SEMI-SUPERVISED LEARNING AND FAIRNESS-AWARE LEARNING UNDER CLASS IMBALANCE

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des akademischen Grades

DOKTOR DER NATURWISSENSCHAFTEN

Dr. rer. nat.

genehmigte Dissertation
von

M. Sc. Vasileios Iosifidis

geboren am 28. Februar 1991, in Athens, Griechenland

Hannover, Deutschland, 2020

Referent: Prof. Dr. Eirini Ntoutsi
Korreferent: Prof. Dr. Salvatore Ruggieri
Korreferent: Prof. Dr. Ujwal Gadiraju
Tag der Promotion: 31.07.2020

ABSTRACT

With the advent of Web 2.0 and the rapid technological advances, there is a plethora of data in every field; however, more data does not necessarily imply more information, rather the quality of data (veracity aspect) plays a key role. Data quality is a major issue, since machine learning algorithms are solely based on historical data to derive novel hypotheses. Data may contain noise, outliers, missing values and/or class labels, and skewed data distributions. The latter case, the so-called *class-imbalance* problem, is quite old and still affects dramatically machine learning algorithms. Class-imbalance causes classification models to learn effectively one particular class (majority) while ignoring other classes (minority). In extend to this issue, machine learning models that are applied in domains of high societal impact have become biased towards groups of people or individuals who are not well represented within the data. Direct and indirect discriminatory behavior is prohibited by international laws; thus, there is an urgency of mitigating discriminatory outcomes from machine learning algorithms.

In this thesis, we address the aforementioned issues and propose methods that tackle class imbalance, and mitigate discriminatory outcomes in machine learning algorithms. As part of this thesis, we make the following contributions:

- *Tackling class-imbalance in semi-supervised learning* – The class-imbalance problem is very often encountered in classification. There is a variety of methods that tackle this problem; however, there is a lack of methods that deal with class-imbalance in the semi-supervised learning. We address this problem by employing data augmentation in semi-supervised learning process in order to equalize class distributions. We show that semi-supervised learning coupled with data augmentation methods can overcome class-imbalance propagation and significantly outperform the standard semi-supervised annotation process.
- *Mitigating unfairness in supervised models* – Fairness in supervised learning has received a lot of attention over the last years. A growing body of pre-, in- and post-processing approaches has been proposed to mitigate algorithmic bias; however, these methods consider error rate as the performance measure of the machine learning algorithm, which causes high error rates on the under-represented class. To deal with this problem, we propose approaches that operate in pre-, in- and post-processing layers while accounting for all classes. Our proposed methods outperform state-of-the-art methods in terms of performance while being able to mitigate unfair outcomes.

Keywords: *class-imbalance, fairness-aware learning, semi-supervised learning, supervised learning*

ZUSAMMENFASSUNG

Mit dem Aufkommen des Web 2.0 und den rasanten technologischen Fortschritten gibt es in jedem Bereich eine Fülle von Daten; mehr Daten bedeuten jedoch nicht unbedingt mehr Informationen, vielmehr spielt die Qualität der Daten (Aspekt der Wahrhaftigkeit) eine Schlüsselrolle. Die Datenqualität ist ein wichtiges Thema, da die Algorithmen des maschinellen Lernens ausschlieSSlich auf historischen Daten basieren, um neue Hypothesen abzuleiten. Die Daten können Rauschen, AusreiSSer, fehlende Werte und/oder Klassenbezeichnungen sowie schiefe Datenverteilungen enthalten. Der letztgenannte Fall, das so genannte Klassenungleichgewichtsproblem, ist ziemlich alt und wirkt sich immer noch dramatisch auf die Algorithmen des maschinellen Lernens aus. Das Klassenungleichgewicht führt dazu, dass Klassifikationsmodelle eine bestimmte Klasse (Mehrheit) effektiv lernen, während andere Klassen (Minderheit) ignoriert werden. Was diese Frage betrifft, so sind die Modelle des maschinellen Lernens, die in Bereichen mit hoher gesellschaftlicher Wirkung angewandt werden, inzwischen gegenüber Personengruppen oder Einzelpersonen, die in den Daten nicht gut repräsentiert sind, verzerrt. Direktes und indirektes diskriminierendes Verhalten ist nach internationalem Recht verboten; daher besteht die Dringlichkeit, diskriminierende Ergebnisse von Algorithmen des maschinellen Lernens abzuschwächen.

In dieser Arbeit befassen wir uns mit den oben genannten Fragen und schlagen Methoden vor, die das Klassenungleichgewicht angehen und diskriminierende Ergebnisse in Algorithmen des maschinellen Lernens mildern. Als Teil dieser Arbeit leisten wir die folgenden Beiträge:

- *Bewältigung des Klassenungleichgewichts beim halbüberwachten Lernen* – Das Problem des Klassenungleichgewichts tritt sehr häufig bei der Klassifizierung auf. Es gibt eine Vielzahl von Methoden, mit denen dieses Problem angegangen werden kann; es mangelt jedoch an Methoden, die sich mit dem Klassenungleichgewicht beim teilüberwachten Lernen befassen. Wir gehen dieses Problem an, indem wir beim halbüberwachten Lernprozess eine Datenanreicherung einsetzen, um die Klassenverteilungen auszugleichen. Wir zeigen, dass halbüberwachtes Lernen in Verbindung mit DatenvergröSSerungsmethoden die Ausbreitung von Klassenungleichgewichten überwinden und den standardmäSSigen halbüberwachten Annotationsprozess deutlich übertreffen kann.
- *Milderung von Ungerechtigkeiten in beaufsichtigten Modellen* – Der Fairness beim betreuten Lernen wurde in den letzten Jahren viel Aufmerksamkeit geschenkt. Es wurde eine wachsende Zahl von Vor-, Ein- und Nachbearbeitungsansätzen vorgeschlagen, um die algorithmische Verzerrung abzuschwächen; diese Methoden betrachten jedoch die Fehlerquote als LeistungsmaSS des Algorithmus des maschinellen Lernens, was zu hohen Fehlerquoten in der unterrepräsentierten Klasse führt. Um dieses Problem zu lösen, schlagen wir Ansätze vor, die in Pre-, In- und Post-Processing-Schichten arbeiten und dabei alle Klassen berücksichtigen. Die von uns vorgeschlagenen Methoden übertreffen den Stand der Technik in Bezug auf die Leistung und sind gleichzeitig in der Lage, unfaire Ergebnisse abzuschwächen.

Schlagwörter: *Klassenungleichgewicht, Fairness-bewusstes Lernen, halbüberwachtes Lernen, überwachtes Lernen*

ACKNOWLEDGMENTS

I would like to thank my colleagues, family, and friends who supported me throughout the course of my PhD studies. Special thanks to my advisor Prof. Eirini Ntoutsi who guided me in every step of my PhD studies. Also, I want to specially thank my friends and colleagues: Nikky, Uj, Boubaki and, Mairy for our countless shared moments, and the unlimited hours we spent together at KBS.

FOREWORD

Throughout the course of my PhD studies, I have worked, published and co-authored several papers in the areas of class imbalance, online learning, fairness-aware learning, semi-supervised learning, supervised learning, and semantic web.

The core contributions of this thesis in the individual chapters have been published in the following venues:

1. The contributions in Chapter 3, which deal with the problem of class imbalance are published in:
   - [IN17] Iosifidis, V. and Ntoutsi, E., 2017, August. Large scale sentiment learning with limited labels. In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1823-1832).
   - [IN19b] Iosifidis, V. and Ntoutsi, E., 2019. Sentiment analysis on big sparse data streams with limited labels. Knowledge and Information Systems, pp.1-40.
2. The contributions in Chapters 4 and 5, which deal with the problem of supervised fairness-aware classification are published in:
   - [IFN19] Iosifidis, V., Fetahu, B. and Ntoutsi, E., 2019, December. FAE: A Fairness-Aware Ensemble Framework. In 2019 IEEE International Conference on Big Data (Big Data) (pp. 1375-1380). IEEE.
   - [IN19a] Iosifidis, V. and Ntoutsi, E., 2019, November. AdaFair: Cumulative Fairness Adaptive Boosting. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 781-790).

The complete list of publications during my PhD is shown below:

1. [IN17] **Iosifidis, V.** and Ntoutsi, E., 2017, August. Large scale sentiment learning with limited labels. In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1823-1832).

2. [FISN17] Fafalios, P., **Iosifidis, V.**, Stefanidis, K., & Ntoutsi, E. (2017, September). Multi-aspect entity-centric analysis of big social media archives. In International Conference on Theory and Practice of Digital Libraries (pp. 261-273). Springer, Cham.

3. [ION17] **Iosifidis, V.**, Oelschlager, A. and Ntoutsi, E., 2017, September. Sentiment classification over opinionated data streams through informed model adaptation. In International conference on theory and practice of digital libraries (pp. 369-381). Springer, Cham.

4. [IN18] **Iosifidis, V.** and Ntoutsi, E., 2018. Dealing with bias via data augmentation in supervised learning scenarios. Jo Bates Paul D. Clough Robert Jäschke, p.24.

5. [MIE+18] Mohapatra, N., **Iosifidis, V.**, Ekbal, A., Dietze, S., & Fafalios, P. (2018). Time-aware and corpus-specific entity relatedness. arXiv preprint arXiv:1810.10004.

6. [FISN18] Fafalios, P., **Iosifidis, V.**, Stefanidis, K., & Ntoutsi, E. (2018). Tracking the history and evolution of entities: entity-centric temporal analysis of large social media archives. International Journal on Digital Libraries, 1-13.

7. [FIND18] Fafalios, P., **Iosifidis, V.**, Ntoutsi, E., & Dietze, S. (2018, June). Tweetskb: A public and large-scale rdf corpus of annotated tweets. In European Semantic Web Conference (pp. 177-190). Springer, Cham.

8. [MCI+18] Melidis, D. P., Campero, A. V., **Iosifidis, V.**, Ntoutsi, E., & Spiliopoulou, M. (2018, June). Enriching Lexicons with Ephemeral Words for Sentiment Analysis in Social Streams. In Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics (p. 38). ACM.

9. [ITN19] **Iosifidis, V.**, Tran, T. N. H., & Ntoutsi, E. (2019, August). Fairness-enhancing interventions in stream classification. In International Conference on Database and Expert Systems Applications (pp. 261-276). Springer, Cham.

10. [IN19b] **Iosifidis, V.** and Ntoutsi, E., 2019. Sentiment analysis on big sparse data streams with limited labels. Knowledge and Information Systems, pp.1-40.

11. [IFN19] **Iosifidis, V.**, Fetahu, B. and Ntoutsi, E., 2019, December. FAE: A Fairness-Aware Ensemble Framework. In 2019 IEEE International Conference on Big Data (Big Data) (pp. 1375-1380). IEEE.

12. [IN19a] **Iosifidis, V.** and Ntoutsi, E., 2019, November. AdaFair: Cumulative Fairness Adaptive Boosting. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 781-790).

13. [GTK+19] Gottschalk, S., Tempelmeier, N., Kniesel, G., **Iosifidis, V.**, Fetahu, B. and Demidova, E., 2019, September. Simple-ML: Towards a Framework for Semantic Data Analytics Workflows. In International Conference on Semantic Systems (pp. 359-366). Springer, Cham.

