

Robust Model Selection for Positive and Unlabeled Learning with Constraints

Tong Wei¹, Hai Wang², Wei-Wei Tu² & Yu-Feng Li^{1*}

¹*Nanjing University, Nanjing 210023, China;*

²*4Paradigm Inc., Beijing 100089, China*

Abstract Positive and unlabeled (PU) learning is the problem in which training data contains only PU samples. Although PU learning is widely used in real-world applications, its model selection remains challenging. Specifically, traditional model selection methods are often highly sensitive to the class prior as well as the data size, resulting in human overhead in hyperparameter optimization. In this paper, we present a method called ODE (rObust moDel sElection) for robust model selection in PU learning. Two novel model evaluators based on the integral probability metric and area under the curve, which are free of the class prior, are introduced, and a variance reduction method is further employed to improve the quality of model selection. In addition, we perform model selection under user-defined constraints and propose a fast halving-style searching algorithm to efficiently identify the most promising model configuration. Extensive empirical studies demonstrate that our proposed method performs more robustly and is more computationally efficient than many state-of-the-art methods.

Keywords Positive and Unlabeled Learning, Model Selection, Automated Machine Learning

Citation Tong Wei, et al. Robust Model Selection for Positive and Unlabeled Learning with Constraints. *Sci China Inf Sci*, for review

1 Introduction

Positive and unlabeled (PU) learning is the problem in which only positive and unlabeled data are available. PU learning has attracted increasing attention in the machine learning literature, as such data naturally arises in real applications. For example, in recommender systems [13, 26], purely positive feedback from users is collected while other feedback is unknown. In webpage categorization [30], only a subset of documents of interest are annotated while other documents are unlabeled. Similarly, many other applications involve PU data, such as software clone detection [38], time-series data [28], and disease gene identification [40].

Many PU learning methods have been developed, which can be roughly divided into two categories. i) The first line of research is based on problem transformation. One direct approach is to identify reliable negative samples from unlabeled data, then PU learning is transformed into supervised learning [22, 24, 25, 42], and solved by employing standard supervised learning algorithms. Another approach is to treat unlabeled data as negative and consider hidden positive samples among unlabeled data as mislabeled samples. The PU learning problem can thus be transformed to the label noise learning [20, 33]. ii) The second line of research involves developing unbiased risk estimators. This type of research can be considered as cost-sensitive classification [7, 9, 11, 17]. For unlabeled samples, different costs are compelled when it is predicted as positive or negative. These approaches typically achieve better performance than approaches based on problem transformation. Nevertheless, they rely on knowledge of the class prior which is usually inaccessible in real-world scenarios.

Although many PU learning algorithms have been proposed, it remains difficult to apply standard model selection for a specific PU task, which requires a large amount of human overhead. Many model selection methods [12, 23, 31, 36] in supervised learning and semi-supervised learning have been studied

* Corresponding author (email: liyf@nju.edu.cn)

to select the best configuration; however, it is nontrivial to directly apply them to PU learning. The difficulties are mainly threefold: 1) supervised model selection methods can easily lead to overfitting when applied to PU learning because labeled data is biased and insufficient; 2) existing performance evaluators [9,17] in PU learning are highly sensitive to the class prior and often lead to inferior performance if the class prior is inaccurate, while robustness is essential in weakly supervised learning [43–46]; and 3) model selection methods must be implemented under given resource constraints, which requires an efficient and controllable algorithm.

To address these challenges, in this paper, we develop a robust PU learning algorithm under user-defined budgets called ODE that attempts to identify the best PU model. First, we propose two criteria for model selection in PU learning based on the integral probability metric (IPM) [37] and area under the curve (AUC) [39]. We demonstrate that the IPM and AUC score are equivalent to unbiased performance estimators when used for PU model selection. The two proposed selection criteria are free of the class prior, thereby mitigating the problem of existing methods’ sensitivity to the class prior. We further attempt to reduce the estimation variance by a control variate method. Then, we formulate the robust PU learning model selection procedure as a constrained optimization problem under user-defined budgets. Two types of resource budgets are considered for the efficiency of ODE: the total running time and the number of training samples used. A halving-style searching algorithm is subsequently developed to efficiently identify promising configurations. Extensive empirical studies on several benchmark and real-world datasets reveal the robustness of the proposed method under varying scenarios.

The contributions of this paper are summarized as follows.

- We propose two unbiased model selection criteria without the need for the class prior. We further reduce the estimation variance by the control variate, which is novel in PU learning.
- We formulate robust model selection as a constrained optimization problem and develop a halving-style algorithm to efficiently identify the most promising configuration.
- We apply the proposed ODE to real-world problems in recommender systems, and it achieves more stable performance than the state-of-the-art methods.

The remainder of this paper is arranged as follows. We start with a brief review of related work. Then, we introduce unbiased risk evaluators and present the proposed robust PU model selection method. Thereafter, we present empirical results followed by the conclusion.

2 Related Work

2.1 Model Selection in PU Learning

Assessing the performance of a learned model constitutes a crucial part of machine learning. However, in some domains, only PU samples are available, which prohibits the use of most standard evaluation metrics.

Previous literature on PU model selection is mainly based on the random selection assumption, that is, observed positive samples are selected from the underlying positive class distribution completely at random. Therefore, the empirical risk of classifiers for negative samples can be approximated by risks of PU data. Based on this assumption, several unbiased risk estimators have been proposed [9,11,17,29,39]. Additionally, [8] proposed using a non-convex surrogate loss function in PU learning to avoid a biased solution. Furthermore, [5] proposed an approach to estimate any metric based on contingency tables. This approach can essentially be reduced to estimating the fraction of (latent) positive samples in the unlabeled set. In addition, [14] demonstrated that many standard evaluation metrics can be correctly recovered with knowledge of the class prior. Nonetheless, their method is restricted to non-traditional classifiers, that is, it directly treats unlabeled data as negative. The above-mentioned estimation methods can facilitate theoretical analysis; however, their main shortcoming is that they require the class prior, which is usually unavailable in real-world applications.

2.2 Hyperparameter Optimization

Hyperparameter optimization is a central part of automated machine learning, which is becoming an emerging area in both industry and academia. Frequently used hyperparameter optimization strategies include simple search and optimization from samples [6,31].

Grid search and random search [2] are two common simple search approaches, with the former being the most traditional method of hyperparameter tuning. To obtain the optimal hyperparameter settings, grid search must enumerate every possible configuration in searching space. Discretization is necessary when the search space is continuous. Random search randomly samples configurations in the searching space. As its name suggests, random search picks one configuration completely at random for evaluation at each step and updates the best hyperparameter setting. On the aspect of optimization from samples, it iteratively generates new configurations based on previously evaluated sampled configurations. Bayesian optimization [3, 18, 34] is the most popular approach. This approach develops a probabilistic model, which maps configurations to their expected performance with uncertainty and defines an acquisition function to balance exploration and exploitation during the search. At each iteration, a new sample is generated by optimizing the acquisition function and used to update the probabilistic model once it is evaluated. However, most hyperparameter search approaches require both positive and negative data for model selection. Therefore, existing model selection methods cannot be directly applied to PU learning due to the absence of negative samples.

