Fairness Learning in Semi-Supervised Setting

Jie Zhang
University of Utah

UUCS-18-008

School of Computing
University of Utah
Salt Lake City, UT 84112 USA

10 December 2018

Abstract

More and more often nowadays, decision systems that rule many aspects of our lives are trained by machine learning algorithms. The trained decision systems are biased in situations where only partial and biased data are available for training. Those biased decision systems will make biased decisions about people and negatively affect people's lives.

In this work, we study the possibility of involving unbiased unlabeled data to learn an accurate and fair classifier. To this end, two fair learning algorithms are proposed. Those two algorithms employ a iterative label swapping procedure to discover more fair and accurate labelings of test points closest to decision boundary. The algorithms then use the new labelings to refit a fairer classifier. The closeness of test points to decision boundary is measured empirically by a threshold. Such a threshold is also a key element through which we can quantify a known trade-off between accuracy and fairness in the fair learning setting. With access to this threshold, a user of proposed fair learning algorithms can fine-tune and make informed decisions when weighing accuracy against fairness. Our experiments show classifiers learned with proposed fair algorithms give fairer predictions than classifiers learned with pure semi-supervised algorithms and supervised algorithm.

FAIRNESS LEARNING IN SEMI-SUPERVISED SETTING

by

JIE ZHANG

A THESIS submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Bachelor of Science

School of Computing

The University of Utah

November 2018

Copyright © JIE ZHANG 2018
All Rights Reserved

The University of Utah School of Computing

The thesis of **JIE ZHANG**

has been approved by the following supervisory committee members:

Suresh Venkatasubramanian Advisor	11/29/2018 Date
H. James de St. Germain Director of Undergraduate Studies School of Computing	Date
Ross Whitaker Director School of Computing	Date

ABSTRACT

More and more often nowadays, decision systems that rule many aspects of our lives are trained by machine learning algorithms. The trained decision systems are biased in situations where only partial and biased data are available for training. Those biased decision systems will make biased decisions about people and negatively affect people's lives.

In this work, we study the possibility of involving unbiased unlabeled data to learn an accurate and fair classifier. To this end, two fair learning algorithms are proposed. Those two algorithms employ a iterative label swapping procedure to discover more fair and accurate labelings of test points closest to decision boundary. The algorithms then use the new labelings to refit a fairer classifier. The closeness of test points to decision boundary is measured empirically by a threshold. Such a threshold is also a key element through which we can quantify a known trade-off between accuracy and fairness in the fair learning setting. With access to this threshold, a user of proposed fair learning algorithms can fine-tune and make informed decisions when weighing accuracy against fairness. Our experiments show classifiers learned with proposed fair algorithms give fairer predictions than classifiers learned with pure semi-supervised algorithms and supervised algorithm.

CONTENTS

ABSTRACT . . . . . **iii**

LIST OF FIGURES . . . . . **v**

CHAPTERS

1. INTRODUCTION **1**

1.1 Introduction 1
1.2 Different Ways to Measure Fairness 3
1.3 Prior Work 4
1.4 Problem Setup 5

2. MAIN RESULTS **7**

2.1 fair-TSVM 7
2.2 fair-LabelPropagation 10

3. EXPERIMENTS **14**

3.1 Data 14
3.2 Implementation and Code 15
3.3 Performance Measures 16
3.4 Results 16

4. DISCUSSION **21**

4.1 Representation of Group Features 21
4.2 Label Swapping on One Pair vs. More Pairs 21
4.3 Summary and Conclusions 22

REFERENCES **24**

LIST OF FIGURES

2.1 Label Propagation 12
3.1 Comparison of Algorithms 18
3.2 Error of Fair-TSVM 18
3.3 Statistical Parity of Fair-TSVM 19
3.4 Error of Fair Label Propagation with rbf gamma=1000 19
3.5 Statistical Parity of Fair Label Propagation with rbf gamma=1000 20

CHAPTER 1

INTRODUCTION

1.1 Introduction

Many decision systems today, such as the systems that decide jail time, loan approval, hiring and others are based on, or partially based on machine learning algorithms ([2], [4], [19]). In situations where only partial and biased data are available for the system to train on, often the classifiers after training from supervised learning algorithms are biased. When these biased classifiers are used to predict if new applicants qualify for loan or hiring, as a result, biased decisions are made. These biased decisions affect millions of people and result in social injustice.

Why are data biased and what can we do to eliminate biases? There could be a few reasons. One reason is affirmative action or other equality policies have only been implemented for a short period of time. This means in labeled dataset used to train classifiers, especially in the positively labeled class, only a small portion of data points are from protected demographic group. Another reason is a reinforcement effect between biased data and biased classifiers. Biased dataset when used in training will produce biased classifiers. Biased classifiers will produce biased predictions. Biased new predictions together with historical data form new biased datasets which will be used for future training tasks. We could hope that affirmative actions or such policies would gradually correct the biases. However, given the reasons mentioned above and the reinforcement effect, improvement process could be long and slow. To accelerate advent of fairness and equality in areas where decision systems prevail, we need to inject fairness manually.

There were various studies in recent years about how to eliminate biases in decisions made by machine learning systems. They fall into a few categories: pre-process the dataset used to train classifiers so that its predictions are fair; during training, introduce fairness

considerations so classifiers learned are fair; post process predictions made by an unfair classifier so predictions become fair. Existing algorithms in these categories have their limitations. Specifically, they do not explicitly quantify the known trade-off between accuracy and fairness of learned classifier. As a result, adopters of existing fair learning algorithms cannot make an informed decision with respect to the trade-off. Our proposed algorithms fall in the second category in that we inject fairness in training step. Our algorithms differ from existing ones in that there is one hyper-parameter which associates with both accuracy and fairness. Thus adopts are able to quantify the trade-off.

Our proposed solution is based on the assumption that population at large are equal in terms of the qualities relevant to classification. Therefore unlabeled datasets containing only features describing qualities of population and no labels, are unbiased representations of the population. We hope to utilize unbiased unlabeled datasets to mediate biases in labeled datasets. Specifically, we propose to introduce unlabeled data into the process of training a classifier. This motivation coincides with the fundamental idea of semi-supervised learning.

Semi-supervised learning is a class of algorithms that make use of unlabeled data for training. It was shown that when labeled datasets are small, geometry of unlabeled data may radically change intuition about classification boundary learned from only labeled data. A classic example is where unlabeled data form two concentric circles and labeled data comprises only two points of two classes, one on each circle. Classifier learned from only labeled data points is drastically different from the true classifier that separates both labeled and unlabeled points. In some sense, unlabeled points provide a fuller picture, comparing to a partial picture given by labeled data. Beyond the fairness setting, in general machine learning, semi-supervised learning is often argued as more efficient because it can produce considerable improvement in accuracy without the time and costs needed for supervised learning. Often acquisition of labeled data for supervised learning requires tremendous human efforts.