14. [NFG+20] Ntoutsi, E., Fafalios, P., Gadiraju, U., **Iosifidis, V.**, Nejdl, W., Vidal, M.E., Ruggieri, S., Turini, F., Papadopoulos, S., Krasanakis, E. and Kompatsiaris, I., 2020. Bias in datadriven artificial intelligence systemsAn introductory survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, p.e1356.

15. [HIL+20] Hu, H., **Iosifidis, V.**, Liao, W., Zhang, H., YingYang, M., Ntoutsi, E. and Rosenhahn, B., 2020. FairNN-Conjoint Learning of Fair Representations for Fair Decisions. In Proceedings of the 23rd International Conference on Discovery Science.

16. [IN20] **Iosifidis, V.** and Ntoutsi, E., 2020. FABBOO - Online Fairness-aware Learning under Class Imbalance. In Proceedings of the 23rd International Conference on Discovery Science.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

The rise of Web 2.0 and the rapid technological advances have caused a data explosion which is beneficial for decision support systems. Such systems rely on historical data in order to derive novel hypotheses to fit the data without being explicitly coded; thus, making such systems applicable in a wide variety of fields. Decision support systems benefit from large amounts of data, which makes the data availability extremely valuable. Nonetheless, more data does not necessarily imply more information, rather the quality of data plays a key role. In many real-world problems, data are incomplete, contain noise and/or outliers, have encoded biases and/or skewed class distributions (class-imbalance), and so on and so forth. It is crucial to deal with such problems, since machine learning algorithms, which are trained on real-world data, are prone to amplify existing errors (garbage in, garbage out) [Par14].

The problem of class-imbalance [HM13] is quite old, and it is frequently observed in all volume ranges of data. Class-imbalance refers to uneven class distributions i.e., when one class (majority) is exceeding by far the other classes (minority classes) w.r.t number of instances. This issue can lead to disproportional error-rates among different classes. Conventional machine learning algorithms do not take this issue into consideration, and propagate it to their outcomes. Another problem which lies in imbalanced datasets, is the problem of rare cases [Ste13] (an example of rare cases can be seen in Figure 1.1). Rare cases (also called *within-class imbalance* or *group-imbalance*) are often treated as outliers or noise since they are not very similar to the majority of the instances within a class e.g., a medical dataset which contains healthy and sick patients may contain sick patients that suffer from a very rare disease.

Another common problem with data, is that they are often unlabeled. To obtain qualitative labels for the data, one has to rely on human annotators; however, the task of human annotation is time-consuming and expensive. In addition, nowadays data velocity has surpassed petabytes per day (velocity refers to data generation rates)

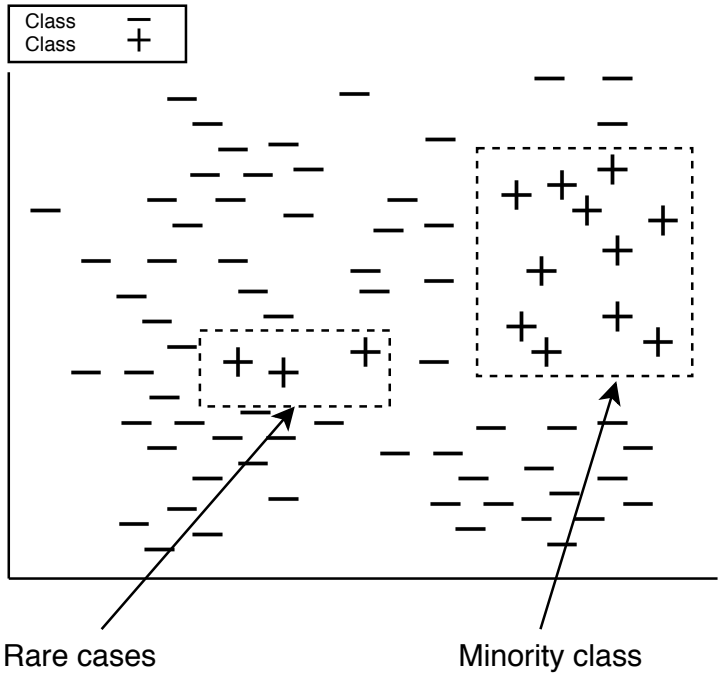


Figure 1.1: Example of rare cases in class-imbalanced data

which makes it impossible to annotate all these data by relying on human annotation. In many fields the labeled data are limited, and the unlabeled data are disproportionately more than the labeled (an example can be seen in Figure 1.2). A whole field of research in machine learning, called semi-supervised learning [CSZ09], focuses on combining the labeled and unlabeled data to estimate labels for the unlabeled data, which can be used afterwards by conventional decision support systems. Figure 1.2 is an example that demonstrates how the unlabeled data can help to estimate efficiently the decision boundary.

Tackling class-imbalance in semi-supervised learning. The first aspect of this thesis is to focus on the problem of class-imbalance propagation in the semi-supervised setting. Semi-supervised methods do not take into consideration the uneven class distributions; thus, they are prone to propagate class imbalanced outcomes which can deteriorate the overall performance. By equalizing class distributions, we force these methods to effectively learn all classes to avoid such deterioration.

Finally, in areas of high societal impact, data and machine learning usage raise concerns regarding privacy, fairness, legal and ethical guidelines [otPP14]. Over the past few years, many cases have been reported, in which machine learning algorithms produced discriminatory outcomes [DARW+19, ASB+19, VHLY15, IS16, EL14, DTD15]. In Amazon's case [IS16], the decision support system was deciding which regions of New York City would receive prime services and which would not. Although Amazon's system did not use the protected attribute *race*, it excluded areas such as Bronx,

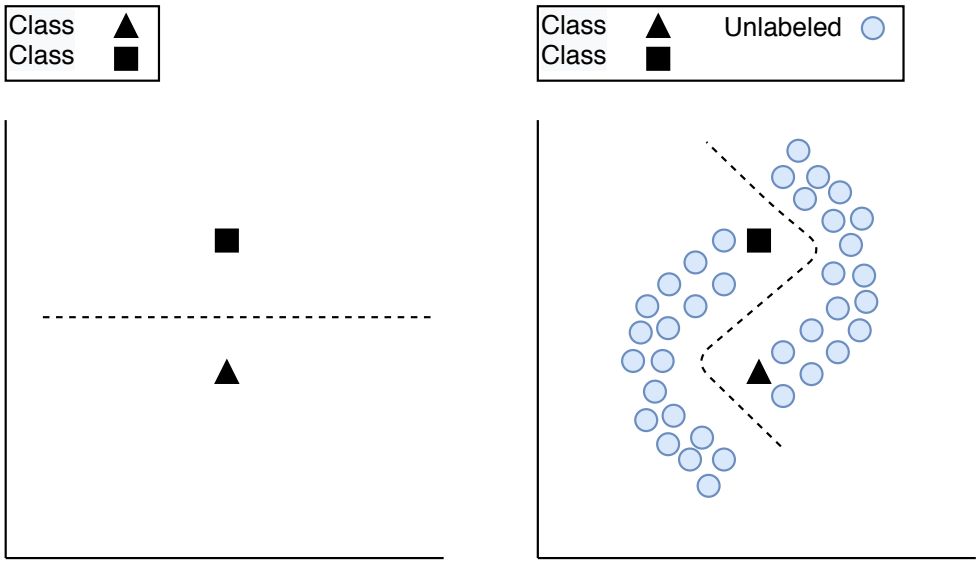


Figure 1.2: Example of labeled and unlabeled data. The decision boundary can change significantly with the use of unlabeled data.

which were predominantly African-American ZIP codes.

Data often contain *proxy-attributes* to other attributes such as *race or gender* attributes (also called *protected attributes*), which can lead a decision support system to produce discriminatory outcomes. Moreover, data reflect *societal biases*, and are not representative of the whole population (*sample bias*). In addition, *system bias* might lead to generation of biased data causing models that further reinforce such discriminatory policies like in predictive policing [LI16]. Finally, machine learning models come with their own assumptions and biases (*model bias*), which clearly affect their generalization performance. Due to the complexity of big data and machine learning algorithms, and their complex interactions, integrating fairness-enhancing interventions into the learning process is essential [FSV+19].

Multiple bias sources can cause machine learning algorithms to discriminate namely: *class imbalance*, *rare cases*, and *class overlap*. A demonstration of these issues is shown via the toy example of Figure 1.3: the *protected* and *non-protected* groups w.r.t. some protected attribute are depicted using different symbols, the class assignments are color-coded.

i) *rare cases* defined w.r.t. some protected attribute(s): the protected group (●, ●) is much smaller than the non-protected (◆, ◆). As a real example, women in the STEM workforce are considerably less than men (28% vs 72% in USA[1]).

ii) *class imbalance*: the negative class (●,◆)) dominates the positive class (●,◆)), which

[1] https://ngcproject.org/sites/default/files/ngcp_the_state_of_girls_and_women_in_stem_2018a.pdf

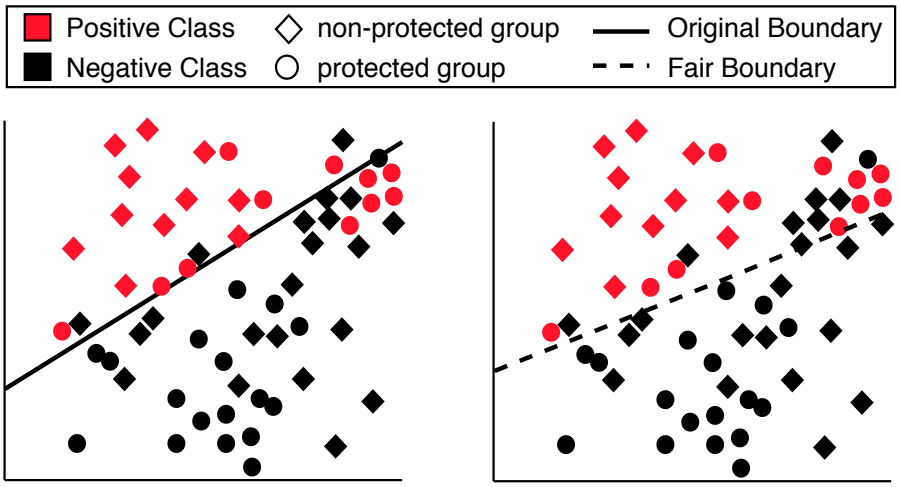


Figure 1.3: Original classifier (left) vs fair classifier (right)

corresponds to granting a benefit (also known as *target class*), and

iii) *class overlap, esp. for the minority class*: the positive instances of the protected group (●) have similar values (or are closer in the *feature space*) with the negative instances of the non-protected group (◆). This is related to the societal biases mentioned before e.g., parental leave is typically longer for mothers compared to fathers.

In the same figure, the decision boundaries of a traditional and a fairness-aware linear classifier are depicted. A traditional learner (left, solid line) optimizes for predictive accuracy - where nearly all protected instances are predicted as negative. A fair-learner (right, dashed line) apart from accuracy, it also considers fairness - where both groups are represented in both classes. It is clear from this example, that a decision support system has to tackle the aforementioned learning challenges.