3 Proposed PU Risk Estimator

To alleviate the problem of the absence of negative samples, it is necessary to develop alternative performance evaluation metrics. In this section, we first provide the basic setting of PU learning. Then, we introduce two representative unbiased risk estimators from existing literature, i.e., uPU [9] and nnPU [17]. Finally, we propose two equivalent performance evaluators: the IPM_{PU} and AUC_{PU} .

3.1 Problem Setting

Given N samples $\mathcal{D} = \{\mathbf{x}_i, s_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $s_i \in \{0, 1\}$, \mathbf{x}_i is regarded as a positive sample if $s_i = 1$; otherwise it is regarded as an unlabeled sample. We denote the set of positive samples \mathcal{P} and unlabeled set $\mathcal{U} = \mathcal{D} - \mathcal{P}$. The positive set \mathcal{P} is sampled independently from the underlying joint density $p(\mathbf{x} | y = 1)$, and the unlabeled set \mathcal{U} is sampled from a mixture density $p(\mathbf{x}) = \pi p(\mathbf{x} | y = 1) + (1 - \pi)p(\mathbf{x} | y = 0)$, where π represents the *class prior probability* and y represents the true class label of instance \mathbf{x} . As in conventional supervised learning, PU learning aims to learn a classifier $p(y = 1 | \mathbf{x})$ that distinguishes positive and negative data.

3.2 Unbiased PU Risk Estimator

Evaluating PU models using only PU data is very challenging, thus, [9] proposes an unbiased risk estimator called uPU. Let $R_p^+(f) = \mathbb{E}_p[\ell(f(\mathbf{x}), +1)]$ and $R_n^-(f) = \mathbb{E}_n[\ell(f(\mathbf{x}), -1)]$ represent the empirical risk of positive and negative samples, respectively. For brevity, we use supervised (PN) learning as the short for supervised learning in the rest of this paper. In PN learning, owing to the availability of positive and negative samples, empirical risk of classifier f , i.e., $R(f)$, can be directly approximated by

$$\widehat{R}_{\text{pn}}(f) = \pi \widehat{R}_p^+(f) + (1 - \pi) \widehat{R}_n^-(f). \quad (1)$$

In PU learning, although $\widehat{R}_n^-(f)$ is unavailable, it can be indirectly approximated owing to the property $(1 - \pi)R_n^-(f) = R_u^-(f) - \pi R_p^-(f)$, and we can obtain the approximation of the PU learning risk as follows:

$$\widehat{R}_{\text{pu}}(f) = \pi \widehat{R}_p^+(f) - \pi \widehat{R}_p^-(f) + \widehat{R}_u^-(f), \quad (2)$$

which can be calculated from PU data.

3.3 Non-negative PU Risk Estimator

Based on the observation that the value of Equation (2) can be negative which often results in overfitting, [17] proposed a non-negative risk estimator for PU learning called nnPU:

$$\widetilde{R}_{\text{pu}}(f) = \pi \widehat{R}_p^+(f) + \max\left\{0, \widehat{R}_u^-(f) - \pi \widehat{R}_p^-(f)\right\}. \quad (3)$$

Briefly, nnPU works by explicitly constraining the training risk of uPU to be non-negative. This estimator helps evaluate candidate models more precisely. However, both uPU and nnPU rely on knowledge of the class prior π , which is usually unknown in practical applications [1]. Although several approaches have been proposed to estimate π from PU data [1, 10, 11, 27, 32], inaccurate estimation usually results in severe performance degradation. In Figure 1, we illustrate the performance of nnPU with the true and estimated value of class prior. As the class-frequency $c = \frac{|\mathcal{P}|}{|\mathcal{P}| + \pi|\mathcal{U}|}$ varies, nnPU with the true value of π consistently outperforms nnPU with the estimated value of π , especially when the cardinality of the positive set \mathcal{P} is small. We state this empirical finding in Observation 1.

Observation 1. Inaccurate estimation of the class prior degrades performance particularly when the number of positive samples is limited, thus it is critical to study model evaluators that are free of the class prior.

3.4 Proposed Equivalent PU Risk Estimators

Although uPU and nnPU can be used for performance evaluation, these unbiased evaluators are susceptible to the estimation of class prior π in real-world problems. In what follows, two equivalent model selection criteria, namely, the IPM and AUC, are described in a unified form without knowledge of π . We first introduce the IPM and AUC score in PN learning and then extend them to the PU setting.

3.4.1 Integral Probability Metric (PM)

We first introduce the IPM [37], which is a widely used metric between two probability measures P and Q defined on \mathcal{X} as follows.

$$\text{IPM}(P, Q; \mathcal{F}) := \sup_{f \in \mathcal{F}} \left| \int_{\mathcal{X}} f(x) dP(x) - \int_{\mathcal{X}} f(x) dQ(x) \right|, \quad (4)$$

where \mathcal{F} is a set of bounded measurable functions. In PN learning, Lemma 1 provides an important theoretical result connecting the IPM with empirical risk minimization [35].

Lemma 1 (Relationship between L -risk and IPM). Let \mathcal{F} be a symmetric hypothesis space and $L(y, \alpha) = \frac{2y\alpha}{y(1-2t)-1}$ where $0 < t < 1$ and $\alpha \in \mathbb{R}$. Then, we have

$$\inf_{f \in \mathcal{F}} R_L(f) = 1 - \text{IPM}(p(\mathcal{X}|y=1), p(\mathcal{X}|y=-1); \mathcal{F}). \quad (5)$$

The IPM used for model selection can be defined as follows:

$$\text{IPM} = \max_{f \in \mathcal{F}} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}_P} [f(\mathbf{x})] - \mathbb{E}_{\mathbf{x}' \in \mathcal{X}_N} [f(\mathbf{x}')], \quad (6)$$

where $\mathbb{E}_{\mathbf{x} \in \mathcal{X}_P} [f(\mathbf{x})]$ and $\mathbb{E}_{\mathbf{x}' \in \mathcal{X}_N} [f(\mathbf{x}')]$ represents the expected prediction score of positive and negative samples, respectively. The IPM can be intuitively explained as maximizing the margin between the expected score over positive and negative samples. It should be noted that (6) can be easily rewritten as

$$\text{IPM} = \max_{f \in \mathcal{F}} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}_P} \left[\mathbb{E}_{\mathbf{x}' \sim \mathcal{X}_N} [f(\mathbf{x}) - f(\mathbf{x}')] \right]. \quad (7)$$

3.4.2 AUC Score

The AUC is one of the most frequently used evaluation metrics in practice, and is defined as

$$\mathbb{E}_{\mathbf{x} \sim \mathcal{X}_P} \left[\mathbb{E}_{\mathbf{x}' \sim \mathcal{X}_N} \left[\ell(f(\mathbf{x}) - f(\mathbf{x}')) \right] \right], \quad (8)$$

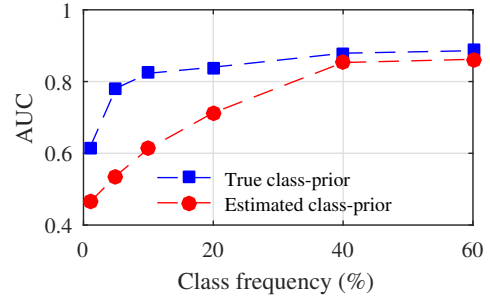


Figure 1 AUC of nnPU [17] with true and estimated value of the class prior.

where ℓ indicates ℓ_{0-1} , which is defined as

$$\ell_{0-1}(z) = \begin{cases} 1 & z > 0 \\ 1/2 & z = 0 \\ 0 & z < 0 \end{cases}. \quad (9)$$

It should be noted that maximizing Equation (8) in the hypothesis space is identical to the IPM when $\ell(z) = z$. Given the definition of the IPM and AUC, we now present our main results that both the IPM and AUC score serve as tailored model evaluators for PU learning. The result is stated in Theorem 1.