In fair learning settings, the cost problem is coupled with bias in labeled data. We therefore want to involve unbiased unlabeled data and employ semi-supervised algorithms to produce fair classifiers.

1.2 Different Ways to Measure Fairness

The research of fairness in machine learning has received unprecedented attention in the past few years. Various definitions and measurements for fairness have emerged. Below we review the most prominent fairness measures.

Given a distribution D over a data set X with labels given by $l : X \to \{-1, 1\}$ and a protected subset $S \in X$, bias of the labeling function l with respect to D is defined as a difference:

$$\Pr_{x\ D|_S^c}(l(x) = 1) - \Pr_{x\ D|_S}(l(x) = 1)$$

The bias of any hypothesis h is the same formulation with $h(x)$ replacing $l(x)$. We say a hypothesis is fair if absolute value of above difference is small. In which case we say the hypothesis achieves *Statistical Parity*. S in this setting represents the protected group which is discriminated against.

Besides statistical parity, *Individual Fairness* [7] describes fairness as the principle that two individuals who are similar with respect to a particular task should be classified similarly. In order to define similarity between individuals, a distance metric needs to be defined. With such a metric, k-nearest-neighbor algorithms are used to ensure labels are consistent between similar individuals. Oftentimes, the difficulty with this fairness notion is in defining similarity metric.

The legal doctrine of *Disparate Impact* is introduced to fairness learning realm by [8]. This notion is used to determine unintended discrimination in the U.S., which is sometimes referred to as indirect discrimination. The opposite of indirect discrimination is direct discrimination, or intended discrimination, which is legally known as *Disparate Treatment*. Mathematically, Disparate Impact is defined as given a data set $D = (X, Y, C)$, with X as the protected attribute (e.g. race, sex, religion, etc.), Y as remaining attributes and C (1 as positive label and -1 as negative label) as binary class prediction, the data set D has disparate impact if

$$\frac{\Pr(C = 1|X = 0)}{\Pr(C = 1|X = 1)} \leq \tau = 0.8$$

Because of the threshold τ, disparate impact is also known as "80% rule".

Equalized Odds [10] is another fairness measure. It is defined with respect to a predictor $\hat{Y}$, a protected attribute A and the true outcome Y. We say $\hat{Y}$ satisfies equalized odds if

$$\Pr(\hat{Y} = 1|A = 0, Y = y) = \Pr(\hat{Y} = 1|A = 1, Y = y), y \in \{0, 1\}$$

Intuitively, equalized odds can be understood as when knowing Y, $\hat{Y}$ and group feature A should be independent. More exactly, this fairness measure requires that for a given true outcome Y, results from predictor $\hat{Y}$ has equal true positive rates and equal false positive rates across two groups.

To summarize, statistical parity and disparate impact care about fairness with respect to the whole group. Thus they are sometimes referred informally as group fairness. Hypothesis that passes these measures are described to have achieved group fairness. A previous study [7] points out that by flipping a certain number of data points' labels arbitrarily chosen from protected group, a classifier can achieve statistical parity. However, such a classifier disregards individuals' qualities in relation to classification task. It can be argued as creating another kind of discrimination. Accuracy of the classifier in this case is reduced. In this work we will examine statistical parity and accuracy of classifiers at the same time. Interestingly, proposed fair learning algorithms in this work has assumptions that match definition of individual fairness. Therefore our work in some sense can be seen as respecting both group and individual fairness.

1.3 Prior Work

As briefly mentioned in Introduction, fairness classification techniques fall into three categories. The first category includes aglorithms that pre-process data [15][16]. Kamiran in [15] massages data by changing labels of discrimination victims and beneficiaries, and then trains a classifier from modified unbiased data. In selecting the best candidates for relabeling, a ranker based on naive Bayes classifier is used. Naive Bayes classifier is also used as the model for learner. In [16], for massaging the dataset and training a classifier, arbitrary combinations of rankers and leaners are used. The second category of techniques include algorithms that inject fairness considerations during training process [17][10][3]. In [17] and [3], regularizers that penalize discriminating classifiers are introduced into training loss. The third category includes techniques that post-process learned classifiers. In [9], proposed algorithm flips predicted labels of points close to decision boundary while keeping group feature visible. As a result, learned classifier improves fairness and does not compromise accuracy too much. Our fair-SSL algorithms are similar to this perturbation

process. Instead of learning a classifier and swapping labels once, we iteratively repeat a process of two steps: swapping labels and refitting model.

There are also other approaches. For example in [14], fair learning problem is formulated as a multi-arm bandit problem. The setup consists of an context x_j^t representing information about an individual from population group j at round t, and a reward of choosing x_j^t in round t. Fair learning is to find out choices at each step t that maximizes total rewards across steps. Further studies with the same approach are discussed in [12] and [13]. There are also studies that try to find non-discriminating classifiers with unsupervised learning algorithms [18][5].

In this work, we build fairness aware algorithms from two well-known semi-supervised learning algorithms: TSVM [11] and Label Propagation [23]. For comparison, we use results from SVM as baseline.

There is a line of work in the 2000s that discuss semi-supervised learning algorithms and theory. [1] introduces a PAC-style framework that models assumptions of semi-supervised learning. The framework defines a type of compatibility between target concept and data distribution. Another work [22] shows that under the manifold assumption and cluster assumption, unlabeled data help recover the true decision boundary. In another line of work about casual and anti-causal learning, [21] mentions semi-supervised learning is not feasible in some settings. This is because unlabeled data will not help, if marginal distribution of features $\Pr(X)$ is independent of goal of learner $\Pr(Y|X)$. [22] preempts the independent case by translating the connection between $\Pr(X)$ and $\Pr(Y|X)$ into manifold and cluster assumptions mentioned above.

1.4 Problem Setup

In this section we define notations and the fair learning problem setup. We define X to be a feature vectors of a dataset. Each data point in X is represented by a vector $\mathbf{x}$ in R^d. We use $y \in \{0, 1\}$ to represent label of each data point. In semi-supervised learning setting, both labeled training data and unlabeled test data will be utilized in training process. We use S_{train} to denote training data set of n examples, and S_{test} to denote test data set without labels containing k examples. $X = S_{train} \bigcup S_{test}$. We use A to denote protected group feature, i.e. race, gender, etc. A corresponds to an element in feature vector $\mathbf{x}$. We allow A

to have two values: 1 to represent majority group and -1 to represent minority group. We assume every instance of $\mathbf{x}$ belongs to exactly one of two demographic groups. We use x_a to denote the value of protected group feature in a feature vector $\mathbf{x}$.

