Adversarial Balancing for Causal Inference

Michal Ozery-Flato [*1] **Pierre Thodoroff** [*2] **Matan Ninio** [1] **Michal Rosen-Zvi** [1] **Tal El-Hay** [1]

Abstract

Biases in observational data of treatments pose a major challenge to estimating expected treatment outcomes in different populations. An important technique that accounts for these biases is reweighting samples to minimize the discrepancy between treatment groups. We present a novel reweighting approach that uses bi-level optimization to alternately train a discriminator to minimize classification error, and a balancing weights generator that uses exponentiated gradient descent to maximize this error. This approach borrows principles from generative adversarial networks (GANs) to exploit the power of classifiers for measuring two-sample divergence. We provide theoretical results for conditions in which the estimation error is bounded by two factors: (i) the discrepancy measure induced by the discriminator; and (ii) the weights variability. Experimental results on several benchmarks comparing to previous state-of-the-art reweighting methods demonstrate the effectiveness of this approach in estimating causal effects.

1. Introduction

Causal inference deals with estimating expected outcomes for treatments or interventions. The gold standard for causal inference studies is randomized controlled trials (RCTs), in which treatment and control groups come from the same data distribution, due to a randomized treatment assignment process. However, RCTs are often costly, sometimes impractical to implement, and may raise ethical questions. An appealing alternative is to infer expected treatment outcomes using the abundance of observational treatment data. Alas, in such data treatment populations are likely to differ from each other. This difference, or bias, between treatment groups and the lack of knowledge on the treatment assignment mechanism, hinder the inference of expected outcomes for the treatment in the entire population.

A common approach for inferring expected treatment outcomes from observational treatment data is by balancing the bias between treatment groups via reweighting of the individuals in these groups. The challenging task of computing balancing weights has applications not only for causal inference, but also for transfer learning, and is highly related to the field of density ratio-estimation (see Related work section).

Motivated by the immense success of generative adversarial networks (GANs) in producing simulated data that highly resembles real world samples, we propose a novel framework that adapts the objective of GANs to the task of generating balancing weights. Similar to GANs, our framework is based on a two-player game that involves a discriminator that measures the bias, or discrepancy, between two data samples, and a generator that aims to produce data indistinguishable, by the discriminator, from another given dataset. The key difference from GANs is that our data generator produces "new" data by reweighting a given dataset, where the weights are obtained by a simple step of exponentiated gradient ascent step on the discriminator loss. Using this framework allows us to harness the complete arsenal of classification methods to the task of generating balancing weights. We evaluated the performance of this algorithm on a range of published causal-inference benchmarks, and assessed the ability to select an appropriate classifier for the input datasets in a standard cross-validation routine.

2. Problem setup

Consider a population where each individual received a single treatment from a finite set of treatments $\mathcal{A}$. The received treatment and the resulting outcome for every individual are indicated by the variables A and Y, respectively. For every treatment $a \in \mathcal{A}$, Y^a denotes the potential outcome for the treatment. The variable Y^a is observed only when $A = a$. Let X denote the vector of observed pre-treatment covariates used to characterize the individuals. Let $\mathcal{D}$ be the distribution over $(X, A, \{Y^a\}_{a\in\mathcal{A}})$ in the population. The expected outcome of a treatment $a \in \mathcal{A}$ in the population is:

$$\mathbb{E}_{Y^a\sim\mathcal{D}}\left[Y^a\right] = \mathbb{E}_{X\sim\mathcal{D}(X)}\left[\mathbb{E}_{Y^a\sim\mathcal{D}(Y^a|X)}\left[Y^a|X\right]\right] . \quad (1)$$

[*]Equal contribution [1]IBM Research [2]McGill University; This work was done while the author was an intern in IBM Research-Haifa. Correspondence to: Tal El-Hay <talelh@il.ibm.com>.

For brevity, we denote $\mathbb{E}\left[Y^a\right] \equiv \mathbb{E}_{Y^a \sim \mathcal{D}}\left[Y^a\right]$.

The goal of many observational studies is to estimate $\mathbb{E}\left[Y^a\right]$ from a finite data sample from $\mathcal{D}$. However, Y^a is observed only in the subpopulation that actually received treatment a, where the distribution over X is $\mathcal{D}(X|A=a) \neq \mathcal{D}(X)$. To overcome this hurdle, we employ the standard assumptions of *strong ignorability*: $Y^a \perp\!\!\!\perp A|X$, and *positivity*: $0 < p(A=a|X=x) < 1, \forall a \in \mathcal{A}$ (Rosenbaum & Rubin, 1983). Strong ignorability, often stated as "no hidden confounders", means that the observed covariates contain all the information that may affect treatment assignment. These assumptions allow rewriting Equation 1:

$$\mathbb{E}\left[Y^a\right] = \mathbb{E}_{X\sim\mathcal{D}(X)}\left[\mathbb{E}_{Y\sim\mathcal{D}(Y|X,A=a)}\left[Y|X,A=a\right]\right] . \tag{2}$$

Equation 2 suggests that $\mathbb{E}\left[Y^a\right]$ can be estimated by a sample from the subpopulation corresponding to $A=a$ under the condition that its distribution over X is $\mathcal{D}(X)$. A common approach to handle this sampling challenge is to use a weighting function $\omega^a(X)$ such that $\mathcal{D}(X|A=a)\omega^a(X) = \mathcal{D}(X)$. The weighting function that satisfies this condition is clearly $\omega^a(X) = \frac{\mathcal{D}(X)}{\mathcal{D}(X|A=a)} = \frac{\mathcal{D}(A=a)}{\mathcal{D}(A=a|X)}$ and therefore

$$\mathbb{E}\left[Y^a\right] = \mathbb{E}_{X\sim\mathcal{D}(X|a)}\left[\omega^a(X)\mathbb{E}_{Y\sim\mathcal{D}(Y|X,A=a)}\left[Y|X,A=a\right]\right] \tag{3}$$

Given a finite sample $S = \{(x_i, a_i, y_i)\}_{i=1}^N$ from $\mathcal{D}$ we would like to produce weights w_i for each $i \in \{i : a_i = a\}$ that approximate $\omega^a(X_i)$. Following Equation 3, given such weights we estimate $\mathbb{E}\left[Y^a\right]$ by:

$$\widehat{\mathbb{E}}[Y^a] = \sum_{i:a_i=a} w_i y_i \, . \tag{4}$$

3. Background on adversarial framework for learning generative models

The adversarial framework, which was introduced by Goodfellow et al. (Goodfellow et al., 2014), aims to learn a generative model of an unknown distribution $\mathcal{D}_{\text{data}}$ using a class of discriminators that gauge the similarity between data distributions. This framework can be described as a game in which a generator simulates data and a discriminator tries to distinguish samples of true data from simulate data samples. The generator employs generative models with an input random variable Z from a predefined distribution $\mathcal{D}_Z$ and a deterministic mapping $g(\mathbf{z})$ to the data space $\mathcal{X}$. Simulated data are generated by sampling data from $\mathcal{D}_Z$ and transforming them through g. At the end of each round of the game, the generator observes the predictions of the discriminator and updates the model for $g(\mathbf{z})$. Given the generator model $g(\mathbf{z})$ the prediction model of the discriminator, $d(\mathbf{x})$, attempts to minimize the expected classification error in the real and simulated samples :

$$L(g,d) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\text{data}}}\left[l(d(\mathbf{x}),1)\right] + \mathbb{E}_{\mathbf{z}\sim\mathcal{D}_Z}\left[l(d(g(\mathbf{z})),0)\right] \tag{5}$$

where l is the loss function. Given the prediction model, $d(\mathbf{x})$, the generator attempts to maximize the expected error, and its objective is to find

$$g^* = \arg\max_g \left(\min_d L(g,d)\right) \tag{6}$$

Examples for loss functions are the Log-loss $l(d(x),c) = c\cdot\log d(x) + (1-c)\cdot\log(1-d(x))$, which was used in (Goodfellow et al., 2014); and the 0-1 loss [1], $l_{0\text{-}1}(d(x),c) = \mathbb{1}\left[\mathbb{1}\left[d(x) > \frac{1}{2}\right] \neq c\right]$, which was used in (Gutmann et al., 2014; Mohamed & Lakshminarayanan, 2016) for likelihood-free inference and in (Lopez-Paz & Oquab, 2016) for classifier two-sample tests. In the next section we adapt the adversarial framework and its key principle of maximizing the discrimination loss to the task of generating balancing weights.