Mitigating class-imbalance and unfairness in supervised models: The second aspect of this thesis is to focus on the combined problem of class-imbalance and discriminatory outcomes in supervised learning. Although there is a plethora of works which aim to mitigate discriminatory outcomes from supervised models, they do not take into consideration the problem of class-imbalance; thus, these methods reject the vast majority of qualified instances which belong to the minority class due to their inability to learn effectively the minority class.

1.2 Overview of Contributions

Tackling class-imbalance in semi-supervised learning: Semi-supervised methods are widely used to enhance classification performance by leveraging large amounts

of unlabeled data. However, these methods are prone to propagate class-imbalance in case that the data they use, have skewed class distributions.

In this thesis, we extensively evaluate semi-supervised methods such as Co-Training, Expectation Maximization, and Self-Learning on the task of sentiment analysis. We show the impact of the unlabeled data to the overall predictions as well as the impact of different confidence thresholds. In addition, we compare the batch annotation, in which all the data are available in advance, with the stream annotation where data are becoming available over the course of the stream.

Furthermore, we use semi-supervised methods to annotate a large scale corpus of opinionated short texts, called *TSentiment15*, and make it publicly available[2] to the community. TSentiment15 consists of more than 200 million English short texts; thus making it the very first large scale dataset of opinionated short texts. We also provide crowdsource evaluation which was performed on a sample of TSentiment15.

To deal with the class-imbalance propagation, we combine semi-supervised methods with augmentation techniques such as over-sampling, under-sampling, distortion, and semantic-similarity. We show the impact of each method in the overall predictions, and analyze the advantages and disadvantages of each method. Our experiments indicate that coupling semi-supervised and augmentation methods significantly outperform default semi-supervised methods.

Mitigating class-imbalance and unfairness in supervised models: Direct and indirect discrimination is prohibited by international laws [DIR98], which raises the urgency of mitigating unfair machine learning outcomes. There is a variety of reasons which cause machine learning algorithms to become discriminatory e.g., data might encode societal biases, data can contain feedback loops, data can contain different data distributions for different segments, and so on and so forth.

In this thesis, we mitigate unfair behavior from supervised machine learning models by making pre-, in- and post-processing interventions. In particular, we study how class- and within-class imbalance affect the decisions of a model. We propose a fairness-aware ensemble framework, called FAE, which tackles class- and within-class imbalance to mitigate unfair outcomes. FAE performs two fairness-aware interventions: i) the pre-processing step, in which data are partitioned and re-balanced to tackle class- and within-class imbalance and ii) the post-processing step, where the decision boundary of the ensemble is shifted to mitigate unfair outcomes. Our experiments show that models which are trained upon data containing disproportional distributions for each segment, are highly discriminatory, in contrast to our approach.

Furthermore, we study fairness in sequential models such as AdaBoost [Sch99]. We propose the *cumulative fairness* notion which is employed by sequential models to mitigate unfair outcomes by modifying the data distributions w.r.t fairness. In addition, our method optimizes for balanced error rate by selecting a sequence of models which minimizes a loss function that combines the balanced error rate and

[2] https://www.l3s.de/~iosifidis/TSentiment15/

discriminatory behavior. The cumulative fairness notion assesses the fairness behavior of a sequential model from the beginning up to the current round. We show that by assigning fairness related weights to misclassified instances based on our cumulative fairness notion, the induced model is able to produce fair outcomes. Our proposed model, called AdaFair, is able to mitigate discrimination as well as tackle the problem of class imbalance, and outperform recent state-of-the-art fairness-aware approaches.

1.3 Outline of the Thesis

The rest of the thesis is organized as follows. In Chapter 2, we introduce the technical background which is necessary to understand the work of this thesis.

In Chapter 3, we explore the problem of limited labels in big data. Furthermore, we investigate how semi-supervised methods can be coupled with augmentation techniques. We show that default semi-supervised methods are prone to propagate class-imbalance in each iteration, and we provide a mechanism that tackles this issue. The work presented in this chapter is contained in the following publications:

- [IN17] Iosifidis, V. and Ntoutsi, E., 2017, August. Large scale sentiment learning with limited labels. In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1823-1832).
- [IN19b] Iosifidis, V. and Ntoutsi, E., 2019. Sentiment analysis on big sparse data streams with limited labels. Knowledge and Information Systems, pp.1-40.

In Chapters 4 and 5, we investigate the combined problem of unfair outcomes and class imbalance in supervised models. More analytically, in Chapter 4 we study the impact of class- and within-class imbalance w.r.t discriminatory outcomes. We show that when the data suffer from severe class- and within-class imbalance, the induced machine learning model is producing highly discriminatory outcomes. We present a fairness aware ensemble, called FAE, which combines pre- and post-processing fairness aware interventions to handle class imbalance and unfair outcomes. In Chapter 5, we investigate fairness in sequential models such as AdaBoost [Sch99], and introduce the cumulative fairness notion. We show that by assigning fairness related costs in each boosting round based on the cumulative fairness notion, the model is able to produce a strong learner which is able to mitigate discriminatory outcomes.

The works presented in these chapters are contained in the following publications:

- [IFN19] Iosifidis, V., Fetahu, B. and Ntoutsi, E., 2019, December. FAE: A Fairness-Aware Ensemble Framework. In 2019 IEEE International Conference on Big Data (Big Data) (pp. 1375-1380). IEEE.

- [IN19a] Iosifidis, V. and Ntoutsi, E., 2019, November. AdaFair: Cumulative Fairness Adaptive Boosting. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 781-790).

Finally, in Chapter 6 we conclude our work, summarize the main contributions, and discuss about future directions.

2 Technical Background

In this chapter, we introduce all the necessary technical background in order to understand the following chapters of the thesis. We begin with introducing the task of supervised learning and all the employed methods in this work. Also, we give a brief overview of ensemble learning which is part of supervised learning, and mention two well-known methods in this category. Afterwards, we highlight two clustering techniques that have been employed in our work. We continue with semi-supervised learning and its two main methods. In addition, we define the problem of class-imbalance. Finally, we provide an overview of the fairness-aware learning task.

2.1 Supervised Learning

In this section, we introduce the task of supervised learning as well as the employed algorithms in this thesis.

We assume a set of historical data $X = ((x_1, y_1), (x_2, y_2)...(x_n, y_n))$, that consist of n i.i.d. examples drawn from a joint distribution $P(A, Y)$. For the ease of simplicity, we consider the binary classification scenario, where $y \in \{-1, 1\}$ are the class labels, and $A = \{A_0, ..., A_m\}$ being the feature space. The goal of supervised classification is to find a map function $f(X) \rightarrow y$ to predict the class labels of unseen instances.

2.1.1 Decision Trees

Decision tree learning [Utg89] is a very popular method, which is used for supervised learning tasks. A decision tree consists of three main components: the root node, the internal nodes and finally the leaf nodes (Figure 2.1). Given a labeled set of historical data, the decision tree splits the instances, based on a splitting criterion, into different leaves. The internal nodes contain the decisions of each split. The expansion of the decision tree is affected by two factors: i) the split criterion, and ii) the stop criterion.

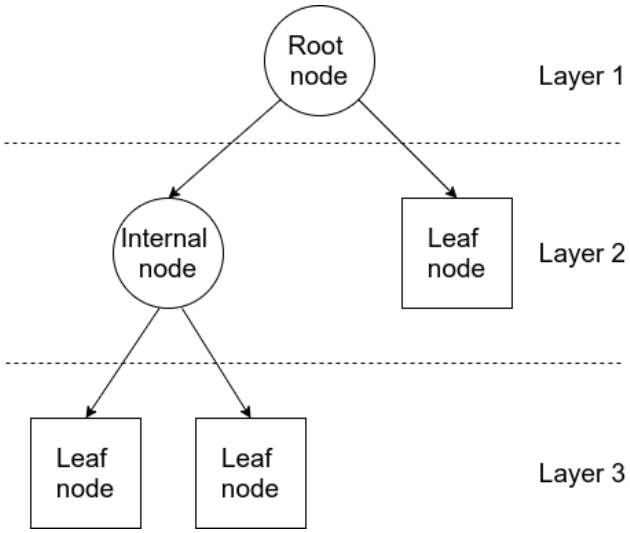


Figure 2.1: Tree structure

The splitting criterion can be defined using a specific measure. The most famous ones are *gini impurity*, *information gain*, *chi-square* and *variance reduction*. In this work, we employ the information gain measure which is used by ID3 [Qui86], C4.5 [Qui14] and C5.0 [KJ13] decision trees. Information gain is an entropy-based measure. By splitting the nodes based on a feature, decision trees try to decrease the entropy on the node. Entropy is defined as:

$$H(X) = -\sum p(X)logp(X) \tag{2.1}$$

where p(X) is a fraction of instances in a given class. Information gain is defined below as:

$$IG(Y, X) = H(Y) - H(Y|X) \tag{2.2}$$

where $H(Y|X)$ is the conditional entropy of class Y given X.

Regarding the stopping criterion, a simple but efficient method is to stop the node splitting as soon as the a minimum threshold of instances inside a leaf is reached. Other methods, called *pruning* methods, have also been proposed and are applied after the construction of the tree. They are bottom-up methods, that cut or merge nodes to simplify the tree and make it more generalizable.

In order to make a prediction, decision trees insert a new instance, which does not contain a class label, into their structure. The instance traverses the tree based on the attribute splits (assuming no missing values, otherwise imputations methods are employed [Twa09]) and ends up in a leaf node m. The estimated probability of the instance to belong to a class k is:

$$p_k = \frac{1}{N_m} \sum_{x_i \in leaf_m} 1 \cdot \mathbf{I}[y_i = k] \tag{2.3}$$

where N_m is the number of instances inside leaf m, $p_k \in Y$, and $\mathbf{I}$ is the identity function which results in 1 if the argument is true, else 0. The predicted class label of an instance is based on the class which dominates inside the leaf.

2.1.2 Naïve Bayes

Naïve Bayes [Mar61] is a probabilistic classifier which relies on Bayes theorem; however, it uses a rather naive assumption that the features of an instance, within a given class, are conditionally independent of each other w.r.t the class labels. Naïve Bayes classifier is particularly popular since it can be constructed very efficiently, it is highly scalable and it produces probabilistic predictions.

Bayes theorem can be described as:

$$P(Y|A_1, ..., A_n) = \frac{P(A_1, ..., A_m|Y)P(Y)}{P(A_1, ..., A_m)} \tag{2.4}$$

However the assumption of Naive Bayes assumes that the features are conditionally independent w.r.t the class label; therefore, it can be reformulated as:

$$P(Y|A_1, ..., A_m) = \frac{P(A_1|Y)...P(A_m|Y)P(Y)}{P(A_1, ..., A_m)}$$
$$P(Y|A_1, ..., A_m) \propto P(Y)\Pi_{i=1}^{m} P(A_i|Y) \tag{2.5}$$

Naïve Bayes classifier performs optimally when the assumption of independence holds. On the other hand, if the assumption of independence does not hold, which may happen in many real-world problems, then the method under-performs.

2.1.3 Ensembles

Ensemble learning combines multiple classifiers (also called weak learners, whose performance is barely better than random guessing) of the same type (homogeneous ensemble) or different types (heterogeneous ensemble) to derive a better hypothesis. Bootstrap aggregating [Bre96] (Bagging) and Adaptive Boosting [Sch99] are two very well-known methods in this category.