The biased labeled data and unbiased unlabeled data assumption can be stated as follow:

1. The bias in labeled data comes from the way its labels are discovered. Specifically when these data receive their current labels, data points in majority groups are more likely to receive positive labels. This is to say in existing data set, $Pr(y = 1|x_a = 1) > Pr(y = 1|x_a = -1)$.

2. For unlabeled data, there is no bias in distribution of classification-relevant quality in whole population. Assuming y is true label for an unlabeled point $\mathbf{x}$ in S_test, $\Pr(y = 1|x_a = 1) = \Pr(y = 1|x_a = -1)$.

With this setup, our goal is to learn a classifier that assigns positive label with equal probability, given a point comes from either group. This means for a data point $\mathbf{x}$, for a classifier represented by f, $\Pr(f(\mathbf{x}) = \mathbf{1}|\mathbf{x_a} = \mathbf{1}) = \Pr(\mathbf{f}(\mathbf{x}) = \mathbf{1}|\mathbf{x_a} = -\mathbf{1})$. In terms of existing fairness definitions, our goal can be stated as hoping to minimize statistical parity to 0.

CHAPTER 2

MAIN RESULTS

In this chapter we present two fair semi-supervised algorithms.

2.1 fair-TSVM

Transductive Support Vector Machines(TSVM) is a semi-supervised learning algorithm proposed in 1999. It is studied and used extensively since then. Given a training set of labeled data $(\boldsymbol{x_i}, y_i)$ for $i \in [1, \ldots, n]$ and a test set of unlabeled data points of $\boldsymbol{x_j}$, $j \in [1, \ldots, k]$. TSVM learns a classifier utilizing both labeled and unlabeled data. Output of TSVM includes the learned classifier and labels for test data points.

TSVM starts by fitting a classifier on only labeled data. It then uses the initial classifier to label unlabeled test points. Because unlabeled data may appear in different geometry from labeled data, classifier learned from only labeled data may misclassify some points in test set. TSVM tries to find out such misclassified points and assign them correct labels. When no misclassified points exist anymore, the algorithm is considered to have converged. The labeling for test data is returned as test prediction. Classifier fit with both training data and test data with final labeling is returned. Such a labeling on test points and learned classifier is the optimal on below optimization problem. Here ξs are slack variables in soft margin SVM [6]. They allow points to fall into classifier margin or appear on the side of classifier opposite from their true label. C and C^* are parameters that give slacks of labeled and unlabeled data different weights.

OP2 TSVM (non-seperable case)

Minimize over $(y_1, \ldots, y_n^*, \boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*)$:

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i + C^*\sum_{j=0}^{k}\xi_j^*$$

subject to:

$$\forall_{i=1}^{n} : y_i[\boldsymbol{w} \cdot \boldsymbol{x_i} + b] \geq 1 - \xi_i$$

$$\forall_{j=1}^{k} : y_j^*[\boldsymbol{w} \cdot \boldsymbol{x_j} + b] \geq 1 - \xi_j^*$$

$$\forall_{i=1}^{n} : \xi_i > 0$$

$$\forall_{j=1}^{k} : \xi_j^* > 0$$

Below we give the original TSVM algorithm [11].

Algorithm 1: TSVM

Input : -training examples, $(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_n}, y_n)$
-test examples, $\boldsymbol{x_1}^*, \ldots, \boldsymbol{x_k}^*$
- C, C^*: parameters from OP2
- num_+: number of test examples to be assigned to class +

Output: predicted labels for test examples, $y_1^*, \ldots, y_k^*$

$(\boldsymbol{w}, b, \boldsymbol{\xi}, _) := solve_svm_qp([(\boldsymbol{x_1}, y_1) \ldots (\boldsymbol{x_n}, y_n)], [], C, 0, 0)$;
Classify the test examples using $< \boldsymbol{w}, b >$. The num_+ test examples with the highest value of $< \boldsymbol{w}, \boldsymbol{x_j^*} > +b$ are assigned to the class + ($y_j^* := 1$); the remaining test examples are assigned to class - ($y_j^* := -1$) ;
$C_-^* := 10^{-5}$;
$C_+^* := 10^{-5} * \frac{num_+}{k - num_+}$;
while $(C_-^* < C^*) || (C_+^* < C^*)$ **do**
    $(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) :=$
    $solve_svm_qp([(\boldsymbol{x_1}, y_1) \ldots (\boldsymbol{x_n}, y_n)], [(\boldsymbol{x_1}^*, y_1) \ldots (\boldsymbol{x_n}^*, y_n)], C, C_-^*, C_+^*)$;
    while $(\exists m, l, s.t. (y_m^* * y_l^* < 0) \& (\xi_m^* > 0) \& (\xi_l^* > 0) \& (\xi_m^* + \xi_l^* > 2))$ **do**
        $y_m^* := -y_m^*$;
        $y_l^* := -y_l^*$;
        $(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) :=$
        $solve_svm_qp([(\boldsymbol{x_1}, y_1) \ldots (\boldsymbol{x_n}, y_n)], [(\boldsymbol{x_1}^*, y_1) \ldots (\boldsymbol{x_n}^*, y_n)], C, C_-^*, C_+^*)$;
    end
    $C_-^* := min(C_-^* * 2, C^*)$;
    $C_+^* := min(C_+^* * 2, C^*)$;
end
return $(y_1^*, \ldots, y_k^*)$;

First, TSVM learns a decision function from training data. This step is exactly as in SVM. TSVM then uses such a decision function to label test data. After test data receive their initial labeling, the algorithm enters into a loop. At beginning of the loop, TSVM uses both training data and now labeled test data to fit a new classifier. Then pairs of points which meet a specified condition are selected and have their labels swapped. Selection condition includes the following two aspects which have to satisfy at the same time: selected pair has to contain two points from two classes; selected pair has to contain at least one point which is labeled wrong with respect to current classifier, and the remaining

point is either also classified wrong, or falls within current classifier margin. After label swapping step, test data labels are updated. TSVM then leans a new classifier with both training data and test data. It then goes back to beginning of the loop.

Swapping condition of the loop is related to weights assigned to ξ of labeled training data and unlabeled test data. Note that at beginning of swapping process, mistakes made on test data are assigned less weights, corresponding to small C^*s in optimization objective. As the swapping and model updating process continues, the algorithm is more confident about labels given to unlabeled data. And C^*s increase. When C^*s reach predefined value, loop stops. TSVM was proved to coverage because each swapping step reduces sum of ξs for test data. Reduction of sum of ξs means reduction of incompatibility between current classifier and data.