4. Adversarial balancing weights

In this section we present our adversarial framework for generating balancing weights, and a novel algorithm that applies it. Similar to GAN, our goal is to generate a sample that resembles data coming from a distribution $\mathcal{D}(X)$. However, while in the original GAN framework the generated sample is *simulated* by applying a transformation on unlimited random data, our balancing framework is constrained to *reweight* a finite samples from the distribution $\mathcal{D}(X|a)$. More generally we consider the problem of reweighting a data sample coming from a source distribution $\mathcal{D}_S$ on X such that it becomes indistinguishable from a sample of a target distribution $\mathcal{D}_T$. The input to our problem are two finite samples from the two distributions:

$$S = \{\mathbf{x}_i\}_{i=1}^n \sim (\mathcal{D}_S)^n;\; T = \{\mathbf{x}_i\}_{i=n+1}^{n+n'} \sim (\mathcal{D}_T)^{n'} \, .$$

This is a general framework for balancing with respect to any target population; therefore, it can be used to estimate different types of causal effects. For example, the *average treatment effect* (ATE) is defined as $\mathbb{E}_{X\sim\mathcal{D}(X)}\left[Y^{a=1}\right] - \mathbb{E}_{X\sim\mathcal{D}(X)}\left[Y^{a=0}\right]$. In this case we estimate $\mathbb{E}_{X\sim\mathcal{D}(X)}\left[Y^a\right]$ using $\mathcal{D}_S := \mathcal{D}(X|a)$ and $\mathcal{D}_T := \mathcal{D}(X)$. Another example is the *average treatment effect in the treated* (ATT), which is defined as $\mathbb{E}_{X\sim\mathcal{D}(X|A=1)}\left[Y^{a=1}\right] - \mathbb{E}_{X\sim\mathcal{D}(X|A=1)}\left[Y^{a=0}\right]$. In the latter example, we estimate $\mathbb{E}_{X\sim\mathcal{D}(X|A=1)}\left[Y^0\right]$ by reweighting a sample from the distribution $\mathcal{D}_S := \mathcal{D}(X|A=0)$ and using $\mathcal{D}(X|A=1)$ as the target distribution.

[1] $\mathbb{1}\left[c\right]$ is the indicator function which is 1 is predicate c if true, and 0 otherwise.

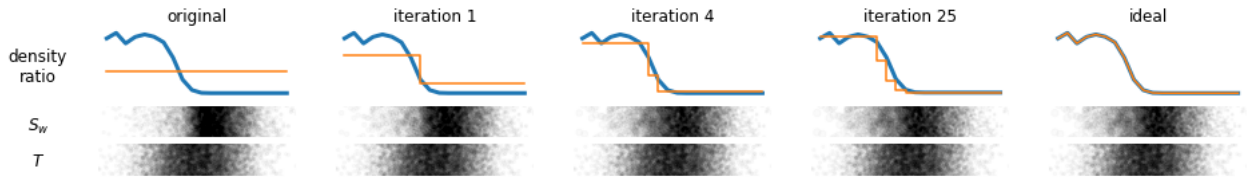


Figure 1. Illustration of the adverasarial balancing algorithm. The thick blue line represents the density ratio $w^*(x) \equiv \frac{\mathcal{D}_T(x)}{\mathcal{D}_S(x)}$. The thin orange line represents the estimated weights in different iterations. The bottom strip is a scatter plot of samples drawn from the target distribution (the density in the x-axis is $\mathcal{D}_T$ and uniform in the y-axis for visualization purpose). This strip is constant in all iterations. The top strip shows a scatter plot of the source distribution where the size of point is proportional to its weight, thereby visualizing S_w. The algorithm starts by uniform weights and a high bias. At each iterations the weights are updated according to a classifier that minimizes L_n to maximize the loss in the next iteration.

4.1. Discrepancy objective

Let $w(X)$ be a non-negative function that reweights samples from $\mathcal{D}_S$ resulting in a new distribution, $\mathcal{D}_{S_w}(X) = w(X)\mathcal{D}_S(X)$. For $\mathcal{D}_{S_w}$ to be a valid distribution, $w(X)$ must satisfy the constraint

$$\mathbb{E}_{X\sim\mathcal{D}_S}\left[w(X)\right] = 1\,. \tag{7}$$

Using a similar loss to the one in Equation 5 and replacing the generator distribution by $\mathcal{D}_{S_w}(X)$, we obtain

$$L(w,d) = \mathbb{E}_{X\sim\mathcal{D}_T}\left[l(d(X),1)\right] + \mathbb{E}_{X\sim\mathcal{D}_S}\left[w(X)l(d(X),0)\right] \tag{8}$$

We can confine the representation of $w(X)$ to a family of models and optimize the loss with respect to this family. However, for the estimation problem defined in Equation 4, it suffices to infer point estimates of $w(X)$ for the given sample from $\mathcal{D}_S$. We denote such point estimates by $w_i \equiv w(x_i)$ and use them in Equation 7 to obtain the following normalization constraint:

$$\frac{1}{n}\sum_{i=1}^{n} w_i = 1\,. \tag{9}$$

Similar to the constraint in 7. The discriminator error becomes

$$L_n(\mathbf{w},d) = \frac{1}{n'}\sum_{i=n+1}^{n+n'} l(d(X_i),1) + \frac{1}{n}\sum_{i=1}^{n} w_i l(d(X_i),0)\,, \tag{10}$$

where $\mathbf{w} \equiv (w_1,\ldots,w_n)$. Note that Equation 10 is the empirical error of the discriminator, when the samples from $\mathcal{D}_S$ and $\mathcal{D}_T$ are given the same importance. The aim of the discriminator is to minimize the error in Equation 10. Weights $\mathbf{w}$ leading to large errors of the discriminator imply its inability to distinguish between the sample T and the weighted samples S. We formulate the objective of the adversarial balancing framework as solving the following optimization problem:

$$\mathbf{w}^* = \arg\max_{\frac{\mathbf{w}}{n}\in\Delta}\left(\min_d L_n(\mathbf{w},d)\right) \tag{11}$$

where Δ be the unit simplex $\Delta = \{\mathbf{u}\in\mathbb{R}^n : \mathbf{u}\geq 0,\ \sum_{i=1}^n u_i = 1\}$.

4.2. Weight learning algorithm

To search for a solution to the max-min objective in Equation 11, we propose the following iterative process. At each step we train a discriminator to minimize the empirical loss of Equation 10. We then update the weights w_i to increase this loss using a single step of exponentiated gradient descent (Kivinen & Warmuth, 1997), which maintains the weight normalization constraint. Figure 1 illustrates this process.

