This chapter is based on the following publication:

Bekker, J., and Davis, J. Positive and Unlabeled Relational Classification through Label Frequency Estimation. In *Inductive Logic Programming (Revised Selected Papers of ILP 2017; Orléans, France; 4-6 September 2017)* (2018b), pp. 16–30. ✸ *Most promising late-breaking student paper.*

Chapter 5 proposes a new, more realistic assumption for learning from positive and unlabeled data: *Selected At Random (SAR)*. It furthermore proposes two methods for learning in this setting: one for when the labeling mechanism is understood and another for when the labeling mechanism is unknown. This chapter is based on the following paper:

Bekker, J., and Davis, J. Beyond the Selected Completely At Random Assumption for Learning from Positive and Unlabeled Data. In *arXiv:1809.03207* (2018d).

Chapter 6 summarizes the contributions presented in this dissertation and provides possible directions for future work.

Chapter 2

Background

This section provides an comprehensive overview of PU learning. The most relevant sections for this dissertation are the 'Selected Completely At Random' and 'Assumptions for an Identifiable Class Prior' in Section 2.2, and, 'Incorporation of the Class Prior' and 'Class Prior Estimation from PU data' in Section 2.4.

Motivated by the significant applications of PU learning, researchers have taken a keen interest in analyzing the PU learning setting. Within PU learning, people have addressed a number of different tasks using a variety of techniques. Despite the breadth, at a high level, the key research questions about PU learning can be formulated rather straightforwardly as:

1. How can we formalize the problem of learning from PU data?
2. What assumptions are typically made about PU data in order to facilitate the design of learning algorithms?
3. Can we estimate the class prior from PU data and why is this useful?
4. How can we learn a model from PU data?
5. How can we evaluate models in a PU setting?
6. When and why does PU data arise in real-world applications?
7. How does PU learning relate to other areas of machine learning?

This survey is structured around giving a comprehensive overview about how the PU learning research community is tackling each of these questions. It concludes with some perspectives about future directions for PU learning research.

The content of this chapter is based on the following publication currently under review for the Machine Learning journal (MLj) [4]:

Bekker, J., and Davis, J. Learning From Positive and Unlabeled Data: A Survey. In *arXiv:1811.04820* (2018e).

2.1 Preliminaries on PU Learning

Learning from positive and unlabeled data (PU learning) is a special case of binary classification. Therefore, we first review binary classification before formally describing the PU learning setting. Then we introduce the labeling mechanism, which is a key concept in PU learning. Finally, we distinguish between two PU learning settings: the single-training-set and case-control scenarios.

2.1.1 Binary Classification

The goal of binary classification is to train a classifier that can distinguish between two classes of instances, based on their attributes. By convention, the two classes are called “positive” and “negative”. To train a binary classifier, the machine learning algorithm has access to a set of training examples. Each training example is a tuple (x, y), where x is the vector of attribute values and y is the class value. An example is positive if $y = 1$ and negative if $y = 0$. Traditional learning algorithms work in a supervised setting, where the training data is assumed to be fully labeled. That is, the class value for each training example is observed. Table 2.1 shows an example of a fully labeled training set. To enable training a correct classifier, the training data is assumed to be an independent and identically distributed (i.i.d.) sample of the real distribution:

$$\begin{aligned}\mathbf{x} &\sim f(x)\\ &\sim \alpha f_+(x) + (1-\alpha) f_-(x), \qquad (2.1)\end{aligned}$$

with class prior $\alpha = \Pr(y = 1)$ and probability density functions of the true distribution f and the positive and negative examples f_+ and f_- respectively.

Table 2.1: **Labeled training set example.** The vector of attribute values are the first 5 rows: $x =$ [age, diabetes family, fatigue, pee/day, blurred vision]

age	diabetes family	fatigue	pee/day	blurred vision	y
25	yes	yes	7	no	0
63	no	yes	10	no	1
49	no	no	4	no	0
34	no	yes	6	yes	1

Table 2.2: **Positive and Unlabeled training set example** for the same dataset as the on in Table 2.1.

age	diabetes family	fatigue	pee/day	blurred vision	y	s
25	yes	yes	7	no	?	0
63	no	yes	10	no	1	1
49	no	no	4	no	?	0
34	no	yes	6	yes	?	0

2.1.2 PU Learning

The goal of PU learning is the same as general binary classification: train a classifier that can distinguish between positive and negative examples based on the attributes. However, during the learning phase, only some of the positive examples in the training data are labeled and none of the negative examples are.

We represent a PU dataset as a set of triplets (x, y, s) with x a vector of attributes, y the class and s a binary variable representing whether the tuple was selected to be labeled. The class y is not observed, but information about it can be derived from the value of s. If the example is labeled $s = 1$, then it belongs to the positive class: $\Pr(y = 1|s = 1) = 1$. When the example is unlabeled $s = 0$, then it can belong to either class. Table 2.2 gives an example of a positive and unlabeled version of a training set. Table 2.3 gives an overview of the notation used in this article.

2.1.3 Labeling Mechanism

The labeled positive examples are selected from the complete set of positive examples according to a probabilistic labeling mechanism, where each positive example x has the probability $e(x) = \Pr(s = 1|y = 1, x)$ of being selected to be labeled, called the *propensity score* [2]. Hence, the labeled distribution is a

Table 2.3: **Notation** used in this article.

Symbol	Description
x	The vector of attributes of an example
$\mathbf{x}$	A set of vectors of attributes of examples
y	Indicator variable for an example to be positive
$\mathbf{y}$	A set of indicator variables for examples to be positive
s	Indicator variable for an example to be labeled
$\mathbf{s}$	A set of indicator variables for examples to be labeled
α	Class prior $\alpha = \Pr(y = 1)$
c	Label frequency $c = \Pr(s = 1\|y = 1)$
e	Propensity score function $e(x) = \Pr(s = 1\|y = 1, x)$
$f(x)$	Probability density function of the instance space (true population)
$f_+(x)$	Probability density function of the positive instance space
$f_-(x)$	Probability density function of the negative instance space
$f_l(x)$	probability density function of the labeled instance space
$f_u(x)$	Probability density function of the unlabeled instance space
P	Number of positive examples
N	Number of negative examples
L	Number of labeled examples
U	Number of unlabeled examples
T	Total number of examples
$\hat{\bullet}$	An estimate for $\bullet$.

biased version of the positive distribution:

$$f_l(x) = \frac{e(x)}{c} f_+(x), \tag{2.2}$$

with $f_l(x)$ and $f_+(x)$ the probability density functions of the labeled and positive distributions respectively. The normalization constant c is the *label frequency*, which is the fraction of positive examples that are labeled $c = \Pr(s = 1|y = 1)$. This can be seen from the following derivation:

$$\begin{aligned}
f_l(x) &= \Pr(x|s = 1) \\
&= \Pr(x|s = 1, y = 1) && \textit{\#by PU definition} \\
&= \frac{\Pr(s = 1|x, y = 1)}{\Pr(s = 1|y = 1)} \Pr(x|y = 1) && \textit{\#Bayes' rule} \\
&= \frac{e(x)}{c} f_+(x)
\end{aligned}$$

2.1.4 The Single-Training-Set and Case-Control Scenarios

The positive and unlabeled examples in PU data can originate from two scenarios. Either they come from a single training set, or they come from two independently drawn datasets, one with all positive examples and one with all unlabeled examples. These scenarios are called the single-training-set scenario and the case-control scenario respectively.

The *single-training-set scenario* assumes that the positive and unlabeled data examples come from the same dataset and that this dataset is an i.i.d. sample from the real distribution, like for supervised classification. A fraction c from the positive examples are selected to be labeled, therefore, the dataset has a fraction αc of labeled examples.

$$\begin{aligned}
\mathbf{x} &\sim f(x) \\
&\sim \alpha f_+(x) + (1-\alpha) f_-(x) \\
&\sim \alpha c f_l(x) + (1-\alpha c) f_u(x). \qquad (2.3)
\end{aligned}$$

The *case-control scenario* assumes that the positive and unlabeled examples come from two independent datasets and that the unlabeled dataset is an i.i.d. sample from the real distribution:

$$\begin{aligned}
\mathbf{x}|\mathbf{s}=\mathbf{0} &\sim f_u(x) \\
&\sim f(x) \\
&\sim \alpha f_+(x) + (1-\alpha) f_-(x). \qquad (2.4)
\end{aligned}$$

The observed positive examples are generated from the same distribution in both the single-training-set and case-control scenario. Hence, in both scenarios the learner has access to a set of examples drawn i.i.d. from the true distribution and a set of examples that are drawn from the positive distribution according to the labeling mechanism that is defined by the propensity score $e(x)$. As a result, most methods can handle both scenarios, but the derivation differs. Consequently, one must always consider the scenario when interpreting results and using software.

The single-training-set scenario has received substantially more attention in the literature. Therefore, this survey assumes this scenario. When methods that were originally proposed in a case-control scenario are discussed on a level where this distinction is necessary, we either convert them to the single-training-set scenario or explicitly state that the case-control scenario is assumed.

2.1.5 Relationship Between the Class Prior and the Label Frequency

The class prior α and the label frequency c are closely related to each other. Given a PU dataset, if one is known, the expected value of the other can be calculated. The label frequency is defined as the fraction of positive examples that are labeled in all the data:

$$\begin{aligned} c &= \Pr(s=1|y=1) \\ &= \frac{\Pr(s=1, y=1)}{\Pr(y=1)} \\ &= \frac{\Pr(s=1)}{\Pr(y=1)}. && \textit{\#by PU definition} \end{aligned}$$

The probability $\Pr(s=1)$ can be counted in the data as the fraction of labeled examples. The probability $\Pr(y=1)$ is related to the class prior. In the single-training-set scenario, it is equal to the class prior. However, in the case control scenario, the class prior is defined in the unlabeled data: $\alpha = \Pr(y=1|s=0)$. Here, the probability $\Pr(y=1)$ is the following:

$$\begin{aligned} \Pr(y=1) &= \Pr(y=1|s=0)\Pr(s=0) + \Pr(y=1|s=1)\Pr(s=1) \\ &= \alpha \Pr(s=0) + \Pr(s=1). \end{aligned}$$

To summarize, the conversions between c and α are done as follows:

$$c = \frac{\Pr(s=1)}{\alpha} \qquad \textit{\# single-training-set scenario} \tag{2.5}$$

$$c = \frac{\Pr(s=1)}{\alpha\,(1-\Pr(s=1)) + \Pr(s=1)} \qquad \textit{\# case-control scenario} \tag{2.6}$$

$$\alpha = \frac{1-c}{c}\frac{\Pr(s=1)}{1-\Pr(s=1)}. \qquad \textit{\# case-control scenario} \tag{2.7}$$

2.2 Assumptions to Enable PU Learning

Learning from PU data is not straightforward. There are two possibilities to explain why an example is unlabeled, either:

1. It is truly a negative example; or

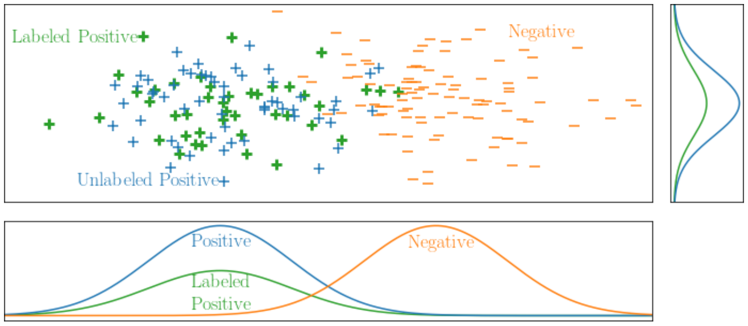

Figure 2.1: **Example of SCAR PU data.** The labeled examples are selected uniformly at random from the positive examples.

2. It is a positive example, but simply was not selected by the labeling mechanism to have its label observed.

Therefore, in order to enable learning with positive and unlabeled data, it is necessary to make assumptions about either the labeling mechanism, the class distributions in the data, or both. The class prior plays an important role in PU learning and many PU learning methods require it as an input. To enable estimating it directly from PU data, additional assumptions need to be made. This section discusses the most commonly made labeling mechanism and data assumptions to enable PU learning as well as the assumptions made to enable estimating the class prior from PU data.

2.2.1 Label Mechanism Assumptions

One approach is to make assumptions about the labeling mechanism. That is, how the examples with an observed positive label were selected.

Selected Completely At Random

The Selected Completely At Random (SCAR) assumption lies at the basis of most PU learning methods, for example, biased learning methods (Section 2.4.2) and methods that directly incorporate the class prior (Section 2.4.3). It assumes that the set of labeled examples is a uniform subset of the set of positive examples [30]. Figure 2.1 shows an examples of a PU dataset under the SCAR assumption.

Definition 1 (Selected Completely At Random (SCAR))**.** *Labeled examples are selected completely at random, independent from their attributes, from the positive distribution. The propensity score $e(x)$, which is the probability for selecting a positive example is constant and equal to the* label frequency *c:*

$$e(x) = \Pr(s = 1|x, y = 1) = \Pr(s = 1|y = 1) = c.$$

Under this assumption, the set of labeled examples is an i.i.d. sample from the positive distribution. Indeed, Equation 2.2 simplifies to $f_l(x) = f_+(x)$.

Under the SCAR assumption, the probability for an example to be labeled is directly proportional to the probability for an example to be positive:

$$\Pr(s = 1|x) = c \Pr(y = 1|x).$$

This enables the use of *non-traditional classifiers*, which are classifiers that predict $\Pr(s = 1|x)$, which are learned by considering the unlabeled examples as negative [30]. These non-traditional classifiers have various interesting properties:

- Non-traditional classifiers preserve the ranking order [30]:

$$\Pr(y = 1|x_1) > \Pr(y = 1|x_2) \Leftrightarrow \Pr(s = 1|x_1) > \Pr(s = 1|x_2).$$

- Training a traditional classifier subject to a desired expected recall, is equivalent to training a non-traditional classifier subject to that recall [72, 6]
- Given the label frequency (or class prior), a probabilistic non-traditional classifier can be converted to a traditional classifier, by dividing the outputs by the label frequency $\Pr(y = 1|x) = \Pr(s = 1|x)/c$ [30].

The SCAR assumption was introduced in analogy with the *Missing Completely A Random assumption (MCAR)* that is common when working with missing data [96, 70]. However, there is a notable difference between the two assumptions. In MCAR data, the missingness of the variable cannot depend on the value of the variable, where in PU learning this is necessarily the case because all negative labels are missing. The class values are missing completely at random only if just the population of positive examples is considered. Moreno et al. (2012) proposed a new missingness class: *Missing Completely At Random-Class Dependent (MAR-C)*, SCAR belongs to this category.

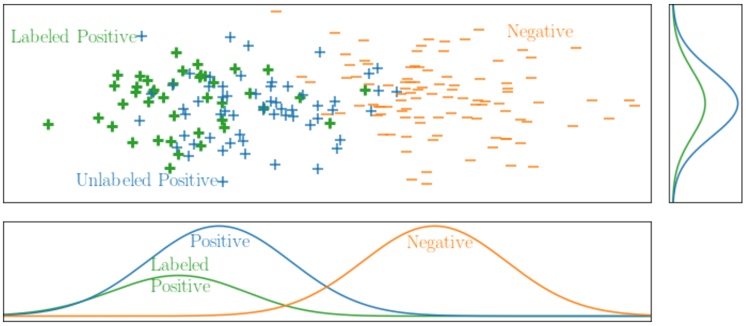

Figure 2.2: **Example of SAR PU and PGPU data.** The labeled examples are a biased sample of the positive examples. The larger the probabilistic gap, the more likely a positive example is selected to be labeled. This means that positive examples which resemble negative examples more, are less likely to be labeled

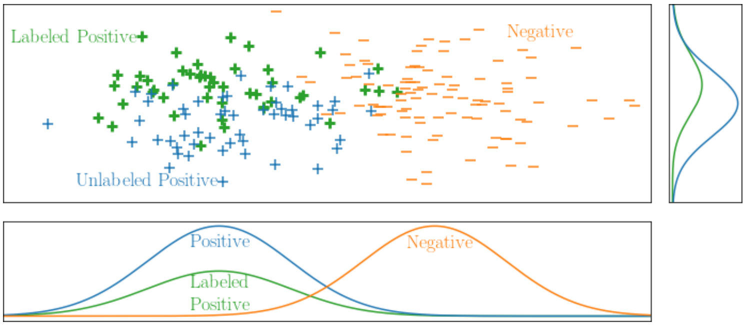

Figure 2.3: **Example of SAR PU data.** The labeled examples are a biased sample of the positive examples. In this case, the labeling mechanism is independent of the probabilistic gap.

Selected At Random

The Selected At Random (SAR) assumption, is the most general assumption about the labeling mechanism: the probability for selecting positive examples to be labeled depends on its attribute values [2]. We propose this assumption in chapter 5 of this dissertation. Figures 2.2 and 2.3 show examples of PU datasets under the SAR assumption.

Definition 2 (Selected At Random (SAR))**.** *Labeled examples are a biased sample from the positive distribution, where the bias completely depends on the* *attributes and is defined by the* propensity score $e(x)$*:*

$$e(x) = \Pr(s = 1|x, y = 1).$$

When the labeling mechanism is understood, incorporating it during the learning phase enables learning an unbiased classifier from SAR PU data. However, when it is not known, additional assumptions are needed to enable learning [2].

Probabilistic Gap

Here, it is assumed that positive examples which resemble negative examples more, are less likely to be labeled. The difficulty of labeling is defined by the *probabilistic gap* $\Delta\Pr(x) = \Pr(y = 1|x) - \Pr(y = 0|x)$ [38]. The labeling mechanism depends on the attribute values x and is therefore a specific case of SAR, which is illustrated in Figure 2.2.

Definition 3 (Probabilistic Gap PU (PGPU))**.** *Labeled examples are a biased sample from the positive distribution, where examples with a smaller probabilistic gap* $\Delta\Pr(x)$ *are less likely to be labeled. The propensity score is a non-negative, monotone decreasing function* f *of the probabilistic gap* $\Delta\Pr(x)$*:*

$$e(x) = f\left(\Delta\Pr(x)\right) = f\left(\Pr(y = 1|x) - \Pr(y = 0|x)\right), \qquad \frac{d}{dt}f(t) < 0.$$

Under the probabilistic gap assumption, the observed probabilistic gap $\Delta\tilde{\Pr}(x) = \Pr(s = 1|x) - \Pr(s = 0|x)$ is related to the real probabilistic gap in two ways: 1) A negative observed probabilistic gap implies a negative real probabilistic gap $\Delta\tilde{\Pr}(x) < 0 \Rightarrow \Delta\Pr(x) < 0$, 2). A larger observed probabilistic gap implies a larger real probabilistic gap $\Delta\tilde{\Pr}(x_1) > \Delta\tilde{\Pr}(x_2) \Rightarrow \Delta\Pr(x_1) > \Delta\Pr(x_2)$. These properties can be used for extracting reliable positive and negative examples [38].

2.2.2 Data Assumptions

The common assumptions about the data distribution are that all unlabeled examples are negative, the classes are separable and the classes have a smooth distribution.

Negativity

The most simple, and most naive, assumption is to assume that the unlabeled examples all belong to the negative class. Despite the fact that this assumption

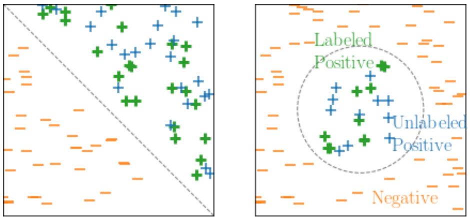

Figure 2.4: **Examples of separable classes.** The first example is linearly separable by a function $f(x_0, x_1) = x_0 + x_1$. The second example is separable by a circle, i.e., by a function $f(x_0, x_1) = -\sqrt{x_0^2 + x_1^2}$.

obviously does not hold, it is often used in practice. In the context of knowledge bases, this assumption is commonly referred to as the *closed-world assumption*. The reason why this assumption is popular is because it enables the use of standard machine learning methods for supervised binary classification [85]. This assumption is simply cited for completeness, and is ignored for the remainder of this survey.

Separability

Under the separability assumption, it is assumed that the two classes of interest are naturally separated. This means that a classifier exists that can perfectly distinguish positive from negative examples. Figure 2.4 shows some examples of separable classes.

Definition 4 (Separability)**.** *There exists a function f in the considered hypothesis space that maps all the positive examples to a value that is higher or equal to a threshold τ and all negative examples to a value that is lower than threshold τ:*

$$f(x_i) \geq \tau, \quad y_i = 1$$

$$f(x_i) < \tau, \quad y_i = 0.$$

Under this assumption, the optimal classifier can be found by looking for the classifier that classifies all labeled examples as positive and as few as possible examples as negative [72, 6]. This idea is exploited by the two-step techniques (Section 2.4.1).

Smoothness

According to the smoothness assumption, examples that are close to each other are more likely to have the same label.

Definition 5 (Smoothness)**.** *If two instances x_1 and x_2 are similar, then the probabilities* $\Pr(y = 1|x_1)$ *and* $\Pr(y = 1|x_2)$ *will also be similar.*

This assumption allows identifying reliable negative examples as those that are far from all the labeled examples. This can be done by using different similarity (or distance) measures such as tf-idf for text [63] or DILCA for categorical attributes [45]. This assumption is important for two-step techniques (Section 2.4.1). It is also used for graph-based approaches [89, 124], local learning [52] and to cluster the data into super-instances where all the instances are assumed to have the same label [67].

2.2.3 Assumptions for an Identifiable Class Prior

The class prior $\alpha = \Pr(y = 1)$ can be an important tool for PU learning under the SCAR assumption. Therefore, it would be useful if it could be estimated directly from PU data. Unfortunately, this is an ill-defined problem because it is not identifiable: the absence of a label can be explained by either a small prior probability for the positive class or a low label frequency [99]. In order for the class prior to be identifiable, additional assumption are necessary. This section gives an overview on possible assumptions, listed from strongest to strictly weaker.

1. *Separable Classes/Non-overlapping distributions* Here, the positive and negative distributions are assumed not to overlap [30, 28, 88]. The positive examples in the unlabeled data are then all those that are likely to be generated by the same distribution as the labeled examples. When all the unlabeled positive examples are identified, class prior estimation becomes trivial.

2. *Positive subdomain/anchor set* Instead of requiring no overlap between the distributions, it suffices to require a subset of the instance space defined by partial attribute assignment (called the anchor set), to be purely positive [99, 24, 3]. The ratio of labeled examples in this subdomain is equal to the label frequency, while in other parts of the positive distribution, the ratio can be lower.

3. *Positive function/separability* This is a more general version of the positive subdomain assumption, where the subdomain can be defined by any

function instead of being limited to partial variable assignments [94]. When this assumption was introduced, it was named 'separability', which we find confusing and thus recommend the more intuitive name 'positive function'.

4. *Irreducibility* The negative distribution cannot be a mixture that contains the positive distribution [6, 48]. All the previous assumption imply irreducibility.

2.3 PU Measures

It is non-obvious how to compute most standard evaluation metrics, such as accuracy, F_1 score, mean square error, etc. from positive and unlabeled data. This introduces challenges both in terms of model evaluation and hyperparameter tuning. The first attempts for addressing this issue focused on proposing metrics that could be computed based on the total number of examples and the number of positive examples. More recent work has explored hypothesis testing and situations where it may be possible to compute standard metrics.

2.3.1 Metrics for PU Data

The most commonly used metric for tuning using PU data is based on the F_1 score, which is defined as:

$$F_1(\hat{\mathbf{y}}) = \frac{2pr}{p+r},$$

with precision $p = \Pr(\mathbf{y}=1|\hat{\mathbf{y}}=1)$ and recall $r = \Pr(\hat{\mathbf{y}}=1|\mathbf{y}=1)$. Under the SCAR assumption, the recall can be estimated from PU data: $r = \Pr(\hat{\mathbf{y}}=1|\mathbf{s}=1)$, however, the precision cannot. The F_1 score cannot be estimated directly from the PU data, but something similar can be. Note that the F_1 score is high when both precision and recall are high. The following performance criterion has the same property and can be estimated from PU data [61]:

$$\begin{aligned}\frac{pr}{\Pr(\mathbf{y}=1)} &= \frac{pr^2}{r\Pr(\mathbf{y}=1)}\\ &= \frac{\Pr(\mathbf{y}=1|\hat{\mathbf{y}}=1)r^2}{\Pr(\hat{\mathbf{y}}=1,\mathbf{y}=1)}\\ &= \frac{r^2}{\Pr(\hat{\mathbf{y}}=1)}.\end{aligned} \tag{2.8}$$

2.3.2 Hypothesis Testing

The G-test is and independence test based on mutual information that can be used for structure learning or feature selection. It turns out that the result of observing independence with the G-test is the same from supervised and PU data. However, the power of the test differs with a constant correction factor $\frac{1-\alpha}{\alpha}\frac{\Pr(s=0)}{1-\Pr(s=0)}$. Because the correction factor is a constant that depends on the amount of labeled data, one can calculate how much more data is required to get the desired power [103]. The conditional test of independence, which was used for learning the PTAN trees, has similar properties [12, 101]. For feature selection, one is interested in ranking the features in order of mutual information between the features and the label. Interestingly, this order remains the same when the unlabeled examples are considered as negative [102].

2.3.3 Computing Standard Evaluation Metrics

More recently, it has been shown that under certain conditions it is possible to compute (bounds on) traditional metrics used to evaluate learned models [18, 49]. Effectively, making the SCAR assumption leads to two important insights. First, by estimating the label frequency or class prior, it is possible to compute the expected number of positive examples in the unlabeled data. Second, the rank distributions of the observed positives and the positive examples contained within the unlabeled data should be similar. Combining these two pieces of information enables reasoning about the total number of positive examples (i.e., the sum of the observed positives and the expected number of positives in the unlabeled data) below (above) a given rank. This is precisely the information needed to construct contingency tables, which can be used to derive standard machine learning metrics such as accuracy, the true positive rate, the false positive rate, and precision. Hence, it is possible in this circumstance to report estimates of these metrics.

2.4 PU Learning Methods

This section provides an overview of the methods that address PU learning. Most methods can be divided into the following three categories: Two-step techniques, biased learning and class prior incorporation. The two-step technique consists of two steps: 1) identifying reliable negative examples, and 2) learning based on the labeled positives and reliable negatives. Biased learning considers PU data as fully labeled data with class label noise for the negative class. Class prior incorporation modifies standard learning methods by applying the mathematics from the SCAR assumption directly, using the provided class prior. Additionally, methods for learning from relational PU data are discussed.

2.4.1 Two-Step Techniques

The two-step technique builds on the assumptions of separability and smoothness. Because of this combination, it is assumed that all the positive examples are similar to the labeled examples and that the negative examples are very different from them. Based on this idea, the two-step technique consists of the following steps [71]:

Step 1 Identify reliable negative examples. Optionally, additional positive examples can also be generated [33].

Step 2 Use (semi-)supervised learning techniques with the positively labeled examples, reliable negatives, and, optionally, the remaining unlabeled examples.

Step 3 (when applicable) Select the best classifier generated in step 2.

Several methods exist for each one of the steps, which are discussed in the following paragraphs. Despite the possibility of choosing the method freely per step [71], most papers propose a fixed combination of methods, which are listed in Table 2.4.

Step 1: Identifying Reliable Negatives (and Positives) In the first step, unlabeled examples that are very different from the positive examples are selected as reliable negatives. Many methods have been proposed to address this problem. They differ from each other in the way distance is defined and when something is considered as different enough. Many two-step papers addressed text classification problems, therefore, many distance measures originate from that domain [72, 63, 122, 65, 33, 64, 66, 76, 73]. The following methods have been proposed to identify reliable negative and possibly positive examples:

Spy Some of the labeled examples are turned into spies by adding them to the unlabeled dataset. Then, a Naive Bayes classifier is trained, considering the unlabeled examples as negative, and updated once using expectation maximization. The reliable negative examples are all the unlabeled negative examples for which the posterior probability is lower than the posterior probability of any of the spies [72]. For this method, it

Table 2.4: **Two-step techniques.** Despite the possibility of choosing the method freely per step, the following combinations where proposed in the literature. Variations of methods are indicated with $*$.

Method	Step 1	Step 2	Step 3
S-EM [72]	Spy	EM NB	ΔE
Roc-SVM [63]	Rocchio	Iterative SVM	$FNR > 5\%$
Roc-Clu-SVM [63]	Rocchio*	Iterative SVM	$FNR > 5\%$
PEBL [123, 122]	1-DNF	Iterative SVM	Last
A-EM [65]	Augmented Negatives	EM NB	ΔF
LGN [64]	Single Negative	BN	/
PE_PUC [124]	PE	(EM) NB	Unspecified
WVC/PSOC[90]	1-DNF*	Iterative SVM	Vote
CR-SVM [66]	Rocchio*	SVM	/
MCLS [16]	k-means	Iterative LS-SVM	Last
C-CRNE [73]	C-CRNE	TFIPNDF	/
Pulce [45]	DILCA	DILCA-KNN	/
PGPU [38]	PGPU	biased SVM	/

is important to have enough labeled examples, otherwise the set of spies is too small and hence unreliable.

1-DNF First, strong positive features are learned by searching for features that occur more often in the positive data than in the unlabeled data. The reliable negative examples are the examples that do not have any strong positive features [122]. Because the requirements for positive features are so weak, there might be too many, resulting in very few reliable negative examples. To resolve this, 1-DNFII proposes to discard positive features with an absolute frequency above some threshold [90].

Rocchio Based on Rocchio classification, this methods builds a prototype for both the labeled and the unlabeled examples. The prototype is the weighted difference of the mean vector of the tf-idf feature vectors of the objective class and the mean vector of the tf-idf feature vectors of the other class. The unlabeled examples that are closer to the unlabeled prototype than the positive prototype are chosen to be the reliable negatives [63]. In addition to Rocchio, k-means clustering can be applied to be more selective: every reliable negative that is closer to a positive prototype than a negative one is removed in this step [63]. Another modification with the aim of being more selective only uses potential unlabeled examples, selected using the cosine similarity, for the negative prototype [66]. Yet another modification is to combine Rocchio with k-means to extract also reliable positive examples in addition to more reliable negatives [76].

PNLH The Positive examples and Negative examples Labeling Heuristic (PNLH) aims to extract both reliable negative and positive examples. First, reliable negatives are extracted using features that more frequently occur in positive data. Subsequently, the sets of reliable positives and negatives are iteratively enlarged by clustering the reliable negatives. Examples that are close to the positive cluster and to no negative cluster are added to the reliable positives. Examples that are close to a negative cluster and not to the positive one are added to the reliable negatives [33].

PE Positive Enlargement aims to extract reliable negative and positive examples. A graph-based semi-supervised learning method is used to extract reliable positives and Naive Bayes for reliable negatives [130].

PGPU Under the probabilistic gap assumption (see Section 2.2.1), all examples with a negative observed probabilistic gap can confidently be considered as negative, and all examples with an observed probabilistic gap that is larger than the probabilistic gap of any observed positive example can confidently be considered as positive [38].

k-means All the examples are clustered using k-means. Reliable negative examples are selected from the negative clusters as the furthest ones from the positive examples [16].

kNN The unlabeled examples are ranked according to their distance to the k nearest positive examples. The unlabeled examples at the greatest distance are selected as reliable negatives [126].

C-CRNE Clustering-based method for Collecting Reliable Negative Examples (C-CRNE) is a method that clusters all the examples and takes the clusters without any positive examples as the reliable negatives [73].

DILCA Reliable negatives are selected based on a trainable distance measure DIstance Learning for Categorical Attributes (DILCA), which is designed specifically for categorical attributes [46]. This distance measure is learned from the positive examples and then used to detect reliable negatives as the furthest examples.

GPU Generative Positive-Unlabeled (GPU) learns a generative model for the positive distribution, based on the labeled set of positives. The reliable negatives are the unlabeled examples with the lowest probability of being generated by the generative model. The number of reliable negatives is set to be equal to the number of labeled positives [1].

Augmented Negatives Instead of selecting reliable negative examples, the unlabeled set is enriched with new examples that are most likely negative.

All the unlabeled and added examples are then initialized as negative [65]. This method is intended for the one-class classification setting where the distribution of negative examples can be different at test time.

Single Negative This method generates a single artificial negative example. This method is intended for an outlier detection setting where very few negative examples are expected in the unlabeled data [64].

Step 2: (Semi-)Supervised Learning In the second step, the labeled positive examples and reliable negatives are used to train a classifier. Any supervised method, like support vector machines (SVM) or Naive Bayes (NB), can be used for this. Semi-supervised methods, like Expectation Maximization on top of Naive Bayes (EM NB), can also incorporate the remaining unlabeled examples. If semi-supervised methods are used, some methods use the extracted reliable examples from the first step as an initialization that can be changed during the learning process [72, 65, 16], while others fix them and only consider the remaining unlabeled examples for possibly belonging to both classes [63, 122]. Apart from existing methods, a few custom methods for PU learning have been proposed:

Iterative SVM In each iteration, an SVM classifier is trained using the positive examples and the reliable negatives. The unlabeled examples that are classified as negative by this classifier are then added to the set of reliable negatives for the next iteration [121].