Bootstrap aggregating

Bootstrap aggregating (Bagging) ensemble [Bre96] is a method which splits the data into multiple different versions (bags), each of which is used to train a weak learner and

afterwards combines these weak learners for an aggregated prediction. The ensemble combines the weak learners by using either majority vote, weighted majority vote, or averaging the estimated probabilities. For the splitting part, the ensemble generates a set of M bags, given that the original training set of size n, each of which has the same size n' ($n' \leq n$). Data from the original training set are sampled with replacement and uniformly. Bagging method is illustrated in Figure 2.2.

Bagging's main advantage is that it reduces the variance; thus, prevents the final model from overfitting. On the other hand, due to the combination of different and independent weak learners, the model is not easy to interpret.

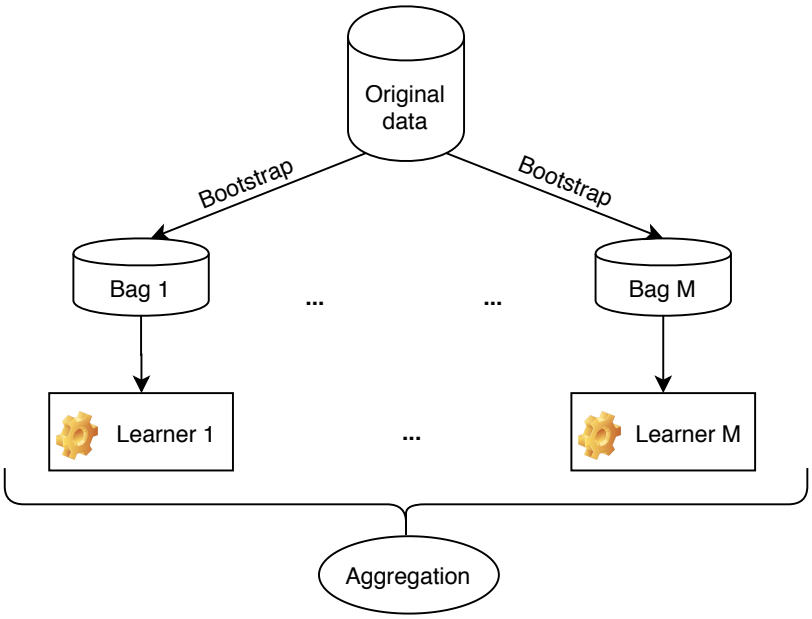


Figure 2.2: Bootstrap aggregating method

Adaptive Boosting

Adaptive Boosting (AdaBoost) [Sch99], is an ensemble of weak learners, where each weak learner is trained upon the errors of its predecessor (Figure 2.3). The idea of AdaBoost is to focus more on the hard-to-learn instances which are misclassified more often than others, by assigning higher weights to them. After each iteration, misclassified instances receive higher weights so that on the next iteration (boosting round), the weak learner will be forced to focus on the hard-to-learn instances. This iterative learning process will result in a strong learner which will use a weighted majority vote for prediction tasks. The pseudo-code of AdaBoost is shown in Alg. 1.

More analytically, the error rate of weak learner h_t in the t^{th} boosting round, is

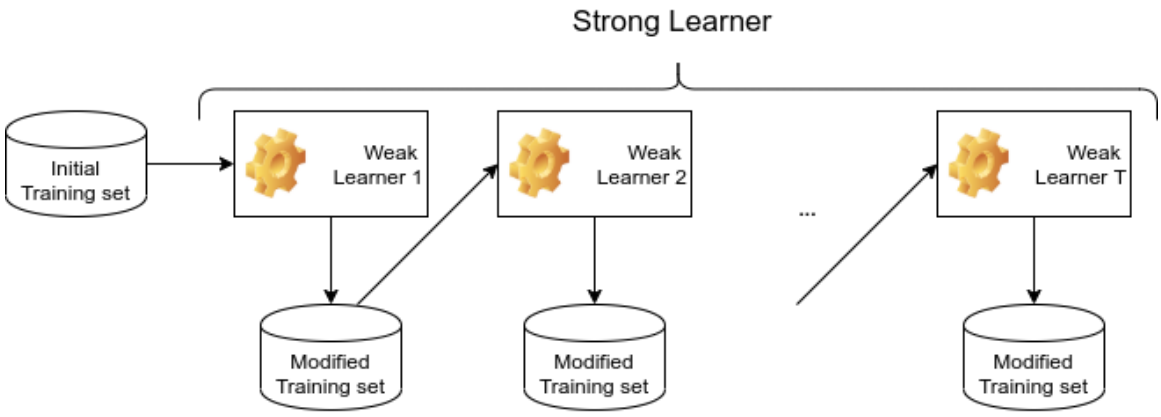


Figure 2.3: Boosting method

measured as follows:

$$err_t = \frac{1}{N}\sum_{i=1}^{N} I[y_i \neq h_t(x_i)] \tag{2.6}$$

where I is the identity function which returns 1 if the condition within is true, otherwise 0. The error rate of each weak learner can be better or worse but never equal to random guessing, i.e., 50% error rate. For the re-weighting, a special parameter α_t is used:

$$\alpha_t = \frac{1}{2} ln\left(\frac{1 - err_t}{err_t}\right) \tag{2.7}$$

and the new instance weights for the next round $t + 1$, are calculated as:

$$w_i^{t+1} = w_i^t \exp(\alpha_t y_i h_t(x_i)) \tag{2.8}$$

In the extreme case that $err = 50\%$ then $\alpha = 0$, which means that the data distributions will not change in the following boosting rounds.

AdaBoost improves the overall accuracy by combining many weak learners that produce a strong learner. It has been proved based on the margin theory that AdaBoost does not overfit [SS99]. However, AdaBoost is sensitive to noise and outliers, since it tries to fit all the data.

2.2 Clustering Methods

In this section, we introduce the employed clustering methods which will be used in this work.

Clustering is the task of separating instances into groups based on their similarity. Clustering methods are used in a wide variety of areas e.g., marketing, fraud detection, bio-medical, and so on. Clustering methods can be classified into different

Algorithm 1: AdaBoost

Input: $D = (x_i, y_i)_1^N, T$
Output: Ensemble H

1. Initialize $w_i = 1/N$ for $i = 1, 2, \ldots, N$
2. For $j = 1$ to T:
   (a) Train a classifier h_j to the training data using weights w_i.
   (b) Compute the error rate $\text{err}_j = \frac{\sum_{i=1}^{N} I(y_i \neq h_j(x_i))}{\sum_{i=1}^{N} w_i}$
   (c) Compute the weight $\alpha_j = \frac{1}{2} \cdot \ln(\frac{1-\text{err}_j}{\text{err}_j})$
   (d) Update the distribution as
   $w_i^{j+1} \leftarrow \frac{w_i^j \cdot \exp(-\alpha_j y_i h_j(x_i))}{Z_j}$ // Z_j is normalization factor
3. Output $H(x) = \sum_{j=1}^{T} \alpha_i h_j(x)$

categories: i) density-based methods, ii) hierarchical based methods, iii) distribution-based methods, iv) grid-based methods, and v) centroid based methods.

In this work, we employ k-Means (centroid-based) and Expectation-Maximization (distribution-based) methods that are described below.

2.2.1 k-Means

K-Means [M+67] is a very popular centroid-based clustering method. The algorithm tries to partition the data by iteratively minimizing the sum of squared distances of the instances inside a cluster.

K-means takes in the parameter k which defines the number of clusters to be estimated. It begins by assigning random points to the centroids of each cluster (Figure 2.4). Afterwards, it assigns instances to the clusters based on a distance function e.g., euclidean, manhattan, etc. The sum of squared distance among the instances and the centroids of the clusters is minimized in each iteration. The less variation a cluster has, the more homogeneous are the instances inside that cluster. More formally, k-Means' goal is:

$$\arg\min \sum_{i=1}^{k} \sum_{x \in K_i} ||x - \mu_i||^2 \tag{2.9}$$

where μ_i is the mean of instances x in cluster K_i.

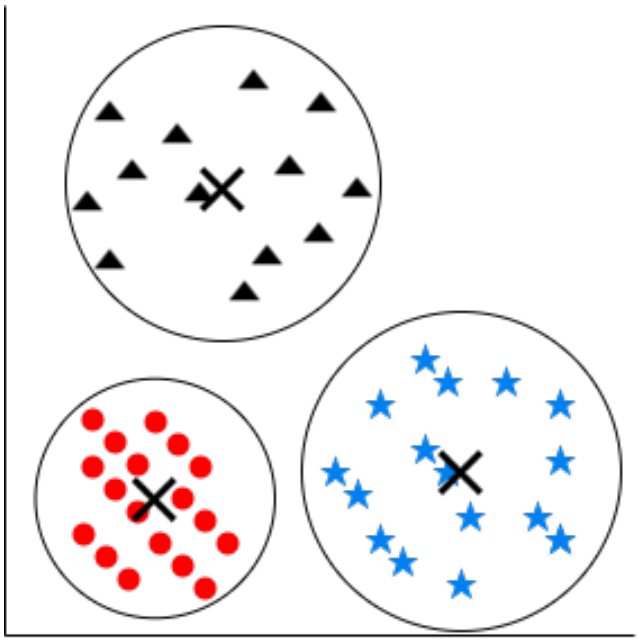


Figure 2.4: K-means example, where $k = 3$ (Xs symbolize the centroids)

K-means is a well-known clustering method, since it is a simple, efficient, and easy to interpret; however, it comes with some limitations. It is sensitive to outliers and k is a user parameter which has to be selected carefully. There are methods that can help with parameter selection, such as *elbow*, *silhouette*, and *gap statistic* methods.

2.2.2 Expectation-Maximization

Expectation-Maximization (EM) [DLR77] is similar to K-means. EM assigns instances to clusters by computing the probabilities of the instances to belong to a cluster, based on probability distributions (Figure 2.5). K-means' objective is to minimize the sum of square distances, while EM's goal is to maximize the likelihood of the data.

EM has two basic operations: i) expectation step (E) in which it generates an expectation of the log-likelihood of the data which is evaluated by the current estimate of the parameters, and ii) maximization step (M) in which the parameters that maximize the expected log-likelihood are calculated. These two steps are repeated until the log-likelihood of the data has converged, or a maximum number of iterations has been reached. More formally, the objective of EM is to maximize data likelihood as follows:

$$p(X|\Theta) = \sum_{z} p(X, Z|\Theta) \tag{2.10}$$

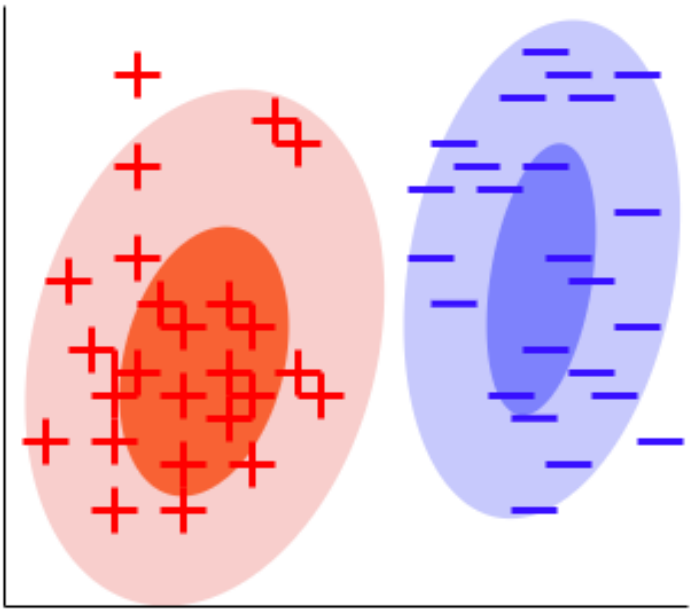


Figure 2.5: Expectation maximization method

where Θ are the unknown parameters which are calculated in each iteration, and Z is a set of unobserved latent variables.