We define the augmented labeled dataset by assigning a class label 0 and weights to $\mathcal{D}_S$, and a class label 1 and uniform weights to $\mathcal{D}_T$:

$$\left\{(\mathbf{x}_i,0;w_i)\right\}_{i=1}^{n} \bigcup \left\{(\mathbf{x}_i,1;w_i=1)\right\}_{i=n+1}^{n+n'}\,. \tag{12}$$

Note that the uniform weights we assigned to $\mathcal{D}_T$ will not be modified by our algorithm. The discriminator predicts the class label, C, of the samples in $\mathcal{D}_T$ using a classifier $d(\mathbf{x})\in\mathcal{F}$, where $\mathcal{F}$ is a predefined classifier family. Recall that the final objective of the adversarial framework is to find $\mathbf{w}$ that maximizes the objective in Equation 11. Following Equation 10, for a fixed classifier d, the generator's loss is linear in $\mathbf{w}$ and $\frac{\partial L_n}{\partial w_i} = l(d(\mathbf{x}_i),0)$ is constant. To maximize the objective in Equation 11, which refers to *any* classifier from the considered family, we update the weights using a single step of exponentiated gradient ascent:

$$w_i^{t+1} = n\frac{w_i^t \exp\left(\alpha\cdot l(d(x_i),0)\right)}{\sum_j w_j^t \exp\left(\alpha\cdot l(d(x_j),0)\right)} \tag{13}$$

Algorithm 1 shows the complete details of the adversarial framework for non-parametric generation of balancing weights. Only the weights for $\mathcal{D}_S$ are updated, while weights for the sample units in $\mathcal{D}_T$ are constantly set to 1. In each iteration the sum of weights in $\mathcal{D}_S$ equals n, ensuring the same importance with respect to the discriminator loss. The predictions of the discriminator (Step 6 in Algorithm

Algorithm 1 Adversarial balancing weights

input $S = \{\mathbf{x}_i\}_{i=1}^{n}, T = \{\mathbf{x}_i\}_{i=n+1}^{n+n'}$
parameters classifier family $\mathcal{H}$, update rule for learning rate α, number of iterations n_{iter}, loss function l
output Balancing weight vector $\mathbf{w}$ for $\mathcal{D}_S$

$\mathbf{c} \leftarrow [\underbrace{0,0,\ldots,0}_{n\text{-times}},\underbrace{1,1,\ldots,1}_{n'\text{-times}}]$
$\mathbf{w} \leftarrow [\underbrace{1,1,\ldots,1}_{n\text{-times}},\underbrace{1,1,\ldots,1}_{n'\text{-times}}]$
$w_i \leftarrow \frac{n}{n'} w_i \;\; , \forall i > n$ {equal class importance}
for n_{iter} iterations **do**
  $\widehat{\mathbf{c}} \leftarrow$ get_predictions$(\mathcal{H}, [S,T], \mathbf{c}, \mathbf{w})$
  $w_i \leftarrow w_i \exp\left(\alpha_i \cdot l(\widehat{c}_i, 0)\right) \;\; , \forall i \leq n$
  $w_i \leftarrow n \frac{w_i}{\sum_{j \in \mathcal{I}_a} w_j} \;\; , \forall i \leq n$
end for
return $\mathbf{w}[i \leq n]$

1) should preferably be obtained with cross validation, to better approximate the generalization error in Equation 8.

The choice of the classifiers family and its hyper parameters is important to enable us to approximate the minimal loss defined in Equation 8 with the empirical loss in Equation 10. On the one hand, we would like to reduce the estimation error of L_n due to over-fitting of the classifier. On the other hand, the family of classifiers should be rich enough to distinguish between "non-similar" (weighted) datasets. In Section 7, we describe our experiments with different classification algorithms, ranging from the low-capacity logistic regression to the large-capacity class of neural networks. We test the ability of our framework to tackle the challenge of bias-variance trade-off by applying a preliminary step of hyper-parameter selection using cross-validation, prior to running Algorithm 1.

5. Theoretical results

In this section we provide theoretical results for the 0-1 loss function, formulating the link between the classifier family and the estimation error. We start with introducing the two-sample divergence measure induced by the discriminator error, namely the $\mathcal{H}$-divergence. We use this divergence measure in the bound we provide for the estimation error.

5.1. The two-sample $\mathcal{H}$-divergence

Let $\mathcal{H}$ denote the family of binary classifiers $h : X \to 0, 1$ considered by the discriminator. Similar to (Kifer et al., 2004; Ben-David et al., 2007; 2010) we define the $\mathcal{H}$-divergence between S_w and T as:

$$d_{\mathcal{H}}(\mathbf{w}) = 2 \max_{h \in \mathcal{H}} \left| \frac{1}{n} \sum_{i=1}^{n} w h(x_i) - \frac{1}{n'} \sum_{i=n+1}^{n+n'} h(x_i) \right| . \quad (14)$$

That is, the $\mathcal{H}$-divergence relies on the capacity of the hypothesis class $\mathcal{H}$ to distinguish between examples from S_w and T. Adapting a result from (Ben-David et al., 2007), the following lemma links the $\mathcal{H}$-divergence between S_w and T, with the minimal error of the discriminator that attempts to classify them, as defined in Equation 10.

Lemma 1. If $\mathcal{H}$ is symmetric, that is, for every $h \in \mathcal{H}$, the inverse hypothesis $1 - h$ is also in $\mathcal{H}$, and the loss function l in is the $0 - 1$ loss, then

$$d_{\mathcal{H}}(\mathbf{w}) = 2 \left[1 - \min_{h \in \mathcal{H}} L_n(\mathbf{w}, h) \right]$$

Proof. See appendix. □

In the following we assume that $\mathcal{H}$ is symmetric.

5.2. Bounds for estimation error

Suppose that $\mathcal{D}_S$ are assigned with labels $\{y_i\}_{i=1}^{n}$ corresponding to the observed treatment outcomes Y^a. Denote $f_Y(X) \equiv \mathbb{E}_{\mathcal{D}_T}[Y^a|X]$. In this section we provide a bound for the estimation error for the case where $f_Y(\mathbf{x})$ is bounded, with $M_Y = \sup_{\mathbf{x}} |f_Y(\mathbf{x})|$. The estimation error can be decomposed to two sources of error by adding and subtracting terms, using $\mathbb{E}_{\mathcal{D}_T}[Y^a] = \mathbb{E}_{\mathcal{D}_T(X)}[f_Y(X)]$ and the triangle inequality:

$$\left| \frac{1}{n} \sum_{i=1}^{n} w_i y_i - \mathbb{E}_{X \sim \mathcal{D}_T}[Y^a] \right| \leq \quad (15)$$
$$\left| \frac{1}{n} \sum_{i=1}^{n} w_i y_i - \frac{1}{n'} \sum_{i=n+1}^{N} f_Y(x_i) \right|$$
$$+ \left| \frac{1}{n'} \sum_{i=n+1}^{N} f_Y(x_i) - \mathbb{E}_{X \sim \mathcal{D}_T}[f_Y(X)] \right|$$

The second term, which does not depend on the weights $\mathbf{w}$, relates to the approximation of the expected value of $f_Y(X)$ by a sample mean of $f_Y(\mathbf{x})$. Following Hoeffding's inequality it is bounded by $2M_Y \sqrt{\frac{\ln \frac{2}{\delta}}{2n}}$ with probability $1-\delta$. For the remaining of this section we focus on bounding the first term, which involves the weights $\mathbf{w}$. We start by decomposing this term using the triangle inequality

$$\left| \frac{1}{n} \sum_{i=1}^{n} w_i y_i - \frac{1}{n'} \sum_{i=n+1}^{N} f_Y(x_i) \right| \leq \left| \frac{1}{n} \sum_{i=1}^{n} w_i (y_i - f_Y(x_i)) \right|$$
$$+ \left| \frac{1}{n} \sum_{i=1}^{n} w_i f_Y(x_i) - \frac{1}{n'} \sum_{i=n+1}^{N} f_Y(x_i) \right| \quad (16)$$

The first term in Equation 16 depends on the variability of the outcome Y^a *given* X, as well as on the weights $\mathbf{w}$. We will address this term in Theorem 1 below. The second term in this equation depends on the difference between the weighted average of $f_Y(X)$ on S and the unweighted average of $f_Y(X)$ on T. Following Equation 14, if $f_Y \in \mathcal{H}$ then this term is smaller or equal to half of the $\mathcal{H}$-divergence. The following lemma extends this observation for a larger family than $\mathcal{H}$, noted as $C(\mathcal{H})$, which contains all functions that can be represented by a bounded linear combination of members in $\mathcal{H}$. More formally we define this larger family as $C(\mathcal{H}) = \{f : f = \sum_j \alpha_j h_j(x) \text{ s.t. } h_j \in \mathcal{H} \text{ and } \sum_j |\alpha_j| \leq M_Y\}$.