Theorem 1 (Equivalent PU Model Evaluator). Given PU dataset $\mathcal{P} \cup \mathcal{U}$, the IPM and AUC score calculated for \mathcal{P} and \mathcal{U} are equivalent estimations in model selection to their supervised counterparts with a linear transformation.

Proof. According to the definition of our evaluators and the linearity of the expectation, we have

$$\begin{aligned} & \frac{1}{1-\pi} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}_{\mathcal{P}}} \mathbb{E}_{\mathbf{x}' \sim \mathcal{X}_{\mathcal{U}}} \ell(f(\mathbf{x}) - f(\mathbf{x}')) \\ &= \frac{1}{1-\pi} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}_{\mathcal{P}}} [\pi \mathbb{E}_{\bar{\mathbf{x}} \sim \mathcal{X}_{\mathcal{P}}} \ell(f(\mathbf{x}) - f(\bar{\mathbf{x}})) + (1-\pi) \mathbb{E}_{\hat{\mathbf{x}} \sim \mathcal{X}_{\mathcal{N}}} \ell(f(\mathbf{x}) - f(\hat{\mathbf{x}}))] \\ &= \frac{\pi}{1-\pi} \underbrace{\mathbb{E}_{\mathbf{x} \sim \mathcal{X}_{\mathcal{P}}} \mathbb{E}_{\bar{\mathbf{x}} \sim \mathcal{X}_{\mathcal{P}}} \ell(f(\mathbf{x}) - f(\bar{\mathbf{x}}))}_I + \underbrace{\mathbb{E}_{\mathbf{x} \sim \mathcal{X}_{\mathcal{P}}} \mathbb{E}_{\hat{\mathbf{x}} \sim \mathcal{X}_{\mathcal{N}}} \ell(f(\mathbf{x}) - f(\hat{\mathbf{x}}))}_{II} \end{aligned}$$

The first term marked by I describes the measurement between \mathcal{P} and hidden positive samples in \mathcal{U} . Term I can be estimated via \mathcal{P} since $\forall \mathbf{x} \in \mathcal{P}$ are obtained from the underlying distribution of positive samples completely at random. Term II is the measurement between \mathcal{P} and hidden negative samples in \mathcal{U} , which is the supervised counterpart of the IPM and AUC. Letting $\ell(z) = z$ be the identity function, we obtain

$$\text{IPM}_{\text{PN}} = \frac{1}{1-\pi} \text{IPM}_{\text{PU}},$$

where IPM_{PU} and IPM_{PN} represent the IPM in PU learning and PN learning, respectively. Letting $\ell(\cdot)$ be ℓ_{0-1} , AUC_{PU} is an equivalent estimation of AUC_{PN} and we obtain

$$\text{AUC}_{\text{PN}} = \frac{1}{1-\pi} \text{AUC}_{\text{PU}} - \frac{\pi}{2(1-\pi)},$$

where AUC_{PU} and AUC_{PN} , represent the AUC score of PU learning and PN learning, respectively. Note that both $\frac{1}{1-\pi}$ and $\frac{\pi}{2(1-\pi)}$ are constants. In other words, IPM_{PU} and AUC_{PU} preserve the ordering of candidate models with respect to IPM_{PN} and AUC_{PN} , respectively.

3.5 Variance Reduction by Control Variate

We have demonstrated that IPM_{PU} and AUC_{PU} are equivalent performance estimators of IPM_{PN} and AUC_{PN} , respectively. However, given an unknown parameter ς and its unbiased estimator z , i.e., $\mathbb{E}[z] = \varsigma$. It will not be accurate if the variance $\text{Var}[z]$ is high. To reduce the variance, we can adopt the control variate method [41] and identify another related unbiased estimator z' such that $\mathbb{E}[z'] = \varsigma$, where ς is the parameter that z' tries to estimate. We construct a new estimator parameterized by a constant η as follows:

$$z^* = z + \eta(z' - \varsigma). \quad (10)$$

It is straightforward to demonstrate that z^* remains unbiased owing to the linear property of the expectation operation:

$$\mathbb{E}[z^*] = \mathbb{E}[z] + \eta \mathbb{E}[z' - \varsigma] = \varsigma + \eta(\mathbb{E}[z'] - \mathbb{E}[\varsigma]) = \varsigma. \quad (11)$$

The variance of z^* can be computed as:

$$\begin{aligned} \text{Var}[z^*] &= \text{Var}[z + \eta(z' - \varsigma)] \\ &= \eta^2 \text{Var}[z'] + 2\eta \text{Cov}(z, z') + \text{Var}[z], \end{aligned} \quad (12)$$

which is a quadratic form of η and achieves a global optimum $\min \text{Var}[z^*] = (1 - \rho_{z,z'}^2) \text{Var}[z]$ when $\eta = -\frac{\text{Cov}(z,z')}{\text{Var}[z']}$, where $0 \leq |\rho_{z,z'}| \leq 1$ is the correlation coefficient of z and z' . Therefore, the variance is reduced, i.e., $\text{Var}[z^*] \leq \text{Var}[z]$.

To reduce the variance $\text{Var}[R_{\text{PU}}]$ of interest, where R represents the IPM or AUC, we construct a non-traditional classifier (NTC) that discriminates samples between \mathcal{P} and \mathcal{U} as $g: \mathbf{x} \in \mathbb{R}^d \rightarrow \tilde{y} \in [0, 1]$. The NTC treats samples in \mathcal{U} as negative and \mathcal{P} as positive. Let $q_p(\tilde{y})$ and $q_u(\tilde{y})$ be the probability density functions of the distribution of $g(\mathbf{x})$ with $\mathbf{x} \sim p(\mathbf{x} | y = 1)$ and $\mathbf{x} \sim p(\mathbf{x})$, respectively. Note that $p(\mathbf{x} | y = 1)$ and $p(\mathbf{x})$ are the distributions in which \mathcal{P} and \mathcal{U} are sampled. The expectation of the density ratio of $\tilde{y} \sim q_p$ is

$$\mathbb{E}_{\tilde{y} \sim q_p} \frac{q_u(\tilde{y})}{q_p(\tilde{y})} = \int \frac{q_u(\tilde{y})}{q_p(\tilde{y})} q_p(\tilde{y}) d\tilde{y} = 1. \quad (13)$$

By substituting this into Equation (10) as the control variate z' for the target model selection criteria estimation, we obtain

$$\begin{aligned} R_{\text{PU}}^* &= \frac{1}{|\mathcal{P}||\mathcal{U}|} \sum_{\mathbf{x} \in \mathcal{P}} \sum_{\mathbf{x}' \in \mathcal{U}} \ell(f(\mathbf{x}) - f(\mathbf{x}')) \\ &\quad + \frac{\eta}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \left(\frac{1 - g(\mathbf{x})}{g(\mathbf{x})} - \mathbb{E}_{\tilde{y} \sim q_p} \frac{q_u(\tilde{y})}{q_p(\tilde{y})} \right) \\ &= \frac{1}{|\mathcal{P}||\mathcal{U}|} \sum_{\mathbf{x} \in \mathcal{P}} \sum_{\mathbf{x}' \in \mathcal{U}} \ell(f(\mathbf{x}) - f(\mathbf{x}')) + \frac{\eta}{|\mathcal{P}|} \sum_{\mathbf{x} \in \mathcal{P}} \left(\frac{1 - g(\mathbf{x})}{g(\mathbf{x})} - 1 \right). \end{aligned}$$

Here, we have $\eta = -\frac{\widehat{\text{Cov}}(R_{\text{PU}}, \frac{q_u(\tilde{y})}{q_p(\tilde{y})})}{\widehat{\text{Var}}[\frac{q_u(\tilde{y})}{q_p(\tilde{y})}]}$, which can be computed from the validation set.