Iterative LS-SVM In each iteration, a non-linear least Squares SVM (LS-SVM) [114] classifier is trained. During the first iteration, the positive and negative examples come from the initialization. In the later iterations, they come from the classification of the previous iteration. In every iteration, the bias is determined by the desired class ratio [16].

DILCA-KNN For both the positive and reliable negative examples, a DILCA distance measure is trained [46]. For each example, the k nearest positives and k nearest reliable negatives are selected and the average distance to those are calculated with the appropriate distance measure. The class is the one for which it has the lowest average distance [45].

TFIPNDF Term Frequency Inverse Positive-Negative Document Frequency is a tf-idf-improved method that weights the terms in documents according to their appearance in positive and negative documents [73].

Step 3 (Optional): Classifier selection Expectation Maximization (EM) generates a new model during every iteration. The local maximum to which

EM converges might not be the best model in the sequence. Therefore, different techniques have been proposed to select a model from the sequence:

ΔE The chosen model is the one from the last iteration where the estimated change in the probability of error $\Delta E = \Pr(\hat{y}_i \neq y) - \Pr(\hat{y}_{i-1} \neq y)$ is negative, i.e., the last iteration where the model improved [72].

ΔF The chosen model is the one from the last iteration where the estimated change in the F_1 score $\Delta F = F_i/F_{i-1}$ is larger than 1, i.e., the last iteration where the model improved [65].

FNR > 5% Stops iterating if more than 5% of the labeled positive examples are classified as negative [63].

Vote All the intermediate classifiers are used and their results are combined through weighted voting. The optimal weights can be found through Particle Swarm Optimization (PSO) [90].

Last The selected model is the one from the last iteration, when the model has converged or the maximum number of iterations was reached.

2.4.2 Biased Learning

Biased PU learning methods treat the unlabeled examples as negatives examples with class label noise, therefore, this section refers to unlabeled examples as negative. Because the noise for negative examples is a constant, this setting makes the SCAR assumption. The noise is taken into account by, for example, placing higher penalties on misclassified positive examples or tuning hyperparameters based on an evaluation metric that is suitable for PU data. Usually the misclassification penalties or other hyperparameters are chosen through tuning using Equation 2.8 [71, 20, 128, 105] or another measure [106]. Alternatively, they are set based on the true class prior [44] or so that a balanced classifier is preferred [82, 61]. This approach has been applied to classification, clustering and matrix completion.

Classification

A large fraction of the biased learning methods are based on *support vector machine (SVM)* methods. The original one is biased SVM which is a standard

SVM method that penalizes misclassified positive and negative examples differently [71]. As an extension, multiple iterations of biased SVM can be executed where misclassified confident unlabeled examples receive an extra penalty [54]. Weighted unlabeled samples SVM (WUS-SVM) assigns a weight to each unlabeled example, on top of the class penalty, that indicates how likely this examples is to be negative. The weight is the minimum distance to a positive example [75].

The noisiness of the negative data makes the learning harder: too much importance might be given to a negative example that is actually positive [100]. This problem has been addressed by using bagging techniques or using least-square SVMs (LS-SVM) [114]. Bagging SVM learns multiple biased SVM classifiers which are trained on the positive examples and a subset of the negative examples [82]. Robust Ensemble SVM (RESVM) builds on bagging SVMs by also resampling the positive examples and using a bootstrap approach. Biased least squares SVM (BLSSVM) is a biased version of LS-SVM, which, additionally, enables local learning by using an extra regularization term that favors close-by examples having the same label, using the smoothness assumption [52]. BLSSVM has been extended to MD-BLSSVM by using the Mahalanobis [77] distance instead of the Euclidean distance [53].

RankSVM (RSVM) is an SVM method that minimizes a regularized margin-based pairwise loss [105]. In this method, the two classes do not get a different penalty, but the regularization parameter and threshold for classification are set by tuning on Equation 2.8. Other hyperplane optimization methods are Biased Twin SVMs [117], nonparallel support vector vector machines (NPSVM) [128], and the Laplacian Unit-Hyperplane classifier (LUHC) [106].

Weighted *logistic regression* favors correct positive classification over correct negative classification by giving larger weights to positive examples [61]. The positive examples are weighted by the negative class prior $\Pr(s = 0)$ and the negative examples by the positive class prior $\Pr(s = 1)$. They show that as a result, the conditional probability that a positive example belongs to the positive class is larger than 0.5 while a negative example will have a conditional probability smaller than 0.5. In principle, a correct classifier would thus be learned. However, when the classes are not separable, the overlapping parts of the instance space might be attributed to the wrong class. This is because the weighting is equivalent to setting the target probability threshold for the non-traditional classifier to $c\Pr(y = 1)$, while it should be $0.5c$ [29]. Separable classes can handle this by having $0, 1$ probabilities, but non-separable classes are only correctly classified if they are balanced. This is discussed in more detail in Section 2.4.3.

Clustering

Topic-Sensitive pLSA (probabilistic latent semantic analysis) is a weighted *constraint clustering* method that introduces must-link constraints between pairs of positive examples and cannot-link constraints between examples from different classes [132]. The must-link constraints have stronger weights than the cannot-link constraints. This method is expected to work well when the number of labeled positive examples is small.

Matrix Completion

Binary *matrix completion* can also be seen as a PU learning problem: the ones in the matrix are the known positives and the zeros are unlabeled [44]. They assume that in reality, there is a probability matrix of the same size which generated the complete binary matrix. Two binary matrix generation settings are considered: 1) The non-deterministic setting where the complete binary matrix was generated by sampling from the probability matrix, and 2) The deterministic setting where the complete binary matrix was generated by thresholding the probability matrix. The observed matrix is generated by uniform sampling from the complete binary matrix.

In the non-deterministic setting, it is possible to recover the probability matrix, if the true class prior is known. To this end, Shifted Matrix Completion (ShiftMC) minimizes an unbiased estimator for the mean square error loss. This is a special case of the general empirical-risk-minimization based method for incorporating the class prior by preprocessing the data (see Section 2.4.3).

In the deterministic setting, the probability matrix cannot be recovered, but the complete binary matrix can. To this end, the matrix factorization method Biased Matrix Completion (BiasMC) penalizes misclassified positives more than misclassified negatives. The penalties are derived from the class prior. Section 2.4.3 shows how this is a special case of the rebalancing method for incorporating the class prior by preprocessing the data. An extension to BiasMC for graphs uses the additional information that neighbors are likely similar [84].

2.4.3 Incorporation of the Class Prior

Under the SCAR assumption, the class prior can be used. There are three categories of methods: postprocessing, preprocessing and method modification. Postprocessing trains a non-traditional probabilistic classifier by considering the unlabeled data as negative and modifies the output probabilities, preprocessing changes the dataset by using the class prior, and method modification modifies the methods to incorporate the class prior.

Remember from Section 2.1.5 that knowing the class prior is equivalent to knowing the label frequency c, which is the proportion of labeled positive examples $c = \Pr(s = 1)/\alpha$. The class prior can be determined using methods discussed in Section 2.5 or it can be tuned using evaluation metrics for PU data, which are discussed in Section 2.3.

Under the SAR assumption, in a similar fashion, the propensity score can be incorporated to enable learning. Currently, this has only been explored for the empirical-risk-minimization-based preprocessing method.

Postprocessing

The probability of an example being labeled is directly proportional to the probability of that example being positive, with the label frequency c as the proportionality constant:

$$\Pr(s = 1|x) = c \Pr(y = 1|x).$$

From this result, it follows directly that a non-traditional probabilistic classifier that is trained to predict $\Pr(s = 1|x)$ by considering the unlabeled data as negative can be used to predict the class probabilities $\Pr(y = 1|x) = \frac{1}{c} \Pr(s = 1|x)$ [30]. Alternatively, when the probabilities are of no importance, the non-traditional classifier can be used directly by changing the target probability threshold τ to $\tau^{PU} = c\tau$. The commonly used $\tau = 0.5$ then results in the decision function $\Pr(s = 1) > 0.5c$. This is equivalent to the decision function $\operatorname{sgn}(\Pr(y = 1|x) - \Pr(y = 0|x)) = \operatorname{sgn}(\frac{2-c}{c} \Pr(s = 1|x) - \Pr(s = 0|x))$ from Zhang and Lee (2005) [127].

Preprocessing

The goal of preprocessing, is to create a new dataset from a PU dataset, which can be used by methods that expect fully supervised data to train the best possible model for the PU data. The proposed methods can be ordered into three categories: rebalancing methods, methods that incorporate the label probabilities and, empirical-risk-minimization-based methods.

Rebalancing Methods As seen before, a non-traditional classifier, trained on the positive and unlabeled data, gives the same classification as a traditional classifier, if the target probability threshold τ is set appropriately. Instead of changing the threshold, the rebalancing method from Elkan (2001) can be employed to weight the data so that the classifier trained on the weighted data will give the same classification with the same target probability threshold as the traditional classifier [29]. Given the target probability threshold for the traditional classifier τ, the target probability threshold for the non-traditional classifier would be $\tau^{PU} = c\tau$. To move the target probability from τ to τ^{PU} in the non-traditional classifier, the data needs to be weighted as follows:

$$\begin{aligned} w^+ &= \tau(1 - \tau^{PU}) & w^- &= (1 - \tau)\tau^{PU} \\ &= \tau(1 - c\tau) & &= (1 - \tau)c\tau \\ &= (1 - c\tau) & &= (1 - \tau)c, \end{aligned}$$

where w^+ and w^- are the weights for positive and negative examples respectively. In the last step, both weights were divided by τ to simplify the formula as this does not affect the learning result. When the target probability is $\tau = 0.5$, this reduces to

$$w^+ = 1 - c/2 \qquad\qquad w^- = c/2,$$

which is equivalent to the result used for BiasMC [44]. If the true class prior is $\alpha = 0.5$, the result reduces to

$$\begin{aligned} w^+ &= 1 - c\alpha & w^- &= c\alpha \\ w^+ &= Pr(s = 0) & w^- &= \Pr(s = 1) \end{aligned}$$

which are the weights used for weighted logistic regression [61].

Rank Pruning was proposed to be more robust to noise. To this end, it first cleans the data based on the class prior and the expected positive label noise (both of which are estimated in a first phase, see Section 2.5), with the goal of only keeping confident positive and negative examples. The confident examples are then weighted to get the correct class prior [88].

Rebalancing methods are only appropriate when one is interested in classification on the given target threshold τ, but not for returning the unbiased estimates of the probability $\Pr(y = 1|x)$.

Incorporation of the Label Probabilities Elkan and Noto (2008) proposed to duplicate the unlabeled examples to let them count partially as positive and partially as negative. The weights are the probabilities of the unlabeled examples being positive and negative respectively [30]. The labeled examples are certain to be positive and are therefore added as positive examples with weight 1. The probability for an unlabeled example to be positive is

$$\Pr(y=1|s=0,x)=\frac{1-c}{c}\frac{\Pr(s=1|x)}{1-\Pr(s=1|x)}.$$

To generate the weighted dataset like this, first a non-traditional classifier to predict $\Pr(s=1|x)$ needs to be trained.

Empirical-Risk-Minimization Based Methods The goal of preprocessing the PU data is that the classifier learned from the resulting dataset is expected to be equal to the classifier trained from a fully labeled dataset. In an empirical risk minimization framework, this means finding the classifier g that minimizes the risk, given some loss function L

$$R(g)=\alpha\mathbb{E}_{f_+}\left[L^+(g(x))\right]+(1-\alpha)\mathbb{E}_{f_-}\left[L^-(g(x))\right],$$

where $L^+(\hat{y})$ and $L^-(\hat{y})$ are the losses for positive and negative examples respectively. The following are some popular loss functions:

$$\begin{aligned}
\text{MAE}: &\quad L^+(\hat{y})=1-\hat{y} &\quad L^-(\hat{y})=\hat{y},\\
\text{MSE}: &\quad L^+(\hat{y})=(1-\hat{y})^2 &\quad L^-(\hat{y})=\hat{y}^2\\
\text{Log Loss}: &\quad L^+(\hat{y})=-\ln\hat{y} &\quad L^-(\hat{y})=-\ln(1-\hat{y}).
\end{aligned}$$

Empirical-Risk-Minimization based-methods, such as SVMs, logistic regression and deep networks, minimize the empirical risk, which is calculated from the data as follows:

$$\begin{aligned}
\hat{R}(g|\mathbf{x},\mathbf{y}) &= \alpha\frac{1}{|\mathbf{y}=\mathbf{1}|}\sum_{x:\mathbf{x}|\mathbf{y}=\mathbf{1}}L^+(g(x))+(1-\alpha)\frac{1}{|\mathbf{y}=\mathbf{0}|}\sum_{x:\mathbf{x}|\mathbf{y}=\mathbf{0}}L^-(g(x))\\
&= \frac{1}{|\mathbf{y}|}\left(\sum_{x:\mathbf{x}|\mathbf{y}=\mathbf{1}}L^+(g(x))+\sum_{x:\mathbf{x}|\mathbf{y}=\mathbf{0}}L^-(g(x))\right). \qquad (2.9)
\end{aligned}$$

In PU data, the empirical risk cannot be calculated directly because not all the class values are observed. However, the PU data and the labeling mechanism can be used to create a new, weighted dataset that is expected to give the same empirical risk as the fully labeled data. Next, the risk is rewritten in terms of expectations over the labeled and unlabeled distributions. Then, it is shown how to create the data which gives the same empirical risk when using the standard formula 2.9 which is used by standard methods and implementations.

The expectation over the negative distribution can be formulated in terms of expectations over the general and the positive distributions, using Equation 2.1. The expectation over the positive distribution can be formulated in terms of an expectation over the labeled distribution and the propensity score, using Equation 2.2:

$$\begin{aligned}
R(g) &= \alpha \mathbb{E}_{f_+}\left[L^+(g(x))\right] + (1-\alpha)\mathbb{E}_{f_-}\left[L^-(g(x))\right] \\
&= \alpha \mathbb{E}_{f_+}\left[L^+(g(x))\right] + \mathbb{E}_{f}\left[L^-(g(x))\right] - \alpha \mathbb{E}_{f_+}\left[L^-(g(x))\right] \\
&= \alpha \mathbb{E}_{f_+}\left[L^+(g(x)) - L^-(g(x))\right] + \mathbb{E}_{f}\left[L^-(g(x))\right] \\
&= \alpha \mathbb{E}_{f_l}\left[\frac{c}{e(x)}\left(L^+(g(x)) - L^-(g(x))\right)\right] + \mathbb{E}_{f}\left[L^-(g(x))\right].
\end{aligned}$$

In the case-control scenario, the expectation over the general distribution can simply be replaced by the expectation over the unlabeled distribution. Therefore, the empirical risk is calculated as follows:

$$\begin{aligned}
\hat{R}(g|\mathbf{x},\mathbf{s}) = \frac{\alpha}{|\mathbf{s}=\mathbf{1}|} \sum_{x:\mathbf{x}|\mathbf{s}=\mathbf{1}} \left(\frac{c}{e(x)}\left(L^+(g(x)) - L^-(g(x))\right)\right) \\
+ \frac{1}{|\mathbf{s}=\mathbf{0}|} \sum_{x:\mathbf{x}|\mathbf{s}=\mathbf{0}} L^-(g(x)). \qquad \text{\# \textit{case-control}}
\end{aligned}$$

Hence, the new dataset is created by adding all unlabeled examples as negative with weight $\frac{1}{|\mathbf{s}=\mathbf{0}|}$, and all labeled examples both as positive with weight $\frac{1}{|\mathbf{s}=\mathbf{1}|}\frac{\alpha c}{e(x)}$ and as negative with weight $-\frac{1}{|\mathbf{s}=\mathbf{1}|}\frac{\alpha c}{e(x)}$.

For the single-training-test scenario, the general distribution is a combination of the labeled and unlabeled distributions (Equation 2.3), which reduces the risk to:

$$\begin{aligned}
R(g) = \alpha c \mathbb{E}_{f_l}\left[\frac{1}{e(x)} L^+(g(x)) + \left(1 - \frac{1}{e(x)}\right) L^-(g(x))\right] \\
+ (1-\alpha c)\mathbb{E}_{f_u}\left[L^-(g(x))\right]. \qquad \text{\# \textit{single-training-set}}
\end{aligned}$$

And the empirical risk to:

$$
\begin{aligned}
\hat{R}(g|\mathbf{x},\mathbf{s}) &= \frac{\alpha c}{|\mathbf{s}=\mathbf{1}|} \sum_{x:\mathbf{x}|\mathbf{s}=\mathbf{1}} \left(\frac{1}{e(x)} L^{+}(g(x)) + \left(1 - \frac{1}{e(x)}\right) L^{-}(g(x)) \right) \\
&\quad + \frac{1-\alpha c}{|\mathbf{s}=\mathbf{0}|} \sum_{x:\mathbf{x}|\mathbf{s}=\mathbf{0}} \left(L^{-}(g(x)) \right) \\
&= \frac{1}{|\mathbf{s}|} \left(\sum_{x:\mathbf{x}|\mathbf{s}=\mathbf{1}} \left(\frac{1}{e(x)} L^{+}(g(x)) + \left(1 - \frac{1}{e(x)}\right) L^{-}(g(x)) \right) \right. \\
&\quad \left. + \sum_{x:\mathbf{x}|\mathbf{s}=\mathbf{0}} \left(L^{-}(g(x)) \right) \right). \qquad \#\ \textit{single-training-set}
\end{aligned}
$$

Hence, the new dataset is created by adding all unlabeled examples as negative with weight 1 and all labeled examples both as positive with weight $\frac{1}{e(x)}$ and as negative with weight $(1 - \frac{1}{e(x)})$.

This general weighting method was proposed in the single-training-set scenario as the first SAR PU learning method [2] (in chapter 5 of this thesis) but it already existed before under the SCAR assumption [111, 25, 57]. The ShiftMC method for matrix completion is also a special case of this method under the SCAR assumption, using the MSE loss [44].

du Plessis et al. (2014) proposed another risk estimator, which simply reweights the examples and does not introduce duplicates [26]. However,the derivation is limited to 0-1 predictions and the method is biased, unless the loss functions sum to one $L^{+}(\hat{y}) + L^{-}(\hat{y}) = 1$, which can only be achieved with non-convex functions.

Method Modification

Many machine learning methods are based on counts of positive and negative examples in subsets of the data. The counts are used to calculate (conditional) probabilities, support, coverage or other metrics that are used to make decisions or set parameters. The counts can be estimated using the same rationale as were used for data weighting [30].

The PU tree learning algorithm POSC4.5, one of the first PU learning methods, needs the count of positive and negative examples in every considered split for the three. They estimate the number of positives in node i as $\hat{P}_i = \min\{\frac{1}{c}L_i, T_i\}$

and the negatives as $\hat{N}_i = T_i - \hat{P}_i$, where L_i and T_i are the number of labeled and total examples in that node [22]. This corresponds to empirical-risk-minimization-based weighing.

Ward et al. (2009) proposed an expectation maximization method on top of logistic regression [115]. The expectation step finds the expected class labels and the maximization step trains the logistic regression model using the expected class labels, followed by rebalancing the model according using the class prior.

For Naive Bayes methods, the probabilities $\Pr(x^{(i)}|y)$, with $x^{(i)}$ the ith attribute of x, are key. For $y = 1$, these can be directly estimated from the labeled data as

$$\Pr(x^{(i)}|y = 1) = \Pr(x^{(i)}|s = 1), \tag{2.10}$$

and for $y = 0$ these can be calculated, somewhat less straightforwardly, as follows:

$$\Pr(x^{(i)}|y = 0) = \frac{\Pr(x^{(i)}) - \alpha \Pr(x^{(i)}|y = 1)}{1 - \alpha}. \tag{2.11}$$

This insight was used to develop PNB, the first Naive Bayes algorithm for PU learning [23]. It was originally proposed for document classification, but was later generalized to general discrete attributes and incorporate the of Laplace correction [12]. In that same paper an averaging method is presented that can incorporate a distribution over the class prior instead of an exact value. Positive Tree Augmented Naive Bayes (PTAN) builds further on PNB, but also needs to calculate the conditional mutual information between variables i and k for structure learning:

$$\sum_j \sum_l \Pr(x^{(i)} = j, x^{(k)} = l, y = 1) \log \frac{\Pr(x^{(i)} = j, x^{(k)} = l|y = 1)}{\Pr(x^{(i)} = j|y = 1) \Pr(x^{(k)} = l|y = 1)}$$
$$+ \Pr(x^{(i)} = j, x^{(k)} = l, y = 0) \log \frac{\Pr(x^{(i)} = j, x^{(k)} = l|y = 0)}{\Pr(x^{(i)} = j|y = 0) \Pr(x^{(k)} = l|y = 0)},$$

all these probabilities can be calculated by using Equations 2.10, 2.11, and:

$$\Pr(x^{(i)} = j, x^{(k)} = l, y = 1) = \alpha \Pr(x^{(i)} = j, x^{(k)} = l|s = 1)$$
$$\Pr(x^{(i)} = j, x^{(k)} = l, y = 0) = (1 - \alpha) \Pr(x^{(i)} = j, x^{(k)} = l|y = 0).$$

Similarly, PU learning methods have been proposed for other Bayesian classifiers. Averaged One-Dependence Estimator (AODE) [116] has been extended to PAODE, Hidden Naive Bayes (HNB) [51] to PHNB, and Full Bayesian network

Classifier (FBC) [113] to PFBC [40]. Some of these methods were further extended to uncertain Bayesian methods, where the attribute values are uncertain: UPNB [39] and UPTAN [35], where this last method uses Uncertain Conditional Mutual Information (UCMI) for structure learning [69].

2.4.4 Relational Approaches

A common task for relational data is to complete automatically constructed knowledge bases or networks by finding new relationships. This task can be seen as PU learning, because everything that is already in the knowledge base or network is known to be true and everything that can possibly be added is unlabeled. Most methods make the *closed-world* assumption and learn models by assuming everything that is not in the knowledge base is negative. However, a few methods have been proposed that do make the *open-world* assumption, which makes it explicit that the data is incomplete.

In chapter 4 of this thesis, we show that when the SCAR assumption holds in the relational PU data, then, relational versions of classic class prior incorporation methods can be used to enable learning [5]. TI*c*ER, a relational version of TI*c*E (Section 2.5.3) can estimate the class prior directly from the relational PU data.

The PosOnly setting of the relational rule learning system Aleph [110] makes the separability assumption and looks for the simplest theory that covers all positive examples and introduces as few new facts as possible [83].

RelOCC is a relational one-class classification method which, based on the smoothness assumption, introduces a tree-based distance method [56]. They do not use unlabeled examples at training time, so, although related, it is not truly PU learning.

The AMIE+ rule learning system for knowledge base completion introduces the partial completeness assumption. It assumes that if for a subject and relationship at least one object is known, then all objects for this subject and relationship are known. For example, if `taughtby(bigdata,jesse)`, then it is assumed that the knowledge base contains all Jesse's classes. Using the partial completeness assumption, the confidence of potential rules can be estimated more precisely [34]. The RC confidence score makes an even more precise estimate, by making a rule-specific SCAR assumption and taking the expected relation cardinalities, i.e., the number of objects/subjects per subject/object and rule combination, into account [133].

PULSE, a relational PU learning algorithm for disjunctive concepts was proposed in the context of relational grounded language learning [7]. In their setting, the positive class can have a limited number of k subclasses. They assume that for each subclass, the SCAR assumption holds, but do not necessary have the same label frequencies.

2.4.5 Other Methods

For completeness, this section lists PU methods that do not fit in any of the considered categories.

Generative Adversarial Networks (GANs) have recently been introduced for PU learning, where they can model the positive and negative distributions [43, 17].

Co-training is a semi-supervised learning technique that learns two models, based on two views of the data, where the goal is to find two models that agree [10]. This idea has been applied to PU learning as well [23, 131].

Data stream classification with PU data has been addressed by multiple works [67, 86, 93, 69, 14].

2.5 Class Prior Estimation from PU Data

Knowledge of the class prior significantly simplifies PU learning under the SCAR assumption. Therefore, it is very useful to estimate it from PU data directly. To this end, a number of methods have been proposed.

2.5.1 Non-traditional Classifier

When the classes are separable, in principle a non-traditional classifier $g(x)$ that predicts $\Pr(s=1|x)$ can be trained that maps all negative examples to 0 and all positive examples to $\Pr(s=1|y=1)=c$. Based on this insight, Elkan and Noto (2008) suggest to train a classifier on part of the data while keeping a separate validation set. Then, they estimate the label frequency as the average predicted probability of a labeled validation set example [30]. When the labeled positive examples can be noisy, i.e., some of them are negative, these false positives can ruin the estimation. Rank Pruning handles this by also estimating the positive noise, using the most confident examples [88]. Both these methods require well-calibrated probabilistic classifiers. Methods such as Platt scaling [91], isotonic regression [125] or beta calibration [58] can be used to calibrate classifiers that do not output well-calibrated probabilities.

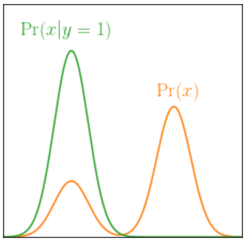

Figure 2.5: **Partial matching.** The goal of partial matching is to find the class prior α that minimizes the divergence between the scaled distributions. This figure is based on Figure 1 in [28].

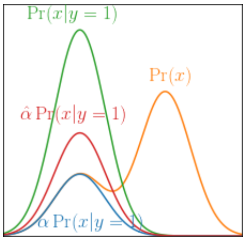

Figure 2.6: **Partial matching with overlap.** When the classes overlap, the original partial mapping method would result in an overestimate for alpha $\hat{\alpha} > \alpha$, like the red line. Using a penalized divergence makes sure that the α-scaled positive distribution does not surpass the total distribution.

2.5.2 Partial Matching

The partial matching approach assumes non-overlapping classes. It uses a density estimation method to estimate the positive distribution, based on the labeled examples, and the complete distribution, based on all the data [28]. The class prior is found by minimizing the difference between the scaled positive distribution, where the scale factor is the class prior. The method is illustrated in Figure 2.5.

The partial matching approach does not work well when the positive and negative distribution overlap. In this case, the correct class prior would give a large divergence in the regions with overlap. By minimizing the divergence, these regions will favor an overestimate of the class prior. To relax the non-overlapping distributions assumption to the positive subdomain assumption, penalized divergences were introduced [24]. These give higher penalties to class priors that result in $\alpha \Pr(x|y=1) > \Pr(x)$ for some x. Intuitively, this finds the class prior that scales the positive distribution as closely to the total distribution, without ever surpassing it. The method is illustrated in Figure 2.6

2.5.3 Decision Tree Induction

In chapter 3 of this dissertation, we propose Tree Induction for c Estimation (TIcE), which estimates the label frequency c under the positive subdomain assumption [3]. It makes the observation that the label frequency remains the same when considering a subdomain of the data and that the fraction of labeled examples in that subdomain provides a natural lower bound on the label frequency. Using a decision tree induction method, it searches for the subdomain that implies the largest lower bound and returns that as the label frequency estimate. Under the positive subdomain assumption, this lower bound is indeed expected to be the label frequency.

2.5.4 Receiver Operating Characteristic (ROC) Approaches

In the ROC setting, one aims to maximize the true positive rate $\text{TPR} = \Pr(\hat{y} = 1|y = 1)$ while minimizing the false positive rate $\text{FPR} = \Pr(\hat{y} = 1|y = 0)$. The TPR can be calculated in PU data, by using the labeled positive set. While the FPR cannot be calculated from PU data, for a given TPR, minimizing the FPR within a hypothesis space $\mathcal{H}$ is equivalent to minimizing the probability of predicting the positive class $Pr(\hat{y} = 1)$:

$$\begin{aligned}
\min_{\hat{y}:\mathcal{H},\text{TPR}} \Pr(\hat{y} = 1) &= \min_{\hat{y}:\mathcal{H},\text{TPR}} \alpha \Pr(\hat{y} = 1|y = 1) + (1-\alpha) \Pr(\hat{y} = 1|y = 0) \\
&= \min_{\hat{y}:\mathcal{H},\text{TPR}} \alpha\text{TPR} + (1-\alpha) \Pr(\hat{y} = 1|y = 0) \\
&= \alpha\text{TPR} + (1-\alpha) \min_{\hat{y}:\mathcal{H},\text{TPR}} \Pr(\hat{y} = 1|y = 0).
\end{aligned}$$

If classifier f exists that minimizes the FPR to zero, then the class prior can be calculated as $\alpha = \Pr(f = 1)/TPR = \Pr(f = 1)/\Pr(f = 1|s = 1)$. In fact, for any classifier f, this is an upper bound:

$$\alpha \geq \frac{\Pr(f = 1)}{\Pr(f = 1|s = 1)}.$$

As a result, maximizing $\Pr(f = 1)/\Pr(f = 1|s = 1)$ over the space of all classifiers gives the class prior [6]. This result is valid under the irreducibility assumption. However, without extra assumptions, infinite examples are required for convergence. The stricter positive subdomain assumption allows for practical algorithms. Scott (2015) implements this idea by building a conditional probability classifier [99]. The same idea is approached from a different angle by Jain et al. (2016) [50, 48]. They use k-kernel density estimation to approximate the positive and total distributions, given different values for the class prior α, in a second step, they select α as the largest value (i.e., minimal $\Pr(\hat{y} = 1)$ and thus minimal FPR) that results in the optimal log likelihood for both densities (i.e., maximal TPR).

2.5.5 Kernel Embeddings

All previous methods, except TI*c*E, aim to model the entire domain with either discriminative or generative models. However, this might be overkill for estimating one constant, especially since the label frequency is equal for every example. Based on this insight, a class prior estimation method using kernel embeddings is proposed that aims to separate part of the positive distribution from the total distribution, under the positive function assumption. This means that they look for functions that map all negative examples to zero. Given a class prior, the minimal proportion from the negative distribution that is selected by any function is estimated. The class prior is the largest value for which that proportion is below a given threshold [94].

2.5.6 Other Sources For the Class Prior

Estimating the class prior from PU data is hard. Therefore, it can be useful to obtain it in another way. For some domains, the class prior can be known from domain knowledge or previous studies. If there is access to a smaller dataset for the same domain that does have both possible and unlabeled labels, these can be used to estimate the class prior from. Or finally, one can just not estimate it but treat is as a hyperparameter and use a validation set and tune for it using a PU evaluation metric from Section 2.3.

2.6 Sources of PU Data and Applications

There are many classification situations where PU data naturally occurs and various machine learning tasks can be phrased as PU learning problems. The following subsection lists some of these situations and tasks. Next, applications that were explicitly addressed as PU learning problems are discussed.

2.6.1 Sources of PU Data

PU data naturally arises in the following settings.

An *automatic diagnosis* system aims to predict if a patient has a disease. The data for such a system would consist of patients that were diagnosed with the disease and patients that were not. However, not being diagnosed is not equal to not having it. Many diseases, like diabetes, often go undiagnosed [19]. Diagnoses patients are thus positive examples, while undiagnosed are unlabeled.

Sometimes, *positive examples are easier to obtain.* Recommendation systems, for example, can use previous purchases or likes as examples for items of interest. Similarly, some spam mails will be tagged as such. Purchased or tagged items are thus positive examples, while the others are unlabeled.

Indirect labels can be used to get some labeled examples. For example, to classify active students based on university records, the students that are registered in university sport classes are active. Other students are unlabeled.

The *case-control* scenario comes from the setting where two datasets are used and one is known to only have positive examples. For example, to predict one's socioeconomic status from her health record, positive examples could be gathered from health centers in upper-class neighborhoods and unlabeled examples from a random selection of health centers.

Negative-class dataset shift occurs when the distribution of the negative examples changes while the positive distribution remains the same. This happens, for example, in adversarial scenarios. In this case it might be easier to obtain a new representative sample from the entire distribution than to label characteristic examples from the new negative distribution [25].

In surveys, *under-reporting* occurs when participants are likely to give false negative responses [104]. This occurs for issues that have social stigma, such as maternal smoking. Research has shown that smoking may be underestimated by up to 47% [37]. In this setting, a negative response is really an unlabeled example.

The goal of *one-class classification* is to recognize examples from the class of interest, i.e., the positive class, from the entire population. When an unlabeled dataset is available that represents the entire population, then this can be seen as learning from positive and unlabeled data [55]. In this case, the negative class often has a large variety, for which it is difficult to label a representative sample [62].

Inlier-based outlier detection has access to a representative sample of inliers, in addition to the standard unsupervised data. With this information, more powerful outlier detection is possible [41, 108]. This task can be phrased as PU learning, with the inliers as the positive class [6].

Automatic *knowledge base completion* is inherently a positive and unlabeled problem. Automatically constructed knowledge bases are necessarily incomplete and only contain true facts [34, 85]. The unlabeled examples are the facts that are considered to be added to the knowledge base.