Lemma 2. Suppose that $f_Y \in C(\mathcal{H})$. Then for every S and $\mathbf{w}$,

$$\left| \frac{1}{n} \sum_{i=1}^{n} w_i f_Y(x_i) - \frac{1}{n'} \sum_{i=n+1}^{N} f_Y(x_i) \right| \leq \frac{M_Y}{2} d_{\mathcal{H}}(Sw, T)$$

Proof. See appendix. □

Lemma 2 leads to the following bound:

Theorem 1. Given $\mathbf{w}$ and $S = \{x_i\}_{i=1}^n$, If $f_Y \in C(\mathcal{H})$ then for any $\delta \in (0, 1)$, with probability of at least $1 - \delta$ we have

$$\left| \frac{1}{n} \sum_{i=1}^{n} w_i y_i - \frac{1}{n'} \sum_{i=n+1}^{N} f_Y(x_i) \right| \leq \frac{M_Y}{2} d_{\mathcal{H}}(Sw, T) + 2M_Y \sqrt{2 \left\| \frac{\mathbf{w}}{n} \right\|_2^2 \ln \frac{2}{\delta}}$$

Proof. Define a set of random variables $Z_i \sim \mathcal{D}(w_i(Y^a - f_Y(x_i))|x_i)$. Each Z_i is bounded by $2w_i M_Y$. Applying Hoeffding's inequality (Mohri et al., 2018) yields that for any $\delta \in (0, 1)$, with probability at least $1 - \delta$

$$\left| \frac{1}{n} \sum_{i=1}^{n} w_i y_i - \frac{1}{n} \sum_{i=1}^{n} w_i f_Y(x_i) \right| < 2M_Y \sqrt{2 \left\| \frac{\mathbf{w}}{n} \right\|_2^2 \ln \frac{2}{\delta}}$$

See appendix for details. Following Equation 16, this inequality together with Lemma 2 provides the desired proof. □

The first term in the bound given in Theorem 1 corresponds to the $\mathcal{H}$-divergence, which is the objective that our weights generator aims to minimize. This implies a tradeoff induced by selection of the discriminator. Using a rich family of classifiers allows to have a good approximation of the function family $C(\mathcal{H})$. On the other hand a compact family leads to low $\mathcal{H}$-divergence and allows to avoid overfitting. Note that the empirical $\mathcal{H}$-divergence can be examined after running the algorithm and thus provide an indication of potential errors.

The second term in this bound is dominated by $\left\| \frac{\mathbf{w}}{n} \right\|_2^2$, indicating the variability in the weights. Observe that $\frac{\mathbf{w}}{n} \in \Delta$ and that $\min_{\mathbf{u} \in \Delta} \|\mathbf{u}\|_2^2 = \frac{1}{n}$. This minimum is obtained for $\mathbf{u}^* = \frac{\mathbf{e}}{n}$, where $\mathbf{e} = (1, \ldots, 1)$. Therefore, when the weights are close to uniform this term converges at the rate of $\sqrt{n}$. Therefore it is desirable to maintain low variability of weights. Note that this variability is bounded by $\|\mathbf{u}\|_2^2 \leq KL(\mathbf{u}, \frac{\mathbf{e}}{n})$, where KL is the Kullback-Leibler divergence (Shalev-Shwartz et al., 2012; Beck & Teboulle, 2003) (see supplemental material). The exponentiated gradient ascent, which maintains the normalization constraint of the computed weights, has a desired property of generating weights with minimal Kullback-Leibler divergence to previous weights (Kivinen & Warmuth, 1997). Therefore, our algorithm is expected to produce weights that remain as close as possible (in an entropy sense) to the initial uniform weights.

In the supplemental material we provide a bound for the general case in which f_Y is not necessarily in $C(\mathcal{H})$. In this case the bound includes an additional term corresponding to a proximity measure of f_Y to $C(\mathcal{H})$.

6. Related work

Inverse propensity weighting (IPW) (Rosenbaum, 1987) is a widely-used balancing method that models the conditional treatment probability given pre-treatment covariates. If the model is correctly specified, then the computed weights are balancing (Horvitz & Thompson, 1952). However, a misspecified model may generate weights that fail to balance the biases, potentially leading to erroneous estimations. In recent years, various methods were developed to generate weights that directly minimize the different objectives used to measure the discrepancy between compared populations (e.g., (Hainmueller, 2012; Graham et al., 2012; Imai & Ratkovic, 2014; Zubizarreta, 2015; Chan et al., 2016; Kallus, 2017; Zhao, 2016)). Each of these methods provides alternative solutions to the following elementary problems: (i) how to measure the bias between two distributions, and (ii) how to generate weights that minimize it. Some of the methods, (e.g., (Graham et al., 2012) and (Imai & Ratkovic, 2014)), fit propensity score models with balance constraints, to guarantee that even if the propensity model is misspecified, these balancing constraints are met. The other algorithms, including the one presented here, are designed to minimize a selected imbalance measure without considering the related propensity scores.

A widely used criterion for assessing the imbalance between two treatment groups is the *standardized difference* in the mean of each covariate (Rosenbaum & Rubin, 1985). Many of the algorithms, such as (Hainmueller, 2012; Imai & Ratkovic, 2014; Chan et al., 2016), focus on minimizing the difference between the first-order moments of the covariates

or their transformations. The algorithm in (Kallus, 2017) uses the *maximum mean discrepancy* (MMD) measure (see (Gretton et al., 2007) for definition), which can account for an infinite number of higher order moments based on kernel methods. Very recently, an independent study (Kallus, 2018) presented a similar idea of using GANs to generate balancing weights. However, the discrepancy objective, the weights model, and the entire algorithm in this study differ from the ones we introduce in this paper.

The problem of finding balancing weights has been studied in the field of density ratio estimation (Sugiyama et al., 2012a;b; Mohamed & Lakshminarayanan, 2016). A similar problem has also been studied in the context of transfer learning under the assumption of covariate shift, where the task is to learn a prediction model from a labeled training data drawn from a source domain different from the target domain (Huang et al., 2007; Sugiyama et al., 2008; Mansour et al., 2009).

Our algorithm, and the other weighting methods we reviewed above, balance covariates without using outcome data. The resulting balanced data may be used for subsequent causal inference analysis involving multiple outcomes. Other state-of-the-art methods for causal inference, such as BART (Chipman et al., 2010) and Causal Forests (Wager & Athey, 2017) focus on training outcome models that allow causal inference for a specific outcome. Finally, there are causal inference methods that combine a treatment assignment model with an outcome model, such as the augmented inverse probability weighting (AIPW) (Robins et al., 1994; Scharfstein et al., 1999; Robins, 2000; Glynn & Quinn, 2010). Recent works in causal inference that took this approach use deep neural networks for learning a new representation of the data that improves outcome prediction on one hand, and on the other hand minimizes the discrepancy between the source and target data (Johansson et al., 2016; Shalit et al., 2017). The approach of learning a representation that minimizes the discrepancy between source and target domains in an adversarial manner, while optimizing label prediction, has recently become very popular in transfer learning, with vast applications in computer vision (Ganin et al., 2016; Tzeng et al., 2017).

7. Experiments

We evaluated our adversarial weighting method on three previously published benchmarks of simulated data by "plugging-in" various classifiers. We compared our method to IPW with the same classifier, and tested against more recent methods for balancing weights. We focused on methods that do not use information on the outcome for estimating the weights.

7.1. Experimental setting

The results reported in this section are based on the zero-one loss function. We considered the following "plug-in" classifiers as the discriminator: **LR**: Logistic regression (default parameters by `Scikit-learn`); **SVM**: a support vector machine with RBF kernel (default parameters by `Scikit-learn`); **MLP**: a multilayer perceptron with 1-3 internal layers. The number of nodes in each internal layer is set to twice the number of variables in the input layer. The exact number of internal layers is selected as the one that minimizes the zero-one prediction error (generalization error) evaluated in a 5-fold cross-validation procedure; **LR/SVM/MLP**: a classifier that is selected from the previously described classifiers as the one minimizing the zero-one prediction error evaluated in a 5-fold cross-validation procedure.