EM is a powerful clustering method. One of its main advantages is that the data likelihood is guaranteed to increase in each iteration. However, it can fall into the pitfall of a local maximum. To avoid that, EM can be initialized multiple times with different initial points. Finally, EM may converge slowly.

2.3 Semi-Supervised Learning

In this section, we introduce the task of semi-supervised learning as well as the employed methods in this work, namely *Co-Training* and *Self-Learning*.

Nowadays, data are generated at tremendous speeds; however, these data suffer from label scarcity. In order to obtain labels, human annotation process has to take place; however, the human annotation process is expensive and slow, thus it is prohibitively expensive. Semi-supervised learning aims to solve this issue by combining a few labeled data with large amounts of unlabeled data during the learning process. Semi-supervised methods assume that instances which are in the same neighborhood (close to each other) probably have the same class label. Such methods can increase classification performance by exploiting the structure of unlabeled data and incorporate them into the classification process. By propagating class labels to the unlabeled data, semi-supervised methods improve their hypotheses.

Semi-supervised methods can be separated into three different categories: i) generative models, ii)) low-density separation, and iii) graph-based models. We briefly review works in each of these categories below.

Generative models: These methods try to estimate the data distribution of points that correspond to each class i.e., $P(X|Y)$. They assume that the data distributions can be expressed as $P(X|Y,\theta)$, where θ is a vector of parameters which they try to estimate. If this assumption holds, the unlabeled data can improve the performance [RV95]. To estimate θ, the models try to maximize the log-likelihood of the labeled and unlabeled data:

$$\arg\max_{\theta} \left(\sum_{i} log(P(x_i|y_i,\theta)P(y_i|\theta)) + \sum_{i} \sum_{y} P(y|x'_i,\theta) log(P(x'_i,y|\theta)P(y|\theta)) \right) \tag{2.11}$$

Popular works in this category are [NMTM00] in which authors used expectation-maximization combined with Naïve Bayes to classify textual data, [MLH04] that used two classifiers (Self-Learning) to classify textual data with binary labels, [BM98] that employed two models (Co-Training) to classify website domains and so on.

Low-density separation: Low-density separation methods try to set boundaries in regions using limited amounts of data. A very popular method is the transductive SVM (TSVM), which tries to maximize the margin over all data by labeling the unlabeled data. For the maximization of the margin on both labeled and unlabeled points, TSVM minimizes the function below:

$$\min_{w,b} w^2 + c \sum L(y_i(wx_i + b)) + C' \sum L'(wx'_i + b) \tag{2.12}$$

where w is the regularizer, L and L' are the loss functions for labeled and unlabeled data. TSVM was introduced in [Vap99] and has been investigated and extended by a plethora of works e.g., [Joa99, BD99, DB01, FM01, CZ05], and so on.

Graph-based methods: These type of methods are based on graph representations of the data, where each node depicts a labeled or unlabeled data point. The graph can be generated using instance similarity or domain knowledge. A very common method to generate the graph is the nearest neighbor weighted graph, which aims to minimize the function below:

$$\sum_{i} \sum_{j} W_{i,j} (f(x_i) - f(x_j))^2 \tag{2.13}$$

Popular works in this category are [ZCP05] in which authors constructed graphs by tweaking or removing edges, [BLRR04] where authors used random noise to the weights of the edges for regularization to perturbate the graph, [PL04] in which they used the *mincut* method to improved textual classification under the assumption that instances close to each other have the same class and so on.

2.3.1 Self-Learning

Self-Learning [Fra67] is the earliest semi-supervised method. It works iteratively by labeling the unlabeled data which uses afterwards to enhance its confidence (Figure 2.6). This method usually employs a threshold that is used as a confidence level for the unlabeled predictions. In the beginning, a supervised classifier is trained upon the small training set. Afterwards, the classifier is applied to the unlabeled data to generate labels. Predictions which exceed the threshold are selected to expand the training set, and this process is repeated until all the unlabeled data are labeled, or a number of iterations is reached.

Self-Learning is a simple and efficient method to leverage the unlabeled data; however, it has a serious flaw. If the unlabeled data contain noise or outliers, the method will learn and propagate errors to the upcoming iterations, which can corrupt the overall predictions.

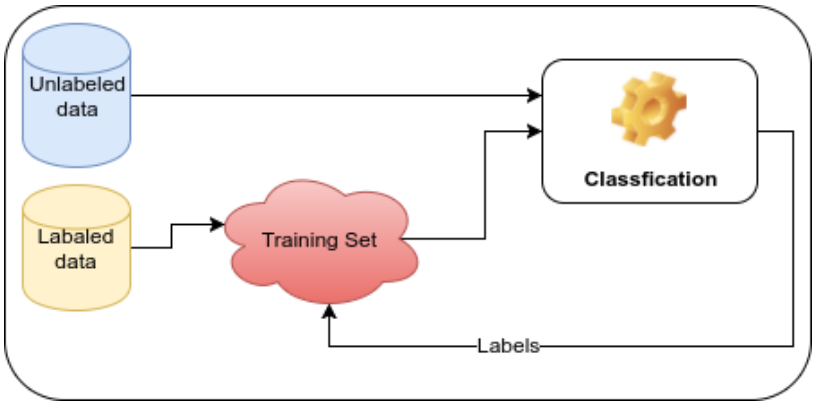


Figure 2.6: Self-Learning method

2.3.2 Co-Training

Co-Training [BM98] has been proposed to tackle the problem of error propagation that takes place in Self-Learning. This method relies on the assumption that the data can be described by two different and conditionally independent sets of features (also called views).

Based on these two different views, this method trains a supervised classifier on each view, and uses them to train each other (Figure 2.7). Iteratively, Co-Training trains these two classifiers on the different views, and the most confident predictions of the unlabeled data will be used to train the opposite classifier. Hence, each classifier will not amplify its own errors; rather, each classifier trains the other classifier. In order for this method to work efficiently, the two views have to describe the data adequately.

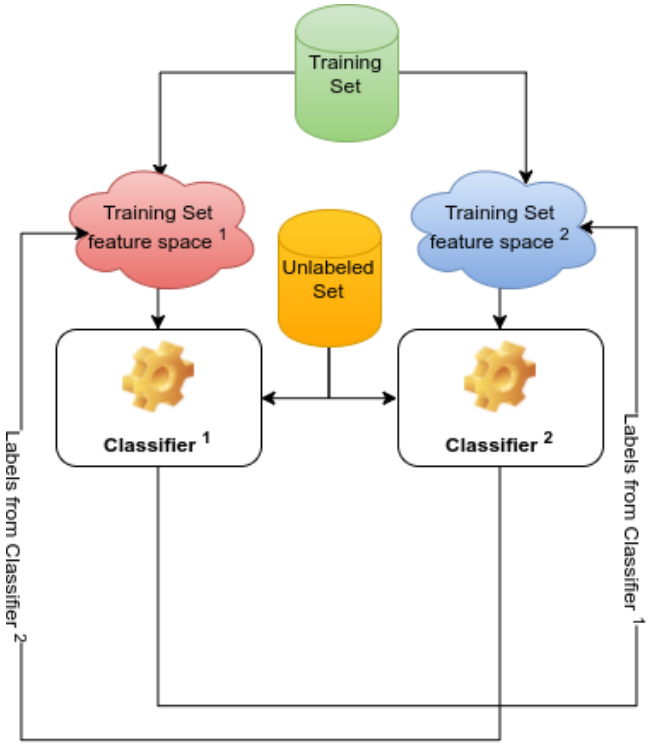


Figure 2.7: Co-Training method

2.4 Class-Imbalance

In this section, we introduce the problem of class imbalance, and categorize the related literature.

The problem of class imbalance is quite challenging and affects a wide variety of fields which rely on machine learning tasks. Class imbalance [HM13] refers to the skewed label distributions within data (between-classes), where one class, called majority class, dominates the other classes, called minority classes, w.r.t the amount of instances. For ease of simplicity, we assume the binary classification setting. The class-imbalance problem affects classification models by making them learn effectively one class (majority) while under-perform on the other class (minority) e.g., given a

dataset that contains 95 negative (majority class) and 5 positive (minority class) instances, a classifier predicts all the instances as part of the majority class, and achieves 95% accuracy; however, it is not able to identify not even one positive instance.

There exists also another type of class-imbalance, called within-class imbalance, which refers to the existence of rare cases. Rare cases can correspond to sub-concepts, which can reside in any class (majority or minority) e.g., in a medical dataset that contains healthy and ill patients, an ill patient may suffer from a rare disease. Rare cases are hard to learn, while they correspond to sub-concepts, which are far less than the common concept of a class. An example of rare cases can be seen in Figure 1.1.

According to [HM13], literature can be separated into three broad categories: i) data level methods, ii) model-based methods, and iii) cost-sensitive methods. For a detailed analysis of each category, the reader is encouraged to see [HM13].

2.4.1 Data level methods

Data augmentation methods operate on the data level i.e., they modify the data distribution before the training phase of a model; thus, making these methods useful for skew-insensitive classifiers. In [JS02], authors investigate the problem of class-imbalance and the impact of re-sampling methods under the inter-dependencies of i) class distribution skewness ii) data' complexities iii) data' volume and iv) classification models. In [LL98], authors propose a combination of standard under-sampling and over-sampling to equalize class distributions, and measure the model's performance using lift analysis. In [IN19b], authors investigate a variety of augmentation methods, for textual data, such as *distortion* and *semantic-similarity* to augment minority class. The distortion method, in this scenario, refers to adding noise to the text by removing terms (words), and the semantic-similarity method refers to swapping terms with terms which are closer in a vector space projection. In [DH+03], authors examine the impact of over-sampling and under-sampling under the cost curve performance metrics and come to the conclusion that under-sampling is significantly more effective than over-sampling in case of decision tree classifiers. In [CBHK02], authors proposed *SMOTE* method, which augments the minority class by taking into consideration the instances' neighborhood, using the k-NN algorithm. SMOTE first locates neighborhoods of minority instances, and then it generates pseudo-instances by combining the attribute values of the neighborhood; thus, pseudo-instances are similar to the instances in the neighborhood.

A different class of approaches which operates on the data level are the generative adversarial networks (GANs). GANs use two neural networks, the *generative network* and the *discriminative network*. The generative network generates data by learning a mapping function from a latent space to the target data distribution and the discriminative network evaluates the generated data w.r.t the original data. A large body of work has been proposed over the recent years for applications like image classification [FAKA+18, Ino18, STR+18], speech recognition [KPP+17, CGK15, SG15],

etc.