4 PU Model Selection with Budget

After developing performance evaluators, we propose a robust model selection approach.

4.1 Formulation and Proposed Solution

Given a configuration set Ω , we aim to search for a configuration \mathcal{C} that maximizes the testing performance under the searching budget. Let $\text{Performance}(\mathcal{C}, \mathcal{D})$ denote the generalization performance of configuration \mathcal{C} trained on dataset \mathcal{D} . We can then formulate our objective as an optimization problem as follows:

$$\begin{aligned} \max_{\mathcal{C} \in \Omega} \quad & \text{Performance}(\mathcal{C}, \mathcal{D}) \\ \text{s.t.} \quad & \text{accumulated_budget} \leq \text{target_budget}. \end{aligned} \quad (14)$$

Specifically, function $\text{Performance}(\cdot, \cdot)$ in Problem (14) can be substituted by the IPM and AUC. We consider that the user-defined budget B is given as the total number of samples for training and B' as the total time allowed for model selection. B and B' can be provided either individually or together, and the budget can be manually assigned by practical resource limits, such as $B \leftarrow N$ or $B \leftarrow 2N$. Since it is infeasible to train all models from the start using the full data and then select the best models, we resort to pruning-during-training which simplifies the optimization objective and reduces the complexity. Given the resource constraints, we develop a new model selection algorithm that iteratively eliminates inferior configurations. By rewriting Problem (14) and we obtain

$$\begin{aligned} \max_{\mathcal{C} \in \Omega} \quad & \text{Performance}(\mathcal{C}, \mathcal{D}) \\ \text{s.t.} \quad & \Omega_1 = \Omega \\ & \Omega_{t+1} \subsetneq \Omega_t, \quad \forall 1 \leq t \leq T \\ & \sum_{t=1}^T |\Omega_t| \cdot |\mathcal{D}_t^{\text{train}}| \leq B \\ & \sum_{k=1}^t \sum_{\mathcal{C}_k \in \Omega_k} \text{Timing}(\mathcal{C}_k, \mathcal{D}_k^{\text{train}}) \leq B'. \end{aligned} \quad (15)$$

Initially, Ω defines a set of candidate configurations. At the t -th iteration, Ω_t consists of a subset of configurations selected to continue to the subsequent iteration. We use $\text{Timing}(\mathcal{C}, \mathcal{D})$ to denote the time

consumption of training \mathcal{C} on \mathcal{D} . To solve Problem (15), we develop a successive halving [15, 21] style algorithm that early-prunes candidate configurations by evaluating trained models using partial training data on the validation set. At each iteration, it uniformly allocates a budget to a set of configurations, evaluates the performance of all configurations, discards the worst half, and repeats until one configuration remains. Algorithm 1 summarizes the details of the proposed ODE.

After pruning less promising configurations, the algorithm allocates exponentially more resources to the remaining configurations. However, directly discarding the worst half of the configurations may be too aggressive, especially in the early stages. Therefore, we take advantage of parameter η to control the pace of pruning. Physically, $1/\eta$ corresponds to the ‘‘age’’ of the trained models: when $1/\eta$ is small, only several models with the largest losses will be discarded; as η increases, a large proportion of candidates will gradually be discarded. To fully investigate the influence of pruning schedules, we consider the following five different halving methods:

- *Successive halving (SH)*: The plain successive-halving method that discards a half of the worst configurations, i.e., $\eta_t = \frac{1}{2}$.
- *log-schedule*: The pruning parameter is set to $\eta_t = (1 - \exp(-\frac{t}{T} * 5)) * (1 - \frac{1}{K}) + \frac{1}{K}$. This parameter increases most rapidly at the beginning of the model selection.
- *exp-schedule*: The pruning parameter is set to $\eta_t = \exp((\frac{t}{T} - 1) * 5) * (1 - \frac{1}{K}) + \frac{1}{K}$. The parameter increases most rapidly at the end of the model selection.
- *linear-schedule*: The pruning parameter is set to $\eta_t = \frac{t}{T} * (1 - \frac{1}{K}) + \frac{1}{K}$. The parameter increases linearly as the pruning progresses.
- *selfpaced-schedule*: The pruning parameter is set to a constant between 0 and 1.

Remark. To apply configuration pruning, we assume a partial order relationship between the training set pair \mathcal{D}_i and \mathcal{D}_j , $\mathcal{D}_i \subset \mathcal{D}_j$. Then, for any configuration \mathcal{C} , we have $\text{Performance}(\mathcal{C}, \mathcal{D}_i) \leq \text{Performance}(\mathcal{C}, \mathcal{D}_j)$. As $|\mathcal{D}_j|$ increases, it approximates the true generalization performance more accurately.

4.2 Analysis

In the following, we first demonstrate that the algorithm never takes a total number of samples that exceeds the budget B by setting $K = |\Omega|$ and $T = \lceil -\log_{\mu}^{\eta_0 K} \rceil + 1$:

$$\sum_{t=0}^{T-1} |\Omega_t| \cdot \left\lfloor \frac{B}{T |\Omega_t|} \right\rfloor \leq \sum_{t=0}^{T-1} \frac{B}{T} \leq B. \quad (16)$$

The rationale for setting $T = \lceil -\log_{\mu}^{\eta_0 K} \rceil + 1$, where μ is the discounting factor of the initial pruning parameter η_0 in *selfpaced-schedule*, is because that $\lceil -\log_{\mu}^{\eta_0 K} \rceil$ is the minimum integer solution of $\eta_0 \mu^T K \leq 1$. In other words, exactly one candidate configuration is retained after the last iteration. For other halving schedules, we return the best configuration in the last iteration. Next, we consider the performance of the algorithm in terms of identifying the best configuration. An intuitive result is provided in [15, 21], as stated in Theorem 2.

Lemma 2 (Performance Guarantee [15]).

Let $\nu_i = \lim_{\tau \rightarrow \infty} \ell_{i,\tau}$, $\bar{\gamma}(t) = \max_{i=1,\dots,K} \gamma_i(t)$ and

$$\begin{aligned} z &= 2 \lceil \log_2(n) \rceil \max_{i=2,\dots,n} i \left(1 + \bar{\gamma}^{-1} \left(\frac{\nu_i - \nu_1}{2} \right) \right) \\ &\leq 2 \lceil \log_2(n) \rceil \left(n + \sum_{i=2,\dots,n} \bar{\gamma}^{-1} \left(\frac{\nu_i - \nu_1}{2} \right) \right). \end{aligned}$$

If budget $B > z$, then the best arm is returned from the algorithm.