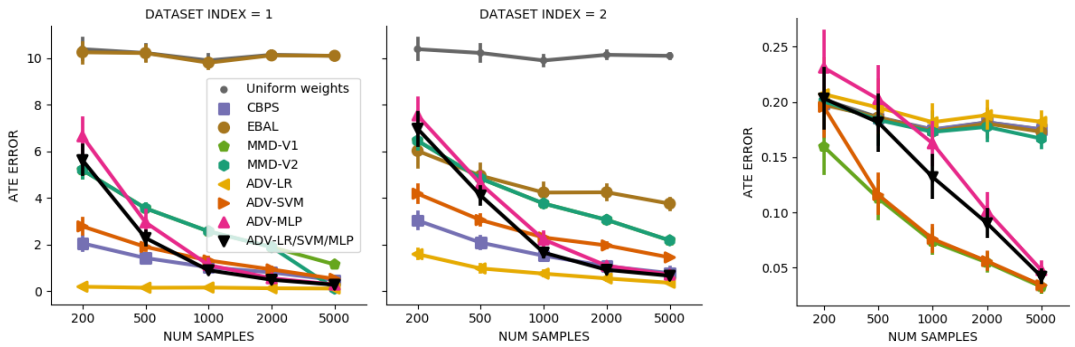
Note that for the classifiers MLP and LR/SVM/MLP, the configuration is set once before running the weighting algorithms. We used a decaying learning rate α in Algorithm 1: $\alpha_{t+1} = \frac{1}{1+0.5 \cdot t}$ and limited the number of iterations T to 20. Finally, to speed running times we configured the function get_predictions in Step 5 of Algorithm 1 to return train predictions.

We compared the results of Algorithm 1 to the results obtained by the following weighting methods: **IPW**: The straightforward inverse propensity weighting, without weight trimming or other enhancements. We tested IPW with the same classifiers we used for the adversarial algorithm; **CBPS**: Covariate Balancing Propensity Score (CBPS) (Imai & Ratkovic, 2014), using its R package (Fong et al., 2014); **EBAL**: Entropy balancing (Hainmueller, 2012), using its R package (Hainmueller & Hainmueller, 2014); **MMD-V1**, **MMD-V2**: An algorithm for minimizing the maximum mean discrepancy (MMD) measure using an RBF kernel (Kallus, 2016; 2017). In MMD-V1 the RBF scale parameter was set to 1. MMD-V2 includes a preliminary step for selecting the RBF scale and a regularization parameter (Kallus, 2016). We implemented MMD-V1 and MMD-V2 using the `quadprog` Python package.

7.2. Benchmarks

We evaluated and compared the different weighting methods on the following benchmarks:

Kang-Schafer benchmark (Kang & Schafer, 2007): The data includes four independently normally distributed covariates: $X_1, X_2, X_3, X_4 \sim N(0,1)$. The outcome covariate Y is generated as $Y = 210 + 27.4X_1 + 13.7X_2 + 13.7X_3 + 13.7X_4 + \epsilon$ where $\epsilon \sim N(0,1)$. The true propensity score is $p(A = 1|X_1, X_2, X_3, X_4) = \text{expit}(-X_1 + 0.5X_2 - 0.25X_3 - 0.1X_4)$. The outcome Y is observed only for $A = 1$. The simulation includes two scenar-



(a) Kang-Schafer benchmark

(b) Circular benchmark

Figure 2. Comparison of weighting algorithms: CBPS, EBAL, MMD and the adversarial algorithm. We compare the adversarial algorithm with two different classifiers: logistic regression (LR) and LR/SVM/MLP. The latter corresponds to the adversarial algorithm with a preceding step of model selection from (i) logistic regression (LR), (ii) support vector machine with RBF kernel (SVM), and (iii) multi-layer perceptrons MLP. MLP corresponds to MLPS with 1/2/3 layers, respectively chosen by cross-validation. Horizontal lines represent 95% confidence intervals computing using bootstrapping.

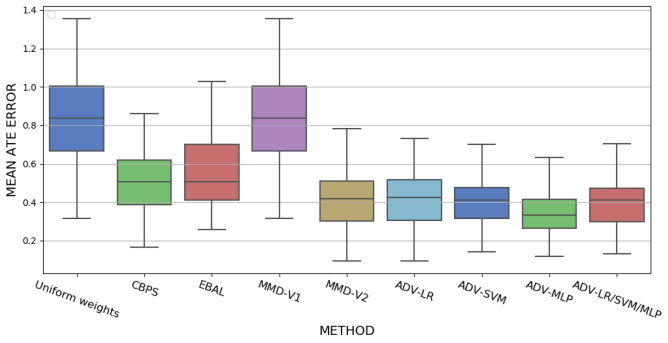


Figure 3. Comparison on ACIC benchmark

ios. In the first, the covariates (X_1, X_2, X_3, X_4) are observed, while in the second the covariates actually seen, (X_1', X_2', X_3', X_4'), are generated as: $X_1' = \exp(X_1/2)$, $X_2' = X_2/(1+\exp(X_1))+10$, $X_3' = (X_1 * X_3/25+0.6)^3$, and $X_4' = (X_2 + X_4 + 20)^2$. As Y is observed only for a biased selection of the data, the task in this benchmark is to estimate the expected potential outcome $E(Y_1)$ for the entire population. In this case we apply Algorithm 1 once to balance the subpopulation of $A = 1$ with the entire population. We generated 5 paired datasets, where each pair corresponds to the 2 scenarios, for data size $n = 200, 500, 1000, 2000, 5000$. Each of the datasets includes 100 random replications. Paired datasets are based on the same randomized covariates (X_1, X_2, X_3, X_4).

Circular benchmark: These simulations are based on the example given in (Kallus, 2016) with a minor modification to accommodate estimation of ATE. The simulations follow a scenario with two covariates X_1 and X_2 independently drawn from a uniform distribution on $[-1, 1]$. The true propensity score is $p(A = 1|X_1, X_2) = 0.95/(1 + \frac{3}{\sqrt{2}}\|(X_1, X_2)\|_2)$. The potential outcomes Y^0 and Y^1 are independently normally distributed with means $\|X\|_2^2 - X_1/2 - X_2/2$ and $\|X\|_2^2$, respectively, and with a standard deviation of $\sqrt{3}$. We generated 5 datasets, each with 100 random replications, for this scenario with data size $n = [200, 500, 1000, 2000, 5000]$.

ACIC benchmark: The Atlantic Causal Inference Conference (ACIC) benchmark (Dorie et al., 2017) includes 77 datasets, simulated with different treatment assignment mechanisms and outcome models. All the datasets use the same 58 covariates with 4802 observations derived from real-world data. These simulations accounted for various parameters, such as degrees of non-linearity, percentage of patients treated, and magnitude of the treatment effect. Each of the 77 datasets includes 100 random replications independently created by the same data generation process, yielding 7700 different realizations in total. For a complete description of this benchmark, see (Dorie et al., 2017).

7.3. Results

For all considered classifiers, the adversarial algorithm outperformed its counterpart IPW in most of the tests, in particular on the large sample size (see Figure A.1 in the supplemental material).

Figure 2 shows the results of Algorithm 1, CBPS, EBAL, MMD-V1 and MMD-V2 on the Kang-Schafer and Circular benchmarks. As a reference, we selected two classifiers for the adversarial algorithm: LR being the simplest classifier and LR/SVM/MLP for its ability to adapt to the data.

As shown, in the Kang-Schafer benchmark, ADV-LR outperforms CBPS, EBAL, and both versions of MMD. In the Circular benchmark, MMD-V1 and ADV-SVM outperformed all compared methods, possibly because it employs Gaussian kernels that can handle the circular contours of the propensity function. Note that the performance of all classifiers improves with data size, and becomes more similar in the final point (n=5000).

Figure 3 presents the results on the ACIC benchmark. These results also support our previous observation that the adversarial framework is better at exploiting classifiers than the IPW method. This plot also provides some evidence for the robustness of the cross-validation procedure, as ADV-LR/SVM/MLP steadily remains one of the top-performing methods. Finally, even our weakest variant ADV-LR exhibited performance superior to CBPS and EBAL, and results comparable to MMD-V2.