2.4.2 Model-based methods

Such methods aim to tackle class-imbalance during training, by employing a mechanism which aims to identify rare patterns. In [CLHB03], authors introduce *SMOTEBoost* which combines SMOTE and AdaBoost [Sch99] to deal with class-imbalance by augmenting the minority class instances which are misclassified in each boosting round. In a similar line of work [SKVHN09], authors present *RUSBoost* which combines AdaBoost and random under-sampling of the majority class instances which are correctly classified in each boosting round. In [GV04], authors introduce *DataBoost-IM*, which locates the hard-to-learn examples from both positive and negative classes during the training phase of AdaBoost, and based on these instances it generates synthetic data at the end of each boosting round. In [Qui91], authors tweak the decision boundary of a decision tree by adjusting the probabilistic estimates of the tree's leaves, while in [ZE01], they propose a class-imbalance sensitive pruning method for decision trees. In [WC03], authors tweak the decision boundary of an SVM based on kernel-alignment ideal. Finally, a class posterior re-balancing framework is proposed in [KSXPK17] to reduce imbalance while retaining classification certainty.

2.4.3 Cost-sensitive methods

These methods do not optimize for standard accuracy, rather try to minimize the overall misclassification costs which have been assigned beforehand. For the cost estimation, a grid search has to be performed w.r.t a given metric such as gmean, f1-score, balanced accuracy, etc. The misclassification cost of minority class is higher than that of the majority class. This category of algorithms is divided into three sub-categories [SKWW07]: i) weighting the data space, ii) making a specific classifier cost-sensitive, and iii) using the Bayes risk theory to assign each instance to a class with the lowest risk. The first sub-category of algorithms aim to alter the data distributions w.r.t a misclassification cost matrix so that the data distribution is biased towards the minority class. The very first method in this line of work is *AdaCost* [FSZC99]. Afterwards many other variations of AdaCost have been introduced over the years such as *CSB1* [Tin00], *CSB2* [Tin00], *RareBoost* [JKA01], *AdaC1* [SKWW07], *AdaC2* [SKWW07], and *AdaC3* [SKWW07]. These methods differ in three main parts: i) error estimation, ii) instance weight assignment, and iii) decision voting schema. An overview of these methods can be seen in Table 2.1. Except for the RareBoost, all the aforementioned methods in this category require a user parameter for the misclassification costs. The second sub-category of algorithms aims to make a specific classifier cost-sensitive. In [NEK+16], authors propose *AdaMEC*, a boosting classifier that uses the misclassification costs only to set thresholds to the decision boundary of AdaBoost, in contrast to the previous methods which use

Method	Initial D^1	D^{t+1} update	a_{t+1} update	Decision H(x)
AdaBoost [Sch99]	$1/N$	$D^t \exp(-a_t h_t y_i)$	$\frac{1}{2}\log\frac{\sum_{i,h_t(x_i)=y_i} D^t(i)}{\sum_{i,h_t(x_i)\neq y_i} D^t(i)}$	$sign(\sum_t a_t h_t(x))$
AdaMEC [NEK+16]	"	"	"	$sign(\sum_{y\in\{-,+\}} c(y) \sum_{t:h_t=y} a_t h_t(x))$
CSB1 [Tin00]	$\frac{C_i}{\sum_i^N C_i}$	$D^t C_i \exp(-h_t y_i)$	"	"
CSB2 [Tin00]	"	$D^t C_i \exp(-a_t h_t y_i)$	"	"
AdaCost [FSZC99]	"	$D^t \exp(-a_t h_t y_i \beta_i)$	$\frac{1}{2}\log\frac{1+\sum_i D^t(i)\exp(-a_t h_t y_i \beta_i)}{1-\sum_i D^t(i)\exp(-a_t h_t y_i \beta_i)}$	"
AdaC1 [SKWW07]	"	$D^t \exp(-C_i a_t h_t y_i)$	$\frac{1}{2}\log\frac{1+\sum_{i,y_i=h_t(x_i)} C_i D^t(i) - \sum_{i,y_i\neq h_t(x_i)} C_i D^t(i)}{1-\sum_{i,y_i=h_t(x_i)} C_i D^t(i) + \sum_{i,y_i\neq h_t(x_i)} C_i D^t(i)}$	"
AdaC2 [SKWW07]	"	$D^t C_i \exp(-a_t h_t y_i)$	$\frac{1}{2}\log\frac{\sum_{i,h_t(x_i)=y_i} C_i D^t(i)}{\sum_{i,h_t(x_i)\neq y_i} C_i D^t(i)}$	"
AdaC3 [SKWW07]	"	$D^t C_i \exp(-C_i a_t h_t y_i)$	$\frac{1}{2}\log\frac{\sum_i C_i D^t(i) + \sum_{i,h_t(x_i)=y_i} C_i^2 D^t(i) - \sum_{i,h_t(x_i)\neq y_i} C_i^2 D^t(i)}{\sum_i C_i D^t(i) - \sum_{i,h_t(x_i)=y_i} C_i^2 D^t(i) + \sum_{i,h_t(x_i)\neq y_i} C_i^2 D^t(i)}$	"
RareBoost [JKA01]	$1/N$	$D^t \exp(-a_t^{p,n} h_t y_i)$	$a_t^p = \frac{1}{2}\ln\frac{TP_t}{FP_t}$, $a_t^n = \frac{1}{2}\ln\frac{TN_t}{FN_t}$	$sign(\sum_{t:h_t(x)\geq 0} a_t^p h_t(x) + \sum_{t:h_t(x)<0} a_t^n h_t(x))$

Table 2.1: Cost Sensitive Boosting Variations

the misclassification costs to change the data distribution in each boosting round. In [QWZZ13], authors introduce a cost-sensitive k-NN classifier, which aims to tackle class-imbalance by using a modified distance function which takes into consideration the misclassification cost matrix. In [LYWZ04], authors use the misclassification cost matrix to define a cost-sensitive splitting criterion in decision trees, while in [BKK+98] they take into account the misclassification costs to determine the pruning criterion of a decision tree. The third sub-category uses the Bayes risk theory to assign each instance to a class with the lowest risk. Few works have been proposed in this direction e.g., in [Dom] the authors swap the class labels of the leaves to minimize the misclassification cost.

2.5 Fairness-Aware Supervised Machine Learning

In this section, we introduce the task of fairness-aware learning and also the most frequently used fairness notions.

We begin by giving the definition of discrimination. Discrimination [Wik] is treatment or consideration of, or making a distinction towards, a person based on a protected attribute to which the person is perceived to belong. Protected attributes considered by the law include: age, disability, race, religion, sex, sexual orientation, etc.

In recent years, it has been observed that machine learning models produce discriminatory outcomes towards individuals or groups of people who share specific characteristics. The task of fairness-aware supervised machine learning extends the supervised learning task by not only considering to find a map function $f(\cdot) \rightarrow y$ that minimizes the error rate, but also mitigates discriminatory outcomes.

Same as in Section 2.1 we consider binary classification with $Y = \{+, -\}$ and $A = \{A_0, ..., A_m\}$ the feature space. We also assume a set of historical data $X = ((x_1, y_1), (x_2, y_2)...(x_n, y_n))$, that consist of n i.i.d examples drawn from a joint distri-

bution $P(A, Y)$. Let $S \in A$ be a binary protected feature $S = \{\bar{z}, z\}$, where $\bar{z}$ is the non-protected group and z is the protected group e.g., $S = gender$, $\bar{z} = male$, and $z = female$. We refer to the positive class as the target class e.g., loan approval. We want to find a map function $f(X) \rightarrow y$, as well as mitigate unfair outcomes at the same time.

A plethora of fairness notions has been proposed to measure discriminatory outcomes. In this work, we present the most frequently used notions of fairness (the interested reader is referred to [RR14, VR18] for a detailed description of the proposed fairness notions).

One of the earliest fairness notions is the **S**tatistical **P**arity (also called demographic parity) [KC09]. It can be formalized as:

$$SP = P(+|\bar{z}) - P(+|z) \tag{2.14}$$

Statistical parity measures the difference between the percentages of non-protected and protected group w.r.t the target class. However, this notion may allow instances from the protected group that may not be qualified to be labeled as positive and reject instances from the non-protected group that may be qualified in order to minimize this difference [DHP+12]. Another fairness notion, called **Eq**ual **Op**portunity [HPS+16], has been introduced to overcome this issue, described as follows:

$$Eq.Op. = P(+|\bar{z}, +) - P(+|z, +) \tag{2.15}$$

Equal opportunity tries to balance the classified difference of the non-protected and protected groups w.r.t the target class. Both statistical parity and equal opportunity focus solely on the target (positive) class. Another fairness notion called **E**qualized **O**dds (or disparate mistreatment) [HPS+16, ZVGRG17] extends equal opportunity by also forcing equal classification percentages w.r.t the negative class:

$$\begin{aligned} \delta TPR &= P(+|\bar{z}, +) - P(+|z, +) \\ \delta TNR &= P(-|\bar{z}, -) - P(+|z, -) \end{aligned} \tag{2.16}$$

During the last decade, a variety of fairness-aware supervised and unsupervised methods have been proposed. These methods can be separated into four main categories: i) pre-processing, ii) in-processing, iii) post-processing, and iv) fair representations. We describe each of these categories below.

Pre-processing methods. Methods in this category [RPT10, LRT11, CWV+17, KC09, CKP09, IN18, KC12], work under the assumption that in order to learn a fair classifier, the training data should be discrimination-free. To this end, they try to i) balance the representation of the different segments in the population, and/or ii) remove encoded societal biases which may exist inside the data. Population segments re-

fer to protected positive/negative and non-protected positive/negative groups. Methods that are balancing the population segments force the model to learn all segments effectively while methods which remove the encoded biases try to eliminate existing discriminatory patterns which occur within a dataset, thus making the model learn *non-discriminatory* patterns. Among the most popular pre-processing methods are class-label swapping (called massaging), instance re-weighting, and preferential sampling that are employed to achieve statistical parity [KC12]. Massaging locates instances which reside near the decision boundary of a classifier, and swaps their class label in order to shift the decision boundary towards a fairer boundary. Re-weighting, on the other hand, generates a fair weight distribution associated to the instances in order to give more weight to population segments which are under-represented. Finally, preferential sampling performs under-sampling and over-sampling to the instances in different population segments based on their distance to the decision boundary of a classifier. Similarly, [IN18] propose augmenting the protected group via semi-synthetically generated instances which reside near the decision boundary of a classifier. In [LRT11], authors propose a k-NN approach which locates and removes instances which have been treated significantly different among their neighborhood instances. Finally, the probabilistic framework of [CWV+17] modifies the instance values such that the instance label is less dependent on the protected attribute(s).

In-processing methods. In-processing methods modify the learning algorithm to eliminate discriminatory behavior. These interventions are typically learner-specific. For instance, [ZVGRG17] add fairness-related constraints in the objective function of a logistic regression model to account for fairness. Their fairness measure aims at minimizing the group differences w.r.t. FPR (false positive rate), FNR (false negative rate), or both. In [KXPK18], authors consider the problem of biased class labels, and assume the existence of an unbiased latent distribution. To learn its parameters, they propose an iterative training approach that re-adjusts instance weights in order to minimize discrimination. In [KCP10], authors propose a discrimination-aware decision tree model by tweaking the splitting attribute criterion to consider not only information gain w.r.t the class attribute but also heterogeneity w.r.t sensitive attribute. The model is optimized for statistical parity w.r.t. ground truth labels. In [DIKL18], authors include sensitive attributes in the learning process by utilizing a joint loss function that makes an explicit trade-off between fairness and accuracy.