Here, the limit $\nu_i = \lim_{\tau \rightarrow \infty} \ell_{i,\tau}$ is assumed to exist and is equal to $-\frac{1}{|\text{VAL}|} \sum_{i=1}^{|\text{VAL}|} R(f_{\theta,t}(\mathbf{x}_i), y_i)$, where VAL denotes the validation set. Without loss of generality, we assume $\nu_1 < \nu_2 \leq \dots \leq \nu_K$. Let $\gamma_i(\tau)$ be the bound of the approximation error of $\ell_{i,\tau}$ with respect to ν_i as a function of τ . γ_i is defined as the point-wise smallest, non-increasing function of τ such that

$$|\ell_{i,\tau} - \nu_i| \leq \gamma_i(\tau), \quad \forall \tau, \quad (17)$$

Algorithm 1 The Framework of ODE**Input:** \mathcal{D} : positive and unlabeled data $\mathcal{D} = \{\mathbf{x}_i, s_i\}_{i=1}^N$ Ω : configuration set η_t : pruning schedule, $0 < \eta_t < 1$ **Process:**

- 1: Set number of iteration $T = \lceil -\log_{\mu}^{\eta_0} |\Omega| \rceil + 1$
- 2: **for** $t = 0, 1, \dots, T - 1$ **do**
- 3: For each configuration \mathcal{C} in Ω_t , feed $r_t = \lfloor \frac{B}{T|\Omega_t|} \rfloor$ training samples to \mathcal{C} .
- 4: Train models for configurations in Ω_t and calculate the IPM or AUC R_{i,r_t} for the i -th configuration in Ω_t on validation set VAL.
- 5: Let σ_t be a bijection on Ω_t such that $R_{\sigma_t(1),r_t} \leq R_{\sigma_t(2),r_t} \leq \dots \leq R_{\sigma_t(|\Omega_t|),r_t}$
- 6: Set $\Omega_{t+1} = \{\mathcal{C}_i \in \Omega_t, 1 \leq i \leq |\Omega_t| : R_{\sigma_t(i),r_t} \geq R_{\sigma_t(\lfloor \eta_t |\Omega_t| \rfloor),r_t}\}$
- 7: Update η_t
- 8: **end for**
- 9: **return** Ω_T

and $\gamma_i^{-1}(\alpha) = \min \{\tau \in \mathbb{N} : \gamma_i(\tau) \leq \alpha\}, \forall i \in [K]$.

5 Experiments

In this section, we examine the performance of the proposed ODE method on benchmark datasets and real-world tasks. Specifically, we evaluate our algorithm in the following four aspects:

- (i) **Shortcomings of existing methods:** we demonstrate that the supervised model selection method is prone to overfitting and existing PU model evaluators are sensitive to the class prior.
- (ii) **Comparisons on benchmark datasets:** we compare ODE with existing model selection methods on benchmark datasets to demonstrate the robustness of ODE.
- (iii) **Real-world task:** we evaluate the effectiveness of the proposed approach on recommender systems, which is a natural application of PU learning.
- (iv) **Ablation studies:** we examine how pruning schedules, the variance reduction method, and different types of budgets affect the performance of ODE.
- (v) **Efficiency:** we compare the efficiency of ODE to that of existing model selection methods.

Table 1 Dataset statistics

Dataset	#Training	#Testing	#Feature	Class ratio	Temporal	Domain	Scale
twonorm	5,180	2,220	20	0.496	×	computer	small
spambase	3,220	1,381	57	0.399	×	email	small
krvskp	2,237	959	36	0.486	×	game	small
sat	2,218	913	36	0.495	×	score	small
ethn	1,840	790	30	0.503	×	image	small
titanic	1,540	661	3	0.324	×	disaster	small
texture	1,418	608	19	0.492	×	sediment	small
zhihu	758,339	533,785	62	0.015	✓	text	large
myhug	470,910	323,192	69	0.046	✓	video	large

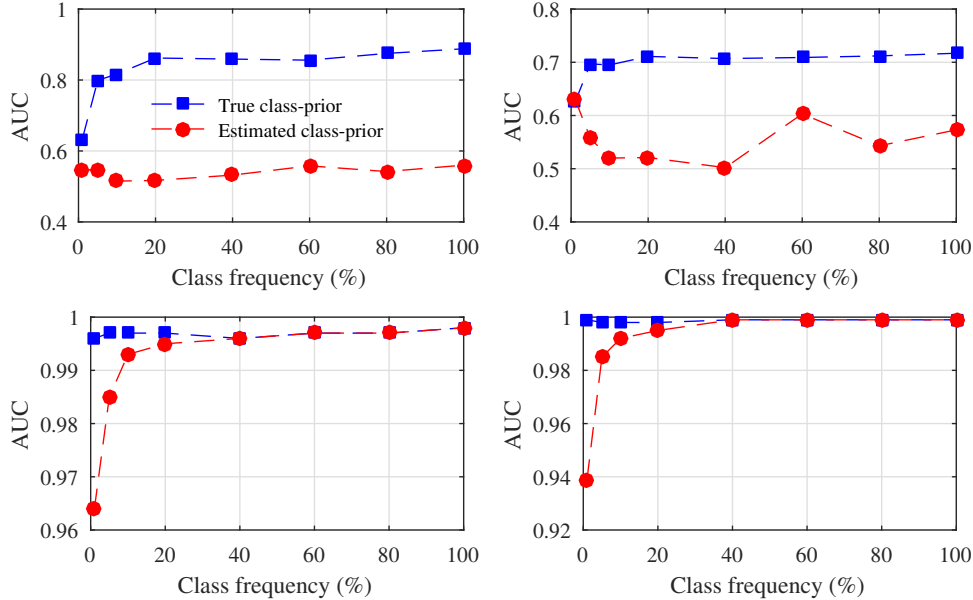


Figure 2 Performance of uPU (left two figures) and nnPU (right two figures) with the true and estimated class prior on four representative datasets.

The comprehensive statistics of the datasets are listed in Table 1. As demonstrated, two larger datasets in real-world problems contain temporal features and are extremely class-imbalanced in comparison with the benchmark datasets. To simulate PU learning problems in practice, we constructed PU data with varying class-frequency c . More specifically, we ran all methods by setting $c \in \{0.01, 0.05, 0.1, 0.2\}$ through random downsampling positive samples. For each c , we repeated the experiment 10 times and reported the average performance. It is observed in Table 2 that using the IPM as the model evaluator typically led to more robust performance than the AUC, therefore we report results of the IPM in following experiments. In the implementation, we applied our model selection algorithm to well established PU learning algorithms, i.e., uPU and nnPU. In particular, the hyperparameters search space is detailed in Table 3.

5.1 Competing Methods

The following methods are compared:

- *Supervised*: This method trains a supervised classifier with the lightGBM [16]. It takes unlabeled data as negative and transforms the problem to standard supervised learning using hyperopt [19] for hyperparameter optimization.
- *WSVM*: This method proposed in [11]. This method treats each unlabeled instance as a combination of positive and negative samples. The LibSVM [4] package is used to train the model.
- *uPU*: This method proposed in [9] uses unbiased PU learning loss.
- *nnPU*: This method proposed in [17] uses non-negative PU learning loss. It is an improved version of uPU which usually overfits because the value of uPU loss can become negative.