We see that each benchmark had a different classifier that obtained the best results in the adversarial framework, with ADV-LR excelling in the Kang-Schafer benchmark, ADV-SVM in the Circular benchmark, and ADV-MLP in ACIC. However, in all three benchmarks, LR/SVM/MLP was the second-best performing classifier, suggesting that it is more robust in unknown scenarios.

8. Discussion

We introduced an adversarial framework for generating balancing weights, which uses a classifier family for measuring the discrepancy between two data samples, and exponentiated gradient ascent, to compute weights that minimize this divergence. Our theoretical results for the estimation error provide further motivation for (i) obtaining weights that maximize the minimal classification error with 0-1 loss for the two samples; and (ii) using exponentiated gradient descent for generating normalized weights that optimize this objective while keeping themselves close to uniform. Our experimental results provide additional support for the effectiveness of exponentiated gradient descent in generating weights that lead to estimates with smaller variance. This setup allows us to easily plug-in a plethora of classification algorithms, each corresponding to a different classifier family, into our framework.

The selection of the classifier family clearly affects the sensitivity of the discriminator in identifying biases between the samples. Low capacity classifier families may weaken the ability of the discriminator to distinguish important biases, while higher capacity families will result in large discrepancy measures even when the samples are balanced. Note that a classification algorithm may use an objective different than the discriminator's error for selecting the best classifier. In this case we assume it may still be used in practice under the assumption that the selected classifier highly correlates with the classifier that would have been selected by the discriminator. In particular, classifiers that incorporate regularization in their objective will have less over-fitting for stronger regularization, yielding a reduced sensitivity of the discriminator, and consequently smaller discrepancy measures.

Selecting a classifier family with an appropriate sensitivity level is a challenging task. We applied a heuristic that uses cross-validation to select the classification algorithm with the lowest estimated generalization error, under the assumption that models with larger generalization error may be either over-sensitive when the error is due to over-fitting, or not sensitive enough when the error is due to large bias. The experiments we conducted on different benchmarks may provide a support for this approach, as the adversarial algorithm with auto-select classifier always ranked second. A future research direction is to improve classifier selection so it reaches comparable results to the first ranked classifier.

The discrepancy measure induced by the discriminator is determined not only by the choice of the classifier family but also by the loss function. When the log loss is used and the family of classifier has enough capacity, then the induced discrepancy measure approximates the Jensen-Shannon divergence (Goodfellow et al., 2014). The discrepancy measures induced by the 0-1 loss, can be viewed as *integral probability metrics* (IPMs) , which are defined with respect to a family, $\mathcal{F}$, of real-valued bounded functions (Sriperumbudur et al., 2012): $\text{IPM}_{\mathcal{F}}(\mathcal{D}_S, \mathcal{D}_T) \equiv \left|\sup_{f\in\mathcal{F}} \mathbb{E}_{X\sim\mathcal{D}_S}\left[f(X)\right] - \mathbb{E}_{X\sim\mathcal{D}_T}\left[f(X)\right]\right|$. Discrepancy measures that can be presented as IPMs include the Wasserstein distance (also known as Earth-Mover distance) and MMD (Sriperumbudur et al., 2012). There are extensions of GANs where the discriminator is replaced by a two sample-test corresponding to the Wasserstein distance (Arjovsky et al., 2017) and the MMD distance (Li et al., 2015; Dziugaite et al., 2015; Li et al., 2017). A future work would be to adapt our framework for estimating these discrepancy measure and minimize them with exponentiated gradient descent.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Beck, A. and Teboulle, M. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*, pp. 137–144, 2007.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F.,

and Vaughan, J. W. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

Chan, K. C. G., Yam, S. C. P., and Zhang, Z. Globally efficient non-parametric inference of average treatment effects by empirical balancing calibration weighting. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(3):673–700, 2016.

Chipman, H. A., George, E. I., McCulloch, R. E., et al. Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1):266–298, 2010.

Dorie, V., Hill, J., Shalit, U., Scott, M., and Cervone, D. Automated versus do-it-yourself methods for causal inference: Lessons learned from a data analysis competition. *ArXiv e-prints: 1707.02641*, 2017.

Dziugaite, G. K., Roy, D. M., and Ghahramani, Z. Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*, 2015.

Fong, C., Ratkovic, M., and Imai, K. Cbps: R package for covariate balancing propensity score. *Comprehensive R Archive Network (CRAN)*, 2014.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Glynn, A. N. and Quinn, K. M. An introduction to the augmented inverse propensity weighted estimator. *Political analysis*, 18(1): 36–56, 2010.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Graham, B. S., de Xavier Pinto, C. C., and Egel, D. Inverse probability tilting for moment condition models with missing data. *The Review of Economic Studies*, 79(3):1053–1079, 2012.

Gretton, A., Borgwardt, K. M., Rasch, M., Schölkopf, B., and Smola, A. J. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pp. 513–520, 2007.

Gutmann, M. U., Dutta, R., Kaski, S., Corander, J., et al. Likelihood-free inference via classification. *arXiv preprint arXiv:1407.4981*, 2014.

Hainmueller, J. Entropy balancing for causal effects: A multivariate reweighting method to produce balanced samples in observational studies. *Political Analysis*, 20(1):25–46, 2012.

Hainmueller, J. and Hainmueller, M. J. Package ebal. 2014.

Horvitz, D. G. and Thompson, D. J. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685, 1952.

Huang, J., Gretton, A., Borgwardt, K. M., Schölkopf, B., and Smola, A. J. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems*, pp. 601–608, 2007.

Imai, K. and Ratkovic, M. Covariate balancing propensity score. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):243–263, 2014.

Johansson, F., Shalit, U., and Sontag, D. Learning representations for counterfactual inference. In *International Conference on Machine Learning*, pp. 3020–3029, 2016.

Kallus, N. Generalized optimal matching methods for causal inference. *arXiv preprint arXiv:1612.08321*, 2016.

Kallus, N. Balanced policy evaluation and learning. *arXiv preprint arXiv:1705.07384*, 2017.

Kallus, N. Deepmatch: Balancing deep covariate representations for causal inference using adversarial training. *arXiv preprint arXiv:1802.05664*, 2018.

Kang, J. D. and Schafer, J. L. Demystifying double robustness: A comparison of alternative strategies for estimating a population mean from incomplete data. *Statistical science*, pp. 523–539, 2007.

Kifer, D., Ben-David, S., and Gehrke, J. Detecting change in data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pp. 180–191. VLDB Endowment, 2004.

Kivinen, J. and Warmuth, M. K. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

Li, C.-L., Chang, W.-C., Cheng, Y., Yang, Y., and Póczos, B. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, pp. 2200–2210, 2017.

Li, Y., Swersky, K., and Zemel, R. Generative moment matching networks. In *International Conference on Machine Learning*, pp. 1718–1727, 2015.

Lopez-Paz, D. and Oquab, M. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016.

Mansour, Y., Mohri, M., and Rostamizadeh, A. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

Mohamed, S. and Lakshminarayanan, B. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of machine learning*. MIT press, 2018.

Robins, J. M. Robust estimation in sequentially ignorable missing data and causal inference models. In *Proceedings of the American Statistical Association*, volume 1999, pp. 6–10. Indianapolis, IN, 2000.

Robins, J. M., Rotnitzky, A., and Zhao, L. P. Estimation of regression coefficients when some regressors are not always observed. *Journal of the American statistical Association*, 89(427):846–866, 1994.

Rosenbaum, P. R. Model-based direct adjustment. *Journal of the American Statistical Association*, 82(398):387–394, 1987.

Rosenbaum, P. R. and Rubin, D. B. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Rosenbaum, P. R. and Rubin, D. B. Constructing a control group using multivariate matched sampling methods that incorporate the propensity score. *The American Statistician*, 39(1):33–38, 1985.

Scharfstein, D. O., Rotnitzky, A., and Robins, J. M. Adjusting for nonignorable drop-out using semiparametric nonresponse models. *Journal of the American Statistical Association*, 94 (448):1096–1120, 1999.

Shalev-Shwartz, S. et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4 (2):107–194, 2012.

Shalit, U., Johansson, F. D., and Sontag, D. Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning*, pp. 3076–3085, 2017.