Post-processing fairness-aware methods Post-processing methods either modify the results of a trained classifier to ensure the chosen fairness criterion is met, or alter the decision boundary of a classifier to mitigate unfair outcomes. In [KCP10], authors investigate how to force a decision tree to make fair predictions by modifying the leaf labels of the tree to opt for statistical parity. In [PRT09a], authors change the confidence values of CPAR classification rules, while in [CV10] authors alter the probabilities in Naïve Bayes models to account for fairness. In [FKL16] authors inves-

tigate fair outcomes in ensemble classifiers such as boosting by shifting the decision boundary of a trained AdaBoost learner until the statistical parity fairness criterion is fulfilled. In [HPS+16], authors set thresholds to predicted outcomes of a classifier in order to achieve same error rates for protected and non-protected group. An extension of this work [PRW+17], analyzes how to obtain calibrated classifiers under the same error rates among groups. Finally, in [ABD+18] they build a fair classifier out of the predictions of another black-box classifier.

Fair representation methods. Fair representation methods try to approximate a transformation function, which transforms the data into a lower *fair dimensional space*. This fair dimensional space may satisfy one or more of the following constrains: i) the correlation between the protected attribute and the non-protected attributes is minimized, ii) the correlation of the protected attribute and the class labels is minimized, and iii) the reconstruction error between the protected and non-protected group is minimized. In [STM+18], authors extend standard PCA and propose the Fair-PCA, which forces the reconstruction error among protected and non-protected groups to be similar. In [LSL+16], authors propose non-linear fair transformations through a variational autoencoder combined with maximum mean discrepancy regularizer to mitigate dependencies between sensitive attributes and latent representations. A holistic approach is presented in [ES16], where the authors combine the fair representations of an adversarial neural network, formulated as a minimax problem, with a neural network to mitigate unfair outcomes. The purpose of the adversarial neural network is to eliminate the dependencies of sensitive attributes to the latent space while the classifier is trained to mitigate unfair outcomes w.r.t statistical parity. These two parts are trained together under a combined loss function. An extension of this work is proposed in [MCPZ18], where the authors obtain fair representations by combining an autoencoder with an adversary, while training a neural network to be fair under a joint loss function, for various fairness notions such as statistical parity, equal opportunity, and equalized odds.

3

Tackling Class-Imbalance in Semi-Supervised Learning

In this chapter, we investigate the problem of class-imbalance in semi-supervised learning. Semi-supervised methods are essential when data suffer from label scarcity since standard supervised learning methods won't work upon such sort of data due to lack of labels and the impracticality of (human) labeling (for large scale datasets).

For the purpose of this study, we focus on a popular task, namely *sentiment analysis*, that aims to gain insights into opinionated textual data. More analytically, we employ textual data from Twitter since huge amounts of unlabeled textual data are generated on a daily basis. We present the insights from our annotation process regarding the effect of different semi-supervised learning approaches, namely Self-Learning, Co-Training, and Expectation-Maximization (EM).

Although semi-supervised methods are the most appropriate ones to handle data with limited labels, as described in this chapter, they are prone to propagate class-imbalanced predictions. To tackle this aggravation by the semi-supervised learning methods, we combine data augmentation such as *sampling methods*, *distortion*, and *semantic-similarity* in the semi-supervised learning process in order to equalize the class distribution. Our results show that semi-supervised learning, coupled with data augmentation, significantly outperforms the default semi-supervised annotation process.

3.1 Introduction

A huge amount of opinions is generated on a daily basis in social media like Twitter and Facebook referring to essentially every entity - products, persons, brands, events or topics. Opinions are valuable for not only consumers, who benefit from the experiences of other consumers, in order to make better buying decisions but also for vendors, who can get insights on what customers like and dislike about their

products [LYAH13].

Such sort of data are freely available nowadays; however, due to their amount and complexity, a proper analysis is required in order to gain insights. The analysis of such sort of data are investigated in the areas of sentiment analysis and opinion mining [PL+08, SNZ16]. Sentiment analysis aims at characterizing the sentiment content of a text as either positive or negative (some approaches also consider the neutral class).

Traditionally, sentiment analysis is investigated in the fully-supervised learning setting, assuming that a fully labeled training set is available, e.g., [PT10, MGL09, PNS+10, PL05, PLV02, YZL09]. However, despite its volume, the amount of labeled data are limited and acquiring (human) labels for all instances at this scale is impractical. Therefore, standard supervised learning methods are not applicable and new methods are required that can exploit both (few) labeled and (huge) unlabeled data for learning a supervised model.

Semi-supervised learning addresses this problem by leveraging unlabeled data, together with the labeled ones, to learn a supervised model. In this work, we employ three well known semi-supervised learning approaches, Self-Learning, Co-Training and Expectation-Maximization (EM), in order to annotate a huge collection of tweets covering the whole year 2015. However, semi-supervised learning does not consider class imbalance, which is a very common problem in many applications [HM13]. In our learning setup, there is a strong imbalance towards the positive sentiment class. The problem of class imbalance in the original dataset is further aggravated via the semi-supervised learning process. To this end, we propose to integrate data augmentation techniques in the semi-supervised learning process to re-balance the classes, and we investigate different forms of data augmentation that are applicable to this domain.

We summarize our findings from the annotation process which cover a variety of interesting aspects, and we make our annotations available to the community in an attempt to provide more complex datasets with temporal characteristics that can facilitate further research in the areas of sentiment analysis and data stream mining, in general. Our contributions are summarized below:

- We label a big Twitter stream dataset with limited labels consisting of 228M tweets without retweets and 275M tweets with retweets; the collection spans the whole year 2015 and is therefore also appropriate for temporal analysis. We make our labels available to the community to facilitate the development of new methods for sentiment analysis and stream mining in general and for evaluation purposes.

- We extensively evaluate the performance of different semi-supervised learning approaches, namely, Self-Learning, Co-Training and EM, and how it is affected by the amount of labeled data, the amount of unlabeled data and the classifier's confidence threshold.

- We employ data-augmentation with semi-supervised learning in order to tackle the problem of class imbalance that exists in the original dataset, and it is further aggravated by the iterative semi-supervised learning approaches. We show that augmentation can help in tackling the class imbalance problem.
- We report on the impact of data redundancy (via retweets) in the performance of the different models. As retweets can be considered as a natural form of data augmentation, we also report on their impact on class imbalance.
- We process the data in two different modes: i) as a batch, where labeled and unlabeled data are available to the algorithm from the beginning and ii) as a stream, where both labeled and unlabeled data are gradually available to the algorithm as the stream progresses. We show that the latter approach with a sliding window of three months achieves a comparable to the batch approach accuracy while being more efficient.
- We provide a qualitative evaluation of our labeling process via crowd-sourcing.

This chapter is an extension of our previous work [IN17]. The major changes include i) including EM in the evaluation as an example of semi-supervised learning method with soft-assignments, ii) tackling the problem of class imbalance in the semi-supervised learning process via data augmentation, iii) qualitative evaluation of the derived labels via crowd-sourcing, and iv) comparison of our annotations to state-of-the-art sentiment annotation tools.

The rest of the chapter is organized as follows: Related work is presented in Section 3.2. In Section 3.3, we describe our dataset and how we derived the ground truth for learning. The semi-supervised learning approaches are described in Section 3.4. In Section 3.5, we describe the augmentation process to handle class imbalance. Our experiments for batch and stream annotation, crowd-source evaluation and comparison to state-of-the-art methods are described in Section 3.6. Finally, conclusions and outlook are presented in Section 3.7.

3.2 Related Work

Our related work comes from the areas of sentiment-analysis, semi-supervised learning and large scale annotations.

3.2.1 Sentiment analysis

Due to the abundance of opinionated texts, there is a lot of research on sentiment analysis regarding the effect of different learners, building domain-specific learners and transferring across different domains. For example, [PLV02] examines the effectiveness of machine learning algorithms such as Naive Bayes classifiers, Maximum

Entropy models and Support Vector Machines to the problem of sentiment classification. In a follow-up work [PL05], the authors investigate the rating-inference problem for which instead of classifying reviews as positive or negative they try to determine the score w.r.t a multi-point scale. [MGL09] propose a framework for domain specific sentiment classification that employs lexical information with a model trained upon a given corpus. A similar approach is proposed by [MCI+18] but for temporal collections. [YZL09] classify reviews from travel blogs using Naive Bayes and SVMs combined with a character based N-gram model. [PT10] employ weighting schemes from information retrieval such as *tf-idf* to improve the classification accuracy on sentiment classification tasks. [PNS+10] investigate sentiment classification across different domains by combining domain-specific words from different domains. Additionally, [HF19] propose an approach that determines phrasing bias, which detect the use of subjective language towards specific events or other entities in terms of words or phrases that are either one sided or inflammatory.

More recently, there is also a lot of work on sentiment analysis over temporal data and data streams. For example, [ION17] propose a classification framework for opinionated streams which adapts to concept drifts that are detected as vocabulary changes over a sliding window. [MSN18] consider the problem of sentiment classification under feature drifts, where different features might undergo different types of temporal drifts and propose an ensemble of different experts, each specialized to capture a particular trend. [UBM+18] propose learning entity-specific models, instead of a global model to facilitate detecting local changes that are not always reflecting in the global stream.

3.2.2 Semi-supervised learning

Semi-supervised learning addresses this problem by leveraging unlabeled data, together with the labeled data, to learn classification models. [NMTM00, NMM06] propose an algorithm for learning from labeled and unlabeled documents based on Expectation - Maximization and Multinomial Naive Bayes (MNB). The algorithm first trains a classifier using the available labeled data, and probabilistically labels the unlabeled ones. It then trains a new classifier using the labels of all documents. The process is repeated until convergence. This basic algorithm was improved by two extensions: by employing a weighting factor to modulate the contribution of unlabeled data and by using multiple mixture components per class, instead of a single one. [SSM11] propose a semi-supervised extension of MNB, called SFE, that uses the estimates of word probabilities obtained from unlabeled data and class conditional word probabilities learned from the labeled data, to learn the parameters of an MNB classifier. [LD13] introduced MNB-FM a method that extends MNB to leverage marginal probabilities of the words, computed over the unlabeled data. The marginal probabilities are used as constraints to improve the class conditional probability estimates for the positive and negative classes. [ZHY+16] proposed MNB-WSC, which

preserves reliable word estimates, as extracted from a sufficient amount of labeled data. We also use MNB as our base model. Despite its class-conditional feature independence assumption, MNB is known to perform moderately and in some cases, it has been reported that its performance for short texts is equal or superior to more complex models [WM12].

A comprehensive survey of semi-supervised learning approaches for Twitter is provided in this recent survey [SCH16]. Similarly to us, they found that Co-Training performed better with limited labels, whereas Self-Learning is the best choice when a significant amount of labeled tweets is available. In contrast to the existing small scale datasets they used for evaluation, we report on a huge collection covering the whole year 2015. [DN09] experimented with a large variety of algorithms for semi-supervised sentiment classification, including active learning, spectral clustering and ensemble learning. Cross-domain sentiment classification is proposed in [AG05], where the authors exploit a small number of labeled and a huge amount of unlabeled data using EM.