Below we present fair-TSVM. As highlighted in blue, the difference with TSVM is mainly in conditions for selecting label swapping points. Instead of selecting pair of points that contains wrong label, fair-TSVM selects pairs of points that are closest to current decision boundary within threshold δ and meet a specific condition. Selection condition in fair-TSVM includes the following two aspects which have to satisfy at the same time: selected pair has to contain two points from two classes; in selected pair positively labeled point has to come from majority group, and negatively labeled point has to come from minority group. The intuition is points that are close to decision boundary are the most probable ones to have been labeled wrong. Thus swapping labels of such pairs of points would hopefully not increase error by a lot if not decrease it. When such pairs of points swap their labels, statistical parity of the classifier is also improving.

Algorithm 2: fair-TSVM

Input : -training examples, $(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_n}, y_n)$
-test examples, $\boldsymbol{x_1}^*, \ldots, \boldsymbol{x_k}^*$
- C, C^*: parameters from OP2
- num_+: number of test examples to be assigned to class +

Output: predicted labels for test examples, $y_1^*, \ldots, y_k^*$

$(\boldsymbol{w}, b, \boldsymbol{\xi}, _) := solve_svm_qp([(\boldsymbol{x_1}, y_1) \ldots (\boldsymbol{x_n}, y_n)], [], C, 0, 0)$;
Classify the test examples using $< \boldsymbol{w}, b >$. The num_+ test examples with the highest value of $< \boldsymbol{w}, \boldsymbol{x_j^*} > +b$ are assigned to the class + ($y_j^* := 1$); the remaining test examples are assigned to class - ($y_j^* := -1$) ;
$C_-^* := 10^{-5}$;
$C_+^* := 10^{-5} * \frac{num_+}{k - num_+}$;
while $(C_-^* < C^*) || (C_+^* < C^*)$ **do**
  $(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) :=$
  $solve_svm_qp([(\boldsymbol{x_1}, y_1) \ldots (\boldsymbol{x_n}, y_n)], [(\boldsymbol{x_1}^*, y_1) \ldots (\boldsymbol{x_n}^*, y_n)], C, C_-^*, C_+^*)$;
  Calculate $1 - \xi^*$, distances of each point to current decision boundary;
  while $\exists m, l, s.t. (m : x_a = 1) \& (l : x_a = -1) \& (y_m^* > 0) \& (y_l^* < 0) \& (1 - \xi_m^* < \delta) \& (1 - \xi_l^* < \delta) \& (1 - \xi_m^* < 1 - \xi_p^*, \forall p! = m, p : x_a = 1 \& y_p^* > 0) \& (1 - \xi_l^* < 1 - \xi_q^*, \forall q! = l, q : x_a = 1 \& y_q^* < 0)$ **do**
    $y_m^* := -1$;
    $y_l^* := 1$;
    $(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) :=$
    $solve_svm_qp([(\boldsymbol{x_1}, y_1) \ldots (\boldsymbol{x_n}, y_n)], [(\boldsymbol{x_1}^*, y_1) \ldots (\boldsymbol{x_n}^*, y_n)], C, C_-^*, C_+^*)$;
    Calculate $1 - \xi^*$, distances of each point to current decision boundary;
  end
  $C_-^* := min(C_-^* * 2, C^*)$;
  $C_+^* := min(C_+^* * 2, C^*)$;
end
return $(y_1^*, \ldots, y_k^*)$;

2.2 fair-LabelPropagation

Label Propagation is an extensively studied semi-supervised learning algorithm. The setup of Label Propagation is the same as TSVM. We are given a training set of labeled data $(\boldsymbol{x_i}, y_i)$, $i \in [1, \ldots, n]$ and a test set of unlabeled data $\boldsymbol{x_j}$, $j \in [1, \ldots, k]$. The goal is to assign labels to test data. Specifically, Label Propagation starts by building a transition matrix to represent a fully connected graph. In the graph, nodes represent all the data points, including training and test points. Edges between each pair of data points i, j is weighted so that the closer those two points are in Euclidean space, the larger their weights w_{ij}. Weights are calculated as:

$$w_{ij} = \exp(-\frac{d_{ij}^2}{\sigma^2}) = \exp(\frac{\sum_{d=1}^{D}(x_i^d - x_j^d)^2}{\sigma^2})$$

σ in above expression is a hyper-parameter. Heuristics about choosing appropriate value for σ is discussed in [23]. The conclusion is σ should be set to a value closely related to margin of dataset.

When a transition matrix of weights is set up, it is dot multiplied with a label matrix. Note label matrix contains current label of each point, from both training data and test data. In training data, labels are from $\{-1, 1\}$, corresponding to two classes. In test set, labels before propagation are set to 0. In repeated dot multiplication of two matrices, labels of training data propagate in the graph and reach test data. Eventually at convergence each test data point receives two real real values, corresponding to probabilities of being labeled into either class.

We first give the original label propagation algorithm below. We will use it as a subroutine in our algorithm fair-Label Propagation.

Algorithm 3: Label Propagation

Input : -training examples, $(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_n}, y_n)$
-test examples, $\boldsymbol{x_1}^*, \ldots, \boldsymbol{x_k}^*$
- σ: parameter for creating transition matrix
-transition Matrix T where $T_{ij} = \Pr(j \to i) = \frac{w_{ij}}{\sum_{l=1}^{k+n} w_{lj}}$

Output: Probability Matrix Y which has dimension $(n + k) * C$. For a two class classification problem, $C = 2$. $C[0] = 0$ represent negative class; $C = 2$. $C[1] = 1$ represent positive class.

1 With C being the number of classes, create a $(n + k) * C$ dimension matrix Y, whose ith row represent the label probability distribution of data point x_i;
2 **while** *Y is not converged* **do**
3 | Propagate $Y \leftarrow TY$;
4 | Row-normalize Y;
5 | Clamp the labeled data;
6 **end**
7 return Y;

As a graph based algorithm, Label Propagation makes the propagating effects more vivid. **Figure 2.1** on the following page from [20] gives intuition about how labels propagate through the graph. Note the construction of graph with weighted edges mentioned above conform with the fairness notion "similar people should be labeled similarly".