Sriperumbudur, B. K., Fukumizu, K., Gretton, A., Schölkopf, B., Lanckriet, G. R., et al. On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics*, 6:1550–1599, 2012.

Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P. V., and Kawanabe, M. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems*, pp. 1433–1440, 2008.

Sugiyama, M., Suzuki, T., and Kanamori, T. *Density ratio estimation in machine learning*. Cambridge University Press, 2012a.

Sugiyama, M., Suzuki, T., and Kanamori, T. Density-ratio matching under the bregman divergence: a unified framework of density-ratio estimation. *Annals of the Institute of Statistical Mathematics*, 64(5):1009–1044, 2012b.

Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, pp. 4, 2017.

Wager, S. and Athey, S. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, (just-accepted), 2017.

Zhao, Q. Covariate balancing propensity score by tailored loss functions. *arXiv preprint arXiv:1601.05890*, 2016.

Zubizarreta, J. R. Stable weights that balance covariates for estimation with incomplete outcome data. *Journal of the American Statistical Association*, 110(511):910–922, 2015.

A Appendix

A.1 Proof of lemma 1

Proof. Due to the symmetry of $\mathcal{H}$ and using $\sum_{i=1}^{n} w_i = n$ we have

$$\max_{h\in\mathcal{H}} \left\{ \frac{1}{n}\sum_{i=1}^{n} w_i h(x_i) - \frac{1}{n'}\sum_{i=n+1}^{n+n'} h(x_i) \right\}$$
$$= \max_{1-h\in\mathcal{H}} - \left\{ \frac{1}{n}\sum_{i=1}^{n} w_i h(x_i) - \frac{1}{n'}\sum_{i=n+1}^{n+n'} h(x_i) \right\} = \max_{h\in\mathcal{H}} \left| \frac{1}{n}\sum_{i=1}^{n} w_i h(x_i) - \frac{1}{n'}\sum_{i=n+1}^{n+n'} h(x_i) \right|$$

And on the other hand we have

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n} w_i h(x_i) - \frac{1}{n'}\sum_{i=n+1}^{n+n'} h(x_i) \\
&= \frac{1}{n}\sum_{i=1}^{n} w_i - w_i \mathbb{1}\left[h(x_i) = 0\right] - \frac{1}{n'}\sum_{i=n+1}^{n+n'} h(x_i) \\
&= 1 - \left(\frac{1}{n}\sum_{i=1}^{n} w_i \mathbb{1}\left[h(x_i) = 0\right] + \frac{1}{n'}\sum_{i=n+1}^{n+n'} \mathbb{1}\left[h(x_i) = 1\right] \right) \\
&= 1 - L_n(\mathbf{w}, h)
\end{aligned}$$

Combining the two results concludes the proof. □

A.2 More details on Theorem 1

We can consider y_i as a random variable whose expected value is $f_Y(x_i)$. Therefore the random variable $w_i y_i - w_i f_Y(x_i)$ has an expected value of zero. Note that $w_i y_i - w_i f_Y(x_i) \in [-2M_Y w_i, 2M_Y w_i]$. Using Hoeffding's inequality we get:

$$P\left(\frac{|\sum_i w_i (y_i - f_Y(x_i))|}{n} \geq \epsilon \right) \leq 2\exp\left(-\frac{2n^2\epsilon^2}{\sum_{i=0}^{n} (4M_Y w_i)^2} \right) = 2\exp\left(-\frac{\epsilon^2}{8M_Y^2 \left\| \frac{\mathbf{w}}{n} \right\|_2^2} \right)$$

Suppose this probability to $\delta = 2\exp\left(\frac{-\epsilon^2}{8M_Y^2 \left\| \frac{\mathbf{w}}{n} \right\|_2^2} \right)$ and solving for ϵ gives

$$\frac{-\epsilon^2}{8M_Y^2 \left\| \frac{\mathbf{w}}{n} \right\|_2^2} = \ln\frac{\delta}{2} \implies \epsilon^2 = 8M_Y^2 \left\| \frac{\mathbf{w}}{n} \right\|_2^2 \ln\frac{2}{\delta} \implies \epsilon = 2M_Y \sqrt{2 \left\| \frac{\mathbf{w}}{n} \right\|_2^2 \ln\frac{2}{\delta}}$$

Then with probability of $1 - \delta$

$$\frac{1}{n}\left|\sum_i w_i y_i - \sum_i w_i f_Y(x_i)\right| < 2M_Y\sqrt{2\left\|\frac{\mathbf{w}}{n}\right\|_2^2 \ln(\frac{2}{\delta})} \qquad (1)$$

Proof of Lemma 2

Suppose that $f_Y(x) = \sum_j \alpha_j h_j(x)$. Then

$$\begin{aligned}
\left|\frac{1}{n}\sum_{i=1}^{n} w(x_i)f_Y(x_i) - \frac{1}{n'}\sum_{i=n+1}^{N} f_Y(x_i)\right| &= \left|\frac{1}{n}\sum_{i=1}^{n} w(x_i)\sum_j \alpha_j h_j(x_i) - \frac{1}{n'}\sum_{i=n+1}^{N}\sum_j \alpha_j h_j(x_i)\right| \\
&= \left|\sum_j \alpha_j\left(\frac{1}{n}\sum_{i=1}^{n} w(x_i)h_j(x_i) - \frac{1}{n'}\sum_{i=n+1}^{N} h_j(x_i)\right)\right| \\
&\leq \sum_j |\alpha_j|\left|\frac{1}{n}\sum_{i=1}^{n} w(x_i)h_j(x_i) - \frac{1}{n'}\sum_{i=n+1}^{N} h_j(x_i)\right| \\
&\leq \sum_j |\alpha_j| \max_h \left|\frac{1}{n}\sum_{i=1}^{n} w(x_i)h(x_i) - \frac{1}{n'}\sum_{i=n+1}^{N} h(x_i)\right| \\
&= \frac{1}{2}M_Y d_{\mathcal{H}}(\mathbf{w})
\end{aligned}$$

A.3 Proof of $\|\mathbf{u}\|_2^2 \leq KL(\mathbf{u}\,\|\,\frac{\mathbf{e}}{n})$ (for completeness)

Let Δ be the unit simplex $\Delta = \{\mathbf{u} \in \mathbb{R}^n : \mathbf{u} \geq 0,\ \|\mathbf{u}\|_1 = 1\}$. Let $\mathbf{e} = (1, ..., 1)$. Note that $\frac{\mathbf{e}}{n} \in \Delta$. Let ψ_e be the entropy function defined by $\psi_e(\mathbf{u}) = \sum_{i=1}^n u_i \ln u_i$. The entropy function ψ_e is known to be 1-strong convex on Δ (e.g., see [Shalev-Shwartz et al.(2012), Beck & Teboulle(2003)Beck and Teboulle]). Additionally, $\frac{\mathbf{e}}{n} = \arg\min_{\mathbf{u}\in\Delta} \psi_e(\mathbf{u})$ and $\psi(\frac{\mathbf{e}}{n}) = -\ln(n)$.