Identification problems aim to identify examples in an unlabeled dataset that are similar to the provided examples. For example, disease gene identification aims to identify new disease-genes [80].

2.6.2 Applications

PU learning has been applied to a variety of problems.

Disease gene identification aims to identify which genes from the human genome are causative for diseases. Here, all the known disease genes are positive examples, while all other candidates, that can be generated by traditional linkage analysis, genes are unlabeled. To check all of the candidates individually would be very costly. With PU learning, a promising subset can be discovered. Several PU methods were developed to this end: ProDiGe is a method based on bagging SVMs [80, 82], PUDI is also a weighted SVM method, but they have different weights for four identified groups of unlabeled examples: reliable negative, likely positive, likely negative and weakly negative [119], EPU uses multiple biological data sources and trains an ensemble model on those [118].

Protein complexes are a set of interacting proteins for specific biological activities. Such complexes can be predicted as subgraphs from protein-protein interaction networks. Known complexes are positive examples and all other possibilities are unlabeled. This problem has been addressed using a non-traditional classifier approach [30, 129].

A *gene regulatory network* is a set of interacting genes that control cell functions. Using the non-traditional classifier method with SVMs, the relationships between activation profiles of gene pairs can be identified [30, 13]. Bagging SVMs have been employed to identify which genes are under control of which transcription factors [82, 81].

In the field of *drug discovery*, the tasks of *drug repositioning*, which looks for interactions between drugs and diseases, and *drug-drug-interactions* are very important. To find these interactions, a pairwise scoring function can be trained so that known interactions score higher than pairs which are not known to interact [74]. The rationale behind this method is similar to RSVM [105].

Ecological modeling of the habitat of species aims to model where certain animals appear. An observed animal at a certain location provides positive examples. However not observing an animal does not mean that it never comes there. An EM algorithm on top of logistic regression that finds the optimal likelihood model, given the class prior, was proposed to address this application [115].

The goal of *targeted marketing* is to only promote products to potential buyers. The difficulty is to identify these customers. A biased SVM approach has been used to identify heat pump owners based on smart meter data, prior sales and weather data [71, 31]. For online retail, purchase data is often used as positive examples. However, for durable goods, like televisions, only a small fraction of potential customers will purchase it, not because they are not interested, but because already have one or are waiting for the right time, etc. A custom algorithm was developed for this application [120].

Remote sensing data, like satellite pictures, can be used to classify certain areas. While examples can be given for the class of interest, it can be hard to identify negative examples, because those are too diverse to be labeled. A non-traditional classifier can be used in such a context [30, 62].

Local descriptors play an important role in *localization* of, for example, mobile robots from laser scanner data. However, in some natural environment, many of the local descriptors might be unreliable and are better filtered out than used. To this end, the non-traditional random forest can be used, where the unlabeled examples are subsampled in a similar way as for bagging SVMs [30, 82, 11, 60].

Recommender systems can suffer from deceptive reviews, which are dishonest positive or negative reviews. These reviews should therefore be filtered out. Some positive examples of such reviews can be provided, but all other reviews to be checked are unlabeled [95].

Focused web crawlers search for relevant web pages given a query. Such a web crawler chooses to follow a link or not, based on the link's context. It is much easier to provide positive examples of such contexts than to provide a good sample of negative examples. Therefore the WVC and PSOC methods have been used to address this problem [90].

2.7 Related Fields

This section briefly discusses the fields that are closely related to PU learning.

2.7.1 Semi-Supervised Learning

The goal of semi-supervised learning is to learn from labeled and unlabeled data [15]. In contrast to PU learning, labeled examples of all classes are assumed to be present in the data. Also, semi-supervised learning can go beyond binary classification tasks. Although semi-supervised methods cannot be applied directly to PU learning, some approaches have been ported from one domain to the other [23, 89].

For semi-supervised learning methods that incorporate the class prior, it is usually assumed that the class prior can be readily estimated from the labeled data, i.e., that positive and negative examples are selected to be labeled with the same probability. However, recently a matching method has been proposed to estimate the class prior when this is not the case [27].

2.7.2 One-Class Classification

The goal of one-class classification is to learn a model that identifies examples from a certain class: the positive class, when only examples of that class are available [55]. It can be seen as training a binary classifier where the negative class consists of all other possible classes. This is in contrast to PU learning, where the domain of interest is defined by the unlabeled data. Also, the unlabeled data enables finding low-density areas which are likely to be classification boundaries under the separability assumption. Under the SCAR assumption, areas with relatively more unlabeled examples than positive ones indicate a negative region, which would not be clear with only positive examples.

2.7.3 Classification in the Presence of Label Noise

Label noise occurs when some of the class labels in the data are erroneous, i.e., when some examples have a class label that does not correspond with its true class value. A common interpretation of PU learning is that it is the specific type of label noise where the positive examples can be incorrectly labeled as negative. All the biased learning methods are based on this interpretation.

Just like the SCAR assumption was proposed in analogy with the MCAR assumption from missing data, a taxonomy for mislabeling mechanisms was proposed in analogy with the missing data taxonomy [32]:

NCAR *Noisy Completely At Random* Every class label has exactly the same probability to be erroneous, independent of the attribute values of the example or the true class value.

NAR *Noisy At Random* The probability for a class label to be erroneous depends completely on the true class value.

NNAR *Noisy Not At Random* The probability for a class label to be erroneous depends on the attribute values

The SCAR labeling mechanism corresponds to the NAR mislabeling mechanism, where the mislabeling probability for the positive and negative class are $1 - c$ and 0 respectively.

2.7.4 Missing Data

When working with missing data, the missingness mechanism that dictates which values are missing plays a crucial role, just like the labeling mechanism for PU learning. The missingness mechanisms are generally divided into three classes [96, 70]:

MCAR *Missing Completely At Random* Every attribute has exactly the same probability to be missing, independent of the other attribute values of the example and the value of the missing attribute.

MAR *Missing At Random* The probability for an attribute to be missing depends completely on the observable attributes of the example.

MNAR *Missing Not At Random* The probability for an attribute to be missing depends on the value that is missing.

The SCAR and SAR assumptions were introduced in analogy with MCAR and MAR. However, it is important to note that within the missing data taxonomy, SCAR and SAR actually both belong to the MNAR class, because positive and negative class values have a different probabilities to be missing: c or $e(x)$ and 0 respectively. The class values are missing (completely) at random only if just the population of positive examples is considered. Moreno et al. (2012) proposed a new missingness class: *Missing Completely At Random-Class Dependent (MAR-C)*, where per class, the data is MCAR, as is the case for SCAR.

2.7.5 Multiple-Instance Learning

The goal of multiple-instance learning is to train a binary classifier. Instead of positive and negative examples, the learner is provided with bags that are labeled positive if at least one of the examples in the bag is positive and negative otherwise. This setting can be phrased as PU learning, or actually NU learning, as the classes are switched. All the examples in a negative bag are known to be negative and can therefore get a negative label, while examples in a positive bag can be both positive and negative and therefore are considered unlabeled. Following this insight, classifiers from either domains can be used to solve the task of the other domain [68].

2.8 Questions Revisited

At the end of the introduction, we posed seven research questions frequently addressed in PU learning research. To conclude, we will revisit these questions and try to synthesize answers to each one.

How can we formalize the problem of learning from PU data? The PU learning literature always assumes one of two learning scenarios: single-training-set or case-control, which are discussed in Section 2.1. The former assumes one dataset that is an i.i.d. sample of the true distribution. A subset of the positive examples of the dataset are labeled while the remaining examples are unlabeled. The latter scenario assumes two independently drawn datasets: an i.i.d. sample of the true distribution (unlabeled) and a sample of the positive part of the true distribution (positive). The labeled examples are selected from the positive subset or the positive distribution according to the labeling mechanism.

What assumptions are typically made about PU data in order to facilitate the design of learning algorithms? As discussed in Section 2.2, assumptions are needed either about the data distribution, or the labeling mechanism, or both. The most common assumptions about the data distribution are separable classes and smoothness, which form the basis for the two-step learning techniques. The most common labeling mechanism assumption is selected completely at random (SCAR) assumption, where postures that the set of labeled examples is a uniformly random subset of the positive examples. It greatly simplifies learning and it serves as the basis of all class-prior based methods. Recently, the more realistic SAR assumption has been proposed which assumes that the labeling mechanism depends on the attributes.

Can we estimate the class prior from PU data and why is this useful? By making assumptions about the data and/or labeling mechanism it is possible to estimate the label frequency and hence class prior in certain conditions (Section 2.2.3). Multiple different techniques have been proposed for this task (Section 2.5). The power and usefulness of this piece of information is that facilitates the design of algorithms for learning from PU data (Section 2.4.3). This is effectively done by estimating the expected number of positive and negative examples of the data, which can be accomplished by either weighting the data and then applying standard algorithms or directly modifying algorithms to work with fractional counts.

How can we learn a model from PU data? Section 2.4 shows that most PU learning methods belong to one of three categories: two-step techniques, biased learning and class prior incorporation methods. Two-step techniques begin by identifying reliable negative (and sometimes positive) examples and then using the labeled and reliable examples to train a classifier. The biased methods treat the unlabeled examples as belonging to the negative class, but attribute a larger loss to false positives than false negatives. Class prior incorporation methods use the class prior to weight the unlabeled data or modify machine learning algorithms to reason about the expected number of positive and negative examples in the unlabeled data.

How can we evaluate models in a PU setting? This is an area that has perhaps received less attention in the literature. This can be approached in two general ways, both of which exploit the SCAR assumption. One is to use the (estimated) class prior and construction bounds for traditional evaluation metrics such as accuracy. The other is to design metrics that can be computed based on the observed information (e.g., could be computed using only positive examples) which are proxies for standard metrics. This was discussed in Section 2.3

When and why does PU data arise in real-world applications? As outlined in Section 2.6, PU data arises in many different fields. At a high-level, it occurs in the following types of situations:

1. When only "positive" information is recorded such as in an electronic medical record or a knowledge base that stores facts, where the absence of information does not imply something is not true;

2. People have a reason to be deceptive and not report such as lying about smoking when pregnant in a survey or an athlete hiding an injury in order to keep playing;

3. Where it is much easier to identify one class than another, such as certain bioinformatics problems or remote sensing.

How does PU learning relate to other areas of machine learning Section 2.7 shows that PU learning is related to numerous areas of machine learning. Most obviously, it is a special case of standard semi-supervised learning. The key differences are that typically semi-supervised approaches have access to at least some examples of all classes, and that semi-supervised approaches go beyond binary classification tasks. Similarly, it can also be viewed through the prism of learning with label noise. Again, PU learning is a specialization in that corresponds to one type of noise: that where positive examples are possibly incorrectly labeled as negative. Some of the nomenclature about labeling mechanisms has been inspired by the long standing field of working with missing data. Finally, it also tied to one-class classification, learning with missing data and multiple-instance learning.

Chapter 3

Scalable Class Prior Estimation from Positive and Unlabeled Data

When learning from positive and unlabeled data under the common *Selected Completely At Random (SCAR)* assumption, every positive example has the same probability for being selected to be labeled, independent of the values of its attributes. This probability is called the *label frequency* and plays a crucial role when learning with this type of data. If the label frequency, or equivalently the class prior, is known, then learning with SCAR PU data can be reduced to the standard binary classification task. The label frequency can be used to either adapt algorithms to incorporate this information during learning [21, 71, 127, 22, 30] or as a preprocessing step to assign weights to the unlabeled examples [30].

Because the label frequency is often unknown, several methods have proposed in the last decade to estimate it from the positive and unlabeled data [30, 28, 24, 48, 50, 94]. Unfortunately, the aforementioned methods are either inaccurate or not scalable in he number of examples. This chapter proposes a simpler and faster, yet equally accurate method for estimating the label frequency from PU data.

Problem Statement

The problem that this chapter addresses is the following:

Given a PU dataset $(\mathbf{x}, \mathbf{y}, \mathbf{s})$, where the class $\mathbf{y}$ is not observed, none of the negative examples are labeled $\Pr(s = 1|y = 0, x) = 0$, and the labeled examples were selected completely at random from the positive set $\Pr(s = 1|y = 1, x) = \Pr(s = 1|y = 1)$,

Estimate the label frequency $c = \Pr(y = 1|s = 1)$.

Contributions of this Chapter

To improve over current methods for estimating the label frequency from PU data, we made the following four concrete contribution:

The first contribution is a theoretical study of lower bounds on the label frequency that can be derived from the data. The theory is based on the simple intuition that any subset of the data cannot contain more positive examples than the total number of examples in that subset. Additionally, we identify when a lower bound is expected to be close to the real label frequency.

The second contribution is a practical algorithm TIcE that estimates the label frequency by looking for the tightest lower bound in the data, using decision tree methods.

The third contribution is an extensive empirical evaluation of TIcE. To evaluate if the decisions made for the practical algorithm behave as expected by a sensitivity analysis was conducted. From a comparison of TIcE against other label frequency estimation algorithms, we could conclude that TIcE's estimates are equivalently accurate as those of the state of the art methods while being an order of magnitude faster.

The fourth contribution is the availability of our TIcE implementation on `https://dtai.cs.kuleuven.be/software/tice`.

The content of this chapter is based on the following publication [3]:

Bekker, J., and Davis, J. Estimating the Class Prior in Positive and Unlabeled Data through Decision Tree Induction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018, New Orleans, Louisiana, United States; 2-7 February 2018)* (2018a), pp. 2712–2719.

3.1 Bounding and Estimating the Label Frequency Using Tree Induction

In this chapter, we aim to estimate the label frequency by considering subdomains of the attributes based on partial assignments. For ease of derivation, we assume discrete variables. The use of subdomains is possible because of the “selected completely at random” which implies label frequencies being equal in any subdomain A.

$$\Pr(s=1|x\in A,y=1)=\Pr(s=1|y=1)=c \tag{3.1}$$

Proof. This can be derived using the law of total probability and the SCAR assumption $\Pr(s=1|x,y=1)=\Pr(s=1|y=1)$:

$$\begin{aligned}
\Pr(s=1|x\in A,y=1) &= \sum_{x'}\Pr(s=1,x=x'|x\in A,y=1)\\
&= \sum_{x'\in A}\Pr(s=1,x=x'|x\in A,y=1)\\
&= \sum_{x'\in A}\Pr(s=1|x=x',y=1)\Pr(x=x'|x\in A,y=1)\\
&= \Pr(s=1|y=1)\sum_{x'\in A}\Pr(x=x'|x\in A,y=1)\\
&= \Pr(s=1|y=1)=c.
\end{aligned}$$

□

Therefore, the label frequency is the ratio of the probabilities to be labeled and to be positive in any subdomain A:

$$c=\frac{\Pr(s=1|x\in A)}{\Pr(y=1|x\in A)}. \tag{3.2}$$

Proof.

$$\begin{aligned}\Pr(s=1|x\in A) &= \sum_{x\in A}\Pr(s=1|x)\Pr(x|x\in A)\\ &= \sum_{x\in A} c\Pr(y=1|x)\Pr(x|x\in A)\\ &= c\sum_{x\in A}\Pr(y=1|x)\Pr(x|x\in A)\\ &= c\Pr(y=1|x\in A)\end{aligned}$$

□

If A is a positive subdomain $\Pr(y=1|x\in A)=1$, the probability of being labeled in this subdomain equals the label frequency. In general, the probability is a lower bound for the label frequency because probabilities are at most 1:

$$c \geq \Pr(s=1|x\in A). \tag{3.3}$$

3.1.1 Label Frequency Lower Bound from Data Subset

Using Equation (3.3), we can use any attribute-conditioned subset of the data, with L labeled and T total examples, to estimate a lower bound on c. Naively, this would be $c \geq L/T$. However, because of the stochastic nature of the labeling, more positive examples might be labeled than expected. Therefore, we include an error term $\epsilon = 1/2\sqrt{(1-\delta/\delta T)}$ that shrinks with the sample size T. The error term is derived from the one-sided Chebyshev inequality. This inequality provides a lower bound on the probability of the number of labeled examples L exceeding the expected number μ by at least λ, based on the labeling variance σ^2. The probability lower bound is set to $\delta = \sigma^2/(\sigma^2+\lambda^2)$.

$$\Pr\left(L \geq \mu+\lambda\right) \leq \frac{\sigma^2}{\sigma^2+\lambda^2}$$

$$\Pr\left(L \geq \mu + \sqrt{\frac{(1-\delta)\sigma^2}{\delta}}\right) \leq \delta.$$

Labeling positive examples follows the Binomial distribution, because, each positive example in the subdomain is independently labeled with a probability

c. Therefore, $\mu = cP$ and $\sigma^2 = c(1-c)P$. Substituting μ and σ, and using $P \leq T$, gives a probabilistic lower bound for c:

$$\Pr\left(L - cT \geq \sqrt{\frac{(1-\delta)c(1-c)T}{\delta}}\right) \leq \delta$$

$$\Pr\left(c \leq \frac{L}{T} - \sqrt{\frac{(1-\delta)c(1-c)}{\delta T}}\right) \leq \delta. \tag{3.4}$$

The error term depends on c. But by using probability properties: $c(1-c) \leq 0.25$, a lower bound with probability at least $1-\delta$ can be calculated using only T and L:

$$\Pr\left(c \leq \frac{L}{T} - \frac{1}{2}\sqrt{\frac{1-\delta}{\delta T}}\right) \leq \delta. \tag{3.5}$$

3.1.2 Interesting Subsets

Using the lower bound in Equation (3.5) to estimate the true label frequency c, requires calculating it in a subset where the bound is tight. This is the case in large (almost) purely positive subsets. The positive probability is directly proportional to the labeled probability, therefore, positive subdomains are expected to have a high ratio of labeled examples. Mostly positive subdomains are therefore likely to be found when looking for highly labeled regions in the data.

If a subset is selected because it has a high number of labeled examples, it is likely that the number of labeled examples will exceed the expected number cP. Therefore, the bound in Equation (3.5) does not hold in such a subset. This issue is resolved by using independent datasets to 1) identify interesting subdomains and 2) calculate the lower bound. To intuitively see that it is resolved, consider datasets $D1$ and $D2$ which are independently sampled from the probability distribution $\Pr(x, y)$. $D1$ is labeled following the "selected completely at random" assumption by throwing a die for each positive example. Now we look for a subdomain that has a high proportion of labeled examples in $D1$. Next, $D2$ is labeled using the same procedure, therefore each positive example in the previously found subdomain has the same probability to be labeled as any other example. The probabilities are unaffected by the subdomain search procedure. Labeling $D2$ first does not change the search procedure, ensuring unaffected probabilities for $D2$ that are usable for calculating the lower bounds.

3.1.3 Tree Induction for Label Frequency Estimation

We propose a novel method TIcE (*Tree Induction for c Estimation*) for estimating the label frequency from PU data, which is summarized in Algorithm 1. The algorithm is based on the insights of the previous sections: It splits the dataset into two separate sets, looks for interesting, i.e., likely positive, subdomains using one set and estimates c using the other set by taking the tightest lower bound that is calculated in the interesting subdomains. Looking for pure subsets in the data is also the objective of decision tree induction, therefore TIcE looks for pure labeled subsets by inducing a decision tree, considering the unlabeled data as negative.

Folds The separate datasets are referred to as the *tree data* and the *estimation data*, the former is used for inducing the tree and the later for estimating the label frequency using the subdomains. To make robust estimates, the process is repeated for k folds: the training data is divided into k random equal-sized subsets and in every fold, 1 subset is used as tree data, the rest as estimation data. The estimate is the average of the estimates of the folds.

Max c_{low} The label frequency is estimated as the maximum lower bound calculated in multiple subsets of the estimation data. The maximum invalidates the lower bound of Equation (3.5) because although each lower bound has a probability $1-\delta$ of holding, the probability of all the lower bounds holding is smaller. The maximum could be avoided by only calculating the lower bound on one subdomain. The most promising subdomain is selected solely based on the *tree data*. This would ensure a correct lower bound with probability $1-\delta$. However, we argue that the maximum would work better in practice. The lower bound is very loose and is only likely to get close to the true c in the rare case of a completely positive subset. By only taking one lower bound, the chance that a lower bound is calculated in a purely positive subset decreases. Especially with a low label frequency, it is hard to predict which subdomains are positive. Moreover, the goal of TIcE is not to find a lower bound, it is to find a close estimate which does not need to be a lower bound. This decision is evaluated in the experiments section.

Split The objective of standard decision tree induction is finding pure nodes. Here, only pure *positive* nodes are of interest. To reflect this, the *maximum biased estimate for the proportion of positives (max-bepp)* score is used [9]. It selects the split that gives the subset with the highest bepp: $\frac{P}{T+k}$, where the parameter k acts like the Laplace parameter to prefer larger subsets.

Algorithm 1: TIcE (k, M, f)

```
Input : $k$: max-bepp parameter
        $M$: maximum number of splits
        $f$: tree/estimation folds
Result: $\hat{c}$
1  $\hat{c} \leftarrow 0.5$
2  for $i = 0$ ; $i < 2$ ; $i{+}{+}$ do
3      $\hat{cs} \leftarrow []$
4      for (tree data, estimation data) $\in f$ do
5          $\delta \leftarrow \max\left[0.025, \frac{1}{1+0.004T(\text{estimation data})}\right]$
6          $c_\text{best} \leftarrow \frac{L(\text{estimation data})}{T(\text{estimation data})}$
7          $q \leftarrow [(\text{tree data, estimation data})]$
8          for $j = 0$ ; $j < M$ and $|q| > 0$ ; $j{+}{+}$ do
9              $(S_t, S_e) \leftarrow \underset{(S_t,S_e)\in q}{\text{argmax}} \left[\frac{L(S_t)}{T(S_t)} - \sqrt{\frac{\hat{c}(1-\hat{c})(1-\delta)}{\delta T(S_t)}}\right]$
10             $q.\text{remove}((S_t, S_e))$
11             $a^* = \underset{a\in\text{atts}(S_e)}{\text{argmax}} \max_{v\in\text{Dom}(a)} \frac{L(\{S_t\ :\ a=v\})}{T(\{S_t\ :\ a=v\})+k}$
12             for $v \in Dom(a^*)$ do
13                 $q.\text{append}((\{S_t : a^* = v\}, \{S_e : a^* = v\}))$
14                 $c_\text{low} = \frac{L(\{S_e:a^*=v\})}{T(\{S_e:a^*=v\})} - \sqrt{\frac{\hat{c}(1-\hat{c})(1-\delta)}{\delta T(\{S_e:a^*=v\})}}$
15                 $c_\text{best} \leftarrow \max(c_\text{best}\ ,\ c_\text{low})$
16         $\hat{cs}.\text{append}(c_\text{best})$
17     $\hat{c} \leftarrow \text{avg}(\hat{cs})$
```

Tighter bound with $\hat{c}$ Equation (3.5) assumes the worst case of $c = 0.5$, making the error term larger than necessary in most cases. An initial estimate c_prior, used in Equation (3.4) makes a more accurate estimate. Therefore, TIcE first induces a tree and to estimate c_prior and then repeats the process to estimate $\hat{c}$, using c_prior.

Choosing δ To calculate lower bounds, the parameter δ needs to be supplied. Its optimal value depends on the application and the data dimensions. δ can be chosen by supplying the minimum number of examples T_R that should be required to calculate a lower bound with some error term ϵ: $\delta = \frac{1}{1+4\epsilon^2 T_R}$.

We propose a simple rule for choosing δ, which we evaluate in the experiments section. Big datasets, that contain more than 10,000 examples require 1,000 examples to update c_{low} with an error of $\epsilon = 0.1$. Smaller datasets need one tenth of their data: $T_R = \min[1000, 0.1T]$. Therefore:

$$\delta = \max\left[\, 0.025\ ,\ \frac{1}{1 + 0.004T}\right]. \tag{3.6}$$

Speed Top-down decision tree induction is an efficient algorithm but can be further sped up by limiting the number of splits to a constant. Limiting the splits with optimal quality preservation is achieved by executing the splits in a best-first order. For this, the nodes need to be scored to indicate which one is most likely to result in good subsets. TIcE uses the lower bound provided by the node data, which needs to be calculated in the tree data to prevent overfitting.

Complexity The worst case time complexity of TIcE is $\mathcal{O}(mn)$, with n the number of examples n and m the number of attributes, assuming that each attribute's domain size is at most d. The size of the queue q can never exceed $(d-1) \cdot M$, therefore, all queue operations (lines 9, 10 and 13) have a constant worst-case time complexity. Nodes are recursively split in lines 9 to 15. Finding the best attribute to split on (line 11), requires going over all available attributes and all the tree data in the node, which has a complexity $\mathcal{O}(mn/|f|) = \mathcal{O}(mn)$. The estimation data will be split on the found attribute (needed in lines 13 and 14), which has complexity $\mathcal{O}(n(|f|-1)/|f|) = \mathcal{O}(n)$. Lines 12 to 15 have complexity $\mathcal{O}(d)$ because for each value of the domain, constant-time operations are executed. The total complexity of splitting a node is thus $\mathcal{O}(nm + n + d) = \mathcal{O}(nm)$. The loops of lines 2,4 and 8 each have a constant number of iterations: 2, $|f|$ and M and, thus, do not alter the complexity.

3.2 Related Work

Elkan and Noto (2008) where the first to consider the label frequency explicitly [30]. Their insight is that a probabilistic classifier trained on PU data is expected to classify positive examples as positive with a probability c. To estimate c, they train a classifier on part of the data and predict the probabilities of the positively labeled examples in the other part. $\hat{c}$ is the average of the predicted probabilities. Another estimator they proposed was the highest predicted probability by the trained classifier on an example in the validation set. They discarded this estimator because of its high variability. This estimator

Table 3.1: **Characteristics of the Datasets.**

Dataset	# Examples	# Vars	$\Pr(y = 1)$
Breast Cancer	683	9	0.350
Mushroom	8,124	21	0.482
Adult	48,842	14	0.761
IJCNN	141,691	22	0.096
Cover Type	536,301	54	0.495
20ng comp vs rec	5,287	200	0.450
20ng comp vs sci	5,279	200	0.450
20ng comp vs talk	4,856	200	0.401
20ng rec vs sci	4,752	200	0.499
20ng rec vs talk	4,329	200	0.450
20ng sci vs talk	4,321	200	0.451

is very related to TIcE, but two important differences make TIcE more reliable. First, their estimator only uses one data point for estimation. Second, the parameters of the model, and thus the prediction, are based on the training data, while TIcE computes its estimate from the validation data.

Several mixture proportion estimation methods have been developed to estimate the class prior from PU data [28, 24, 50, 48, 94]. These methods are discussed in detail in Chapter 2.5. Scott (2015) [99] and du Plessis et al. (2015) [24] make the same assumption as us that a positive subdomain exists in the subspace. Our method differs from theirs in that it concentrate on those regions instead of modeling the entire domain. Ramaswamy et al.(2016) [94] also made the observation that not the entire domain needs to be modeled. Their method corresponds to an exhaustive search over functions that maps the data to subsets, selecting the minimum class prior that makes the subsets purely positive. For smaller sample sizes, this is not a robust method.

3.3 Experiments

We aim to gain insight in the performance of TIcE. First, we check if in practice it is better to take the maximum of lower bounds or to use one lower bound. Second, we evaluate our method for setting δ. Finally, we compare TIcE to other class prior estimation algorithms.

3.3.1 Data

We use 11 real-world datasets that are summarized in Table 3.1. IJCNN was used for the IJCNN 2001 neural network competition[1] [92]. All the others are UCI[2] datasets. Features were generated for the twenty newsgroups data (20ng) using bag of words with the 200 most frequent words, disregarding nltk stop words. Binary classification tasks were defined by classifying pairs of general categories (computer, recreation, science, and talk). All the multivalued features were binarized and the numerical features were scaled between 0 and 1.

3.3.2 Methods

To create PU datasets from the completely labeled datasets, the positive examples are selected to be labeled with label frequencies $c \in [0.1, 0.3, 0.5, 0.7, 0.9]$. Not that varying c does not change the training set size, only the number of observed positive examples in the training data. In addition to PU datasets, NU datasets were generated by labeling negative examples with the same frequencies. Per label frequency and class label, 5 different random labelings were executed. In total, there are $11 \cdot 2 \cdot 5 \cdot 5 = 550$ settings.

We fixed all the hyperparameters of our method TIcE, because, in the considered context, no supervised validation dataset is available for tuning. The max-bepp parameter is $k = 5$ as in the original paper. The maximum number of splits is $M = 500$, which is expected to be large enough in any setting because of the best-first ordering. δ was set using Equation (3.6) on the estimation data in each fold. The number of folds is 5. All experiments are repeated 5 times with different random folds. If a node is split on a numerical feature, the range is split into 4 equal parts. When only the most promising subdomain is considered, this subdomain is selected by calculating lower bounds on the tree data. The implementation and additional results are available online.[3]

We compared to the following class prior estimation methods that also make the "selected completely at random" assumption: EN [30], PE [28], pen-L1 [24], KM1 and KM2 [94], αMax [50] and αMax_N [48].[4] KM1 and KM2 cannot handle large datasets. Therefore, like the authors of those papers, we subsampled the

[1] Available on: `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

[2] `http://archive.ics.uci.edu/ml/`

[3] `https://dtai.cs.kuleuven.be/software/tice`

[4] The code for PE was taken from `http://www.mcduplessis.com/index.php/class-prior-estimation-from-positive-and-unlabeled-data/` and for KM1 and KM2 from `http://web.eecs.umich.edu/~cscott/code/kernel_MPE.zip`. The code for pen-L1 was the same as in [24]. The code for αMax, αMax_N and EN were the same as in [48]. These were acquired through personal communication.

Table 3.2: **Sensitivity to T_R.**

%	$\lvert\hat{\alpha}-\alpha\rvert$ (all)	$\lvert\hat{\alpha}-\alpha\rvert$ (3 largest datasets)
50	0.092	0.084
75	0.089	0.079
100	0.090	0.095
125	0.135	0.108
150	0.170	0.127

datasets to have at most 2000 examples and repeated the process five times. The implementation also does not estimate a class prior from the smallest dataset Breast Cancer.

For all the experiments, the absolute error on the class prior $|\hat{\alpha} - \alpha|$ and CPU time were measured. The label frequency c is converted to the class prior with $\hat{\alpha} = L/(\hat{c}T)$. To evaluate using Equation (3.6) to set δ, its sensitivity to T_R is analyzed. To this end, in addition to using the proposed value $T_R^* = \min[1000, 0.1T]$, the experiments were also executed with $T_R \in \{0.5, 0.75, 1.25, 1.5\} \cdot T_R^*$.

All experiments are executed on a Red Hat Enterprise Linux Compute Node with access to 24G RAM and 1 core of a Xeon E5-2680v2 CPU.

3.3.3 Results

Using the maximum of lower bounds often gives better estimates for the label frequency and is never worse than only using the lower bound of the "most promising" subdomain (Figures 3.1, 3.2). This is in line with our expectations. When little data is available, like for Breast Cancer, predicting the "most promising" subdomain is even more challenging. For Adult, both methods work fine because it has a high class prior.

Changing T_R does not change the estimates much, especially when the the dataset is large (Figures 3.3, 3.4 and Table 3.2). In fact, our rule from Equation (3.6) is usually overly conservative, which probably means that it is unlikely to for such big positive subsets to occur in the data. We did not change the rule because that would be tuning on the test data.

When comparing the absolute error $|\hat{\alpha} - \alpha|$ between different methods, we see that TIcE is equivalently accurate as the state of the art (Figures 3.7, 3.8 and Table 3.4). Its average rank according to $|\hat{\alpha} - \alpha|$ is the second best, after KM2, and it has on average the lowest absolute error $|\hat{\alpha} - \alpha|$. Note that it is also

Table 3.3: **Time comparison between different methods.**

Method	Average time per example (ms)
TIcE	**0.76**
PE	1.10
pen-L1	4.84
EN	21.96
αMax	38.47
KM1/KM2	47.28
αMax_N	52.92

very stable, its predictions are never off much by much. This is reflected in its standard deviation, which is the lowest of all the methods. For comparison, look at the results of KM2. This is usually the most accurate estimator, but when it does not have access to many labels (e.g. 20ng Rec vs Sci, $c = 0.1$), it is biased too much towards 0.5 which results in large errors. TIcE's stability is owed to two reasons: First, overestimates of the label frequency c (which are underestimates of the class prior α) only occur in 21% of the experiments because of the conservative lower bounds. Second, a tight lower bound is likely to be found because it is likely that some subdomain exists that is (close to) purely positive.

TIcE is a fast method, the other estimators that are equivalent or slightly worse (KM2, αMax_N, αMax) are more than an order of magnitude slower (Figures 3.5, 3.6 and Table 3.3). Speed-wise, PE comes closest to TIcE, however, it gives less accurate estimates for the class prior. Note that KM1 and KM2 seem to have reasonable times for big datasets such as Cover Type, this is because it uses a subsample of the data instead of the full dataset.