Now we present fair-Label Propagation which is built upon label propagation but with explicit fairness consideration. Intuitively, fair-Label Propagation can be understood as

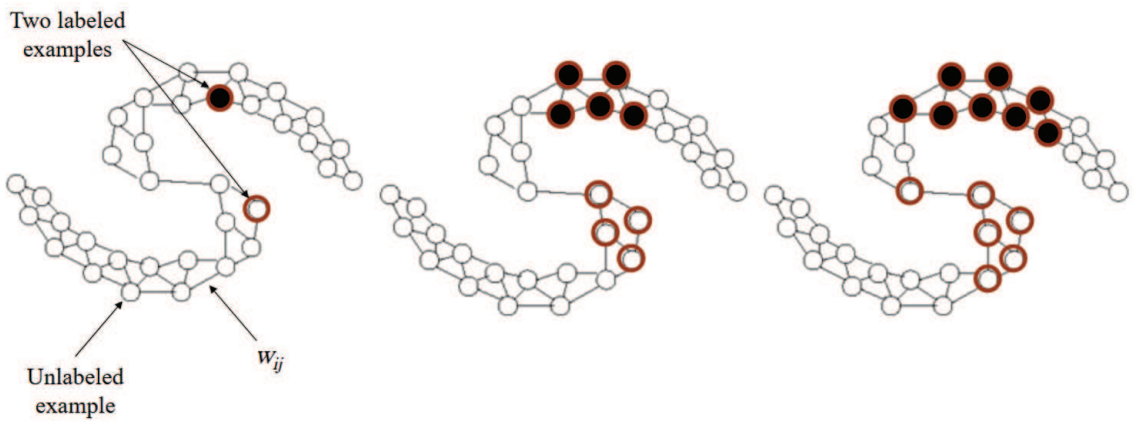

Figure 2.1. Label Propagation

iteratively choosing points chose to current decision boundary to swap labels, and restart propagation until converge. Specifically, label propagation is used as a subroutine to fit a model. After each iteration of label propagation, data points whose two class probability values are close within threshold δ and at the same time satisfying a specified condition will be chosen to swap labels. Here, the specified condition includes the following two possibilities which only one will satisfy for a data point: if the data point being considered is labeled positive, it must be from majority group; if the data point being considered is labeled negative, it must be from minority group. Note the label swapping step is not on a pair of points any more as in TSVM. Here all points whose two class probability are close within threshold δ and meet the conditions would get their labels swapped. Label swapping process itself is worth noting as well. Instead of assigning -1 or 1 to a test point, we assign 100% probability for it to be in either of two classes. If before swap, a point has more than 50% probability to be in positive class, it will receive 100% probability to be in negative class after swap, and vice versa.

The intuition for choosing such points and assign them opposite labels is similar to fair-TSVM. In fair-Label Propagation, close probability values for either class is a proxy of closeness to decision boundary. Points having close probability values after an iteration of Label Propagation are the ones most likely to have been labeled wrong. Changing their class assignment would least likely result in error. Since points that have label swapped also meet specified group and label condition, we are sure statistical parity of the model is improved after label swapping. Below we present fair-Label Propagation. Label swapping

step is highlighted.

Algorithm 4: fair-Label Propagation

Input : -training examples, $(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_n}, y_n)$
-test examples, $\boldsymbol{x_1}^*, \ldots, \boldsymbol{x_k}^*$
- σ: parameter for creating transition matrix
- δ: probability threshold to decide swapping candidates
-transition Matrix T where $T_{ij} = \Pr(j \to i) = \frac{w_{ij}}{\sum_{l=1}^{k+n} w_{lj}}$

Output: Probability Matrix Y which has dimension $(n+k) * C$

With C being the number of classes, create a $(n+k) * C$ dimension matrix Y, whose ith row represent the label probability distribution of data point x_i. For originally unlabeled points, probability of being in either class is 0.;
while *Y is not converged* **do**
    Y = Label Propagation ($[(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_n}, y_n)]$, $[\boldsymbol{x_1}^*, \ldots, \boldsymbol{x_k}^*]$, σ, T) ;
    for *each data point* $\boldsymbol{x}$ **do**
        if $i : x_a = -1 \& Y_i[0] - Y_i[1] <= \delta$ **then**
            $y_i = 1$;
        if $i : x_a = 1 \& Y_i[1] - Y_i[0] <= \delta$ **then**
            $y_i = 0$;
    end

end
list[$\arg\max_c Y_i[c]$];

Note that in both fair-TSVM and fair-Label Propagation, there is one hyper-parameter δ that controls which points are chosen for label swap. It is a parameter that has close relation with both statistical parity and accuracy of the classifier. This is because the larger this parameter is, the points that are further away from current decision boundary are chosen to swap labels. Such points are less likely to have wrong labels to begin with. In other words, we are more confident about their current labels and believe they may not need label swapped. Therefore, swapping their labels means an likely increase in error. Note that the larger this threshold is, the more points are selected to swap labels. For statistical parity, this means larger threshold will improve statistical parity more, for each such swap improves statistical parity of the classifier.

With access to such a tunable hyper-parameter that relates to both accuracy and statistical parity, our algorithms are advantageous in that they allow users to make informed choice on the degree of error and statistical parity they think appropriate.

CHAPTER 3

EXPERIMENTS

In this chapter, we describe experimental results on fair-TSVM and fair-Label Propagation.

3.1 Data

Instead of using existing datasets as appeared in other fair learning literature, we create a dataset that satisfies fair semi-supervised learning assumptions specified in problem setup. In existing datasets, for a data point from minority group, it is likely that its ground truth label as shown in the dataset is biased. Such biased labels create problems when we measure accuracy of learned classifier by comparing predictions with such ground truth labels. Therefore we create a fair dataset which satisfy semi-supervised learning assumptions.

The created dataset contains two classes, positive and negative, and two groups, protected and unprotected. Ground truth labeling in each class with respect to each group achieves statistical parity. We refer to this data set as S. In S, feature vectors $\boldsymbol{x_i}$ have equal possibility of in either group. This is to say we dictates in this data set there are equal number of points from either group. Given an arbitrary data point, when knowing its group feature, its chances of receiving positive label are the same no matter which group it is from. This is to say, $\Pr(y_i = 1 | i : x_a = -1) = \Pr(y_i = 1 | i : x_a = 1)$ for arbitrary data point i. We simulate the skewed labeling process described in problem setup with below described steps. Suppose we want n labeled training points and the remaining $|S| - n$ will be used as unlabeled test points.

1. Create training data. From data set S with both features and labels, specifically from positive class, choose $0.4 * n$ points that are from majority group, and choose $0.1 * n$ points that are from minority group. From negative class, choose $0.4 * n$ points that

are from majority group, and choose $0.1 * n$ points that are from minority group. Combine them to form training data.

2. Create test data. Take out training data points identified in above step from original data set S. Mask labels of remaining data points. Consider remaining data points as test set.

For notation clarity, we restate that training data contains points $(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_n}, y_n)$. They correspond to training instances notations used in algorithms in Chapter 2. Test data contains points $\boldsymbol{x_1}^*, \ldots, \boldsymbol{x_k}^*$. They correspond to test instances notations used in algorithms in Chapter 2.

3.2 Implementation and Code

Fair-TSVM and fair-Label Propagation are implemented and code for both are available upon request.

Fair-TSVM is implemented with the language Julia. We use a optimization solver library in Julia called JuMP to solve the quadratic optimization problem mentioned in Chapter 2. For easier comparison with fair-Label Propagation, we use radial basis function kernel to transform features. The choice of weights C, C^* in optimization objective and kernel bandwidth in radial basis function are cross-validated on pure TSVM implementation. Best performing values for those hyper-parameters are chosen to be used in fair-TSVM.