Table 2 Comparison between the IPM and AUC as model selection criteria. The shown numbers are AUC (%). We set the class frequency $c = 0.01$.

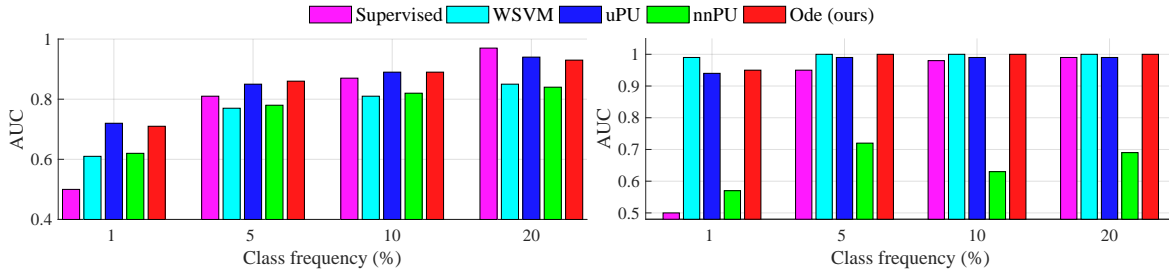
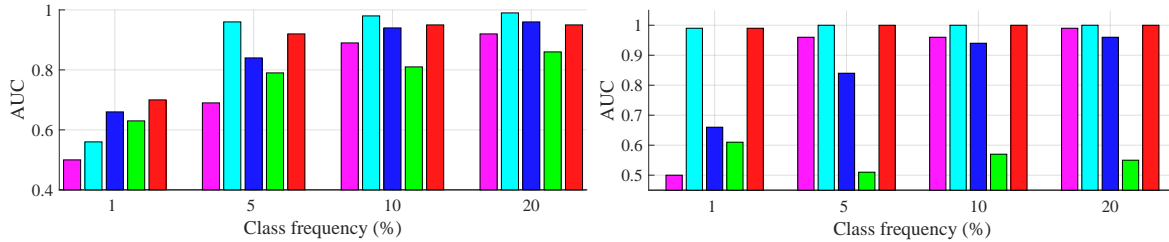
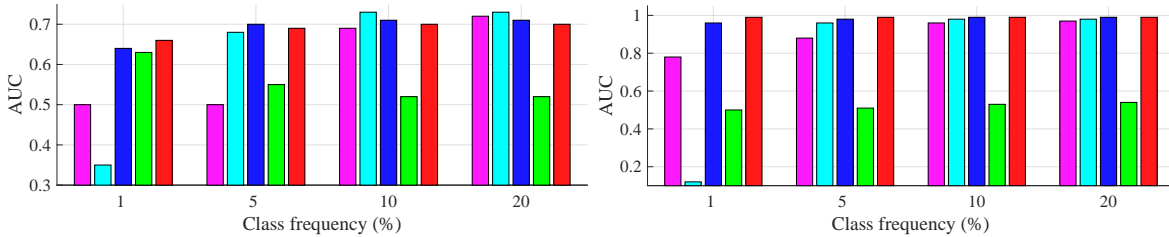
Dataset	ethn	titanic	krvskp	sat	texture	spambase	twonorm
ODE (IPM)	70.29	66.17	70.89	99.73	99.99	82.75	99.65
ODE (AUC)	70.53	65.64	69.88	99.86	89.49	80.89	96.87

Table 3 A summary of hyperparameter search space.

Algorithm	Parameter	Feasible space	Cardinality
uPU	λ	$[-5, 5]$	50
	basis	{“lm”, “gauss”}	2
nnPU	γ	$[0, 1]$	50
	β	$[0, 1]$	10

5.2 Sensitivity to Class Prior of uPU and nnPU

We showcase the performance of uPU and nnPU with varying class-frequency in Figure 2 to emphasize the importance of developing class prior free model evaluators. As stated in Observation 1, the non-negative risk estimation used in nnPU is extremely sensitive to inaccurate estimation of class prior. nnPU degenerates its performance severely when the class-frequency is small. Similarly, the unbiased risk estimation developed in uPU degenerates its performance as much as 50% in the worst cases. The results demonstrate the value of our developed model evaluators.

**Figure 3** Performance comparison on *kruskp* (left) and *texture* (right) datasets with varying class-frequencies.**Figure 4** Performance comparisons on *ethn* (left) and *sat* (right) datasets with varying class-frequency.**Figure 5** Performance comparison on *titanic* (left) and *twonorm* (right) datasets with varying class-frequencies.

5.3 Comparison Results

To verify the effectiveness of ODE, we compare it with the above-mentioned baselines and leading PU learning algorithms. The comparison results are reported in Figures 3, 4, and 5, where the means of testing performance based on 10 random samplings are shown. The results demonstrate that the

supervised model selection baseline performed poorly when only a modest set of positive samples was provided. It achieved competitive results when additional positive samples were given because sampled unlabeled samples are more likely to be negative; however, its performance was extremely unstable. This validates that directly applying a supervised hyperparameter search baseline is prone to overfitting when positive data is scarce. We implemented WSVM using a Gaussian kernel and fit the data well in most cases except when the number of positive samples was limited, facing similar problems to those of the supervised baseline. Since a deep model is employed for nnPU, it is unsurprising that nnPU generally had worse performance on small datasets (e.g., ethn, krvskp) than other algorithms. uPU generally achieves the second-best performance across different settings; however, it performed poorly on some datasets. In comparison, the proposed ODE was more robust across many datasets and class-frequencies.

5.4 Real-World Tasks

Here, we examine the performance of ODE on real-world problems in recommender systems and report the comparison results in Figure 6. The main purpose of these tasks is to predict whether users are interested in documents, videos, or goods. As the WSVM algorithm has considerable training complexity, we employed a linear kernel, replacing the Gaussian kernel. In this part of experiment, we compared uPU, nnPU, and ODE using estimated class prior $\hat{\pi}$ by [1]. Particularly, on the zhihu dataset, uPU was highly sensitive to an inaccurate estimation value of π , and its performance decreased by up to 20% in terms of AUC. In contrary, our ODE outperformed all competing methods with varying class-frequency. On the myhug dataset, ODE and uPU both achieved relatively robust performance; however, that of ODE is slightly higher in the cases $c = 0.1$ and $c = 0.2$. These results verify the robustness of the proposed ODE, which not only achieves promising results on benchmark datasets but also succeeded in real-world tasks.

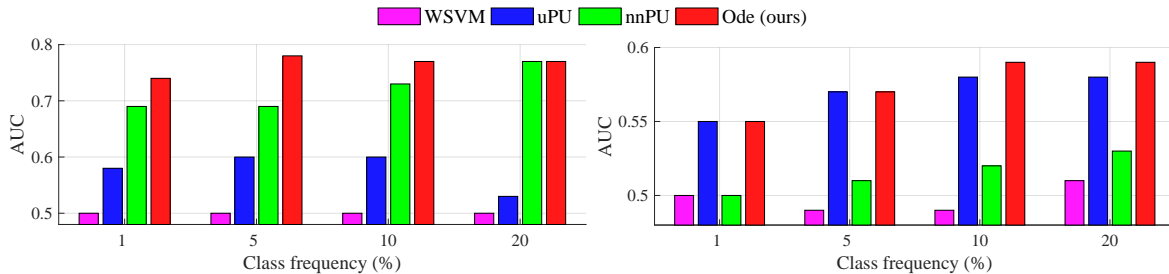


Figure 6 Performance comparison on real-world tasks, zhihu (left) and myhug (right), with varying class-frequencies

Table 4 Ablation study for the configuration pruning strategy. The displayed numbers are AUC (%).