Table 3.4: **Absolute class prior error comparison between different methods.** TIcE is the second-best method when considering the rank in all settings. However, it has the best average performance and lowest standard deviation, therefore, it is more reliable.

Method	$\|\hat{\alpha} - \alpha\|$ rank +/- SD	mean $\hat{\alpha} - \alpha$ +/- SD
KM2	**2.83 +/- 2.03**	0.10 +/- 0.13
TIcE	3.22 +/- 1.82	**0.09 +/- 0.06**
αMax_N	3.42 +/- 1.89	0.13 +/- 0.10
αMax	3.51 +/- 1.38	0.12 +/- 0.09
KM1	4.04 +/- 1.19	0.11 +/- 0.09
EN	5.84 +/- 1.62	0.27 +/- 0.16
PE	6.16 +/- 1.39	0.29 +/- 0.14
pen-L1	6.88 +/- 1.86	0.37 +/- 0.20

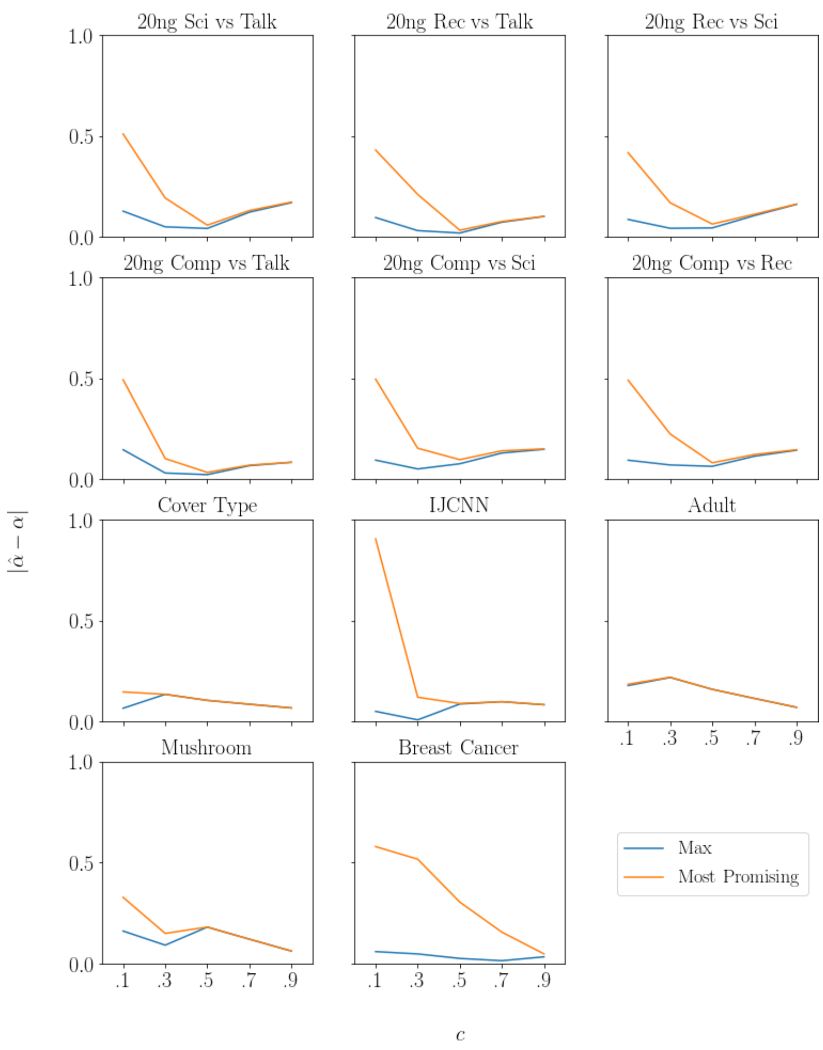

Figure 3.1: **Taking the maximum lower bound vs most promising subdomain for PU datasets**. The true label frequency is varied on the x-axis. The lower the error, the better. Using the maximum gives better results, especially for lower frequencies.

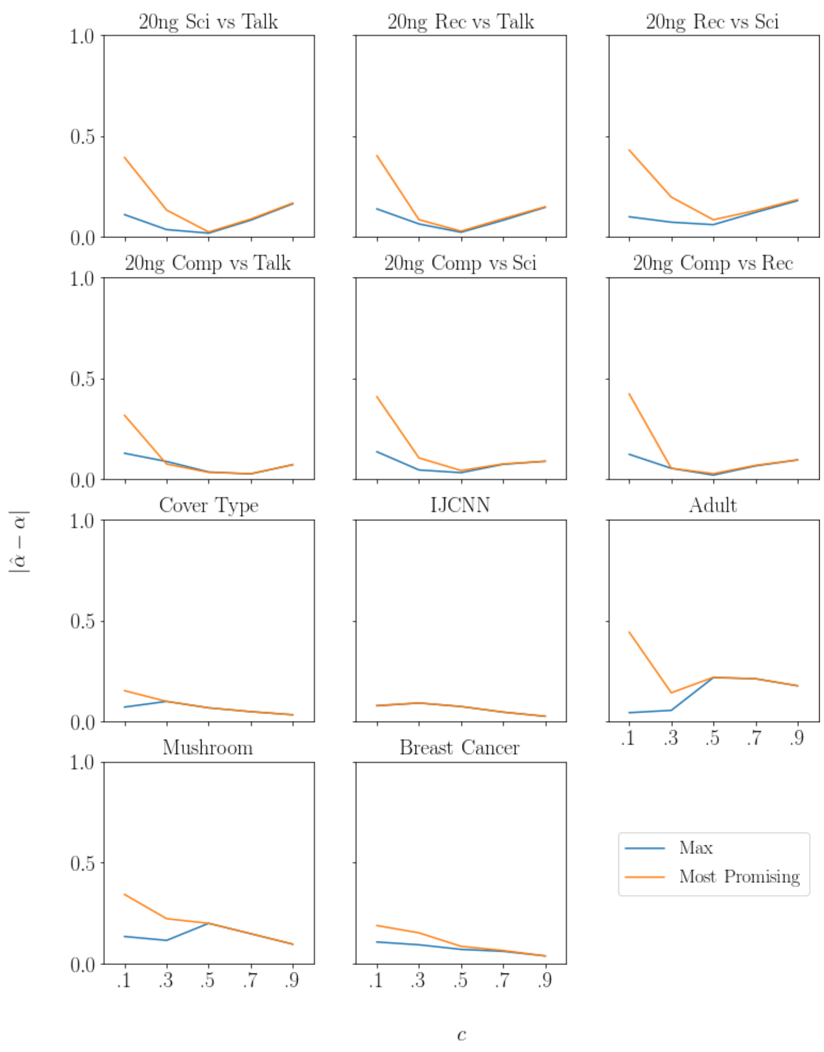

Figure 3.2: **Taking the maximum lower bound vs most promising subdomain for NU datasets**. The true label frequency is varied on the x-axis. The lower the error, the better. Using the maximum gives better results, especially for lower frequencies.

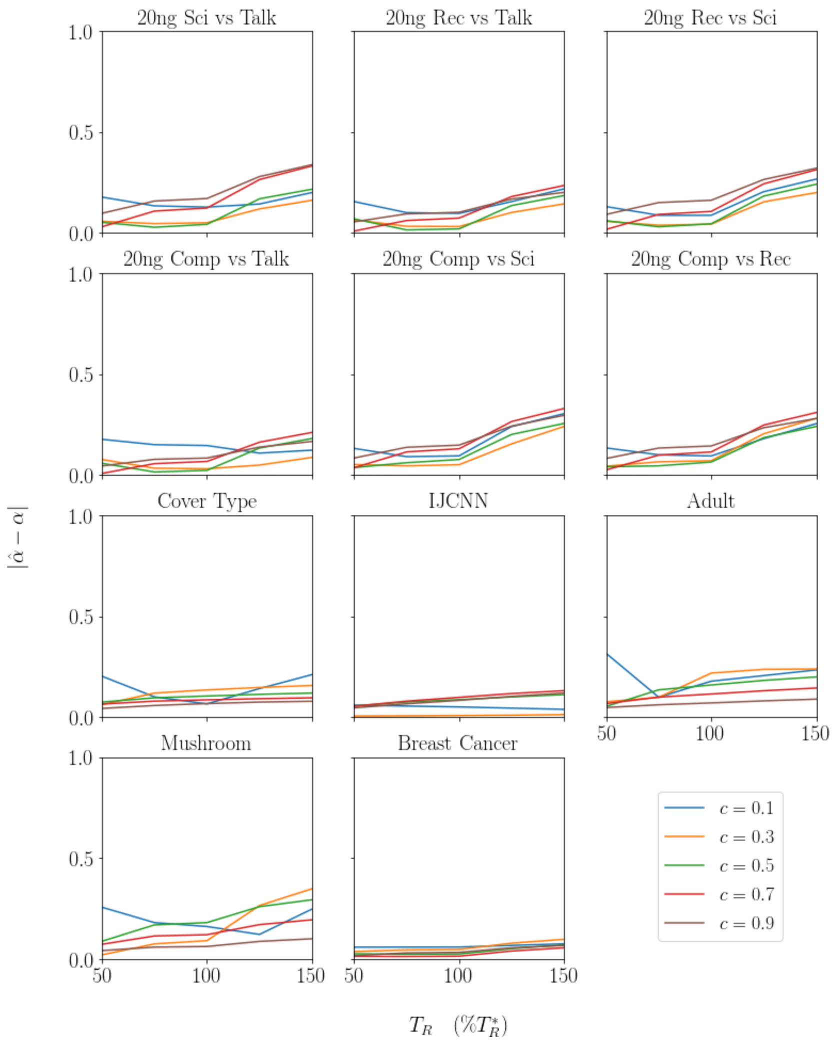

Figure 3.3: **Sensitivity to T_R for PU datasets**. The number of examples to update c_{low} with an error $\epsilon = 0.1$ is varied on the x-axis. TIcE is not very sensitive to T_R.

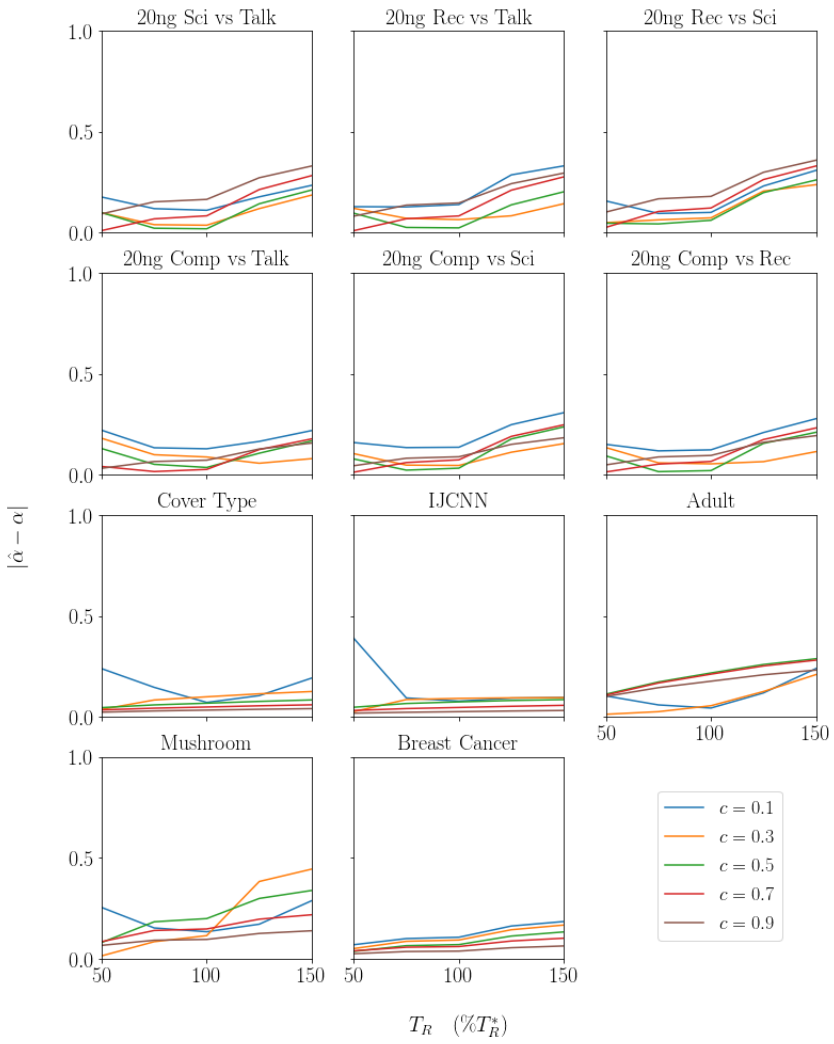

Figure 3.4: **Sensitivity to T_R for NU datasets**. The number of examples to update c_{low} with an error $\epsilon = 0.1$ is varied on the x-axis. TIcE is not very sensitive to T_R.

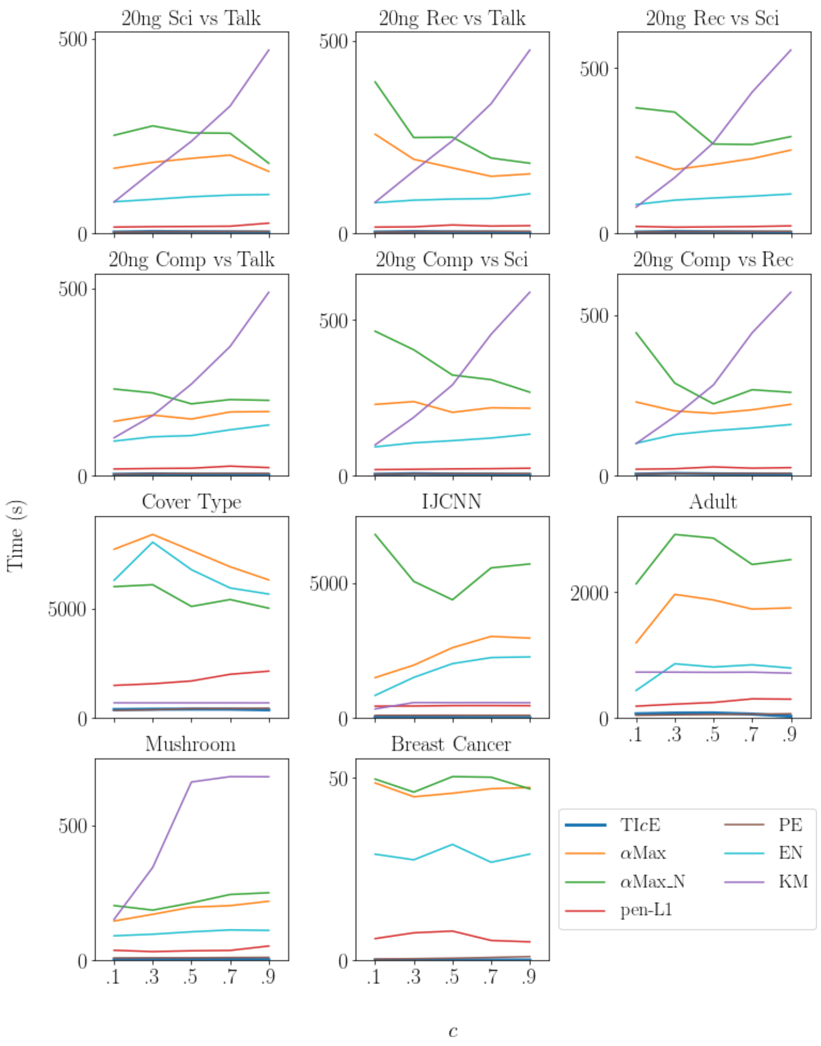

Figure 3.5: **Time comparison between different methods for PU datasets**. The true label frequency is varied on the x-axis. TIcE is consistently fast.

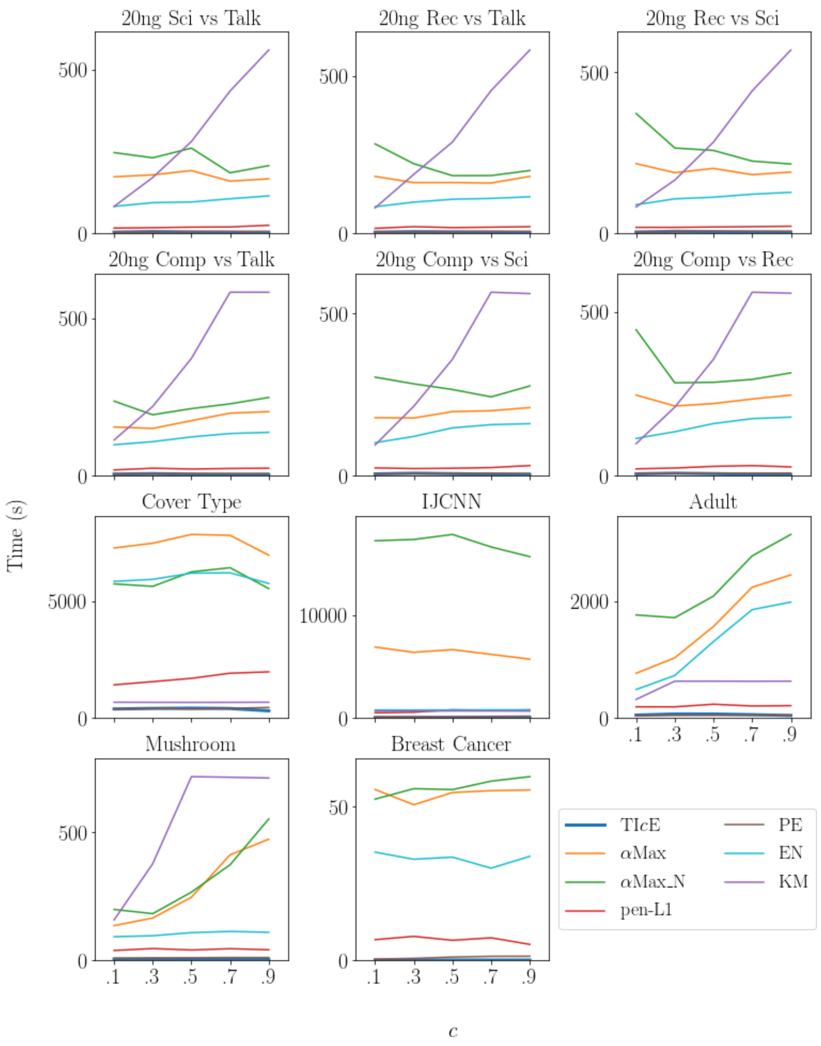

Figure 3.6: **Time comparison between different methods for NU datasets.** The true label frequency is varied on the x-axis. TI*c*E is consistently fast.

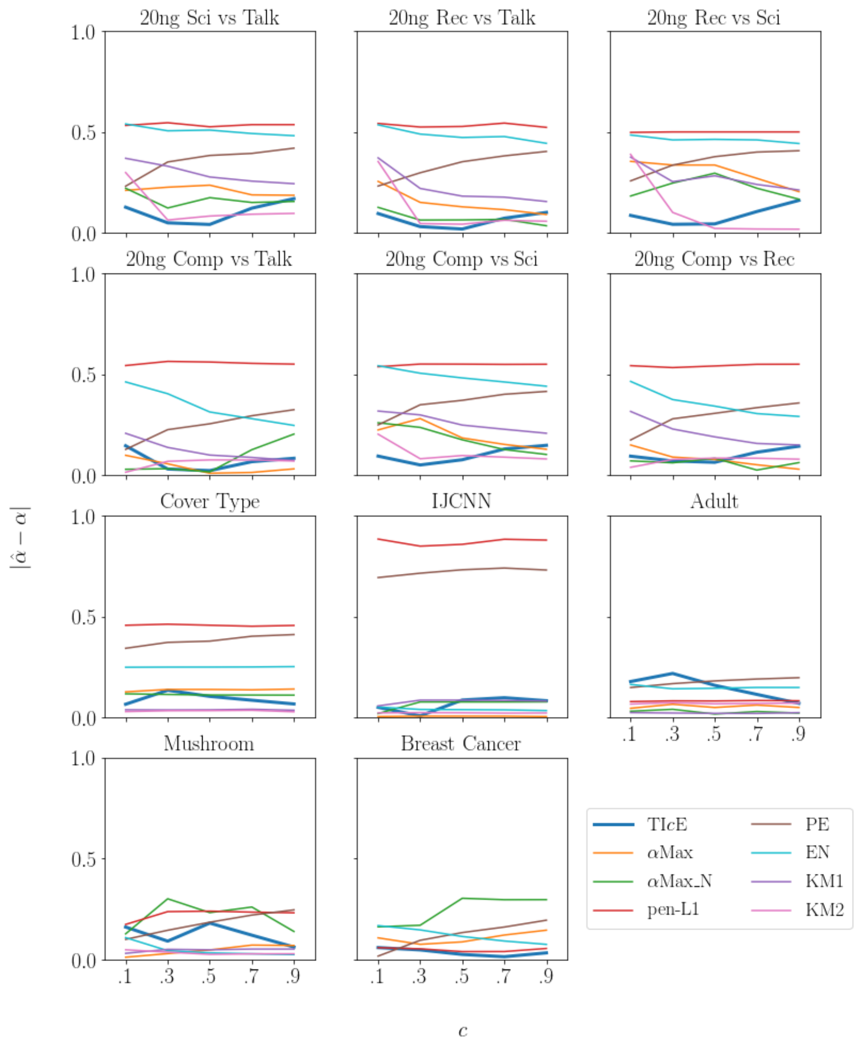

Figure 3.7: **Absolute class prior error comparison between different methods for PU datasets.** The true label frequency is varied on the x-axis. The lower the error, the better. TI*c*E gives stable estimates, with an average error 0.09 and standard deviation 0.06.

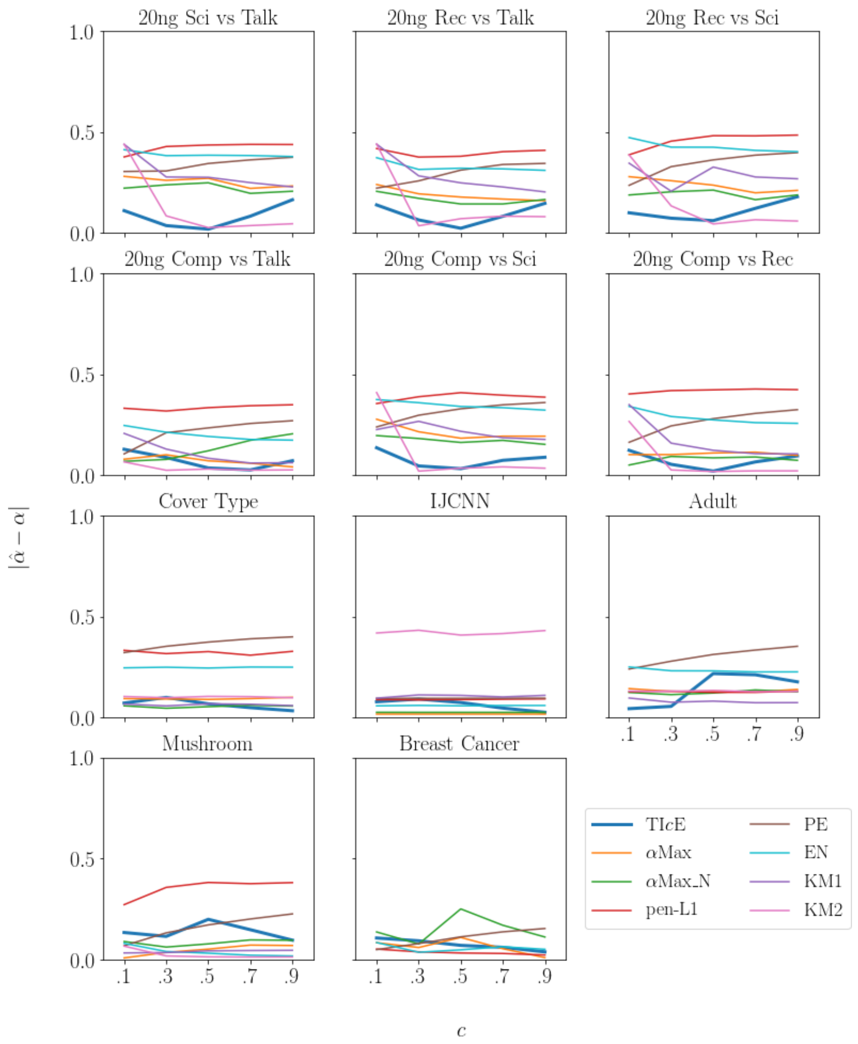

Figure 3.8: **Absolute class prior error comparison between different methods for NU datasets.** The true label frequency is varied on the x-axis. The lower the error, the better. TI*c*E gives stable estimates, with an average error 0.09 and standard deviation 0.06.

3.4 Conclusions

In this chapter, we proposed a simple yet effective method for estimating the class prior. The method is based on the insight that the label frequency (which serves as a proxy for the class prior) is expected to be the same in any subdomain of the attributes. As a result, subsets of the data naturally imply lower bounds on the label frequency. The lower bounds will be tight when the subset belongs to a positive subdomain. Finding likely positive subdomains can easily be done using decision tree induction based on the PU data. Despite the simplicity of the method, it gives good and stable estimates. The experiments show that this method is equivalently accurate to the state of the art but an order of magnitude faster.

Chapter 4

Learning from Positive and Unlabeled Relational Data under the Selected Completely At Random Assumption

The task of learning from relational data significantly differs from traditional machine learning because the data has a completely different structure. Therefore, it is not surprising that learning from positive and unlabeled examples has developed independently in the two domains. In relational learning, a standard assumption is the closed-world assumption, which regards all unlabeled examples as negative [59, 109, 36, 85]. A few methods have been developed that make the separability assumption [83, 79, 98]. This chapter explores if the Selected Completely At Random (SCAR) assumption can be employed in the relational setting. To this end, we investigate whether similar techniques can be used as in the propositional case, when the SCAR assumption applies.

Problem Statement

The problem that this chapter addresses is the following:

Given a relational PU dataset, where the class $\mathbf{y}$ is not observed, none of the negative examples are labeled $\Pr(s = 1|y = 0, \text{facts}) = 0$, and the labeled examples were selected completely at random from the positive set $\Pr(s = 1|y = 1, \text{facts}) = \Pr(s = 1|y = 1)$,

Train a binary classifier that distinguishes between positive and negative examples, using insights from propositional PU learning.

The importance of this problem comes from the independent development of techniques for positive and unlabeled learning in the relational and propositional domain. Most relational methods use the separability assumption, which can be too restrictive. Therefore it is interesting to investigate how ideas and techniques that originate from propositional SCAR PU data can be ported to relational SCAR PU data.

Contributions of this Chapter

This chapter addresses the problem statement by making four main contributions.

The first contribution is to employ insights from learning from positive and unlabeled data in propositional domains to learning from this type of data in relational domains. More specifically, we investigate if the SCAR assumption can be utilized in a similar fashion.

The second contribution is proposing two methods for incorporating the label frequency in relational classifier learning. The methods are applied to two popular relational learning methods: TILDE and Aleph [8, 110].

The third contribution is to modify our propositional method TI*c*E for estimating the label frequency from the data (see Chapter 3) to TI*c*ER, which operates in the relational domain.

The fourth contribution is an extensive empirical evaluation. The main conclusion from the experiments is that when the data is easily separable, then incorporating the estimate from TI*c*ER in TILDE or Aleph performs similar to the established method for relational PU learning, and when the data is not easily separable, then TI*c*ER outperforms it.

The content of this chapter is based on the following publication [5]:

Bekker, J., and Davis, J. Positive and Unlabeled Relational Classification through Label Frequency Estimation. In *Inductive Logic Programming (Revised Selected Papers of ILP 2017; Orléans, France; 4-6 September 2017)* (2018b), pp. 16–30. ✷ *Most promising late-breaking student paper.*

```
professor(jesse)                 hasposition(jesse,faculty)
professor(hendrik)               hasposition(benoit,faculty)
professor(tinne)                 hasposition(benedicte,faculty)
professor(benedicte)             taughtby(uai,tinne,fall_1819)
professor(benoit)                taughtby(bdap,jesse,1819)
student(jessa)                   taughtby(bdap,jessa,1819)
student(sebastijan)              taughtby(ml,hendrik,fall_1819)
advisedby(jessa,jesse)           taughtby(ml,benoit,fall_1819)
advisedby(sebastijan,hendrik)    ta(ml,sebastijan,fall_1819)
advisedby(adrien,benoit)         ta(bdap,jessa,1718)
inphase(defending,jessa)         courselevel(uai,ma)
inphase(defending,sebastijan)    courselevel(bdap,ma)
```

Figure 4.1: **Relational Data.** The relations are represented as functions over objects. Relations can have different arities.For example, `professor` is a unary relation, `advisedby` binary and `taughtby` ternary.

4.1 Background on Learning from Relational Data

For most learning tasks, examples are expected to be fixed-length vectors where each position corresponds to an attribute, like in Table 2.1. Relational data, in contrast has no fixed-length format. A relational dataset consists of objects and relations between those objects. Figure 4.1 shows an example of relational data, loosely based on widely used UW-CSE dataset.[1]

The relational learning task that we consider in this dissertation is to predict unary relations like `professor(benoit)` from this type of data. Two established systems for this task are TILDE, a decision tree learner [8], and Aleph, a rule leaner [110].

TILDE is a relational version of the C4.5 top-down decision tree induction method. The main difference with the standard propositional method is the set of tests considered at a node, which are now relations with constants or variables as objects, where at least one of the objects is the queried object or can be related to the queried objected through a grounding based on a relation higher up the tree. The answer to the test is 'yes' if a relation exists for the queried object. The input data contains examples of objects of the classes that the tree should classify. Figure 4.2 shows the decision tree that TILDE learns from the UW-CSE data.

[1] `http://alchemy.cs.washington.edu/data/`

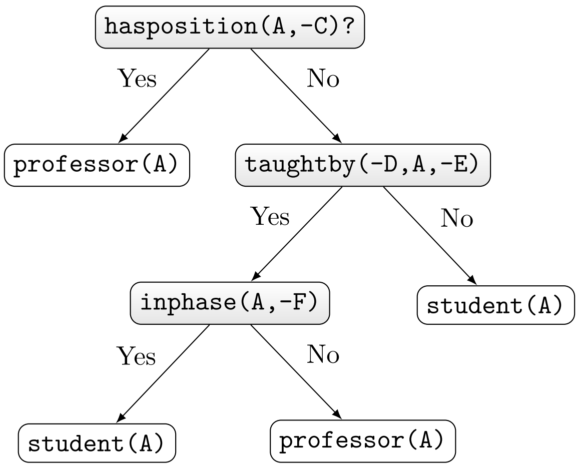

Figure 4.2: **TILDE example.** The learned decision tree predicts the classes `professor(A)` or `student(A)` for objects `A` of type person.

```
professor(A) :- hasposition(A,faculty).
professor(A) :- taughtby(B,A,C), courselevel(B,500), ta(B,D,E).
professor(A) :- hasposition(A,faculty_adjunct).
professor(A) :- advisedby(B,A).
professor(A) :- hasposition(A,faculty_emeritus).
```

Figure 4.3: **Aleph example.** The learned rule set for the concept `professor`. Every object `A` of type person for which one of the rules fires, is predicted to be a `professor`.

Aleph is a relational rule learner that aims to learn rule sets that define concepts, like, for example, `professor`. Each rule is a conjunction of relations, where the objects can be both constants or variables. The input data contains positive examples of objects that belong to the concept and negative examples of objects that to not belong to the concept. If the concept to be learned is `professor`, then the positive examples are all `A` that are in the relation `professor(A)` and the negative examples are all `A` that are in the relation `student(A)`. Each rule in the rule set is learned by building the most specific rule possible to describe a positive example and then iteratively generalizing it by selecting a subset of the relations in the rule. The subset is chosen based on a evaluation function, which, by default, is coverage $P_i - N_i$. Figure 4.2 shows the rule set that Aleph learns from the UW-CSE data.

4.2 Using the Label Frequency for Relational PU Learning

Elkan and Noto propose to use the label frequency directly to modify traditional classifiers for PU learning [30]. Concretely, they proposed the following two methods:

1. **Probabilistic classifier modification:** This method trains a probabilistic classifier to predict the probability for instances to be labeled. To this end, during training, it considers unlabeled examples as negative. The label frequency is then employed to modify the output probabilities. It transforms the probability that an instance is labeled $\Pr(s = 1|x)$ into the probability that an instance is positive: $\Pr(y = 1|x) = \frac{1}{c} \Pr(s = 1|x)$ [127]. This modified classifier can be used directly or to transform the PU dataset into a probabilistically weighted PN dataset.

2. **Score function modification:** Learning algorithms that make decisions based on counts of positive and negative examples data subsets i can be modified to use counts of labeled and unlabeled examples. The positive and negative counts P_i and N_i can be obtained with $P_i = L_i/c$ and $N_i = T_i - P_I$. Decision trees, for example, assign classes to leaves and score splits based on the positive/negative counts in the potential subsets and can, therefore, be transformed to PU learners [22].