Fair-Label Propagation contains pure Label Propagation as a subroutine. We choose to use existing implementation of Label Propagation from Scikit-learn python package. Hyper-parameters in Label Propagation setup include kernel choice, kernel bandwidth, maximum iteration. We choose radial basis function kernel to be consistent with fair-TSVM. For kernel bandwidth and maximum iteration, we cross-validate on pure Label Propagation and the best performing values are chosen to use in fair-Label Propagation.

With all other hyper-parameters cross-validated and values chosen, the only remaining hyper-parameter is a distance threshold δ. It is used in both fair algorithms to select candidate points to swap labels. In our experiment, we hope to show such a distance threshold correlates with both statistical parity and accuracy. We thus try different values for threshold δ and observe changes in both statistical parity and accuracy.

3.3 Performance Measures

Recall there are a few things we hope to show with our experiments. First, we hope to show that classifiers trained with semi-supervised algorithms, and fair semi-supervised algorithms by involving unlabeled unbiased data perform better in fairness and at the same time does not compromise on accuracy. Second, we hope to show label swapping process helps recover a better decision boundary in terms of both accuracy and fairness. With this goal, we hope proposed fair semi-supervised algorithms find classifiers that are better in those aspects than classifier learned with pure semi-supervised algorithms. Third, we hope to show swapping threshold, a hyper-parameter in fair semi-supervised algorithms is correlated with both accuracy and fairness of a classifier. Tuning such a variable, users of proposed fair algorithms can decide best for themselves the degree of error and fairness they want for the classifier and prediction results.

With above mentioned goals, we experiment with these five algorithms: SVM, TSVM, fair-TSVM, Label Propagation, fair Label Propagation. All five algorithms are run on the same data set constructed as described in Data section. We compare learned classifiers in both fairness and accuracy. SVM, because of its fully supervised setup, is used as a baseline. TSVM and Label Propagation are pure semi-supervised algorithms without fairness consideration. We hope to see classifiers learned with these two algorithms show some improvement in terms of accuracy and fairness than classifier learned with SVM. Fair-TSVM and fair-Label Propagation are fair semi-supervised algorithms proposed in Chapter 2. We hope to see classifiers learned with those two algorithms perform better in terms of both accuracy and fairness than pure semi-supervised algorithms. Note from existing fairness notions, we chose Statistical Parity as our fairness measure.

3.4 Results

Our experiments achieved results as expected. Specifically, on dataset described in previous section, classifier learned with fair-Label Propagation has better accuracy and statistical parity than SVM and pure Label Propagation. Shown in **Figure 3.1** on page 18, classifiers learned with fair-Label Propagation with RBF kernel bandwidth $\gamma = 1000$ and swapping threshold $\delta = 0.28$ and $\delta = 0.5$ achieved most satisfactory results. Classifier learned with Fair-TSVM was able to achieve significant fairness improvement than SVM

and TSVM.

From **Figure 3.2** on the following page and **Figure 3.3** on page 19, we can see how error and statistical parity of fair-TSVM classifiers change as swapping threshold δ changes. Error of fair-TSVM classifiers starts to increase with increased threshold value. This verifies our hypothesized relation between swap threshold and error mentioned at end of previous chapter. Statistical parity of fair-TSVM classifiers reaches a minimum value at a particular threshold as threshold increase. This is probably a complication of fair-TSVM working on the particular non-linearly separable dataset. Since each time fair-TSVM only swaps one pair of labels, it could be with larger swap threshold, new classifier fit after label swap reverse some of the statistical effects achieved by classifiers in previous iterations.

Figure 3.4 on page 19 and **Figure 3.5** on page 20 also show a correlation between error and swapping threshold. **Figure 3.4** on page 19 shows error first decreases with increased threshold, and then increases. This is the most ideal case expected. It means until error reaches minimum, points within threshold tested are exactly the ones labeled wrong because of discrimination. Swapping labels of these points improves statistical parity as well as accuracy. **Figure 3.5** on page 20 shows statistical parity decrease as we increase swapping threshold. This confirms hypothesized relation between those two variables mentioned at end of previous chapter.

Our experiments verify existence of trade-off between accuracy and statistical parity. With explicit plots describing such a trade-off, our proposed algorithms quantify the trade-off by associating both variables with value of swapping threshold. This setup allow users of algorithms to explicitly measure accuracy and fairness, and make informed decisions.