Pruning schedule	ethn	titanic	krvskp	sat
SH	71.52	63.99	70.52	91.85
log-schedule	71.90	66.09	70.11	90.24
exp-schedule	72.20	66.30	69.74	99.74
linear-schedule	72.55	65.65	70.09	96.74
selfpaced-schedule	70.64	66.17	71.01	99.73

5.5 Ablations on Pruning Schedule

Here, we study the effect of the successive-halving method on four datasets with different pruning strategies. As illustrated in Table 4, *selfpace-schedule* and *exp-schedule* achieved the best performance when compared to the baseline SH. More specifically, *selfpaced-schedule* improved the AUC score from 91.85% to 99.73% on *sat*. This indicates that *selfpaced-schedule* and *exp-schedule* postponed releasing candidate configurations to the end of the model selection phase. Finally, *selfpaced-schedule* was adopted in the proposed ODE for its stable performance.

Table 5 A summary of average training time in seconds (ratio) for PU learning algorithms based on 10 replications. The best results are in bold.

Dataset	WSVM	uPU	nnPU	ODE
ethn	100.3 (6.4)	15.6 (1.0)	29.2 (1.8)	15.4 (1.0)
sat	101.3 (15.7)	13.5 (2.1)	7.4 (1.1)	6.4 (1.0)
spambase	683.8 (45.0)	36.7 (2.4)	16.9 (1.1)	15.1 (1.0)
texture	29.7 (26.7)	7.4 (4.0)	7.1 (6.4)	1.1 (1.0)
titanic	22.6 (64.8)	1.4 (17.3)	7.7 (22.0)	0.3 (1.0)
twonorm	1182.4 (273.7)	15.2 (3.5)	20.6 (4.7)	4.3 (1.0)

5.6 Ablations on Control Variate

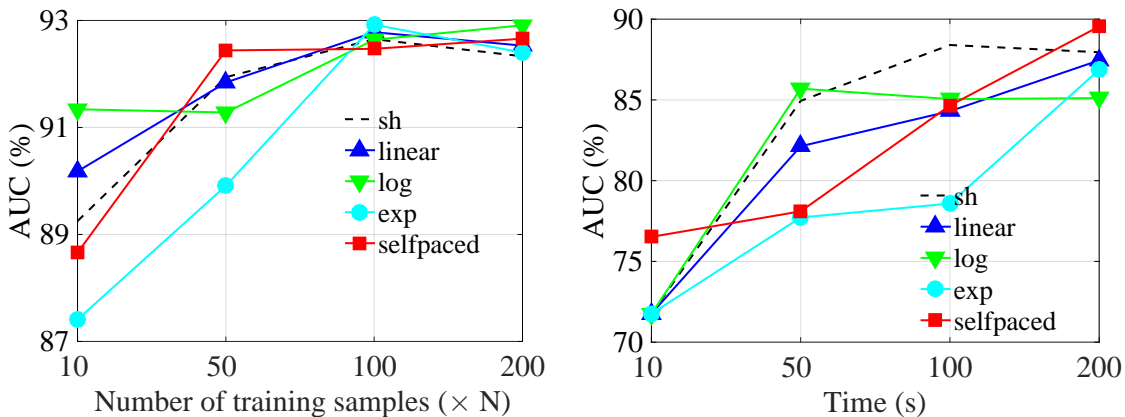
Here, we conducted experiments on four datasets to evaluate the effectiveness of the control variate. The results are reported in Table 6. It can be observed that ODE (w/ control variate) achieved significantly higher performance than ODE (w/o control variate) in three out of four datasets, which empirically confirms the efficacy of the control variate method in improving estimation robustness.

Table 6 Ablation study for control variate. The displayed numbers are AUC (%).

Method	ethn	titanic	krvskp	sat
ODE (w/o control variate)	69.33	67.69	68.15	96.54
ODE (w/ control variate)	72.20	66.30	69.74	99.74

5.7 Ablations on Different Types of Budgets

Here, we investigate the performance of ODE with different types of resources. We consider two types of resources, i.e., the *number of training samples used* and the *running time*. As illustrated in Figure 7, all pruning strategies typically exhibit improved performance as the number of resources increases. It is thus unsurprising to observe that the improvement in performance generally accelerated in the beginning and slowed at the end. This is because, with abundant resources, most strategies can identify promising configurations.

**Figure 7** Performance of different types of pruning strategies with varying number of resources.

5.8 Efficiency of Proposed ODE

In addition to evaluating the performance, we also examine the efficiency of the proposed ODE in comparison with competing methods, i.e., WSVM, uPU, and nnPU. The comparison results are reported in Table 5. For fair comparison, we set the sample budget $B = N$ and did not impose the time budget B' .

We fixed the class-frequency $c = 0.6$ for all datasets. Table 5 reveals that ODE consistently outperformed WSVM, uPU, and nnPU in all cases. In particular, ODE was $72\times$ faster than WSVM on average, over $5\times$ faster than uPU, and $6.2\times$ faster than nnPU. This demonstrates that our proposed algorithm not only achieves robust performance, but can also effectively identify promising configurations.

6 Conclusion

In this work, we formulate model selection in PU learning under resource budgets as a constrained optimization problem. To evaluate model performance based on PU data, two unbiased model evaluators are proposed that are free of the class prior. To facilitate efficient model selection, we propose a fast halving-style algorithm with five different pruning schedules. Finally, we present the results of extensive numerical studies on both benchmark datasets and real-world problems in recommender systems, which demonstrate the robustness and efficiency of the proposed algorithm.

Acknowledgements The authors want to thank the associate editor and reviewers for helpful comments and suggestions. This research was supported by the NSFC (61772262).