These methods can also be applied in the relational domain. Our proposed solutions c-adjusted TILDE and c-adjusted Aleph are described below.

4.2.1 Relational Probabilistic Classifier Modification: c-adjusted TILDE

The first method for using the label frequency requires a probabilistic classifier which predicts the probability that an instance is labeled. The first-order logical decision tree learner TILDE can easily be made probabilistic. Doing so simply requires counting for each leaf i the number of labeled L_i and unlabeled examples U_i that reach i setting the leaf's probability to $\frac{L_i}{U_i+L_i}$ [8]. The tree that predicts the probability for instances to be positive has the same structure as the tree for distinguishing between labeled and unlabeled example, but requires altering the probability in each of its leaves. The new probability in each leaf i is $\frac{1}{c}\frac{L_i}{U_i+L_i}$, where L_i and U_i are defined as above.

4.2.2 Relational Score Function Modification: c-adjusted Aleph

The second method for using the label frequency requires a classifier that makes decisions based on the counts N_i and P_i in a subset of the data i. TILDE satisfies this criterion, and so does the rule learner Aleph [110]. The default evaluation function of Aleph is coverage, which is defined as $P_i - N_i$, where i is the subset of examples that satisfy the rule. To modify Aleph to use the label frequency c, the coverage for each rule r should be calculated as follows:

$$\text{PU coverage} = P_i - N_i = 2P_i - T_i = 2\frac{L_i}{c} - T_i \tag{4.1}$$

where L_i is the number of labeled (i.e., positive) examples covered by the rule and T_i is the total number of examples covered by the rule.

4.3 Label Frequency Estimation

To estimate the label frequency in relational PU data, we will use the insights of a propositional label frequency estimator. We first review the original method and then propose a relational version.

4.3.1 Label Frequency Estimation in Propositional PU data

The propositional estimator is TIcE from Chapter 3 [3]. It is based on two main insights: 1) a subset of the data naturally provides a lower bound on the label frequency, and 2) the lower bound of a large enough positive subset approximates the real label frequency. TIcE uses decision tree induction to find likely positive subsets and estimates the label frequency by taking the maximum of the lower bounds implied by all the subsets in the tree.

The label frequency is the same in subsets of the data because of the SCAR assumption, therefore it can be estimated in a subset of the data. Clearly, the true number of positive examples P_i in a subset i cannot exceed the total number of examples in that subset T_i. This naively implies a lower bound: $c = L_i/P_i \geq L_i/T_i$. To take stochasticity into account, this bound is corrected with confidence $1-\delta$ using the one-sided Chebyshev inequality which introduces an error term based on the subset size:

$$\Pr\left(c \leq \frac{L_i}{T_i} - \frac{1}{2}\sqrt{\frac{1-\delta}{\delta T_i}}\right) \leq \delta \tag{4.2}$$

The higher the ratio of positive examples in the subset, the closer the bound gets to the actual label frequency. The ratio of positive examples is unknown, but directly proportional to the ratio of labeled examples. Therefore, TIcE aims to find subsets of the data with a high proportion of labeled examples using decision tree induction. To avoid overfitting, i.e. finding subsets i where $L_i/P_i > c$, k folds are used to induce the tree and estimate the label frequency on different datasets.

The parameter δ is set such that at least one tenth of the data or 1000 examples are needed to estimate the label frequency with an error term of 0.1: $1/2\sqrt{(1-\delta)/(\delta T_R)} = 0.1$, with $T_R = \min[T/10, 1000]$. This imposed by

$$\delta = \max\left[0.025, \frac{1}{1+0.004T}\right] \tag{4.3}$$

4.3.2 Label Frequency Estimation in Relational PU Data

We propose TIcER (Tree Induction for c Estimation in Relational data). The main difference with TIcE is that it learns a first-order logical decision tree using TILDE [8]. Each internal node splits on the formula which locally optimizes the gain ratio, considering the unlabeled examples as negative. The examples that satisfy the formula go to the left, the others to the right. Each node in the tree, therefore, specifies a subset of the data, and each subset implies a lower bound on the label frequency through Equation (4.2). The estimate for the label frequency is the maximal lower bound implied by the subsets:

$$\hat{c} = \max_{i \in \text{subsets}}\left[\frac{L_i}{T_i} - \frac{1}{2}\sqrt{\frac{1-\delta}{\delta T_i}}\right] \tag{4.4}$$

To prevent overfitting, k folds are used to induce the tree and estimate the label frequency on different datasets. With relational data, extra care should be taken that the data in different folds are not related to each other. The final estimate is the average of the estimates made in the different folds.

Table 4.1: **Characteristics of the datasets.**

Datasets	#Examples	Class 1 (#)	Class 2 (#)	# Folds
IMDB	268	Actor (236)	Director (32)	5
Mutagenesis	230	Yes (138)	No (92)	5
UW-CSE	278	Student (216)	Professor (62)	5
WebKB	922	Person (590)	Other (332)	4

4.4 Related Work

A few relational positive and unlabeled learning methods exist. The method proposed by Muggleton makes the separability assumption and searches for the smallest hypothesis which covers all the positive examples [83]. If the underlying concept is complicated, this is likely to overgeneralize. RelOCC is a positive and unlabeled classification method that incrementally learns a tree-based distance measure which measures the distance to the positive class [56]. The SHERLOCK system is also related because it learns rules from positive examples by evaluating the rules on their statistical relevance and significance, however, it does not utilize the unlabeled data [98].

Knowledge base completion is inherently a positive and unlabeled problem: all the examples that are already in the knowledge base are positive and all the additional facts that could be included are unlabeled [34]. However, many methods make a closed-world assumption when learning models from the original knowledge base and assume everything that is not present to be false [59, 109, 36, 85]. Recently, a new score function for evaluating knowledge base completion rules was proposed [133]. It approximates the true precision of a rule by assuming that the coverage of a rule is equal for labeled and unlabeled positives and by estimating the functionality of the relation.

Concurrently with this work, Blockeel (2017) proposed a relational PU method in the context of relational grounded language learning [7]. In their setting, the positive class can have a limited number of k subclasses and for each of them the SCAR assumption hold but do not necessary have the same label frequencies. A crucial difference with our work is that they make the assumption of separable classes while this is not necessary for our proposed methods.

Table 4.2: **Dataset complexities:** The complexities of the models that are trained on the complete and fully labeled datasets. For TILDE, the number of splits in the tree is shown. For Aleph, the number of rules is reported and the average rule length is given in parentheses.

Dataset	TILDE	Aleph Class1	Aleph Class2
IMDB	1	1 (1)	1 (1)
Mutagenesis	6	7 (2.29)	8 (2.25)
UW-CSE	3	5 (1)	5 (1.4)
WebKB	33	32 (3.19)	38 (2.08)

4.5 Experiments

Our goal is to evaluate if knowing the label frequency makes learning from relational PU data easier and if TIcER provides a good estimate of the label frequency. More specifically, we will answer the following questions:

Q1: Does c-adjusted TILDE, the proposed relational probabilistic classifier modification method, improve over classic TILDE when faced with PU data and how sensitive is it to the correctness of $\hat{c}$?

Q2: Does c-adjusted Aleph, the proposed relational score function modification method, improve over classic Aleph when faced with PU data and how sensitive is it to the correctness of $\hat{c}$?

Q3: How well does TIcER estimate the label frequency? In which cases does it perform better or worse?

Q4: How do label frequency adapted methods compare with Muggleton's PosOnly method [83]?[2]

4.5.1 Datasets

We evaluate our approach on four commonly used datasets for relational classification (Table 4.1). All datasets are available on Alchemy[3], except for Mutagenesis.[4] The classes of WebKB are disjunctive concepts. Person contains web pages from students, faculty and staff and Other contains web

[2]This is the only general PU learning method for relational data which existed at the time of publication of the article that this chapter is based on. PULSE was published simultaneously and could not be compared to when writing the article [7].

[3]http://alchemy.cs.washington.edu/data/

[4]http://www.cs.ox.ac.uk/activities/machlearn/mutagenesis.html

pages from departments, courses, and research projects. To get an intuition of the complexity of the concepts to be learned, Table 4.2 shows how big the TILDE and Aleph models are if they are trained on the complete dataset with labels for all examples. The datasets were converted to PU datasets by selecting some of the positive examples at random to be labeled. The labeling was done with frequencies $c \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Each has five different random labelings.

4.5.2 Methods

For our experiments we used the following PU classifiers:

- c-adjusted TILDE, as described in Section 4.2.1.
- c-adjusted Aleph, as described in Section 4.2.2.
- Aleph, taking unlabeled examples as negative ($\hat{c} = 1$)
- TILDE, taking unlabeled examples as negative ($\hat{c} = 1$)
- PosOnly: Muggleton's approach, implemented in Aleph [83].[5]

All classifiers, including TILDE when used for TIcER, use standard settings, with the exceptions of requiring PosOnly rules to cover at least two examples and allowing infinite noise and exploration in Aleph.

For the c-adjusted methods, an estimate of the label frequency c is required. This estimate $\hat{c}$ can be the correct label frequency c or the estimate obtained by our method TIcER. For the sensitivity experiments, the $\hat{c}$ is varied in $c \pm \Delta$ with $\Delta \in \{0, 0.05, 0.15, 0.25\}$. The naive baseline where unlabeled examples are considered to be negative can be seen as a special case of the c-adjusted methods with $\hat{c} = 1$.

k-fold cross-validation was applied for validation, i.e., $k - 1$ folds were used for learning the classifier and the other fold to evaluate it. The evaluation fold has access to all the labels. TIcER also needs folds for estimation, it used 1 fold for inducing a tree and the other $k - 2$ folds for bounding the label frequency.

The classifiers are compared using the F_1 score and the absolute error of the estimated cs are reported. No time comparison was executed because the time complexity of PosOnly is strictly lower than the time complexity of the combination of TIcER and then c-adjusted Aleph, this follows from PosOnly and c-adjusted Aleph having the same time complexity.

[5]`http://www.cs.ox.ac.uk/activities/machinelearning/Aleph/aleph`

4.5.3 c-adjusted TILDE: Performance and Sensitivity to $\hat{c}$

This section aims to answer **Q1**: Does c-adjusted TILDE, the proposed relational probabilistic classifier modification method, improve over classic TILDE when faced with PU data and how sensitive is it to the correctness of $\hat{c}$? To this end, TILDE was adjusted with different estimates for the label frequency that deviate from the true label frequencies with fixed values Δ. The adjusted versions are compared to the naive method which considers unlabeled examples as negative, i.e., $\hat{c} = 1$. The results are presented in Figure 4.4.

As expected, taking the label frequency into account improves the classifier. A striking observation is that overestimates of the label frequency c can severely degrade performance, while underestimates may even improve performance. This is because of the modification method: only the leaf probabilities are altered. Therefore, an underestimate makes leaves with at least one labeled example more likely to classify instances as positive, while leaves without any labeled examples will always classify instances as negative.

4.5.4 c-adjusted Aleph: Performance and Sensitivity to $\hat{c}$

This section aims to answer **Q2**: Does c-adjusted Aleph, the proposed relational score function modification method, improve over classic Aleph when faced with PU data and how sensitive is it to the correctness of $\hat{c}$? To this end, Aleph was adjusted with different estimates for the label frequency that deviate from the true label frequencies with fixed values Δ. The adjusted versions are compared to the naive method which considers unlabeled examples as negative, i.e., $\hat{c} = 1$. The results are presented in Figure 4.5.

Taking the label frequency into account drastically improves the classifier: the F_1 score barely drops when the label frequency decreases. In most cases, a reasonable approximation of the label frequency yields an equivalent performance to using the true label frequency. Two exceptions are 1) when there are few positive examples in the fully labeled dataset, and 2) when the target concept is very complex. But even in these cases, the performance does not suffer that much, especially when compared to simply assuming that all unlabeled examples belong to the negative class.

4.5.5 TIcER Evaluation

This section aims to answer **Q3**: How well does TIcER estimate the label frequency? In which cases does it perform better or worse? To this end, TIcER was used to estimate the label frequency c and compared to the true label frequency in all the training folds of all the datasets. Based on the theory, it is expected that TIcER works well when it can find subsets in the dataset that are purely positive and contain a sufficient number of examples. To check this, the maximal proportion of true positives over all the used subsets was recorded for each setting. We could look at the size of this purest subset to check if a large subset is found. However, the purest subset could be very small and another subset that is almost as pure could be very big. Therefore, we recorded the largest subset that is at least 90% as pure as the purest subset. Figure 4.6 shows the absolute error $|\hat{c} - c|$, purity $\max(P_i/T_i)$ of the purest subset i and size T_j of the largest subset j with purity close to that of the purest subset, for different label frequencies c. Figures 4.7 and 4.8 compare the performance of TILDE and Aleph respectively when adjusted with the TIcER estimate, the true label frequency and without adjusting it.

TIcER gives reasonable results most of the time. The experiments confirm our expectations: it performs worse when it fails to find subsets with a high ratio of positive examples or when the subsets contain few examples. Although the estimates are not perfect, they still can improve the performance of TILDE and Aleph. Most of the time, the performance using the estimated label frequency is close to the performance of using the true label frequency. TILDE even gives better results with the estimate than with the true label frequency, this is because TIcER estimates the label frequency by looking for the maximum lower bound and hence tends to give underestimates. Aleph performs worse for the cases where it is sensitive to the label frequency. This is notably the case for UW-CSE.

4.5.6 Method Comparison

This section aims to answer **Q4**: How do label frequency adapted methods compare with Muggleton's PosOnly method? To this end, we compare TIcER-adjusted TILDE and Aleph with PosOnly. The results are presented in Figure 4.9.

The label frequency indeed makes learning from PU data easier, as it gives similar or better results than PosOnly. It is especially interesting that for the most complex dataset (WebKB) PosOnly is outperformed. The label frequency-based methods are only outperformed when there are exceptionally few labeled examples or no close-to-pure subsets in the data. Because using the label frequency adjust existing methods, it can benefit from any advancements and optimizations made to traditional classifiers.

4.6 Conclusions

For propositional PU classification tasks, it has long been known that knowing the label frequency greatly simplifies the problem. We transferred this idea to the relational classification tasks and make the same conclusion here. Adjusting established classifiers such as TILDE and Aleph in very simple ways perform equally well as Muggleton's PosOnly method. For the most complex dataset, PosOnly is even outperformed. Because only small adjustments in traditional classifiers are needed, this PU classification method will improve as traditional classifiers improve.

When the label frequency is unknown, it needs to be estimated from the positive and unlabeled data. We propose a TIcER, a relational version of TIcE, which employs decision trees to find pure positive subsets in the data and uses these to estimate the label frequency. This method works well when it can find highly positive subsets of the data that contain enough examples.

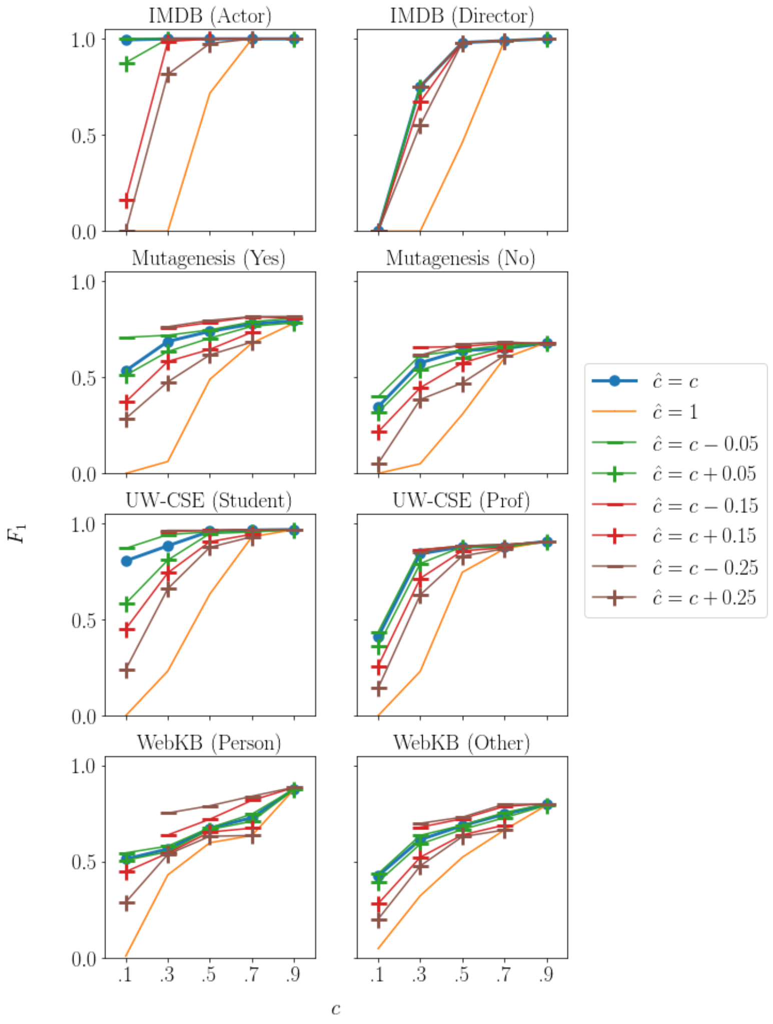

Figure 4.4: **TILDE sensitivity to c.** Taking the label frequency into account clearly improves the classifier. The F_1 does decrease as fewer labeled examples are provided. It is striking that underestimates and overestimates of the label frequency have very different effects on the performance. c-adjusted TILDE is very sensitive to overestimates, but not to underestimates. In fact, in some cases it even benefits from underestimates! When the label frequency is high, most of the positive examples are observed and therefore all the methods work well. Invalid label frequency estimates ($\hat{c} < 0 and \hat{c} > 1$) are omitted.

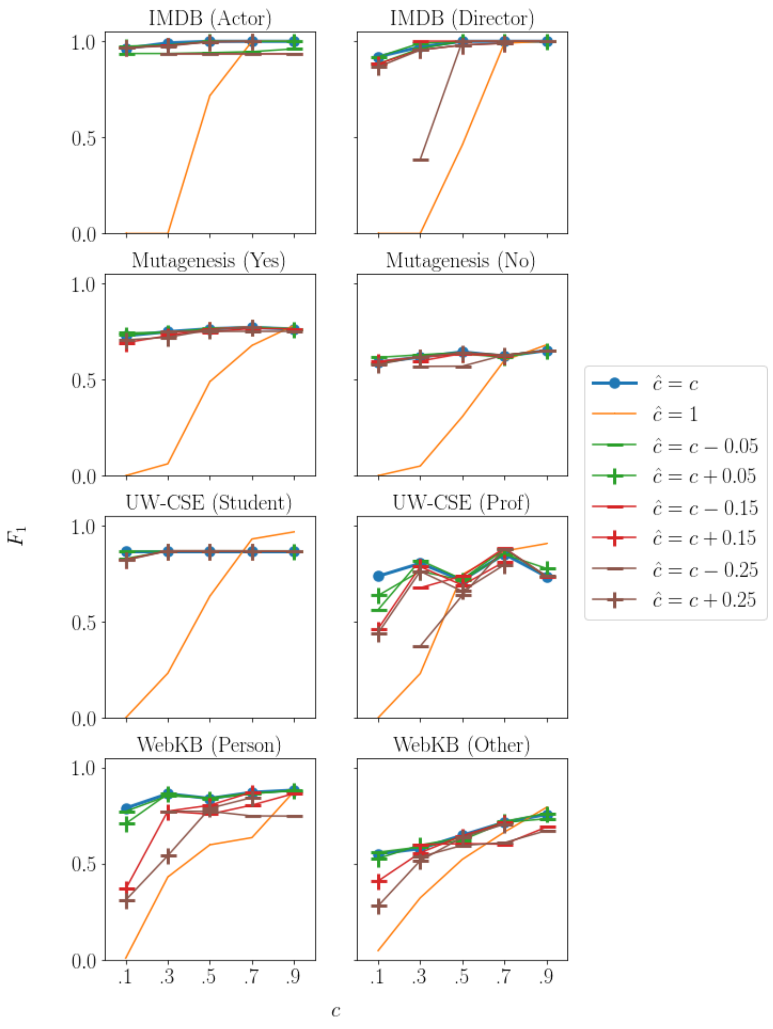

Figure 4.5: **Aleph sensitivity to c.** Considering the label frequency clearly substantially improves the classifier: as the number of labeled examples decreases, the F_1 score barely drops. Aleph is not very sensitive to the label frequency c, except when there are few positive examples to start with (IMDB-director and UW-CSE-Prof) or when the target concept is complex (WebKB). Even in these cases, a bad estimate for the label frequency is better than taking the unlabeled examples as negative ($\hat{c} = 1$). When the label frequency is high, most of the positive examples are observed and therefore all the methods work well. Invalid label frequency estimates ($\hat{c} < 0 and \hat{c} > 1$) are omitted.

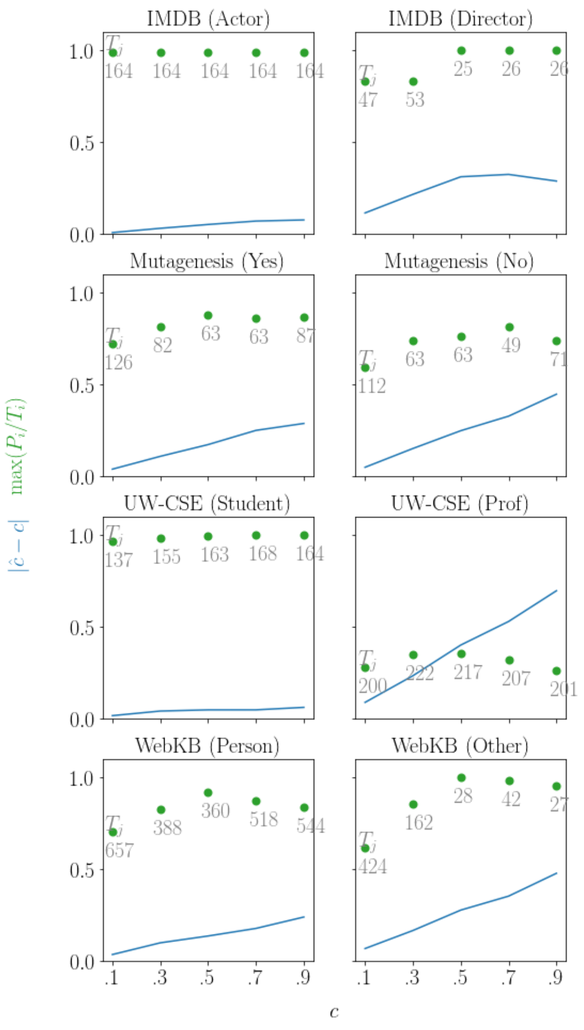

Figure 4.6: **Label Frequency Estimates.** The estimate is expected to be good if a large enough subset with a high proportion of positives was found, this is confirmed by the experiments. For example, the worst results, for UW-CSE (Prof), are explained by the low positive proportions. Subset i is the subset with the maximum purity $max(P_i/T_i)$ and subset j of size T_j is the largest subset that is at least 90% as pure as subset i. For Mutagenesis and WebKB, we observe that a smaller label frequency results in larger T_j. This is not because the purest subset i is less pure than those with a larger label frequency, therefore larger subsets with that low purity can be found. Note that the error $|\hat{c} - c|$ grows as c gets larger. This is because $\hat{c}$ is usually an underestimate and therefore the error is usually smaller than c.

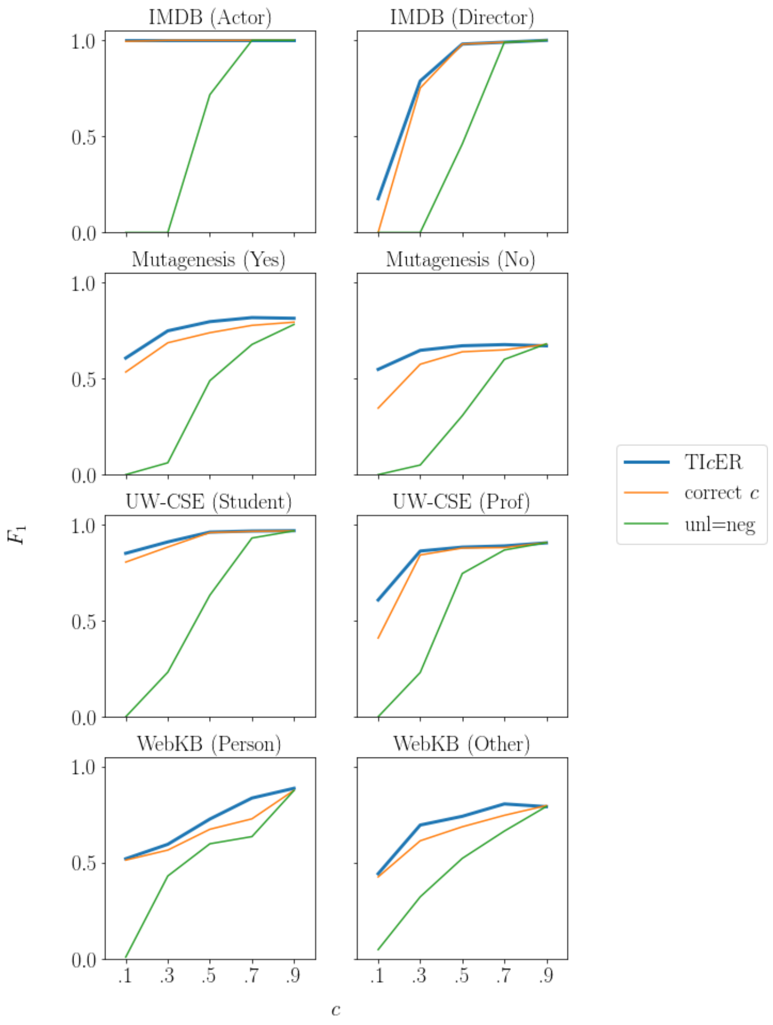

Figure 4.7: **TIcER-adjusted TILDE**. Adjusting TILDE with the TIcER estimate gives very similar results to adjusting it with the true label frequency, sometimes even better. This is explained by TIcER giving underestimates.

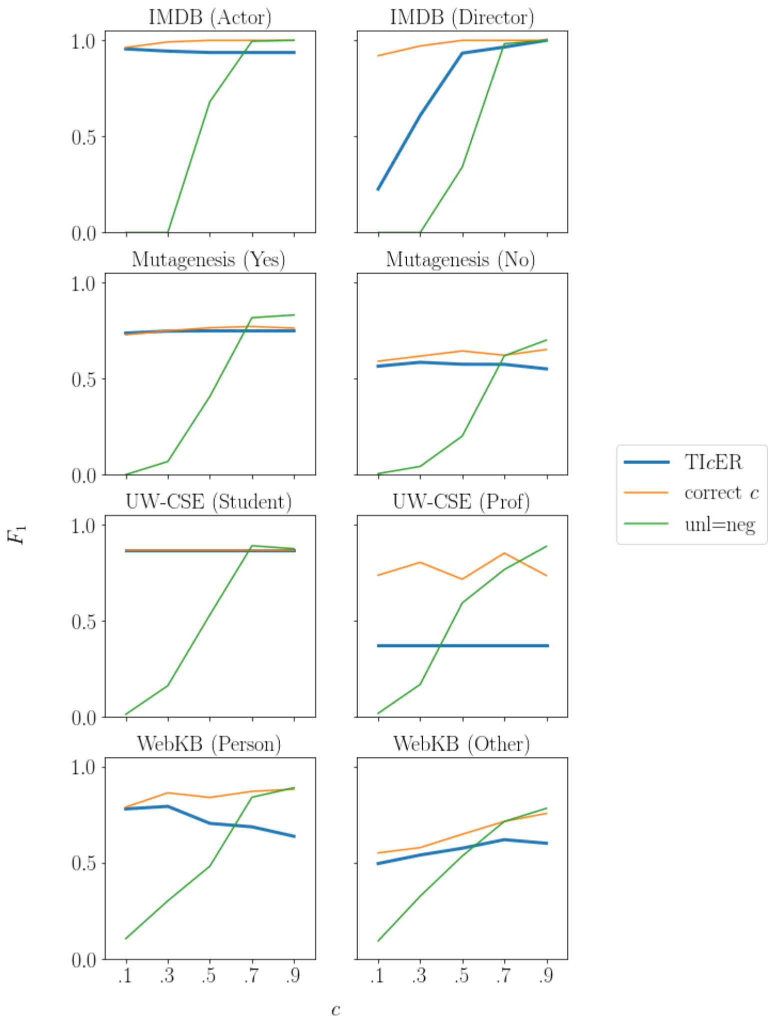

Figure 4.8: **TIcER-adjusted Aleph**. UW-CSE-Prof is doubly problematic because it is both sensitive to the label frequency and the most difficult dataset for TIcER.

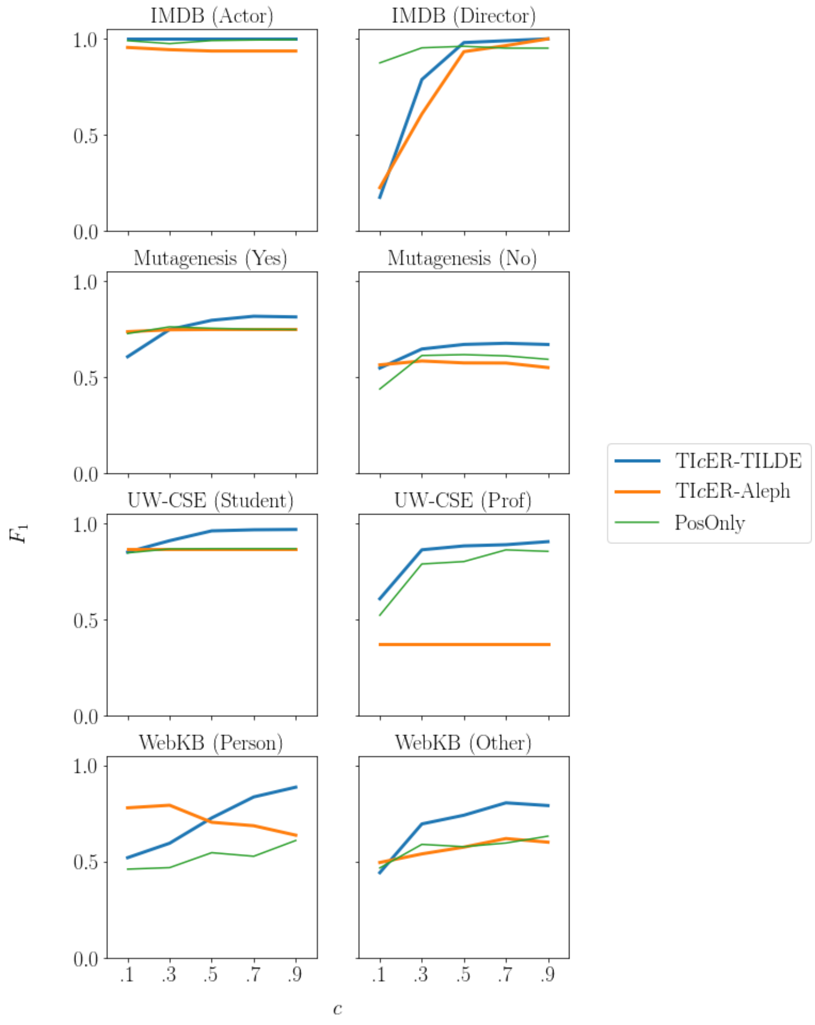

Figure 4.9: **Comparison of methods.** In most cases, the methods give similar results, which supports the claim that using the label frequency simplifies PU learning, also in the relational domain. It is interesting to see that for the most complex dataset (WebKB) Muggleton's PosOnly is outperformed. The only situations where any of the c-adjusted methods perform significantly worse than PosOnly are those with an extremely small number of labeled examples (IMDB-Director with low label frequency) or when the estimate is extremely bad because of the lack of pure positive subsets (UW-CSE)

Chapter 5

Beyond the Selected Completely At Random Assumption

Most positive and unlabeled data is subject to selection biases. The labeled examples can, for example, be selected from the positive set because they are easier to obtain or more obviously positive. The commonly made *Selected Completely At Random (SCAR)* assumption assumes that no such biases are present and is therefore often an unreasonable assumption to make for real-world data. This chapter investigates how learning can be enabled if selection biases are present.

Problem Statement

The problem that this chapter addresses is the following:

Given a PU dataset $(\mathbf{x}, \mathbf{y}, \mathbf{s})$, where the class $\mathbf{y}$ is not observed, none of the negative examples are labeled $\Pr(s = 1|y = 0, x) = 0$, and the labeled examples were selected from the positive set according to a labeling mechanism that may or may not be known,

Train a binary classifier that distinguishes between positive and negative examples. There are two cases: case 1) the labeling mechanism is provided, case 2) no labeling mechanism is provided.