Figure 3.1. Comparison of Algorithms

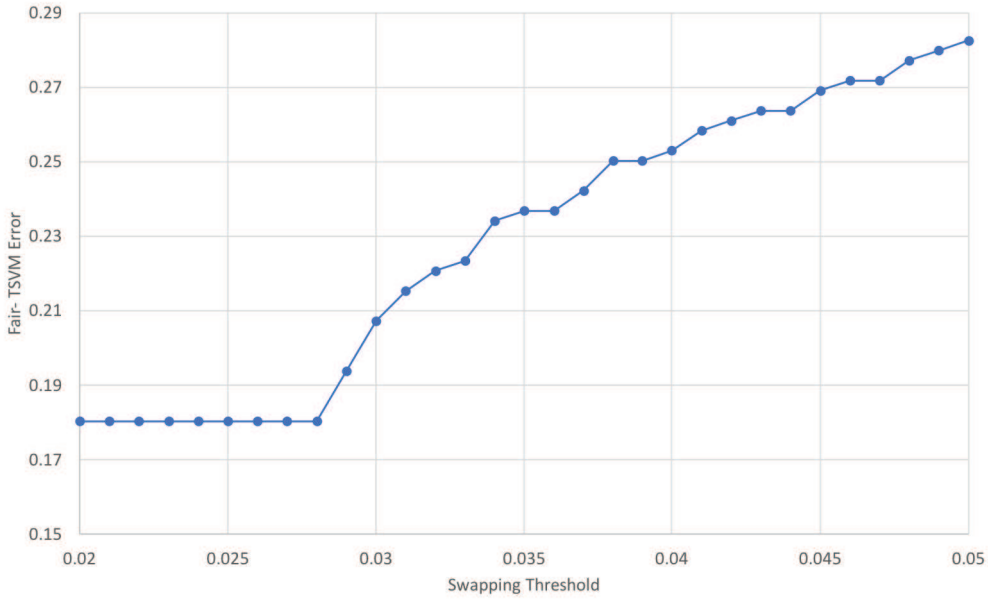

Figure 3.2. Error of Fair-TSVM

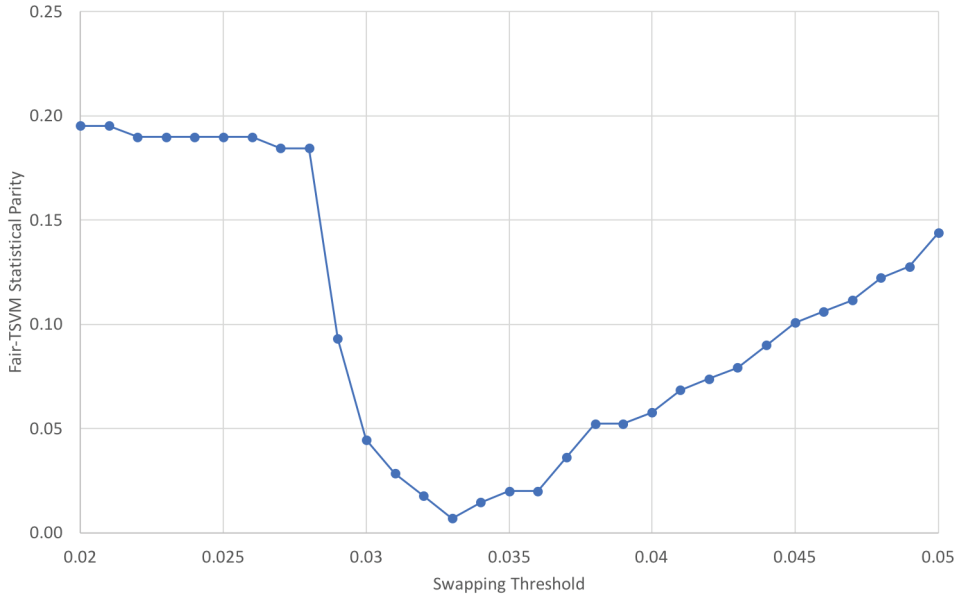

Figure 3.3. Statistical Parity of Fair-TSVM

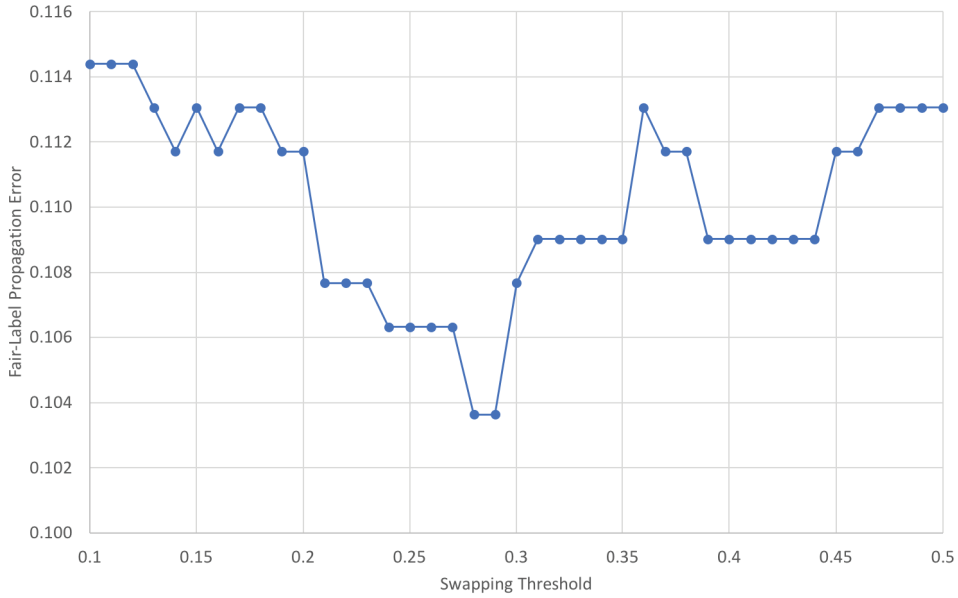

Figure 3.4. Error of Fair Label Propagation with rbf gamma=1000

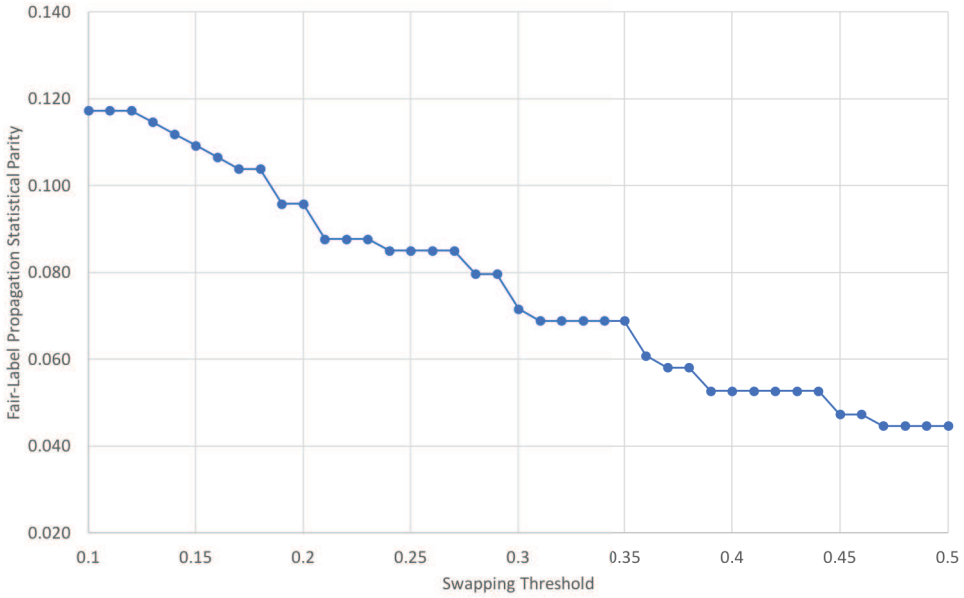

Figure 3.5. Statistical Parity of Fair Label Propagation with rbf gamma=1000

CHAPTER 4

DISCUSSION

There are still many aspects of fair learning in semi-supervised setting that are not fully examined. Below we motivate a few topics which call for further studies.

4.1 Representation of Group Features

The framework and results introduced in [22] show unlabeled data help to learn the correct decision boundary under manifold and cluster assumptions. Manifold assumption and cluster assumptions is the key connection between $\Pr(X)$ and $\Pr(Y|X)$. This connection defines cases when unlabeled data will help. In other words, when this connection is missing, semi-supervised algorithms may not work better than supervised algorithms. However, the manifold assumption may not always hold true. In fact in the realm of fairness learning, or on other learning problems where features contain discrete values, it cannot be assumed that feature space is a manifold. Our work empirically shows that even in such cases, there are ways to involve unlabeled data and learn a more accurate approximation to true decision function, namely with proposed fair-Label Propagation algorithm. However, we could also question the choice of discrete values $\{-1, 1\}$ to represent group features. Does it make sense to choose such values in relation to the rest feature values in the dataset? We would further study the choices of group feature values. With reasonable values for group features, feature space may be considered a closer approximation to a manifold. In such cases, with the algorithms proposed in [22], we may hope to learn classifiers better than those learned with pure semi-supervised algorithms in our experiments.