References

- 1 Jessa Bekker and Jesse Davis. 2018. 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-18), New Orleans, Louisiana, USA*. 2712–2719.
- 2 James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* 13, Feb (2012), 281–305.
- 3 James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems*. 2546–2554.
- 4 Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM TIST* 2, 3 (2011), 27:1–27:27.
- 5 Marc Claesen, Jesse Davis, Frank De Smet, and Bart De Moor. 2015. Assessing binary classifiers using only positive and unlabeled data. *arXiv preprint arXiv:1504.06837*.
- 6 Andrew R. Conn, Katya Scheinberg, and Lu'is N. Vicente. 2009. *Introduction to Derivative-Free Optimization*. MPS-SIAM series on optimization, Vol. 8. SIAM.
- 7 Marthinus Christoffel du Plessis, Gang Niu, and Masashi Sugiyama. 2014. Analysis of Learning from Positive and Unlabeled Data. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems, Montreal, Quebec, Canada*. 703–711.
- 8 Marthinus C Du Plessis, Gang Niu, and Masashi Sugiyama. 2014. Analysis of learning from positive and unlabeled data. In *Advances in neural information processing systems*. 703–711.
- 9 Marthinus Christoffel du Plessis, Gang Niu, and Masashi Sugiyama. 2015. Convex Formulation for Learning from Positive and Unlabeled Data. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France*. 1386–1394.
- 10 Marthinus Christoffel du Plessis, Gang Niu, and Masashi Sugiyama. 2017. Class-prior estimation for learning from positive and unlabeled data. *Machine Learning* 106, 4 (2017), 463–492.
- 11 Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA*. 213–220.
- 12 Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and robust automated machine learning. In *Advances in neural information processing systems*. 2962–2970.
- 13 Cho-Jui Hsieh, Nagarajan Natarajan, and Inderjit S. Dhillon. 2015. PU Learning for Matrix Completion. In *ICML*, Vol. 37. 2445–2453.
- 14 Shantanu Jain, Martha White, and Predrag Radivojac. 2017. Recovering True Classifier Performance in Positive-Unlabeled Learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, California, USA*. 2066–2072.
- 15 Kevin G. Jamieson and Ameet Talwalkar. 2016. Non-stochastic Best Arm Identification and Hyperparameter Optimization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain (JMLR Workshop and Conference Proceedings)*, Vol. 51. JMLR.org, 240–248.
- 16 Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30, Long Beach, CA, USA*. 3149–3157.
- 17 Ryuichi Kiryo, Gang Niu, Marthinus Christoffel du Plessis, and Masashi Sugiyama. 2017. Positive-Unlabeled Learning with Non-Negative Risk Estimator. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, Long Beach, CA, USA*. 1674–1684.
- 18 Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. 2016. Fast bayesian optimization of machine learning hyperparameters on large datasets. *arXiv preprint arXiv:1605.07079*.
- 19 Brent Komer, James Bergstra, and Chris Eliasmith. 2014. Hyperopt-sklearn: automatic hyperparameter configuration for scikit-learn. In *ICML workshop on AutoML*, Vol. 9. Citeseer.
- 20 Wee Sun Lee and Bing Liu. 2003. Learning with Positive and Unlabeled Examples Using Weighted Logistic Regression. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), Washington, DC, USA*. 448–455.
- 21 Lisha Li, Kevin G. Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2017. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. *Journal of Machine Learning Research* 18 (2017), 185:1–185:52.
- 22 Xiaoli Li and Bing Liu. 2003. Learning to classify texts using positive and unlabeled data. In *Proceedings of International Joint Conference on Artificial Intelligence*, Vol. 3. 587–592.

- 23 Yu-Feng Li, Hai Wang, Tong Wei, and Wei-Wei Tu. 2019. Towards automated semi-supervised learning. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*.
- 24 Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and S Yu Philip. 2003. Building Text Classifiers Using Positive and Unlabeled Examples.. In *ICDM*, Vol. 3. 179–188.
- 25 Bing Liu, Wee Sun Lee, Philip S. Yu, and Xiaoli Li. 2002. Partially Supervised Classification of Text Documents. In *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), Sydney, Australia*. 387–394.
- 26 J. McAuley, R. Pandey, and J. Leskovec. 2015. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Sydney, Australia, 785–794.
- 27 Aditya Krishna Menon, Brendan van Rooyen, Cheng Soon Ong, and Bob Williamson. 2015. Learning from Corrupted Binary Labels via Class-Probability Estimation. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France (JMLR Workshop and Conference Proceedings)*, Vol. 37. JMLR.org, 125–134.
- 28 Minh Nhut Nguyen, Xiaoli Li, and See-Kiong Ng. 2011. Positive Unlabeled Learning for Time Series Classification. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Spain*. 1421–1426.
- 29 Gang Niu, Marthinus Christoffel du Plessis, Tomoya Sakai, Yao Ma, and Masashi Sugiyama. 2016. Theoretical Comparisons of Positive-Unlabeled Learning against Positive-Negative Learning. In *Advances in Neural Information Processing Systems 29, Barcelona, Spain*. 1199–1207.
- 30 Ioannis Partalas, Aris Kosmopoulos, Nicolas Baskiotis, Thierry Artieres, George Paliouras, Eric Gaussier, Ion Androutsopoulos, Massih-Reza Amini, and Patrick Galinari. 2015. LSHTC: A benchmark for large-scale text classification. *arXiv preprint arXiv:1503.08581*.
- 31 Yao Quanming, Wang Mengshuo, Jair Escalante Hugo, Guyon Isabelle, Hu Yi-Qi, Li Yu-Feng, Tu Wei-Wei, Yang Qiang, and Yu Yang. 2018. Taking human out of learning applications: A survey on automated machine learning. *arXiv preprint arXiv:1810.13306*.
- 32 Harish G. Ramaswamy, Clayton Scott, and Ambuj Tewari. 2016. Mixture Proportion Estimation via Kernel Embeddings of Distributions. In *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York City, NY, USA*. 2052–2060.
- 33 Hong Shi, Shaojun Pan, Jian Yang, and Chen Gong. 2018. Positive and Unlabeled Learning via Loss Decomposition and Centroid Estimation.. In *Proceedings of International Joint Conference on Artificial Intelligence*. 2689–2695.
- 34 Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*. 2951–2959.
- 35 Bharath K Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, Gert RG Lanckriet, et al. 2012. On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics* 6 (2012), 1550–1599.
- 36 Chris Thornton, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. 2013. Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA*. ACM, 847–855.
- 37 Ilya O. Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. 2018. Wasserstein Auto-Encoders. In *Proceedings of the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada*.
- 38 Huihui Wei and Ming Li. 2018. Positive and Unlabeled Learning for Detecting Software Functional Clones with Adversarial Training.. In *Proceedings of International Joint Conference on Artificial Intelligence*. 2840–2846.
- 39 Zheng Xie and Ming Li. 2018. Semi-Supervised AUC Optimization Without Guessing Labels of Unlabeled Data. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA*. AAAI Press, 4310–4317.
- 40 Peng Yang, Xiaoli Li, Hon-Nian Chua, Chee-Keong Kwoh, and See-Kiong Ng. 2014. Ensemble positive unlabeled learning for disease gene identification. *PloS one* 9, 5.
- 41 Kaichao You, Ximei Wang, Mingsheng Long, and Michael Jordan. 2019. Towards Accurate Model Selection in Deep Unsupervised Domain Adaptation. In *International Conference on Machine Learning*. 7124–7133.
- 42 Tong Wei, Feng Shi, Hai Wang, Wei-Wei Tu, and Yu-Feng Li. 2020. MixPUL: Consistency-based Augmentation for Positive and Unlabeled Learning. *arXiv preprint arXiv:2004.09388*.
- 43 Tong Wei, Lan-Zhe Guo, Yu-Feng Li, and Gao, Wei. 2018. Learning safe multi-label prediction for weakly labeled data. *Machine Learning* 107, 4 703–725.
- 44 Zhi-Hua Zhou. 2019. Abductive learning: Towards bridging machine learning and logical reasoning. *Science China Information Sciences* 62, 7 76–101.
- 45 Yu-Feng Li and De-Ming Liang. 2019. Safe semi-supervised learning: a brief introduction. *Frontiers of Computer Science* 13, 4 669–676.
- 46 Dietterich, Thomas G. 2019. Robust artificial intelligence and robust human organizations. *Frontiers of Computer Science* 13, 1 1–3.