To the best of our knowledge, all existing methods exists for learning from positive and unlabeled data make either the SCAR or the separability assumption. By enabling PU learning under realistic assumptions, this chapter opens up a whole new range of possibilities.

Contributions of this Chapter

This chapter addresses the problem statement by making five main contributions.

The first contribution of this chapter is to propose two new classes of assumptions for learning from positive and unlabeled data: Selected At Random (SAR) and Selected Not At Random (SNAR). Like the SCAR assumption, they have counterparts in the missing data literature. The rest of the chapter focuses on the SAR assumption.

The second contribution is a technique to learn with a known SAR labeling mechanism, which is theoretically analyzed in an empirical risk minimization framework.

The third contribution is a practical algorithm for learning from SAR PU data when the labeling mechanism is unknown.

The fourth contribution is an extensive empirical evaluation. We show that for SAR PU data, our approaches result in improved performance over making the standard SCAR assumption.

The fifth contribution is the availability of the source code of this work (algorithms and SAR PU data generation) on `https://dtai.cs.kuleuven.be/software/sar`.

The content of this chapter is based on the following paper:

Bekker, J., and Davis, J. Beyond the Selected Completely At Random Assumption for Learning from Positive and Unlabeled Data. In *arXiv:1809.03207* (2018d).

5.1 Background on Causal Inference

This chapter uses insights from causal inference, in particular, we borrow the notion of the propensity score from this field. Therefore, we review the basics of causal inference and the propensity score first.

Causal inference aims to discover causal effects from data. Examples of causal effects are the effectiveness of a drug against a disease, or the effect of a job-training on an individual's employment prospects. The subject of which the causal effect is studied, i.e.,the drug or the job-training, is called the *treatment*. In this dissertation, we follow the potential outcomes framework [47]. The potential outcome of the active treatment is the outcome that would happen if the treatment was applied and the potential outcome of the control treatment is the outcome that would happen if the treatment was not applied. The causal effect of a treatment is defined as the difference in potential outcomes.

The difficulty of causal inference, is that only one treatment (active or control) can be applied for each unit and hence, no direct comparison is possible. The simplest way to deal with this is to do a randomized experiment with two groups: the treatment and the control group. The two groups represent the same distribution and the treatment groups gets the treatment while the control group does not. The causal effect can then be estimated as the average difference in outcome for the two groups.

Randomized experiments are not always possible, either because the treatment is out of our control, or if, for some reason, we are inclined to give the treatment to certain units, e.g., the patients that are expected to die without the drug. In this case, the *assignment mechanism*, which stochastically decides which units get the treatment, needs to be taken into account. The probability for unit x of getting the treatment$y$ is called the *propensity score* $e(x) = \Pr(y = 1|x)$. A non-uniform assignment mechanism implies that the causal effect can no longer be estimated as the average difference in outcome for the two complete groups. Instead, a weighted average is considered, where each treated unit is weighted inversely with its propensity score $\frac{1}{e(x)}$ and each control unit is weighted with $\frac{1}{1-e(x)}$.

5.2 Labeling Mechanisms for PU Learning

The mechanism behind selecting positive examples to be labeled is called the labeling mechanism. To date, PU learning has largely focused on the SCAR setting. However, clearly labels are not missing completely at random in most real-world problems. For example, the data included in automatically constructed KBs is biased several ways. One is that it is learned from Web data, and only certain types of information appear on the Web (e.g., it is easier to find text about high-level professional sports teams than low-level ones). Two, the algorithms used to extract information from the Web employ heuristics to ensure that only information that is likely to be accurate (e.g., by using redundancy) is included in the KB. Similarly, biases arise when people decide to like items online, bookmark web pages, or subscribe to mail lists. Therefore we believe it is important to consider and study other labeling mechanisms.

When Elkan and Noto (2008) first formalized the SCAR assumption, they noted the similarity of the PU setting to the general problem of learning in the presence of missing data [30]. Specifically, they noted that the SCAR assumption is somewhat analogous with the missing data mechanism called *Missing Completely At Random (MCAR)* [96]. Apart from MCAR, the two other classes of missing data mechanisms are *Missing At Random (MAR)* and *Missing Not At Random (MNAR)*. To complete this analogy, we propose the following corresponding classes of PU labeling mechanisms:

SCAR *Selected Completely At Random*: The labeling mechanism does not depend on the attributes of the example, nor on the probability of the example being positive, i.e., each positive example has the same probability to be labeled.

SAR *Selected At Random*: The labeling mechanism depends on the values of the attributes of the example, but given the attribute values it does not depend on the probability of the example being positive.

SNAR *Selected Not At Random*: All other cases: The labeling mechanism depends on the real probability of this example being positive, even given the attribute values.

There is one very important difference between PU labeling mechanisms and missingness mechanisms in that the labeling always depends on the class value: only positive examples can be selected to be labeled. According to the missingness taxonomy, all PU labeling mechanisms are therefore MNAR. SNAR is a peculiar class because it depends on the real class probability, while the class needs to be positive by definition. The class probability refers to the probability of an identical instance to this one being positive. Consider, for example, the problem of classifying pages as interesting. If a pages is moderately interesting to you, some days you might say like it while other days that you do not. The labeling mechanism in this case could depends on how much you like them and therefore on the instance's class probability.

5.3 Learning with SAR Labeling Mechanisms

In this chapter, we focus on SAR labeling mechanisms, where the key question is how can we enable learning from SAR PU data? Our key insight is that the labeling mechanism is also related to the notion of a propensity score from causal inference [47]. In causal inference, the propensity score is an instance-specific probability, based on a set of an example's attributes, that it is assigned to the treatment or control group. We use an analogous idea and define the propensity score as the labeling probability for positive examples:

Definition 6 (Propensity Score)**.** *The propensity score, denoted $e(x)$, for x is the label assignment probability for positive instances with $X = x$,*

$$e(x) = \Pr(s = 1|y = 1, x)$$

A crucial difference with the propensity score from causal inference is that our score is conditioned on the class being positive.

We propose a new method for incorporating the propensity score when learning in a PU setting, by using the propensity scores to reweight the data. In causal inference, inverse-propensity-scoring is a standard method where the examples are weighted with the inverse of their propensity score [70, 47, 97]. This cannot be applied when working with positive and unlabeled data because we have zero probability for labeling negative examples. But we can do a different kind of weighting. The insight is that for each labeled example $(x_i, s = 1)$ that has a propensity score e_i, there are expected to be $\frac{1}{e_i}$ positive examples, of which $\frac{1}{e_i} - 1$ did not get selected to be labeled. This insight can be used in algorithms that use counts, to estimate the correct count from the observed positives and their respective propensity scores. In general, this can be formulated as learning with negative weights: every labeled example gets a weight $\frac{1}{e_i}$ and for every labeled example a negative example is added to the dataset that gets a negative weight $1 - \frac{1}{e_i}$. Every unlabeled example is added as a negative example with weight 1. A labeled example with a low label frequency will thus be considered as adding many positive examples and removing many negative ones. Because the label frequency was low, it is only likely that it got selected if there are many similar unlabeled examples. Therefore, the weighting essentially moves those similar unlabeled examples to the positive set.

We now provide a theoretical analysis of the propensity-weighted method, to characterize its appropriateness. We consider two cases: (1) when we know the true propensity scores and (2) when we must estimate them from data.

5.3.1 Case 1: True Propensity Scores Known

Standard evaluation measures, such as Mean Absolute Error (MAE), Mean Square Error (MSE) and log loss, can be formulated as follows:

$$R(\hat{\mathbf{y}}|\mathbf{y}) = \frac{1}{n}\sum_{i=1}^{n} y_i\delta_1(\hat{y}_i) + (1-y_i)\delta_0(\hat{y}_i),$$

with n the size of $\mathbf{y}$ and $\hat{\mathbf{y}}$ and $\delta_y(\hat{y})$ represents the cost for predicting $\hat{y}$ when the class is y, for example:

$$\begin{aligned}\text{MAE}: \delta_y(\hat{y}) &= |y-\hat{y}|,\\ \text{MSE}: \delta_y(\hat{y}) &= (y-\hat{y})^2,\\ \text{Log Loss}: \delta_1(\hat{y}) &= -\ln\hat{y}\ ,\ \ \delta_0(\hat{y}) = -\ln(1-\hat{y}).\end{aligned}$$

We can formulate propensity-weighted variants estimator of these cost functions as follows:

Definition 7 (Propensity-Weighted Estimator)**.** *Given propensity the scores* $\mathbf{e}$ *and PU labels* $\mathbf{s}$*, the propensity weighted estimator of* $R(\hat{\mathbf{y}}|\mathbf{y})$ *is*

$$\hat{R}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s}) = \frac{1}{n}\sum_{i=1}^{n} s_i\left(\frac{1}{e_i}\delta_1(\hat{y}_i) + (1-\frac{1}{e_i})\delta_0(\hat{y}_i)\right) + (1-s_i)\delta_0(\hat{y}_i),$$

where $\mathbf{y}$ *and* $\hat{\mathbf{y}}$ *are vectors of size* n *containing, respectively, the true labels and predicted labels.* $\delta_y(\hat{y})$ *is the cost for predicting* $\hat{y}$ *when the true class is* y*.*

This estimator is unbiased:

$$\begin{aligned}&\mathbb{E}[\hat{R}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s})])\\ &= \frac{1}{n}\sum_{i=1}^{n} y_ie_i\left(\frac{1}{e_i}\delta_1(\hat{y}_i) + (1-\frac{1}{e_i})\delta_0(\hat{y}_i)\right) + (1-y_ie_i)\delta_0(\hat{y}_i)\\ &= \frac{1}{n}\sum_{i=1}^{n} y_i\delta_1(\hat{y}_i) + (1-y_i)\delta_0(\hat{y}_i) = R(\hat{\mathbf{y}}|\mathbf{y})\end{aligned}$$

To characterize how much the estimator can vary from the expected value, we provide the following bound:

Proposition 1 (Propensity-Weighted Estimator Bound)**.** *For any predicted classes* $\hat{\mathbf{y}}$ *and real classes* $\mathbf{y}$ *of size* n*, with probability* $1-\eta$*, the propensity-weighted estimator* $\hat{R}(\hat{\mathbf{y}}|\mathbf{s},\mathbf{e})$ *does not differ from the true evaluation measure* $R(\hat{\mathbf{y}}|\mathbf{y})$ *more than*

$$|\hat{R}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s}) - R(\hat{\mathbf{y}}|\mathbf{y})| \leq \sqrt{\frac{\delta_{max}^2 \ln\frac{2}{\eta}}{2n}},$$

with δ_{max} *the maximum absolute value of cost function* δ_y*.*

Proof. All the examples are selected to be labeled independently from each other. Therefore, the weighted costs of the examples are independent random variables. As a result, the Hoeffding inequality can be applied [42]:

$$\Pr(|\hat{R}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s}) - \mathbb{E}[\hat{R}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s})]| \geq \epsilon) \leq 2\exp\left(\frac{-2n\epsilon^2}{\delta_{\max}^2}\right)$$

$$\Leftrightarrow \Pr(|\hat{R}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s}) - R(\hat{\mathbf{y}}|\mathbf{y})| \geq \epsilon) \leq 2\exp\left(\frac{-2n\epsilon^2}{\delta_{\max}^2}\right)$$

By setting defining the right-hand side of the inequality to η, the bound e can be calculated in terms of η:

$$\eta = 2\exp\left(\frac{-2n\epsilon^2}{\delta_{\max}^2}\right)$$

$$\epsilon = \sqrt{\frac{\delta_{\max}^2 \ln\frac{2}{\eta}}{2n}}.$$

□

The propensity-weighted estimator can be used as the risk for Expected Risk Minimization (ERM), which searches for a model in the hypothesis space $\mathcal{H}$ by minimizing the risk:

$$\hat{\mathbf{y}}_{\hat{R}} = \text{argmin}_{\hat{\mathbf{y}}\in\mathcal{H}}\hat{R}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s})$$

The following proposition characterizes how much the estimated risk for hypothesis $\hat{\mathbf{y}}_{\hat{R}}$ can deviate from its true risk.

Proposition 2 (Propensity-Weighted ERM Generalization Error Bound)**.** *For a finite hypothesis space $\mathcal{H}$, the difference between the propensity-weighted risk of the empirical risk minimizer $\hat{\mathbf{y}}_{\hat{R}}$ and its true risk is bounded, with probability $1-\eta$, by:*

$$R(\hat{\mathbf{y}}_{\hat{R}}|\mathbf{y}) \leq \hat{R}(\hat{\mathbf{y}}_{\hat{R}}|\mathbf{e},\mathbf{s}) + \sqrt{\frac{\delta_{max}^2 \ln \frac{|\mathcal{H}|}{\eta}}{2n}}$$

Proof.

$$\begin{aligned}
\Pr\left(\hat{R}(\hat{\mathbf{y}}_{\hat{R}}|\mathbf{e},\mathbf{s}) - R(\hat{\mathbf{y}}_{\hat{R}}|\mathbf{y}) \geq \epsilon\right) &\leq \Pr\left(\max_{\hat{\mathbf{y}}_i}(\hat{R}(\hat{\mathbf{y}}_i|\mathbf{e},\mathbf{s}) - R(\hat{\mathbf{y}}_i|\mathbf{y})) \geq \epsilon\right) \\
&= \Pr\left(\bigvee_{\hat{\mathbf{y}}_i}(\hat{R}(\hat{\mathbf{y}}_i|\mathbf{e},\mathbf{s}) - R(\hat{\mathbf{y}}_i|\mathbf{y})) \geq \epsilon\right) \\
&\quad \# \textit{ Boole's inequality} \\
&\leq \sum_{i=1}^{|\mathcal{H}|} \Pr\left(\hat{R}(\hat{\mathbf{y}}_i|\mathbf{e},\mathbf{s}) - R(\hat{\mathbf{y}}_i|\mathbf{y}) \geq \epsilon\right) \\
&\quad \# \textit{ Hoeffding's inequality} \\
&\leq |\mathcal{H}| \cdot \exp\left(\frac{-2n\epsilon^2}{\delta_{\max}^2}\right) = \eta
\end{aligned}$$

Solve for ϵ:

$$\epsilon = \sqrt{\frac{\delta_{\max}^2 \ln \frac{|\mathcal{H}|}{\eta}}{2n}}$$

□

5.3.2 Case 2: Propensity Scores Estimated from Data

Often the exact propensity score is unknown, but we have an estimate $\hat{e}$ of it. In this case, the bias of the propensity-weighted estimator is:

Proposition 3 (Propensity-Weighted Estimator Bias)**.**

$$bias(\hat{R}(\hat{\mathbf{y}}|\hat{\mathbf{e}},\mathbf{s})) = \frac{1}{n}\sum_{i=1}^{n} y_i(1-\frac{e_i}{\hat{e}_i})\,(\delta_1(\hat{y}_i)-\delta_0(\hat{y}_i))$$

Proof.

$$\text{bias}(\hat{R}(\hat{\mathbf{y}}|\hat{\mathbf{e}},\mathbf{s})) = R(\hat{\mathbf{y}}) - \mathbb{E}[\hat{R}(\hat{\mathbf{y}}|\hat{\mathbf{e}},\mathbf{s})]$$

$$\begin{aligned}\mathbb{E}[\hat{R}(\hat{\mathbf{y}}|\hat{\mathbf{e}},\mathbf{s})] &= \frac{1}{n}\sum_{i=1}^{n} y_i e_i\left(\frac{1}{\hat{e}_i}\delta_1(\hat{y}_i) + (1-\frac{1}{\hat{e}_i})\delta_0(\hat{y}_i)\right) \\ &\quad + (1-y_ie_i)\delta_0(\hat{y}_i) \\ &= \frac{1}{n}\sum_{i=1}^{n} y_i\frac{e_i}{\hat{e}_i}\delta_1(\hat{y}_i) + (1-y\frac{e_i}{\hat{e}_i})\delta_0(\hat{y}_i)\end{aligned}$$

$$\begin{aligned}\text{bias}(\hat{R}(\hat{\mathbf{y}}|\hat{\mathbf{e}},\mathbf{s})) &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i\frac{e_i}{\hat{e}_i}\right)\delta_1(\hat{y}_i) + \left(1-y_i-1+y\frac{e_i}{\hat{e}_i}\right)\delta_0(\hat{y}_i) \\ &= \frac{1}{n}\sum_{i=1}^{n} y_i\left(1-\frac{e_i}{\hat{e}_i}\right)\delta_1(\hat{y}_i) - y_i\left(1-\frac{e_i}{\hat{e}_i}\right)\delta_0(\hat{y}_i) \\ &= \frac{1}{n}\sum_{i=1}^{n} y_i\left(1-\frac{e_i}{\hat{e}_i}\right)(\delta_1(\hat{y}_i)-\delta_0(\hat{y}_i))\end{aligned}$$

□

From the bias, we draw three conclusions. First,the term y_i shows that scores only need to be accurate for positive examples. Second, the term $(\delta_1(\hat{y}_i) - \delta_0(\hat{y}_i))$ shows that an incorrect propensity score has a larger impact when the predicted classes have more extreme values (i.e., tend towards zero or one), because then this term will be larger. Third, An underestimate $\hat{e}_i$ is expected to result in a larger bias than an overestimate of the same size. Both the terms

$\left(1 - \frac{e_i}{\hat{e}_i}\right)$ and $(\delta_1(\hat{y}_i) - \delta_0(\hat{y}_i))$ are expected to be larger in absolute values for an under estimate. The former is verified trivially by filling in equal-sized over and underestimates (both cumulative $\hat{e}(x) = e(x) \pm \Delta$ or multiplicative $\hat{e}(x) = \Delta e(x)$ and $\hat{e}(x) = e(X)/\Delta$ work). The latter is understood by seeing that lower propensity scores result in learning models that estimate the positive class to be more prevalent than it is, and hence predict more extreme values that are closer to one for positive examples.

Side Note on Sub-Optimality of Expected Risk

Another method that one might be inclined to use when incorporating the propensity score is to minimize the expected risk, which is defined as

$$\begin{aligned}\hat{R}_{\text{exp}}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s}) = \mathbb{E}_{\mathbf{y}|\mathbf{e},\mathbf{s},\hat{\mathbf{y}}}\left[R(\hat{\mathbf{y}}|\mathbf{y})\right] &= \mathbb{E}_{\mathbf{y}|\mathbf{e},\mathbf{s},\hat{\mathbf{y}}}\left[\frac{1}{n}\sum_{i=1}^{n} y_i\delta_1(\hat{y}_i) + (1-y_i)\delta_0(\hat{y}_i)\right] \\ &= \frac{1}{n}\sum_{i=1}^{n}\Pr(y_i=1|e_i,s_i,y_i)\delta_1(\hat{y}_i) \\ &\qquad + (1-\Pr(y_i=1|e_i,s_i,y_i))\delta_0(\hat{y}_i).\end{aligned}$$

With conditional probabilities $\Pr(y_i = 1|e_i, s_i, y_i)$:

$$\begin{aligned}\Pr(y_i=1|e_i,s_i,\hat{y}_i) &= s\Pr(y_i=1|e_i,s_i=1,\hat{y}_i) \\ &\qquad + (1-s)\Pr(y_i=1|e_i,s_i=0,\hat{y}_i) \\ &= s + (1-s)\frac{\Pr(y_i=1|\hat{y}_i,e_i)\Pr(s=0|\hat{y}=1,\hat{y}_i,e_i)}{\Pr(s=0|\hat{y},e_i)} \\ &= s + (1-s)\frac{\hat{y}_i(1-\Pr(s=1|\hat{y}=1,\hat{y}_i,e_i))}{1-\Pr(s=1|\hat{y},e_i)} \\ &= s + (1-s)\frac{\hat{y}_i(1-e_i)}{1-\hat{y}_ie_i},\end{aligned}$$

this results in

$$\hat{R}_{\text{exp}}(\hat{\mathbf{y}}|\mathbf{e},\mathbf{s}) = \frac{1}{n}\sum_{i=1}^{n}\left(s_i + (1-s_i)\frac{\hat{y}_i(1-e_i)}{1-\hat{y}_ie_i}\right)\delta_1(\hat{y}_i) + (1-s_i)\frac{1-\hat{y}_i}{1-\hat{y}_ie_i}\delta_0(\hat{y}_i).$$

Training the model by minimizing the expected risk $\hat{R}_{\text{exp}}$ selects the hypothesis $\hat{\mathbf{y}}_{\hat{R}_{\text{exp}}} = \text{argmin}_{\hat{\mathbf{y}} \in \mathcal{H}} \hat{R}_{\text{exp}}(\hat{\mathbf{y}}|\mathbf{e}, \mathbf{s})$.

However, the expected risk is not an unbiased estimator of the true risk and as a result, $\hat{\mathbf{y}}_{\hat{R}_{\text{exp}}}$ is not expected to be the best hypothesis. In fact, the hypothesis of always predicting the positive class $\forall_i : \hat{y}_i = 1$ always has an expected risk $\hat{R}_{\text{exp}}(\hat{\mathbf{y}}|\mathbf{e}, \mathbf{s}) = 0$. To illustrate the problem, Figure 5.1 shows an example of the expected risk for different hypotheses for a simple learning task, where the model consists of only one parameter: the class prior. With any propensity score, the preferred hypothesis is the one with class prior $\alpha = 1$, i.e. the one that always predict $\hat{y} = 1$.

More generally, any non-probabilistic hypothesis, i.e. one that only predicts $\hat{y} = 0$ or $\hat{y} = 1$, that predicts the positive class for all the labeled examples has an expected risk $\hat{R}_{\text{exp}}(\hat{\mathbf{y}}|\mathbf{e}, \mathbf{s}) = 0$. As a consequence, if the classes are separable, the correct hypothesis will have an expected risk $\hat{R}_{\text{exp}}(\hat{\mathbf{y}}_{\text{correct}}|\mathbf{e}, \mathbf{s}) = 0$, but it will not be preferred over other non-probabilistic hypotheses which correctly classify the labeled examples, this is illustrated in Figure 5.2.

5.4 Learning under the SAR Assumption

If the propensity scores for all examples are known (i.e., the exact labeling mechanism is known), they can be directly incorporated into the learning algorithm. However, it is more likely that they are unknown. Therefore, this section investigates how to permit learning in the SAR setting when the exact propensity scores are unknown. We discuss two such settings. The first is interesting from a theoretical perspective and the second from a practical perspective.

5.4.1 Reducing SAR to SCAR

Learning the propensity scores from positive and unlabeled data requires making additional assumptions: if any arbitrary instance can have any propensity score, then it is impossible to know if an instance did not get labeled because of a low propensity score or low class probability. Therefore, the propensity score needs to depend on fewer attributes than the final classifier [47]. A simple way to accomplish this is to assume that the propensity function only depends on a subset of the attributes: the *propensity attributes* $x_e \in x$:

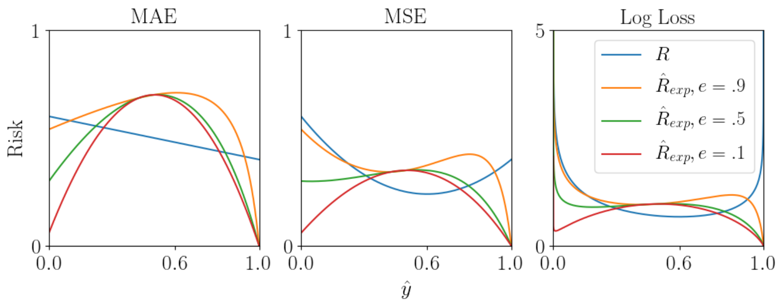

Figure 5.1: **Illustration of sub-optimality of expected risk.** Here, there is no distinction between examples $\forall_{i,j} : x_i = x_j$, 60% of the examples are positive and all examples have the same propensity score $\forall_i : e_i = e$. In this case, learning a model means estimating the class prior $\hat{y} = \hat{\alpha}$, which is expected to be $\alpha = 0.6$. MSE and log loss are proper loss functions and can find this. However, for all propensity scores $e \in (0, 1]$, the expectation of both metrics is minimized at $\hat{y} = 1$.

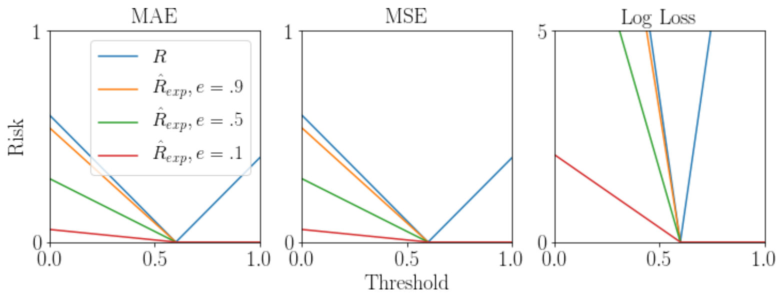

Figure 5.2: **Illustration of expected risk for deterministic hypotheses with different thresholds.** Here, the examples have one attribute with values $x \in [0, 1]$, all the examples with $x_i < 0.6$ are positive and all other negative. All examples have the same propensity score $\forall_i : e_i = e$. In this case, learning a model means estimating the optimal threshold for x to separate the positive examples from the negative examples. The expected risks are zero for the true threshold, as well as any other threshold that classifies the labeled examples as positive.

$$\Pr(s = 1|y = 1, x) = \Pr(s = 1|y = 1, x_e)$$

$$e(x) = e(x_e).$$

Often, this is not an unrealistic assumption. It is trivially true if the labeling mechanism does not have access to all attributes (e.g., because some were collected later). It may also arise if a labeler cannot interpret some attributes (e.g., raw sensor values) or only uses the attributes that are known to be highly correlated with the class.

To see why this can be a sufficient assumption for learning in a SAR setting, consider the case where the propensity attributes x_e have a finite number of configurations, which is true if these attributes are all discrete. In this case, it is possible to partition the data into strata, with one strata for each configuration of x_e. Within a strata, the propensity score is a constant (i.e., all positive examples have the same propensity score) and can thus be determined using standard SCAR PU learning techniques. Note that, as discussed in the preliminaries, the SCAR assumption alone is not enough to enable learning from PU data, and hence one of the additional assumptions [6, 99, 24, 94, 3] must be made.

Reducing SAR to SCAR is suboptimal in practice as it does not work if x_e contains a continuous variable. Even for the discrete case, the number of configurations grows exponential as the size of x_e increases. Furthermore, information is lost by partitioning the data. Some smoothness of the classifier over the propensity attributes is expected, but this is not encouraged when learning different classifiers for each configuration. Similarly the propensity score itself is expected to be a smooth function over the propensity variables.

5.4.2 EM for Propensity Estimation

In practice, due to the problems with reducing the SAR to the SCAR case, it is best to jointly search for a classifier and lower dimensional propensity score function that best explain the observed data. This approach also offers the advantage that it relaxes the additional assumptions: if they hold in the majority of the propensity attributes' configurations, the models' smoothness helps to overcome potential issues arising in the configurations where the assumptions are violated. This subsection presents a simple expectation-maximization method for simultaneously training the classification and the propensity score model. It aims to maximize the expected log likelihood of the combination of models.

Expectation Given the expected classification model $\hat{f}$ and propensity score model $\hat{e}$, the expected probability of the positive class $\hat{y}_i$ of instance x_i with label s_i is:

$$\begin{aligned}\hat{y}_i &= \Pr(y_i = 1|s_i, x_i, \hat{f}, \hat{e}) \\ &= s_i + (1-s_i)\frac{\hat{f}(x_i)\,(1-\hat{e}(x_i))}{1-\hat{f}(x_i)\hat{e}(x_i)}.\end{aligned}$$

Proof.

$$\begin{aligned}\Pr(y=1|s,x,f,e) &= s\Pr(y=1|s=1,x) + (1-s)\Pr(y=1|s=0,x) \\ &= s + (1-s)\frac{\Pr(y=1|x)\Pr(s=0|y=1,x)}{\Pr(s=0|x)} \\ &= s + (1-s)\frac{\Pr(y=1|x)\,(1-\Pr(s=1|y=1,x))}{1-\Pr(s=1|x)} \\ &= s + (1-s)\frac{\Pr(y=1|x)\,(1-\Pr(s=1|y=1,x))}{1-\Pr(y=1|x)\Pr(s=1|y=1,x)},\end{aligned}$$

where the first step follows from the definition of PU data $s=1 \to y=1$ and Bayes' rule. □

Maximization Given the expected probabilities of the positive class $\hat{y}_i$, the models f and e are trained to optimize the expected log likelihood:

$$\begin{aligned}f, e = &\operatorname*{argmax}_{f,e} \sum_{i=1}^{n} \mathbb{E}_{y_i|x_i,s_i,\hat{f},\hat{e}} \ln \Pr(x_i, s_i, y_i|f,e) \\ &= \operatorname*{argmax}_{f} \sum_{i=1}^{n} \left[\hat{y}_i \ln f(x_i) + (1-\hat{y}_i)\ln(1-f(x_i))\right], \\ &\operatorname*{argmax}_{e} \sum_{i=1}^{n} \hat{y}_i\left[s_i \ln e(x_i) + (1-s_i)\ln(1-e(x_i))\right]\end{aligned}$$

Proof.

$$
\begin{aligned}
f, e =& \operatorname*{argmax}_{f,e} \sum_{i=1}^{n} \mathbb{E}_{y_i|x_i,s_i,\hat{f},\hat{e}} \ln \Pr(x_i, s_i, y_i | f, e) \\
=& \operatorname*{argmax}_{f,e} \sum_{i=1}^{n} \mathbb{E}_{y_i|x_i,s_i,\hat{f},\hat{e}} \ln \left[\Pr(x_i) \Pr(y_i|x_i, f) \Pr(s_i|y_i, x_i, e)\right] \\
=& \operatorname*{argmax}_{f,e} \sum_{i=1}^{n} \mathbb{E}_{y_i|x_i,s_i,\hat{f},\hat{e}} \ln \left[\Pr(y_i|x_i, f) \Pr(s_i|y_i, x_i, e)\right] \quad \textit{max not over } \Pr(x_i) \\
=& \operatorname*{argmax}_{f} \sum_{i=1}^{n} \mathbb{E}_{y_i|x_i,s_i,\hat{f},\hat{e}} \ln \Pr(y_i|x_i, f) \\
& + \operatorname*{argmax}_{e} \sum_{i=1}^{n} \mathbb{E}_{y_i|x_i,s_i,\hat{f},\hat{e}} \ln \Pr(s_i|y_i, x_i, e) \\
=& \operatorname*{argmax}_{f} \sum_{i=1}^{n} \left[\hat{y}_i \ln \Pr(y_i = 1|x_i, f) + (1 - \hat{y}_i) \ln \Pr(y_i = 0|x_i, f)\right], \\
& \operatorname*{argmax}_{e} \sum_{i=1}^{n} \left[\hat{y}_i \ln \Pr(s_i|y_i = 1, x_i, e) + (1 - \hat{y}_i) \ln \Pr(s_i|y_i = 0, x_i)\right] \\
=& \operatorname*{argmax}_{f} \sum_{i=1}^{n} \left[\hat{y}_i \ln \Pr(y_i = 1|x_i, f) + (1 - \hat{y}_i) \ln \Pr(y_i = 0|x_i, f)\right], \\
& \operatorname*{argmax}_{e} \sum_{i=1}^{n} \left[\hat{y}_i \ln \Pr(s_i|y_i = 1, x_i, e)\right] \qquad \textit{max not over } \Pr(s_i|y_i = 0, x_i) \\
=& \operatorname*{argmax}_{f} \sum_{i=1}^{n} \hat{y}_i \ln \Pr(y_i = 1|x_i, f) + (1 - \hat{y}_i) \ln \Pr(y_i = 0|x_i, f), \\
& \operatorname*{argmax}_{e} \sum_{i=1}^{n} \hat{y}_i \big[s_i \ln \Pr(s_i = 1|y_i = 1, x_i, e) \\
& \qquad + (1 - s_i) \ln(1 - \Pr(s_i = 1|y_i = 1, x_i, e))\big]
\end{aligned}
$$

□

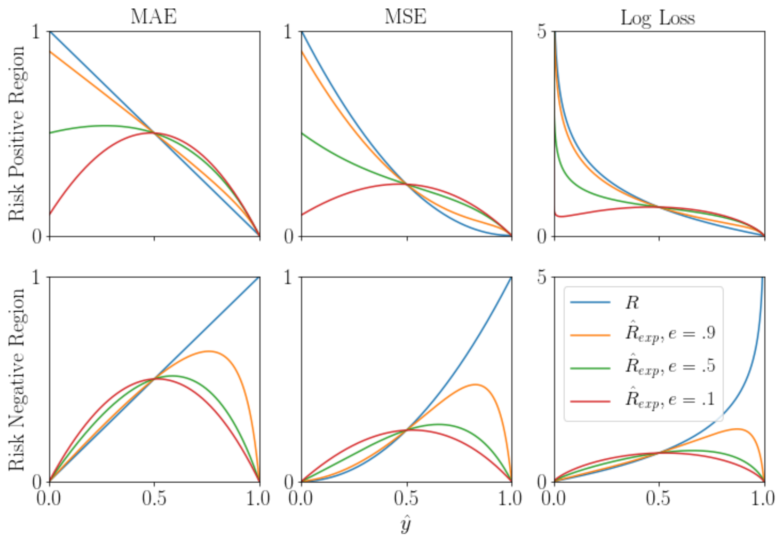

Figure 5.3: **Illustration of expected risk for positive and negative regions in separable data.** In positive regions, all the examples are positive and in negative regions, all the examples are negative. In positive regions, the combination of enough observed labels and a reasonable starting point for hypothesis $\hat{y}$ will converge to the correct hypothesis $\hat{y} = 1$. In negative regions, a reasonable starting point for hypothesis $\hat{y}$ will converge to the correct hypothesis $\hat{y} = 0$, especially when the propensity score in this region is estimated to be large.