Lemma 1. [Shalev-Shwartz et al.(2012), Beck & Teboulle(2003)Beck and Teboulle] For every $\mathbf{u} \in \Delta$

$$\psi_e(\mathbf{u}) + \ln(n) \geq \frac{1}{2}\left\|\frac{\mathbf{e}}{n} - \mathbf{u}\right\|_1^2$$

We now show that the relative entropy, also known as the KL-divergence, of $\mathbf{u}$ w.r.t. to the uniform vector $\frac{\mathbf{e}}{n}$ is an upper bound for L_2 norm of $\mathbf{u}$. Minimizing one will minimize the other...

$$KL(\mathbf{u}\|\frac{\mathbf{e}}{n}) \equiv \sum_{i=1}^{n} u_i \log(u_i) + \sum_{i=1}^{n} u_i \ln(n) = \psi_e(\mathbf{u}) + \ln(n) \geq \frac{1}{2}\left\|\frac{\mathbf{e}}{n} - \mathbf{u}\right\|_1^2 \geq \frac{1}{2}\left\|\frac{\mathbf{e}}{n} - \mathbf{u}\right\|_2^2$$

A.4 Proof of $\left|\frac{1}{n'}\sum_{i=n+1}^{N} f_Y(x_i) - \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_T}\left[f_Y(X)\right]\right| \leq 2M_Y\sqrt{\frac{\ln\frac{2}{\delta}}{2n}}$ (for completeness)

Using Hoeffding's equality:

$$P\left(\left|\frac{1}{n'}\sum_{i=n+1}^{N} f_Y(x_i) - \mathbb{E}_{X\sim\mathcal{D}_T}\left[f_Y(X)\right]\right| \geq \epsilon\right) \leq 2\exp\left(-\frac{n'\epsilon^2}{2M_Y^2}\right)$$

Let compute ϵ when this probability is δ:

$$2\exp\left(-\frac{n'\epsilon^2}{2M_Y^2}\right) = \delta \implies \frac{n'\epsilon^2}{2M_Y^2} = \ln(\frac{2}{\delta}) \implies \epsilon = 2M_Y\sqrt{\frac{\ln(\frac{2}{\delta})}{2n'}}$$

A.5 Relaxation of Theorem 1 for general response functions

To explore the relationship between the discriminator family recall that $C(\mathcal{H})$ is the family of all functions that can be approximated by a bounded linear combination of members in $\mathcal{H}$. More formally, let $C(\mathcal{H}) = \{f : f = \sum_j \alpha_j h_j(x)$ s.t. $h_j \in \mathcal{H}$ and $\sum|\alpha_j| \leq M_Y\}$. Let U_X be the union of the supports of $\widehat{\mathcal{D}}_S$ and $\widehat{\mathcal{D}}_T$. For any function $f(\mathbf{x})$ denote the difference of empirical means by

$$\Delta_{\mathbb{E}}[S_w, T; f] = \left|\sum_{\mathbf{x}\in U_X}\left(\widehat{\mathcal{D}}_{S_w}(\mathbf{x}) - \widehat{\mathcal{D}}_T(\mathbf{x})\right) f\right| .$$

Note that $d_{\mathcal{H}}(S_w, T) = \max_{h\in\mathcal{H}} \Delta_{\mathbb{E}}[S_w, T; h]$. Let $f_Y^* = \arg\min_{f\in C(\mathcal{H})} \Delta_{\mathbb{E}}[S_w, T; f_Y - f]$, that is, $f_Y^*(x)$ is a member from $C(\mathcal{H})$ having the closest $\Delta_{\mathbb{E}}[S_w, T; \cdot]$ value to that of $f_Y(x)$. The crux of the adversarial objective is illustrated by a bound on the difference between the weighted source mean and the target mean. This bound depends on the distance between the outcome function and $C(\mathcal{H})$ as well as on the $\mathcal{H}$-divergence. Specifically,

Lemma 2. For every S and $\mathbf{w}$

$$\left|\frac{1}{n}\sum_{i=1}^{n} w_i f_Y(x_i) - \frac{1}{n'}\sum_{i=n+1}^{N} f_Y(x_i)\right| \leq \Delta_{\mathbb{E}}[f_Y - f_Y^*] + M_Y d_{\mathcal{H}}(Sw, T)$$

Proof.

$$\begin{aligned}
&\left|\frac{1}{n}\sum_{i=1}^{n} w(x_i)f_Y(x_i) - \frac{1}{n'}\sum_{i=n+1}^{N} f_Y(x_i)\right| \\
&= \left|\sum_{\mathbf{x}\in U_X} (\widehat{\mathcal{D}}_{S_w}(x) - \widehat{\mathcal{D}}_T(x)) f_Y(x)\right| \\
&\leq \left|\sum_{\mathbf{x}\in U_X} (\widehat{\mathcal{D}}_{S_w}(x) - \widehat{\mathcal{D}}_T(x))(f_Y(x) - f_Y^*(x))\right| + \left|\sum_{\mathbf{x}\in U_X}\sum_j (\widehat{\mathcal{D}}_{S_w}(x) - \widehat{\mathcal{D}}_T(x)) h_j(x)\alpha_j\right| \\
&= \Delta_{\mathbb{E}}[f_Y - f_Y^*] + \left|\sum_j \alpha_j \sum_{\mathbf{x}\in U_X} (\widehat{\mathcal{D}}_{S_w}(x) - \widehat{\mathcal{D}}_T(x)) h_j(x)\right| \\
&\leq \Delta_{\mathbb{E}}[f_Y - f_Y^*] + \leq \sum_j |\alpha_j| \left|\sum_{\mathbf{x}\in U_X} (\widehat{\mathcal{D}}_{S_w}(x) - \widehat{\mathcal{D}}_T(x)) h_j\right|
\end{aligned}$$

□

This lemma leads to the following bound:

Theorem 1. Given a sample $S \equiv \{x_i\}_{i=1}^n$ and a corresponding weights vector $\mathbf{w}$, for any $\delta \in (0,1)$, with probability of at least $1-\delta$ we have

$$\left|\frac{1}{n}\sum_{i=1}^{n} w_i y_i - \frac{1}{n'}\sum_{i=n+1}^{N} f_Y(x_i)\right| \leq 2M_Y\sqrt{2\left\|\frac{\mathbf{w}}{n}\right\|_2^2 \ln\frac{2}{\delta}} + \Delta_{\mathbb{E}}[f_Y - f_Y^*] + \frac{M_Y}{2} d_{\mathcal{H}}(S_w, T)$$

Proof. Combining Equation 16 with the Hoefdings inequality and Lemma 2 gives the desired result. □

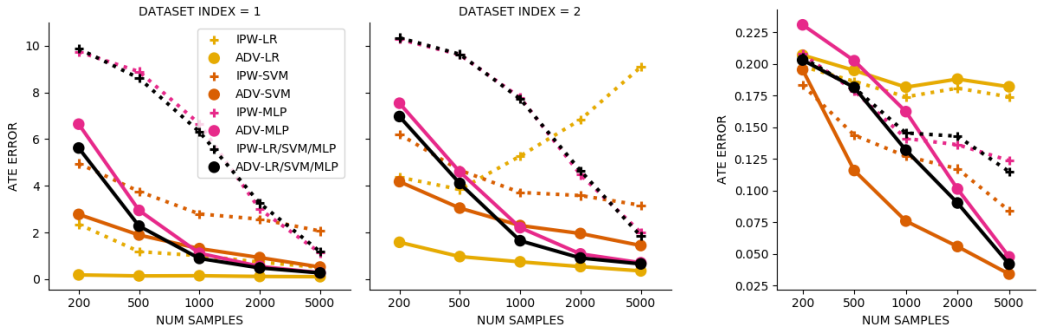
A.6 More experimental results

See figures for a comparison of Adversarial and IPW with the same classifier family.

References

[Beck & Teboulle(2003)Beck and Teboulle] Beck, A. and Teboulle, M. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

[Shalev-Shwartz et al.(2012)] Shalev-Shwartz, S. et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.



(a) Kang-Schafer benchmark

(b) Cirulcar benchmark

Figure A.1: Comparison of Adversarial and IPW with the same classifier families. Horizontal lines represent 95% confidence intervals computing using bootstrapping. Adversarial weighting algorithm versus IPW with different classifiers: logistic regression (LR), support vector machine with RBF kernel (SVM), and multi-layer perceptrons MLP. MLP corresponds to MLPS with 1/2/3 layers, respectively chosen by cross-validation. LR/SVM/MLP indicate a preceding step of model selection, prior to running IPW or ADV.

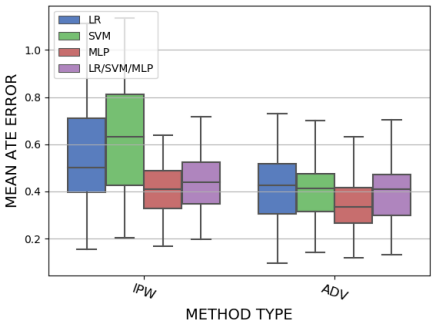


Figure A.2: Comparison of Adversarial and IPW on the ACIC benchmark