4.2 Label Swapping on One Pair vs. More Pairs

There is a difference in label swapping step of the two fair algorithms given in Chapter 2. Specifically, fair-TSVM only swaps one pair of points in each swapping iteration. On

the other hand, fair-Label Propagation swaps labels of all points that are within threshold distance from current decision boundary. We could also have done the same in fair-TSVM, with selecting and swapping labels of all points whose distance from current decision boundary are within threshold. Because RBF kernel fair-TSVM is highly similar with RBF kernel label propagation, we expect swapping labels of more points in each iteration to improve performance of fair-TSVM to be similar to fair-Label Propagation. In the same spirit, we have tried fair-Label Propagation with swapping one pair of points in each iteration, following the same scheme used in fair-TSVM. Classifier learned from such a fair-Label Propagation algorithm has very similar performance in accuracy and fairness as fair-TSVM classifiers. Since fair-Label Propagation shows better and closer to expected performance than fair-TSVM, we believe swapping all points within theshold in one swap iteration is a better heuristic. The less desired performance of fair-TSVM is likely because swapping one pair of labels exert limited correction effects.

In formulating fair-Label Propagation, besides giving candidate points 100% probability of being in opposite class, we could also swap the two probability values.This means, if in the pair selected, majority group point has higher probability of being in positive class, it ends up with less probability of in positive class after the swap, and vice versa for the minority group point in selected pair. This setup is also experimented. It shows less significant improvement in fairness comparing to giving a candidate point 100% probability to be in opposite class. The relatively moderate improvement can be understood as making limited correction in both fairness and accuracy in the training process. And in situations where biased labels of training data are more influential, those moderate influence from swapping probability values will be overwhelmed.

4.3 Summary and Conclusions

In this work, we study the possibility of using unlabeled data points to learn a fair and accurate classifier, in cases where labeled points are biased. With an assumption that classification-relevant qualities are equally distributed in both population groups, unlabeled data can be considered as an unbiased feature representation for classification task.

To show fair learning is possible in above mentioned setting, we propose two fair

learning algorithms. We employ a procedure to swap labels of unlabeled unbiased test points that are most likely labeled wrong by biased classifier and refit a classifier after label swapping. We carry out those two steps repeatedly and iteratively. We show proposed fair algorithm fair-Label Propagation performs better than pure semi-supervised algorithms and supervised algorithm in both accuracy and fairness, and pure semi-supervised algorithm Label Propagation perform better in terms of accuracy and fairness than pure supervised algorithm. We show proposed fair algorithm fair-TSVM performs better than pure semi-supervised algorithms and supervised algorithm in Statistical Parity and in the same time does not compromise accuracy by a lot.

We also show through experiment that on a learning task, we could empirically decide a threshold for label swapping so that we can make informed decision about trade-off between fairness and accuracy of learned classifier and prediction on test points.

REFERENCES

[1] M.-F. Balcan and A. Blum, *A pac-style model for learning from labeled and unlabeled data*, in International Conference on Computational Learning Theory, Springer, 2005, pp. 111–126.

[2] A. Barry-Jester, B. Casselman, and D. Goldstein, *The new science of sentencing*. The Marshall Project, Aug. 2015.

[3] R. Berk, H. Heidari, S. Jabbari, M. Joseph, M. Kearns, J. Morgenstern, S. Neel, and A. Roth, *A convex framework for fair regression*, arXiv preprint arXiv:1706.02409, (2017).

[4] N. Byrnes, *Artificial intolerance*. MIT Technology Review, Mar. 2016.

[5] F. Chierichetti, R. Kumar, S. Lattanzi, and S. Vassilvitskii, *Fair clustering through fairlets*, in Advances in Neural Information Processing Systems, 2017, pp. 5029–5037.

[6] C. Cortes and V. Vapnik, *Support-vector networks*, Machine learning, 20 (1995), pp. 273–297.

[7] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, *Fairness through awareness*, in Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, ACM, 2012, pp. 214–226.

[8] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, *Certifying and removing disparate impact*, in Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 259–268.

[9] B. Fish, J. Kun, and . Lelkes, *A Confidence-Based Approach for Balancing Fairness and Accuracy*, in Proceedings of the 2016 SIAM International Conference on Data Mining, Proceedings, Society for Industrial and Applied Mathematics, June 2016, pp. 144–152.

[10] M. Hardt, E. Price, N. Srebro, et al., *Equality of opportunity in supervised learning*, in Advances in neural information processing systems, 2016, pp. 3315–3323.

[11] T. Joachims, *Transductive inference for text classification using support vector machines*, in ICML, vol. 99, 1999, pp. 200–209.

[12] M. Joseph, M. Kearns, J. Morgenstern, S. Neel, and A. Roth, *Better fair algorithms for contextual bandits*.

[13] ——, *Fair algorithms for infinite and contextual bandits*, arXiv preprint arXiv:1610.09559, (2016).

[14] M. Joseph, M. Kearns, J. H. Morgenstern, and A. Roth, *Fairness in learning: Classic and contextual bandits*, in Advances in Neural Information Processing Systems, 2016, pp. 325–333.

[15] F. Kamiran and T. Calders, *Classifying without discriminating*, in Computer, Control and Communication, 2009. IC4 2009. 2nd International Conference on, IEEE, 2009, pp. 1–6.

[16] ——, *Data preprocessing techniques for classification without discrimination*, Knowledge and Information Systems, 33 (2012), pp. 1–33.

[17] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma, *Fairness-aware classifier with prejudice remover regularizer*, in Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2012, pp. 35–50.

[18] B. T. Luong, S. Ruggieri, and F. Turini, *k-nn as an implementation of situation testing for discrimination discovery and prevention*, in Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2011, pp. 502–510.

[19] C. Miller, *Can an algorithm hire better than a human?* The New York Times, June 2015.

[20] J. Rong, *Semi-supervised learning*. SlidePlayer.

[21] B. Schölkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. Mooij, *On causal and anticausal learning*, arXiv preprint arXiv:1206.6471, (2012).

[22] V. Sindhwani, M. Belkin, and P. Niyogi, *11 the geometric basis of semi-supervised learning*, (2006).

[23] Z. Xiaojin and G. Zoubin, *Learning from labeled and unlabeled data with label propagation*, Tech. Rep., Technical Report CMU-CALD-02–107, Carnegie Mellon University, (2002).