From the maximization formula, it can be seen that to optimize the log likelihood, the models both need to optimize the log loss of a weighted dataset. The classification model f receives each example twice, once as positive, weighted by the expected probability of it being positive $\hat{y}_i$ and once as negative, weighted by the expected probability of it being negative $(1 - \hat{y})$. The propensity score model e receives each example once, positive if the observed label is positive and negative otherwise, weighted by the expected probability of it being positive $\hat{y}_i$.

This method is expected to work best if the classes are separable, because the expected log loss, which is an expected risk, of the classifier is being minimized. In Section 5.3.2 it was shown that using the expected risk is in general suboptimal. However, when the classes are separable, the correct model has an expected risk $\hat{R}_{\text{exp}}(\mathbf{y}|\hat{\mathbf{e}}, \mathbf{s}) = 0$. Convergence to the right model is not guaranteed because there can be other local minima. Yet, with a reasonable initial hypothesis and large enough propensity scores, the right hypothesis is expected to be found. This is because in this case 1) the predictions $\hat{y}$ for the positive regions will converge to the minimum risk $\hat{y} = 1$ and 2) the predictions $\hat{y}$ for the negative regions will converge to the minimum $\hat{y} = 0$. If the hypothesis did not get a good initialization, for example, $\hat{y} = 0.9$ in negative regions, then it might converge to the local minimum of $\hat{y} = 1$. This is illustrated in Figure 5.3.

Because the optimal expected log loss will be reached for a deterministic model that classifies all the labeled data as positive, the trained classification model is not expected to be the correct model when the classes are non-separable. It is then not even expected to be the optimal classifier for the trained propensity score. Therefore, it is advisable to retrain the classifier with the obtained propensity scores, using the propensity-weighted risk estimation method.

The classification model is initialized by fitting a balanced model which considers the unlabeled examples as negative. This is a good starting point because the true class prior is likely to be closer to 0.5 than to the ratio of labeled examples. The propensity score model is initialized by using the classification model to estimate the probability that each unlabeled example is positive.

Classic EM converges when the log likelihood stops improving. However, the likelihood could stop improving before the propensity score model has converged. Convergence is therefore formulated as convergence of both the log likelihood and the propensity model. We measure the change in the propensity score model by the average slope of the minimum square error line through the propensity score prediction of the last n iterations.

5.5 Related Work

As previously noted, almost all PU learning work that we are aware of focuses on the SCAR setting or on separable problems. Therefore, our work is significantly different from the existing work on PU learning. The work of Schnabel et al. (2016) on dealing with biases in the observed ratings for recommender systems is closely related to ours [97]. They also make use of propensity scores to cope with the biases. However, there is a crucial difference in that their approach has access to labels for all classes. This important in practice as that means that the propensity score for each example is non-zero. In contrast, in PU learning, the propensity score for any negative example is zero: You never observe these labels.

5.6 Experiments

The aim of this section is to answer the following questions:

Q1 How does propensity-weighted learning perform with a provided propensity score function or label frequency? And how is the performance affected by:

- **Q1a** The number of propensity attributes?
- **Q1b** A biased estimate of the propensity score function?
- **Q1c** Class correlation of the propensity attributes?

Q2 Can the SAR assumption facilitate better learning from SAR PU data when the propensity score is unknown?

- **Q2a** Can the propensity score function be recovered?
- **Q2b** Does the number of propensity attributes and their correlation with the class affect the performance?

5.6.1 Data

We use eight real-world datasets that are summarized in Table 5.1. For 20 News Groups,[1] we distinguish between computer (pos) and recreational (neg) documents. After removing their headers, footers, quotes, and English stop words, the documents were transformed to 200 word occurrence attributes using the Scikit-Learn[2] count vectorizer. For Cover Type,[1] we distinguish the Lodgepole Pine (pos) from all other cover type (neg). The Diabetes [1] data was preprocessed in a similar manner to [112]. Additionally we dropped attributes with the same value in 95% of the examples, and replaced uncommon attribute values by "other". The positive class is patients being readmitted within 30 days. Image Segmentation[1] was used to distinguish between nature (sky, grass or foliage) and other scenes (brickface, cement, window, path). Adult[1], Breast Cancer[1], Mushroom[1], and Splice[3] were used as is. All the datasets were further preprocessed to have exclusively continuous attributes, scaled between -1 and 1. Multivalued attributes were binarized.

[1] http://archive.ics.uci.edu/ml/

[2] http://scikit-learn.org

[3] Available on LIBSVM Data repository https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Table 5.1: **Characteristics of the Datasets.**

Dataset	# Instances	# Attributes	$\Pr(y=1)$
20 Newsgroups	3,979	200	0.55
Adult	48,842	14	0.24
Breast Cancer	683	9	0.35
Cover Type	581,012	54	0.49
Diabetes	99,492	127	0.11
Image Segmentation	2,310	18	0.43
Mushroom	8,124	111	0.48
Splice	3,175	60	0.52

5.6.2 Methodology and Approaches

Constructing Datasets. The datasets were randomly partitioned into train (80%) and test (20%) sets five times. For each of the five train-test splits, we transformed the data into positive and unlabeled datasets in a number of ways. First, the number of attributes used to derive the propensity score was varied from zero to four. The datasets with zero propensity attributes are only used in the experiments that study the influence of the number of propensity attributes. Only attributes that had a standard deviation of at least 0.6 were considered.[4] The attributes were selected in one of two ways: (1) at random, which was then repeated for three different combinations of attributes or (2) based on the largest correlation with the class.[5] For a given set of selected attributes, positive examples were selected to be labeled according to the following propensity score:

$$e(x_e) = \prod_{i=1}^{k} \left(\mathrm{sc}(x_e^{(i)}, p^-, p^+)\right)^{\frac{1}{k}},$$

$$\mathrm{sc}(x_e^{(i)}, p^-, p^+) = p^- + \frac{x_e^{(i)} - \min \mathbf{x}_e^{(i)}}{\max \mathbf{x}_e^{(i)} - \min \mathbf{x}_e^{(i)}}(p^+ - p^-).$$

This gives propensity scores between p^- and p^+, with all propensity attributes attributing equally to it. In our experiments the propensity scores were between 0.2 and 0.8. These propensity scores emulate the setting where observing a higher attribute value stimulates selecting that example to be labeled. Such settings occur, for example, when the attribute represents a symptom for a disease (e.g. level of pain), or when the attribute represents the presence of

[4]For binary attributes this means that at most 90% of the examples can take on the same value.

[5]Here, the signs of the attribute values were possibly inverted to either get all positively or all negatively correlated attributes.

words that correlate with a subject's interest. For each set of selected propensity attributes, we generated five labelings.

Settings and Approaches. We consider two settings. Within in each setting we consider three approaches: Assuming the data is SAR and using propensity score weighting (denoted SAR), assuming the data is SCAR and using class prior weighting (denoted SCAR), and assuming all unlabeled examples belong to the negative class (denoted (Naive). We also show the performance given fully supervised data. We always uses logistic regression as the base classifier for both the classification and the propensity score model. The choice for logistic regression is motivated by it's ability to predicts well-calibrated probabilities [87]. **Setting 1** assumes that the true propensity scores (SAR) or the true class prior (SCAR) are provided. **Setting 2** assumes that these must be estimated from data. Here, our EM method was to estimate the the propensity score under SAR assumptions.[6] To estimate the class prior, two state-of-the art methods for learning under SCAR assumptions were used with standard settings: KM2 [94][7] and TI*c*E [3].[8]

Question 1b investigates the effect of over- and underestimated propensity score estimates. The correct propensity score or label frequency was altered with a bias $\in \{1.1, 1.3, 1.5\}$, multiplying the score with it for an overestimated or dividing for a negative one.

Unless the correlation of propensity attributes with the class is explicitly studied, propensity models with random attributes are used for the experiments.

Evaluation Metric. The Mean Square Error (MSE) is used to report performance. We evaluate two things: (1) classification performance, and (2) the propensity model performance. When evaluating the propensity models, we compared the predicted propensity score from our model to the true propensity scores.

5.6.3 Results with Provided Propensity Scores

A1&A1a. Either incorporating the propensity score or using the class prior leads to better classification performance than naively assuming all unlabeled examples are negative (Figure 5.4). When the PU data is generated by a SAR labeling mechanism where the propensity scores depends on the attribute values, training a model using the propensity scores results in superior performance

[6]http://dtai.cs.kuleuven.be/software/sar

[7]`http://web.eecs.umich.edu/~cscott/code/kernel_MPE.zip`

[8]`https://dtai.cs.kuleuven.be/software/tice/`

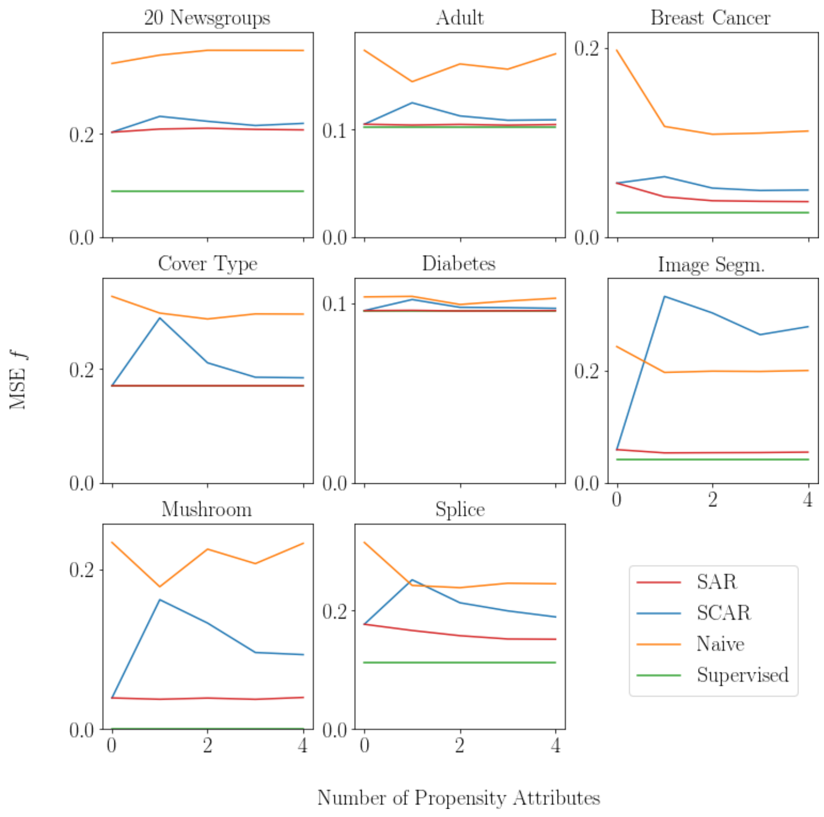

Figure 5.4: **Influence of the number of propensity attributes on propensity-weighted learning.** Propensity-weighted learning performs well for any number of propensity attributes.

compared to making the SCAR assumption. This is true even though the SCAR learner knows the correct label frequency.

A1b. Using an under- or overestimated propensity score clearly affects the performance, however in most cases the biased propensity scores, and even often biased label frequencies, outperform treating the unlabeled examples as negative. Interestingly, an overestimate hurts the performance less. This is in agreement with proposition 3 (Figure 5.5).

A1c. The correlation, either positive or negative, of the propensity attributes with the class has a big influence on the difficulty of the problem, as can be understood from the performance of the naive approach. It has a smaller influence when the label frequency was used and is almost unnoticeable when

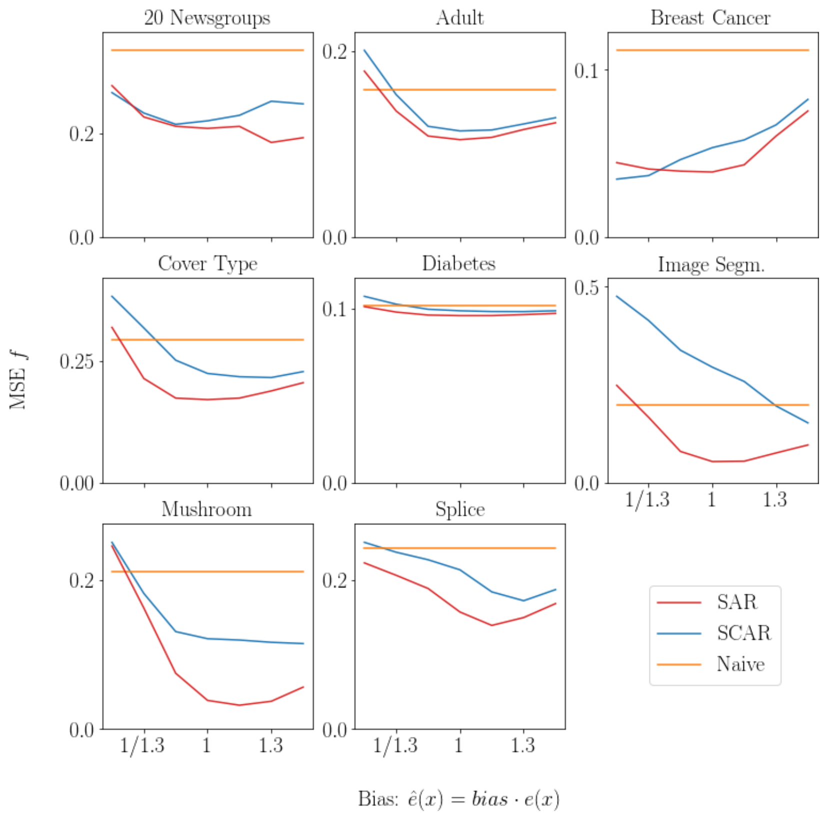

Figure 5.5: **Influence of a biased propensity score estimate on propensity-weighted learning.** Even underestimated (bias<1) and overestimated (bias>1) propensity scores outperform Naive. Underestimates hurt the performance more than overestimates.

using the propensity scores (Figure 5.6).

5.6.4 Results with Learned Propensity Scores

A2. When the labeling mechanism is unknown, but the propensity attributes are, learning both the propensity score and the classification model from the data almost always outperforms learning under the SCAR assumption or the naive method. For the diabetes dataset, the naive method seems outperforms all others. The dataset is very imbalanced and MSE is not the most appropriate measure in this case. The F_1 scores for Diabetes are 0.15 (Supervised), 0.24

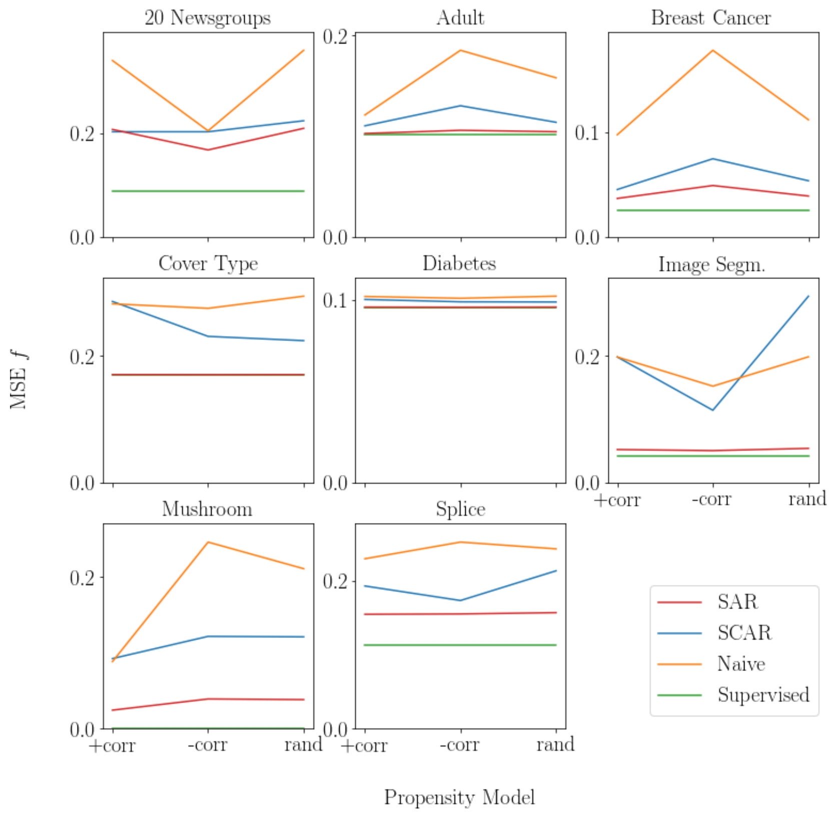

Figure 5.6: **Influence of class correlation of the propensity attributes on propensity-weighted learning.** There are three settings, the attributes are either positively or negatively correlated with the class or they are random attributes. The propensity attributes being correlated with the class has a big influence on the difficulty of the problem. Knowing the label frequency for SCAR or the propensity score SAR reduces the difficulty.

(SAR), 0.20 (KM2), 0.20 (TI*c*E) and 0.09 (Naive), showing that actually SAR is here to the best method (Table 5.2). Note that only the Mushroom dataset is separable, as is needed to have the EM algorithm converged to the correct model. Yet, the imperfect propensity score models learned by the algorithm are still much closer to the real propensity scores than taking the constant label frequency as the propensity score for all examples (Table 5.3). Consequently, the classification models also perform better under the SAR assumption.

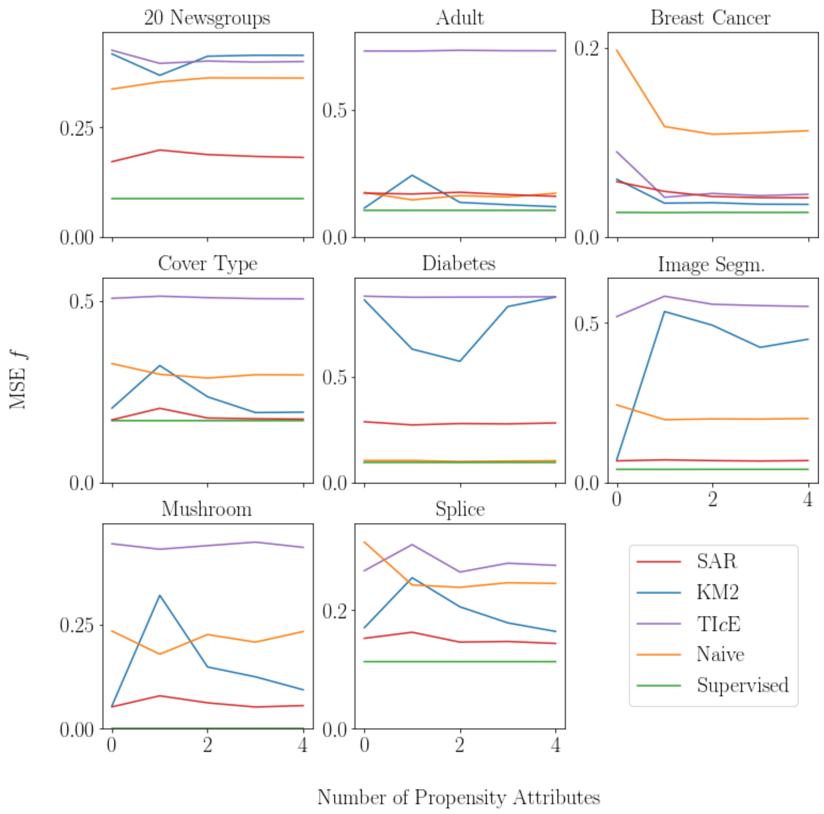

Figure 5.7: **Influence of the number of propensity attributes on the classifier quality when learning with unknown propensity scores.** The quality of the SAR classification model is mostly unaffected by the number of attributes.

A2a. Learning under SAR assumptions always results in a better propensity score model than under SCAR assumptions (Table 5.3).

A2b. When increasing the number of propensity attributes, the MSE of the propensity model improves for most settings, both SAR and SCAR. Like noted before, this is because the more variables, the better the variations in the label frequency can be explained as noise. The MSE of the SAR classification model is mostly unaffected by the number of attributes (Figures 5.7 and 5.8). The correlation of the propensity attributes with the class does not have a large effect on the performance when learning under SAR assumptions (Figures 5.9 and 5.10).

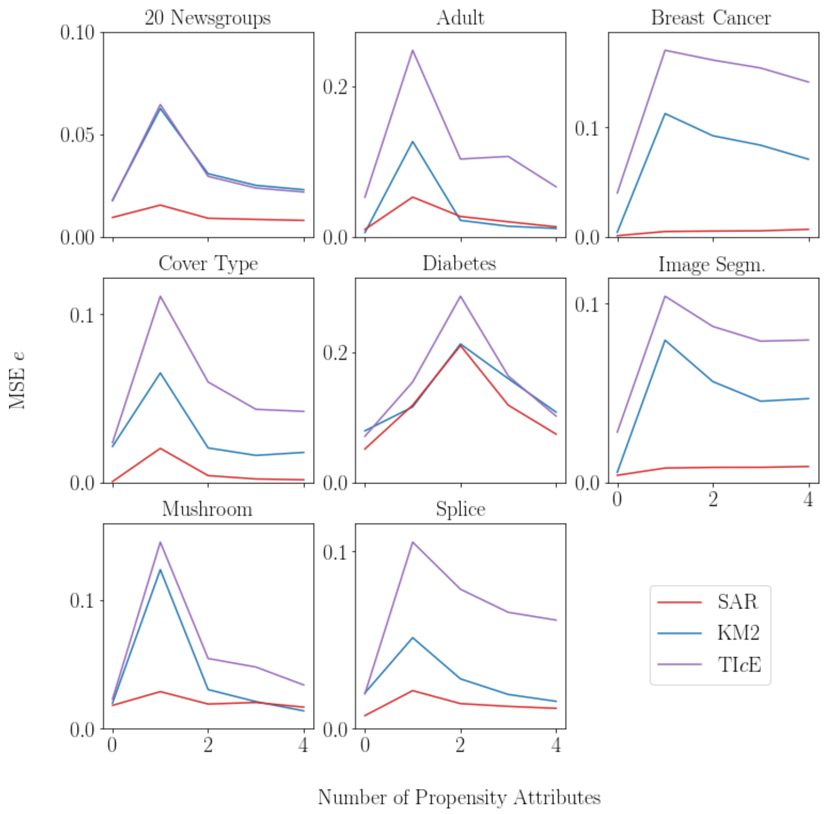

Figure 5.8: **Influence of the number of propensity attributes on the propensity model quality when learning with unknown propensity scores.** The quality of the SAR propensity model is mostly unaffected by the number of attributes. For most settings, the propensity model quality of SCAR methods improves when increasing the number of propensity attributes.

5.7 Conclusions

This chapter investigates learning from SAR PU data: positive and unlabeled data with non-uniform labeling mechanisms. We proposed and theoretically analyzed a empirical-risk-minimization based method for weighting PU datasets with the propensity scores to achieve unbiased learning. Furthermore, we explored which assumptions are necessary to learn from SAR PU data generated by an unknown labeling mechanism. We proposed a practical EM-based method for this setting. Empirically, we demonstrated that for SAR PU data our

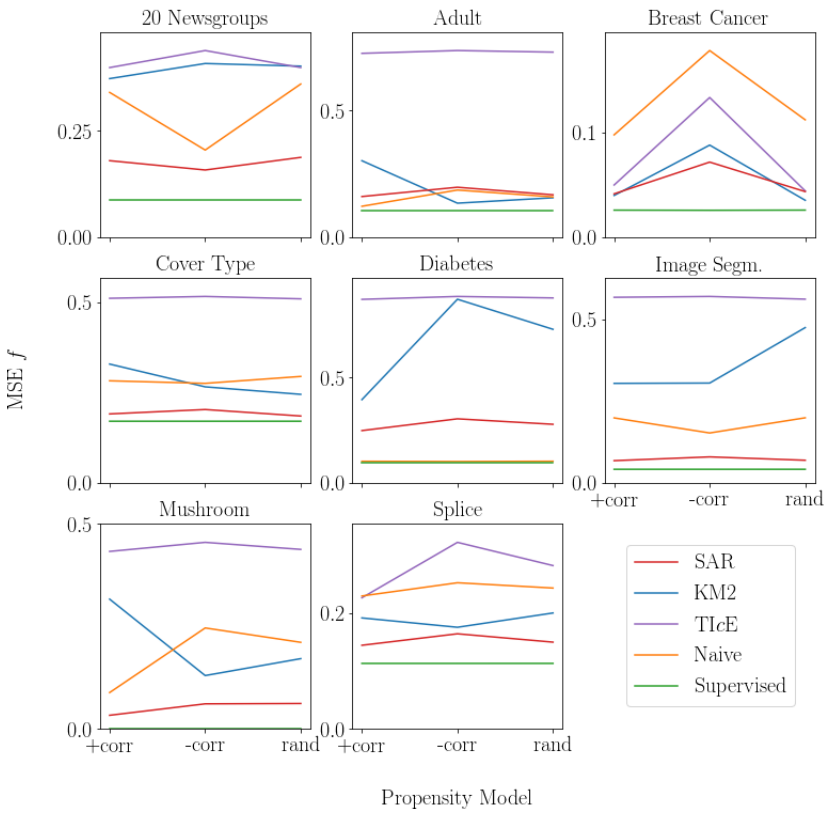

Figure 5.9: **Influence of class correlation of the propensity attributes on the classifier quality when learning with unknown propensity scores.** There are three settings, the attributes are either positively or negatively correlated with the class or they are random attributes. The propensity attributes being correlated with the class has a big influence on the difficulty of the problem. Knowing the label frequency for SCAR or the propensity score SAR reduces the difficulty.

proposed propensity weighted method offers superior performance over making the SCAR. This is true for both given and learned propensity scores. The EM-based algorithm is not expected to converge to the optimal solution for non-separable data, but still outperforms the methods that assume the labels to be SCAR when the data is non-separable. This is because the estimated propensity scores are still better than taking the constant label frequency as the propensity score for all examples.

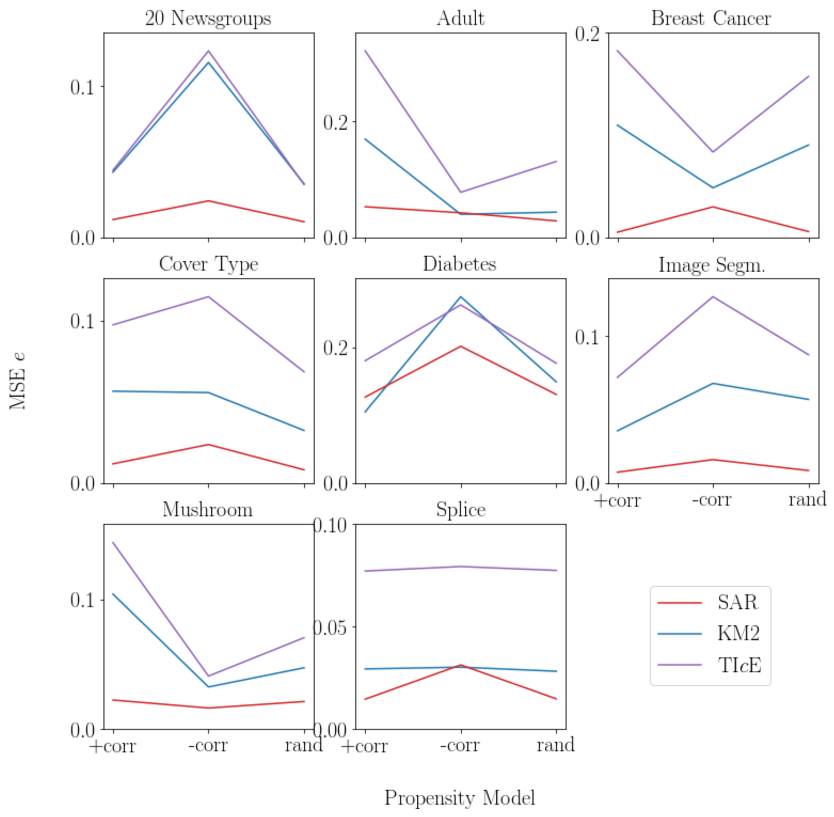

Figure 5.10: **Influence of class correlation of the propensity attributes on the propensity model quality when learning with unknown propensity scores.** There are three settings, the attributes are either positively or negatively correlated with the class or they are random attributes. The propensity attributes being correlated with the class has a big influence on the difficulty of the problem. Knowing the label frequency for SCAR or the propensity score SAR reduces the difficulty.

Table 5.2: **Mean Square Error (MSE) of classification model f.**

Dataset	Supervised	SAR	KM2	TIcE	Naive
20 Newsgroups	0.09	**0.19**	0.40	0.40	0.36
Adult	0.10	0.17	**0.15**	0.73	0.16
Breast Cancer	0.03	0.04	**0.03**	0.04	0.11
Cover Type	0.17	**0.18**	0.24	0.51	0.29
Diabetes	0.10	0.28	0.73	0.88	**0.10**
Image Segmentation	0.04	**0.07**	0.47	0.56	0.20
Mushroom	0.00	**0.06**	0.17	0.44	0.21
Splice	0.11	**0.15**	0.20	0.28	0.24

Table 5.3: **Mean Square Error (MSE) of propensity score model e.**

Dataset	SAR	KM2	TIcE
20 Newsgroups	**0.01**	0.04	0.03
Adult	**0.03**	0.04	0.13
Breast Cancer	**0.01**	0.09	0.16
Cover Type	**0.01**	0.03	0.07
Diabetes	**0.13**	0.15	0.18
Image Segmentation	**0.01**	0.06	0.09
Mushroom	**0.02**	0.05	0.07
Splice	**0.01**	0.03	0.08

Chapter 6

Conclusions

This chapter summarizes the main contributions and conclusions of this dissertation. It also provides possible directions for future work.

6.1 Summary

The objective of this dissertation was to push the boundaries of learning from positive and unlabeled data by studying the commonly made assumptions, more specifically the *Selected Completely At Random (SCAR)* assumption. By taking this perspective, we were able to improve upon current methods for estimating the class prior in SCAR PU data, port SCAR-based techniques and insights to the relational domain and introduce a novel, more realistic, *Selected At Random (SAR)* assumption. This dissertation presented four main contributions, which are discussed in more detail below.

6.1.1 Contribution 1: Literature Survey

Although PU learning is a relatively young field, it has grown a lot over the last two decades. It originated independently from different fields: semi-supervised learning with the absence of labels from one class, one-class classification with the additional information of unlabeled data, and classification with one-sided label noise. Because of these different points of view, the field of PU learning has developed almost independently in different communities, resulting in a lack of overview. In this dissertation, we wrote an extensive survey of the accumulated knowledge. We explicitly structured the field according to the different assumptions that have been made and the types of methods that have been proposed. Furthermore, an overview of published applications and links to related fields have been discussed.

6.1.2 Contribution 2: Scalable Class Prior Estimation in Positive and Unlabeled Data

The ability of estimating the class prior from a PU dataset is an important step for many PU learning methods. Given a known class prior, standard machine learning methods can be used to train a classifier from PU data. The class prior is then used to reweight the data, to adjust the output probabilities of the model, or to modify the learning algorithm.

We proposed a simple yet accurate and scalable method TI*c*E for estimating the class prior in SCAR PU data. The method follows from the observation that the label frequency (which serves as a proxy for the class prior) is expected to be the same in any subdomain of the instance space. Moreover, each subdomain provides a natural lower bound on the label frequency. Our method estimates the label frequency by searching via decision tree induction for the largest lower bound.

Despite the simplicity of the method, its estimates are equivalently accurate as the estimates made by the state-of-the-art methods, while being an order of magnitude faster. This was shown by an extensive empirical evaluation on eleven real-world datasets.

6.1.3 Contribution 3: Learning from Positive and Unlabeled Relational Data under the Selected Completely At Random Assumption

Despite the natural presence of PU data in relational learning tasks, such as knowledge base completion, PU learning has mostly been developed in the propositional field. Moreover, the few existing relational methods were developed independently of the advances in propositional PU learning. As a result, the widely used SCAR assumption, had not been considered yet for relational PU data.

We investigated if the insights from propositional PU learning, and more specifically the SCAR assumption, can be used for relational data. To this end, two methods for incorporating the class prior in relational learning methods were proposed: probabilistic classifier modification and score function modification, which were applied to the widely used TILDE and Aleph systems. Furthermore, we showed how to modify our class prior estimation method TI*c*E to TI*c*ER, to operate in the relational domain.

With an extensive empirical evaluation, we showed that indeed incorporating the class prior can be helpful. Especially, when the data is not easily separable, which is the underlying assumption of the established relational methods.

6.1.4 Contribution 4: Beyond the Selected Completely At Random Assumption

For real problems, the SCAR assumption often falls short. The labeling mechanism often does depend on some of the data attributes. Patients with clearer symptoms, for example, are much more likely to be diagnosed than those with more subtle symptoms. To enable PU learning in realistic scenarios, we introduced the *Selected At Random (SAR)* assumption. This is an important advancement for PU learning as it opens a whole new range of possible applications. We propose two methods for learning under this assumption: 1) a weighting method for when the labeling mechanism is known, and 2) an EM method for when the labeling mechanism is unknown.

Our proposed SAR weighting method is based on an unbiased evaluation measure estimator. It was analyzed in an empirical-risk-minimization based framework. When the labeling mechanism is not exactly know, but an estimate that differs from the true mechanism is used, then the risk will be biased as well. From this bias, we can see that it is safer to overestimate the propensity scores (the label probabilities), than to underestimate them.

To develop a method for learning under the SAR assumption with an unknown labeling mechanism, a theoretical analysis was conducted to understand which additional assumptions are necessary to enable learning. Based on this analysis, the practical EM-based approach was proposed.

From the empirical evaluation, it can indeed be seen that for SAR PU data, making the SCAR assumption is suboptimal compared to making the SAR assumption. This observation holds both when the labeling mechanism is known and when it is not.

6.2 Future Work

The contributions of this dissertation provided valuable new methods and insights for learning from PU data. Nevertheless, there are still many open questions and challenges in the field. This section presents several possible directions for future work.

Method comparison A lot of PU learning methods have been proposed over the last two decades. However, there is no consistent comparison between the techniques. Many papers claim to propose the new state-of-the-art method, while only comparing to a limited number of methods on a limited number of datasets and often without making the underlying assumptions explicit. The used datasets also differ a lot between papers. An extensive comparison between methods, on a range of datasets, where for each of the datasets it is clear which assumptions hold and what the application domain is, would be a great contribution to the field. Such a comparison would contribute in several ways. It would show which methods are really the state of the art for a given set of assumptions and application domains. Also, it might clarify which ideas are indeed helpful to learn good PU classifiers under each set of assumptions, which will spark ideas for novel methods.

PU dataset collection Currently, most PU methods are evaluated on fully labeled datasets, from which PU data was artificially generated. It would be very interesting to obtain freely available real-world PU datasets, where the true classes are known. In these datasets the labeling mechanism would be real.

Unify single-training-set and case-control scenarios PU learning methods are always developed for a specific scenario. Sometimes, it is obvious how to modify the method to the other scenario, but other times, for example when lengthy derivations are involved, it is not at all. It would be interesting to study general techniques to go from the one scenario to the other. Naively, one could turn the available data into the desired data type for the method to be used. A single-training-set dataset can always be converted into a case-control dataset by considering the positive data as case data and taking the complete dataset as control data. The other direction is more complicated, by combining the case and control data into one dataset, the positive class will be overrepresented. Techniques from cost-sensitive classification could be used to address this. However, as many learning methods are not invariant to base rate changes, this would invalidate proven results for these methods [29].

Reasonable assumptions in real-world settings Most of the PU learning research, and specifically the development of learning assumptions, has been theoretical work. The learning assumptions are chosen based on what would be helpful for learning instead of what tends to hold in practice. Therefore, it is not clear how often the separability or SCAR assumption are reasonable in practice. The proposed SAR assumption is probably general enough to be realistic, but it might be so general that it makes learning more complicated than is necessary. A study of real-world PU settings and the assumptions that hold there, might give rise to new methods that are more fit for real problems.

Incorporating relational meta information Relational data possesses more information than the explicitly encoded facts and relations. For example, the distribution over a relation's cardinality can tell us how many of these relations are expected to occur per object, like from the relation `fatherof(father,child)`, it can be calculated how many children fathers tend to have. This information can be exploited for PU learning to, for example, discard models that result in significantly different distributions, like people having 10 children on average.

PU evaluation methods With PU data, most standard evaluation metrics cannot be calculated and hence not be used for tuning or evaluation. Very few alternatives have been proposed that do work with PU data. The most commonly used one is a method based on the F_1 metric that simultaneously aims to optimize the precision and recall. However, the balance between those two metrics is set more or less arbitrarily [61]. When the class prior is known and the SCAR assumption holds, some standard metrics can be used [49]. It would be interesting to see how standard metrics can be calculated under the SAR assumption, given the propensity scores. Generally, more research on PU evaluation metrics would be insightful.

SAR method for non-separable data The proposed EM method for learning from SAR PU data works best when the data is (more or less) separable. More research is needed to find a method that would work in a more general setting.

Bibliography

[1] Basile, T. M., Di Mauro, N., Esposito, F., Ferilli, S., and Vergari, A. Density estimators for positive-unlabeled learning. In *New Frontiers in Mining Complex Patterns: 6th International Workshop, NFMCP 2017, Held in Conjunction with ECML-PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Revised Selected Papers* (2018), vol. 10785, Springer, pp. 49–64.

[2] Bekker, J., and Davis, J. Beyond the selected completely at random assumption for learning from positive and unlabeled data. *arXiv preprint arXiv:1809.03207* (2018).

[3] Bekker, J., and Davis, J. Estimating the class prior in positive and unlabeled data through decision tree induction. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence* (2018), pp. 2712–2719.

[4] Bekker, J., and Davis, J. Learning from positive and unlabeled data: A survey. *arXiv preprint arXiv:1811.04820* (2018).

[5] Bekker, J., and Davis, J. Positive and unlabeled relational classification through label frequency estimation. In *Inductive Logic Programming* (Cham, 2018), N. Lachiche and C. Vrain, Eds., Springer International Publishing, pp. 16–30.

[6] Blanchard, G., Lee, G., and Scott, C. Semi-supervised novelty detection. *Journal of Machine Learning Research 11* (2010), 2973–3009.

[7] Blockeel, H. Pu-learning disjunctive concepts in ilp. In *ILP 2017 late breaking papers* (2017).

[8] Blockeel, H., and De Raedt, L. Top-down induction of first-order logical decision trees. *Artificial intelligence* (1998), 285–297.

[9] Blockeel, H., Page, D., and Srinivasan, A. Multi-instance tree learning. In *Proceedings of the 22nd international conference on Machine learning* (2005), ACM, pp. 57–64.

[10] Blum, A., and Mitchell, T. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory* (1998), ACM, pp. 92–100.

[11] Breiman, L. Random forests. *Machine Learning 45*, 1 (Oct 2001), 5–32.

[12] Calvo, B., Larrañaga, P., and Lozano, J. A. Learning bayesian classifiers from positive and unlabeled examples. *Pattern Recogn. Lett. 28*, 16 (Dec. 2007), 2375–2384.

[13] Cerulo, L., Elkan, C., and Ceccarelli, M. Learning gene regulatory networks from only positive and unlabeled data. *BMC bioinformatics 11*, 1 (2010), 228.

[14] Chang, S., Zhang, Y., Tang, J., Yin, D., Chang, Y., Hasegawa-Johnson, M. A., and Huang, T. S. Positive-unlabeled learning in streaming networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016), ACM, pp. 755–764.

[15] Chapelle, O., Scholkopf, B., and Zien, A. Semi-supervised learning. *IEEE Transactions on Neural Networks 20*, 3 (2009), 542–542.

[16] Chaudhari, S., and Shevade, S. Learning from positive and unlabelled examples using maximum margin clustering. In *Proceedings of the 19th International Conference on Neural Information Processing* (Berlin, Heidelberg, 2012), vol. 3, Springer-Verlag, pp. 465–473.

[17] Chiaroni, F., Rahal, M.-C., Hueber, N., and Dufaux, F. Learning with a generative adversarial network from a positive unlabeled dataset for image classification. In *IEEE International Conference on Image Processing* (2018).

[18] Claesen, M., Davis, J., De Smet, F., and De Moor, B. Assessing binary classifiers using only positive and unlabeled data. *arXiv preprint arXiv:1504.06837* (2015).

[19] Claesen, M., Smet, F. D., Gillard, P., Mathieu, C., and Moor, B. D. Building classifiers to predict the start of glucose-lowering pharmacotherapy using belgian health expenditure data. *CoRR abs/1504.07389* (2015).

[20] Claesen, M., Smet, F. D., Suykens, J. A. K., and Moor, B. D. A robust ensemble approach to learn from positive and unlabeled data using svm base models. *Neurocomputing 160* (2015), 73–84.

[21] Denis, F. Pac learning from positive statistical queries. In *Proceedings of the International Conference on Algorithmic Learning Theory* (1998), Springer, pp. 112–126.

[22] Denis, F., Gilleron, R., and Letouzey, F. Learning from positive and unlabeled examples. *Theoretical Computer Science 348*, 1 (2005), 70–83.

[23] Denis, F., Laurent, A., Gilleron, R., and Tommasi, M. Text classification and co-training from positive and unlabeled examples. In *Proceedings of the ICML 2003 workshop: the continuum from labeled to unlabeled data* (2003), pp. 80–87.

[24] du Plessis, M., Niu, G., and Sugiyama, M. Class-prior estimation for learning from positive and unlabeled data. *Proceedings of the 7th Asian Conference on Machine Learning* (2015), 221–236.

[25] Du Plessis, M., Niu, G., and Sugiyama, M. Convex formulation for learning from positive and unlabeled data. In *International Conference on Machine Learning* (2015), pp. 1386–1394.

[26] du Plessis, M. C., Niu, G., and Sugiyama, M. Analysis of learning from positive and unlabeled data. In *Advances in Neural Information Processing Systems* (2014), pp. 703–711.

[27] du Plessis, M. C., and Sugiyama, M. Semi-supervised learning of class balance under class-prior change by distribution matching. *Neural networks: the Official Journal of the International Neural Network Society 50* (2012), 110–9.

[28] Du Plessis, M. C., and Sugiyama, M. Class prior estimation from positive and unlabeled data. *IEICE TRANSACTIONS on Information and Systems 97*, 5 (2014), 1358–1362.

[29] Elkan, C. The foundations of cost-sensitive learning. In *Proceedings of the seventeenth international joint conference on artificial intelligence* (2001), vol. 17, Lawrence Erlbaum Associates Ltd, pp. 973–978.

[30] Elkan, C., and Noto, K. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (2008), ACM, pp. 213–220.

[31] Fei, H., Kim, Y., Sahu, S., Naphade, M., Mamidipalli, S. K., and Hutchinson, J. Heat pump detection from coarse grained smart meter data with positive and unlabeled learning. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (2013), ACM, pp. 1330–1338.

[32] Frénay, B., and Verleysen, M. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems 25*, 5 (2014), 845–869.

[33] Fung, G. P. C., Yu, J. X., Lu, H., and Yu, P. S. Text classification without negative examples revisit. *IEEE Transactions on Knowledge and Data Engineering 18* (2006), 6–20.

[34] Galárraga, L., Teflioudi, C., Hose, K., and Suchanek, F. M. Fast rule mining in ontological knowledge bases with AMIE+. *The VLDB Journal—The International Journal on Very Large Data Bases 24*, 6 (2015), 707–730.

[35] Gan, H., Zhang, Y., and Song, Q. Bayesian belief network for positive unlabeled learning with uncertainty. *Pattern Recogn. Lett. 90*, C (Apr. 2017), 28–35.

[36] Gardner, M., Talukdar, P., Krishnamurthy, J., and Mitchell, T. Incorporating vector space similarity in random walk inference over knowledge bases. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014), pp. 397–406.

[37] Gorber, S. C., Schofield-Hurwitz, S., Hardt, J. S., Levasseur, G., and Tremblay, M. D. The accuracy of self-reported smoking: a systematic review of the relationship between self-reported and cotinine-assessed smoking status. *Nicotine & tobacco research : official journal of the Society for Research on Nicotine and Tobacco 11*, 1 (2009), 12–24.

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

[39] He, J., Zhang, Y., Li, X., and Wang, Y. Naive bayes classifier for positive unlabeled learning with uncertainty. In *Proceedings of the 2010 SIAM International Conference on Data Mining* (2010), SIAM, pp. 361–372.

[40] He, J., Zhang, Y., Li, X., and Wang, Y. Bayesian classifiers for positive unlabeled learning. In *Proceedings of the 12th International Conference on Web-age Information Management* (Berlin, Heidelberg, 2011), WAIM'11, Springer-Verlag, pp. 81–93.

[41] Hido, S., Tsuboi, Y., Kashima, H., Sugiyama, M., and Kanamori, T. Inlier-based outlier detection via direct density ratio estimation. *2008 Eighth IEEE International Conference on Data Mining* (2008), 223–232.

[42] Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association 58*, 301 (1963), 13–30.

[43] Hou, M., Chaib-draa, B., Li, C., and Zhao, Q. Generative adversarial positive-unlabelled learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18* (7 2018), pp. 2255–2261.

[44] Hsieh, C.-J., Natarajan, N., and Dhillon, I. PU learning for matrix completion. In *International Conference on Machine Learning* (2015), pp. 2445–2453.

[45] Ienco, D., and Pensa, R. G. Positive and unlabeled learning in categorical data. *Neurocomput. 196*, C (July 2016), 113–124.

[46] Ienco, D., Pensa, R. G., and Meo, R. From context to distance: Learning dissimilarity for categorical data clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD) 6*, 1 (2012), 1–25.

[47] Imbens, G. W., and Rubin, D. B. *Causal inference in statistics, social, and biomedical sciences.* Cambridge University Press, 2015.

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

[49] Jain, S., White, M., and Radivojac, P. Recovering true classifier performance in positive-unlabeled learning. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence* (2017), pp. 2066–2073.

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

[51] Jiang, L., Zhang, H., and Cai, Z. A novel bayes model: Hidden naive bayes. *IEEE Transactions on knowledge and data engineering 21*, 10 (2009), 1361–1371.

[52] Ke, T., Jing, L., Lv, H., Zhang, L., and Hu, Y. Global and local learning from positive and unlabeled examples. *Applied Intelligence 48* (2017), 2373–2392.

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

[54] Ke, T., Yang, B., Zhen, L., Tan, J., Li, Y., and Jing, L. Building high-performance classifiers using positive and unlabeled examples for text classification. In *International Symposium on Neural Networks* (2012), Springer, pp. 187–195.

[55] Khan, S., and Madden, M. One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review* (2014).

[56] Khot, T., Natarajan, S., and Shavlik, J. W. Relational one-class classification: A non-parametric approach. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence* (2014), pp. 2453–2460.

[57] Kiryo, R., Niu, G., du Plessis, M. C., and Sugiyama, M. Positive-unlabeled learning with non-negative risk estimator. In *Advances in Neural Information Processing Systems* (2017), pp. 1675–1685.

[58] Kull, M., de Menezes e Silva Filho, T., and Flach, P. A. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Proceedings of the twentieth International Conference on Artificial Intelligence and Statistics* (2017), pp. 623–631.

[59] Lao, N., Subramanya, A., Pereira, F., and Cohen, W. W. Reading the web with learned syntactic-semantic inference rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (2012), pp. 1017–1026.

[60] Latulippe, M., Drouin, A., Giguere, P., and Laviolette, F. Accelerated robust point cloud registration in natural environments through positive and unlabeled learning. In *Proceedings of the 23th International Joint Conference on Artifical Intelligence* (2013), pp. 2480–2487.

[61] Lee, W. S., and Liu, B. Learning with positive and unlabeled examples using weighted logistic regression. In *Proceedings of the Twentieth International Conference on Machine Learning* (2003), pp. 448–455.

[62] Li, W., Guo, Q., and Elkan, C. A positive and unlabeled learning algorithm for one-class classification of remote-sensing data. *IEEE Transactions on Geoscience and Remote Sensing 49* (2011), 717–725.

[63] Li, X., and Liu, B. Learning to classify texts using positive and unlabeled data. In *Proceedings of the eighteenth International Joint Conference on Artifical Intelligence* (2003), vol. 3, pp. 587–592.

[64] Li, X., Liu, B., and Ng, S.-K. Learning to identify unexpected instances in the test set. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence* (2007), vol. 7, pp. 2802–2807.

[65] Li, X.-L., and Liu, B. Learning from positive and unlabeled examples with different data distributions. In *European Conference on Machine Learning* (2005), Springer, pp. 218–229.

[66] Li, X.-L., Liu, B., and Ng, S.-K. Negative training data can be harmful to text classification. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (2010), Association for Computational Linguistics, pp. 218–228.

[67] Li, X.-L., Yu, P. S., Liu, B., and Ng, S.-K. Positive unlabeled learning for data stream classification. In *Proceedings of the 2009 SIAM International Conference on Data Mining* (2009), SIAM, pp. 259–270.

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

[69] Liang, C., Zhang, Y., Shi, P., and Hu, Z. Learning very fast decision tree from uncertain data streams with positive and unlabeled samples. *Information Sciences 213* (2012), 50–67.

[70] Little, R. J., and Rubin, D. B. *Statistical analysis with missing data.* John Wiley & Sons, 2002.

[71] Liu, B., Dai, Y., Li, X., Lee, W. S., and Yu, P. S. Building text classifiers using positive and unlabeled examples. In *Proceedings of the Third IEEE International Conference on Data Mining* (2003), IEEE, pp. 179–186.

[72] Liu, B., Lee, W. S., Yu, P. S., and Li, X. Partially supervised classification of text documents. In *Proceedings of the Nineteenth International Conference on Machine Learning* (2002), vol. 2, Citeseer, pp. 387–394.

[73] Liu, L., and Peng, T. Clustering-based method for positive and unlabeled text categorization enhanced by improved TFIDF. *Journal of Information Science and Engineering 30* (2014), 1463–1481.

[74] Liu, Y., Qiu, S., Zhang, P., Gong, P., Wang, F., Xue, G., and Ye, J. Computational drug discovery with dyadic positive-unlabeled learning. In *Proceedings of the 2017 SIAM International Conference on Data Mining* (2017), SIAM, pp. 45–53.

[75] Liu, Z., Shi, W., Li, D., and Qin, Q. Partially supervised classification–based on weighted unlabeled samples support vector machine. In *Proceedings of the International Conference on Advanced Data Mining and Applications* (2005), Springer, pp. 118–129.

[76] Lu, F., and Bai, Q. Semi-supervised text categorization with only a few positive and unlabeled documents. *2010 3rd International Conference on Biomedical Engineering and Informatics 7* (2010), 3075–3079.

[77] Mahalanobis, P. On the generalised distance in statistics. *National Institute of Science of India* (1936).

[78] Mahdisoltani, F., Biega, J., and Suchanek, F. M. Yago3: A knowledge base from multilingual wikipedias. In *Proceedings of the sixth biennial Conference on Innovative Data Systems Research* (2013).

[79] McCreath, E., and Sharma, A. ILP with noise and fixed example size: A bayesian approach. In *Proceedings of the Fifteenth International Joint Conference on Artifical Intelligence* (1997), pp. 1310–1315.

[80] Mordelet, F., and Vert, J.-P. Prodige: Prioritization of disease genes with multitask machine learning from positive and unlabeled examples. *BMC bioinformatics 12* (2011), 389.

[81] Mordelet, F., and Vert, J.-P. Supervised inference of gene regulatory networks from positive and unlabeled examples. *Methods in molecular biology 939* (2013), 47–58.

[82] Mordelet, F., and Vert, J.-P. A bagging svm to learn from positive and unlabeled examples. *Pattern Recognition Letters 37* (2014), 201–209.

[83] Muggleton, S. Learning from positive data. In *Selected Papers from the 6th International Workshop on Inductive Logic Programming* (1996), pp. 358–376.

[84] Natarajan, N., Rao, N., and Dhillon, I. PU matrix completion with graph information. In *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2015 IEEE 6th International Workshop on* (2015), IEEE, pp. 37–40.

[85] Neelakantan, A., Roth, B., and McCallum, A. Compositional vector space models for knowledge base completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (2015), Association for Computational Linguistics, pp. 156–166.

[86] Nguyen, M. N., Li, X.-L., and Ng, S.-K. Positive unlabeled learning for time series classification. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence* (2011), pp. 1421–1426.

[87] Niculescu-Mizil, A., and Caruana, R. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning* (2005), ACM, pp. 625–632.

[88] Northcutt, C. G., Wu, T., and Chuang, I. L. Learning with confident examples: Rank pruning for robust classification with noisy labels. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence* (2017), UAI'17, AUAI Press.

[89] Pelckmans, K., and Suykens, J. A. Transductively learning from positive examples only. In *Proceedings of the European Symposium on Artificial Neural Networks* (2009), pp. 23–28.

[90] Peng, T., Zuo, W., and He, F. Svm based adaptive learning method for text classification from positive and unlabeled documents. *Knowledge and Information Systems 16* (2007), 281–301.

[91] Platt, J., et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers 10*, 3 (1999), 61–74.

[92] Prokhorov, D. IJCNN 2001 neural network competition. *Slide presentation in IJCNN 1* (2001), 97.

[93] Qin, X., Zhang, Y., Li, C., and Li, X. Learning from data streams with only positive and unlabeled data. *Journal of Intelligent Information Systems 40* (2012), 405–430.

[94] Ramaswamy, H., Scott, C., and Tewari, A. Mixture proportion estimation via kernel embedding of distributions. In *International Conference on Machine Learning* (2016), pp. 2052–2060.

[95] Ren, Y., Ji, D., and Zhang, H. Positive unlabeled learning for deceptive reviews detection. In *Proceedings of the conference on Empirical Methods in Natural Language Processing* (2014), pp. 488–498.

[96] Rubin, D. B. Inference and missing data. *Biometrika 63*, 3 (1976), 581–592.

[97] Schnabel, T., Swaminathan, A., Singh, A., Chandak, N., and Joachims, T. Recommendations as treatments: Debiasing learning and evaluation. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning* (2016), vol. 48, pp. 1670–1679.

[98] Schoenmackers, S., Davis, J., Etzioni, O., and Weld, D. S. Learning first-order Horn clauses from web text. In *Proceedings of Conference on Empirical Methods on Natural Language Processing* (2010).

[99] Scott, C. A rate of convergence for mixture proportion estimation, with application to learning from noisy labels. In *Proceedings of The 18th International Conference on Artificial Intelligence and Statistics* (2015), pp. 838–846.

[100] Scott, C., and Blanchard, G. Novelty detection: Unlabeled data definitely help. In *The 12th International Conference on Artificial Intelligence and Statistics* (2009), pp. 464–471.

[101] Sechidis, K., and Brown, G. Markov blanket discovery in positive-unlabelled and semi-supervised data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (2015), Springer, pp. 351–366.

[102] Sechidis, K., and Brown, G. Simple strategies for semi-supervised feature selection. *Machine Learning 107* (2017), 357–395.

[103] Sechidis, K., Calvo, B., and Brown, G. Statistical hypothesis testing in positive unlabelled data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (2014), Springer, pp. 66–81.

[104] Sechidis, K., Sperrin, M., Petherick, E. S., Luján, M., and Brown, G. Dealing with under-reported variables: An information theoretic solution. *International Journal of Approximate Reasoning 85* (2017), 159–177.

[105] Sellamanickam, S., Garg, P., and Keerthi, S. S. A pairwise ranking based approach to learning with positive and unlabeled examples. In *Proceedings of the 2011 ACM on Conference on Information and Knowledge Management* (2011).

[106] Shao, Y.-H., Chen, W.-J., Liu, L.-M., and Deng, N.-Y. Laplacian unit-hyperplane learning from positive and unlabeled examples. *Information Sciences 314* (2015), 152–168.

[107] Singhal, A. Introducing the knowledge graph: things, not strings. *Official Google Blog 5* (2012).

[108] Smola, A. J., Song, L., and Teo, C. H. Relative novelty detection. In *The 12th International Conference on Artificial Intelligence and Statistics* (2009), pp. 536–543.

[109] Socher, R., Chen, D., Manning, C. D., and Ng, A. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems 26* (2013), pp. 926–934.

[110] Srinivasan, A. The Aleph manual, 2001.

[111] Steinberg, D., and Scott Cardell, N. Estimating logistic regression models when the dependent variable has no variance. *Communications in Statistics-Theory and Methods 21*, 2 (1992), 423–450.

[112] Strack, B., DeShazo, J. P., Gennings, C., Olmo, J. L., Ventura, S., Cios, K. J., and Clore, J. N. Impact of hba1c measurement on hospital readmission rates: analysis of 70,000 clinical database patient records. *BioMed research international 2014* (2014).

[113] Su, J., and Zhang, H. Full bayesian network classifiers. In *Proceedings of the 23rd international conference on Machine learning* (2006), ACM, pp. 897–904.

[114] Suykens, J. A. K., and Vandewalle, J. Least squares support vector machine classifiers. *Neural Processing Letters 9* (1999), 293–300.

[115] Ward, G., Hastie, T., Barry, S., Elith, J., and Leathwick, J. R. Presence-only data and the em algorithm. *Biometrics 65*, 2 (2009), 554–563.

[116] Webb, G. I., Boughton, J. R., and Wang, Z. Not so naive Bayes: Aggregating one-dependence estimators. *Machine Learning 58* (2005), 5–24.

[117] Xu, Z., Qi, Z., and Zhang, J. Learning with positive and unlabeled examples using biased twin support vector machine. *Neural Computing and Applications 25* (2014), 1303–1311.

[118] Yang, P., Li, X., Chua, H.-N., Kwoh, C.-K., and Ng, S.-K. Ensemble positive unlabeled learning for disease gene identification. In *PloS one* (2014).

[119] Yang, P., Li, X., Mei, J.-P., Kwoh, C. K., and Ng, S.-K. Positive-unlabeled learning for disease gene identification. In *Bioinformatics* (2012), pp. 2640–2647.

[120] Yi, J., Hsieh, C.-J., Varshney, K. R., Zhang, L., and Li, Y. Scalable demand-aware recommendation. In *Advances in Neural Information Processing Systems* (2017), pp. 2412–2421.

[121] Yu, H. Single-class classification with mapping convergence. *Machine Learning 61*, 1-3 (2005), 49–69.

[122] Yu, H., Han, J., and Chang, K.-C. PEBL: Web page classification without negative examples. *IEEE Transactions on Knowledge and Data Engineering 16*, 1 (2004), 70–81.

[123] Yu, H., Han, J., and Chang, K. C.-C. PEBL: positive example based learning for web page classification using svm. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge Discovery and Data Mining* (2002), ACM, pp. 239–248.

[124] Yu, S., and Li, C. Pe-puc: A graph based pu-learning approach for text classification. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition* (2007), Springer, pp. 574–584.

[125] Zadrozny, B., and Elkan, C. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (2002), ACM, pp. 694–699.

[126] Zhang, B., and Zuo, W. Reliable negative extracting based on knn for learning from positive and unlabeled examples. *Journal of Computers 4*, 1 (2009), 94–101.

[127] Zhang, D., and Lee, W. S. A simple probabilistic approach to learning from positive and unlabeled examples. In *Proceedings of the fifth Annual UK Workshop on Computational Intelligence (UKCI)* (2005), pp. 83–87.

[128] Zhang, Y., Ju, X., and Tian, Y. Nonparallel hyperplane support vector machine for pu learning. *2014 10th International Conference on Natural Computation (ICNC)* (2014), 703–708.

[129] Zhao, J., Liang, X., Wang, Y., Xu, Z., and Liu, Y. Protein complexes prediction via positive and unlabeled learning of the ppi networks. In *Proceedings of the 13th International Conference on Service Systems and Service Management (ICSSSM)* (June 2016), pp. 1–6.

[130] Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 17* (2004), pp. 321–328.

[131] Zhou, J. T., Pan, S. J., Mao, Q., and Tsang, I. W. Multi-view positive and unlabeled learning. In *Proceedings of the 4th Asian Conference on Machine Learning* (2012).

[132] Zhou, K., Xue, G.-R., Yang, Q., and Yu, Y. Learning with positive and unlabeled examples using topic-sensitive plsa. *IEEE Transactions on Knowledge and Data Engineering 22* (2010), 46–58.

[133] Zupanc, K., and Davis, J. Estimating rule quality for knowledge base completion with the relationship between coverage assumption. In *Proceedings of the Web Conference 2018* (2018), pp. 1–9.

Curriculum vitae

Jessa Bekker was Born in Leuven on October 13th 1990. She received secondary education from Sint-Albertuscollege in Heverlee. Both her bachelors and masters diploma in Engineering Science where obtained at KU Leuven, the bachelor in 2012 and the master, specialized in Computer Science with the option Artificial Intelligence, in 2014. Her Master's thesis was on *Learning Markov Random Fields with Tractable Representations.*

She started her doctoral studies under supervision of Prof. dr. Jesse Davis in the Machine Learning group within the DTAI lab (Declarative Languages and Artificial Intelligence). The Agency for Innovation by Science and Technology in Flanders (IWT), that is now part of the Agency for Innovation and Entrepreneurship (VLAIO), supported her research by awarding her with a PhD fellowship. She was a visiting scholar for four months at the University of California, Los Angeles (UCLA) in 2016, by invitation of Prof. dr. Adnan Darwiche and Prof. dr. ir. Guy Van den Broeck. In her free time, she was busy promoting machine learning and computer science by, amongst other activities, teaching at the S^3 Summer School of Science, being a CoderDojo coach and giving talks about her research field to non-expert audiences at Meetups. On December 13, 2018, she defended her doctoral dissertation entitled *Learning from Positive and Unlabeled Data.*

List of publications

Conference Papers

Bekker, J., Davis, J., Choi, A., Darwiche, A., and Van den Broeck, A. Tractable Learning for Complex Probability Queries. In *Advances in Neural Information Processing Systems 28 (NIPS 2015; Montréal, Québec, Canada; 7-12 December 2015)* (2015a), pp. 2242–2250.

Liang, Y., Bekker, J., and Van den Broeck, G. Learning the Structure of Probabilistic Sentential Decision Diagrams. In *Proceedings of the 33rd Conference on Uncertainty in Artificial Intelligence (UAI 2017; Sydney, Australia; 11-15 August 2017)* (2017).

Bekker, J., and Davis, J. Estimating the Class Prior in Positive and Unlabeled Data through Decision Tree Induction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018, New Orleans, Louisiana, United States; 2-7 February 2018)* (2018a), pp. 2712–2719.

Bekker, J., and Davis, J. Positive and Unlabeled Relational Classification through Label Frequency Estimation. In *Inductive Logic Programming (Revised Selected Papers of ILP 2017; Orléans, France; 4-6 September 2017)* (2018b), pp. 16–30. ✸ *Most promising late-breaking student paper.*

Papers Under Review

Bekker, J., and Davis, J. Beyond the Selected Completely At Random Assumption for Learning from Positive and Unlabeled Data. In *arXiv:1809.03207* (2018d).

Bekker, J., and Davis, J. Learning From Positive and Unlabeled Data: A Survey. In *arXiv:1811.04820* (2018e). (Submitted to Machine Learning Journal)

Workshop Papers

Bekker, J., Hommersom, A., Lappenschaar, M., and Davis, J. Measuring adverse drug effects on multimorbity using tractable Bayesian networks. In *Proceedings of Machine Learning for Health @ NIPS 2016 (NIPS ML4HC 2016; Barcelona, Spain; 9 December 2016)* (2016a).

Bekker, J., and Davis, J. Learning from Positive and Unlabeled Data under the Selected At Random Assumption. In *Proceedings of Learning with Imbalanced Domains @ ECML 2018 (LIDTA 2018; Dublin, Ireland; 10 September 2018)* (2018c).

Abstracts

Bekker, J., Van den Broeck, G., and Davis, J. Ordering-Based Search for Tractable Bayesian Networks. In *Proceedings Workshop of Women in Machine Learning (WiML 2015; Montréal, Québec, Canada; 7 December 2015)* (2015b).

Bekker, J., Choi, A., and Van den Broeck, G. Learning the Structure of Probabilistic SDDs. In *Proceedings Workshop of Women in Machine Learning (WiML 2016; Barcelona, Spain; 5 December 2016)* (2016b).

Bekker, J., and Davis, J. Beyond the Selected Completely At Random Assumption for Learning from Positive and Unlabeled Data. In *Proceedings Workshop of Women in Machine Learning (WiML 2018; Montréal, Québec, Canada; 5 December 2018)* (2018f).

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
DECLARATIVE LANGUAGES AND ARTIFICIAL INTELLIGENCE (DTAI)
Celestijnenlaan 200A box 2402
B-3001 Leuven
jessa.bekker@cs.kuleuven.be
http://people.cs.kuleuven.be/∼jessa.bekker