Self-Learning is categorized as a form of semi-supervised learning [Fra67]. The central idea behind Self-Learning is that we can expand the training set by using the most confident predictions of the classifier. Self-Learning might cause error propagation as the predictions of the classifier are then used for its training [HZ11, ZNS14]. To deal with these issues, Co-Training was introduced in [BM98] that combines two different views of the data in two classifiers, which are then used together for the expansion of the training set. The intuition behind this approach is that different classifiers will make different errors and therefore one classifier can learn from the other instead of just learning by itself as in Self-Learning. In [LWZL11] the authors study class imbalance for semi-supervised classification. They use under-sampling in order to generate multiple balanced training sets and during the iterations of the semi-supervised process they dynamically change the classifiers by varying the feature space. [XWDL15] use negations and antonyms to generate an opposite view of the training data; the original and the opposite view are exploited afterwards via Co-Training.

3.2.3 Large scale annotation

TSentiment [GBH09] is a dataset of 1.6 million tweets covering the period from April 6, 2009 to June 25, 2009, which are annotated as positive or negative through distant supervision. The training set consists of tweets with emoticons, which serve as noisy labels. To the best of our knowledge this is the largest Twitter dataset for sentiment analysis and is used extensively also in stream mining due its temporal aspects [BF10, WZNS15]. TweetsKB [FIND18] is another interesting dataset, which contains entity-related annotations such as co-entities, popularity and also includes sentiment annotations via SentiStrength tool [KCWF12]. The dataset can be exploited to analyze social media archives e.g., [FISN18]. Finally, HSpam14 [SS15] is

a dataset of 14 million tweets in English which are annotated with spam and ham (or, non-spam) labels. The annotation process consists of four steps: a heuristic-based selection of tweets that are more likely to be spam, a cluster-based manual annotation, a reliable ham tweet detection and finally, EM-based label prediction for the remaining unlabeled tweets. Our dataset covers a larger period, of one year and therefore is more appropriate for such tasks.

3.3 Dataset description

The dataset and the different preprocessing steps as well as their effect on the dataset are described in Section 3.3.1. In Section 3.3.2, we describe how we derive the training set, i.e., labeled instances for the classification task. In Section 3.3.3 we provide an exploratory analysis of the dataset with a focus on its temporal characteristics.

3.3.1 Data collection and preprocessing

The dataset has been collected[1] from the 2015 Twitter stream using its public streaming API[2], which provides a random selection of tweets (about 1% of all tweets). In total, 1.9 billion tweets were crawled, in all different languages (English, Japanese, Spanish, Greek, etc.). We selected only the English tweets which were not re-tweets (as flagged by the API); the filtered dataset consists of 384 millions tweets (20%), which generate 269 million distinct words.

We applied several preprocessing steps that are described below:

- *Slang words replacement*: We mapped slang words to normal expressions using a slang word dictionary[3]. For example, “lol" was mapped into “laughing out loud". This resulted in a slight increase of the words.
- *Links and mentions*: Links and mentions, e.g., “https://example.com", “@bbc", were removed.
- *Negation handling*: We consider negations on verbs and adjectives. For the former, we concatenated to a single verb, e.g., “don’t work" → “not_work". For the latter, we replaced the negation with its antonym, e.g., “not bad"→ “good", using WordNet list[4].
- *Special character removal*: We removed punctuation and numbers. We removed the symbol ‘#’ from hashtags and we treated them as normal words. We replaced

[1]The Twitter crawling collection project is part of the L3S research center initiative.
[2]https://dev.twitter.com/streaming/overview
[3]www.noslang.com
[4]https://wordnet.princeton.edu/

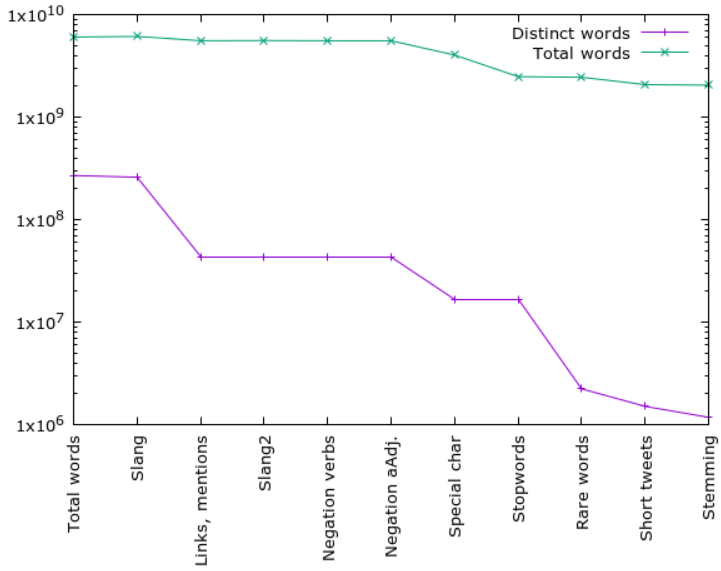


Figure 3.1: Preprocessing effects

repeated letters occurring more than two times in a row with two letters; e.g., “huuuungry" → “huungry".

- *Removal of emoticons*: We also removed the emoticons from the training data, aiming at classifiers that can learn from the word features. In general, we removed all non-ASCII characters most of which were special types of emoticons. But, we use the emoticons to derive the labels for the training set (c.f., Section 3.3.2).
- *Stopword removal*: We removed stopwords using Weka’s stopwords list [5].
- *Stemming*: We applied Porter stemmer.
- *Removal of rare words*: We removed rare words from the corpus, using a frequency of 10 as the cut-off value.
- *Removal of short tweets*: Finally, we removed tweets with less than four (< 4) words after the aforementioned steps, similarly to [TV14].

The pre-processing resulted in a reduction of the corpus and of the vocabulary (i.e., distinct words), c.f., Figure 3.1. In particular, the corpus was reduced by 41% (from 384M to 228M tweets) and the vocabulary was reduced by 99,5% (from 269M to 1,17M distinct words). Figure 3.2 shows the frequency distribution of the unique words in the corpus, a total of 1.6B words. As we see, the majority of unique words has less than 100 occurrences (almost 1M words). Around 150K unique words are

[5] http://weka.sourceforge.net/doc.dev/weka/core/stopwords/Rainbow.html

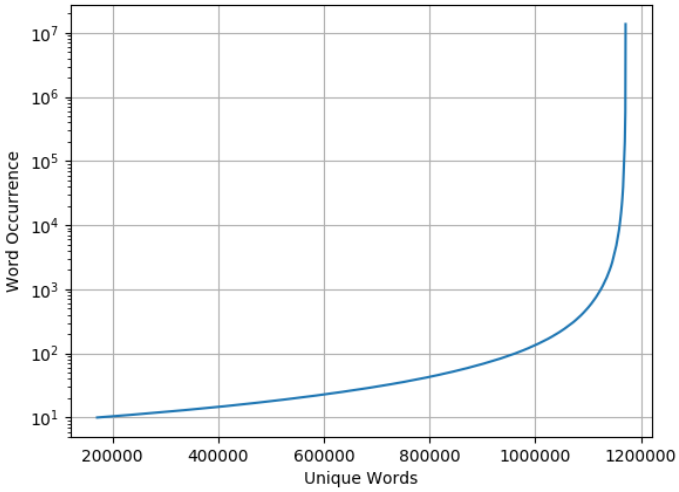


Figure 3.2: Frequency distribution of unique words

occurring from 100 to 1K times, while only 18K unique words are occurring more than 1K times.

3.3.2 Building the ground truth for learning

In the absence of human labels, we derive our ground truth for learning by combining two approaches/experts: i) an emoticon-based approach and a sentiment lexicon approach (using SentiWordNet[6]). Our idea is to consider as ground truth those tweets for which both experts agree in their labellings.

Deriving labels from emoticons Using a list of positive[7] and negative[8] emoticons we identify tweets with clear sentiment; those are tweets with only positive or only negative emoticons, which we then classify accordingly. This approach is similar to [GBH09].

In our 220M tweets dataset, only 10,1M (4.4%) contain emoticons. Out of 10.1M tweets with emoticons, 3,8M (37%) were classified as clear positive (those with only positive emoticons), 1,5M (15%) as clear negative (those with only negative emoticons), 4,8M (48%) as mixed cases (both positive and negative emoticons). Only tweets with clear emoticon-based sentiment, a total of 5.3M (=3.8M+1.5M) tweets, were used for building the ground truth.

[6]http://sentiwordnet.isti.cnr.it/

[7]Positive emoticons : c) =] :] :} ;> :>) :^) : D =) ;) :) 8) (: (; : o) : −) : P < 3 : 3 ^_^

[8]Negative emoticons : $-c$: [: { :< : −(: / : −[: c : − < : (:′ { >: [

	SWN. Pos.	SWN. Neg.	SentiW. Neutral
Emot. Pos.	**2,211,091**	840,787	807,887
Emot. Neg.	1,032,536	**316,662**	157,322

Table 3.1: Emoticon-vs SentiWordNet-based labellings. Agreements marked in boldface.

Deriving labels from SentiWordNet SentiWordNet [BES10] is a dictionary of words and their associated sentiment. The words might appear multiple times as different part of the speech (POS). Therefore, we first find the POS of each word in a tweet using Stanford's POS tagger [TKMS03] and considering the whole tweet to derive the context. Then, we calculate for each tweet its overall score by aggregating the scores of its component words from SentiWordNet; for the aggregation, each word is weighted with a harmonic series [GGT16]. A word's sense is associated to a score that is calculated by computing a weighted average of the differences between the positivity and negativity scores assigned to the various senses of the word. Same as in [BESS13], we employ Harmonic mean function (3.1) to aggregate the different senses of a given word since the senses are sorted according to frequency in descending order [GGT16].

$$H = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + ... + \frac{1}{x_n}} \tag{3.1}$$

Building the ground truth The results of juxtaposing the emoticons- and SentiWordNet-based labels are presented in Table 3.1. We considered as our ground truth the true positives, i.e., tweets where both emoticon-based and SentiWordNet-based labeling agree. SentiWordNet supports also neutral class as shown in Table 3.1, however due to the inability of emoticon-based approach to distinguish efficiently the neutral class, we employ only the positive and negative class tweets hereafter. Our final ground truth dataset consists of 2,527,753 tweets. From these roughly 2,5M tweets, 87.47% are positive (2,211,091 tweets) and the rest 12.52% (316,662 tweets) are negative.

An interesting observation from Table 3.1 is the disagreement between the two labeling experts: distant supervision using emoticons and SentiWordNet. A profound reason is the existence of the extra neutral class in case of SentiWordNet. However the sources of disagreement extend beyond this; in particular, SentiWordNet is a static database which has been generated upon WordNet Gloss Corpus[9], a corpus of manually annotated WordNet synset definitions. Such a database does not capture the large variability of words in a social stream (due to e.g., the creation of new medium-specific words like hashtags), in fact the coverage is pretty narrow [MCI+18]. On the other side, SentiWordNet contains high quality sentiment annotations comparing to

[9]http://wordnetcode.princeton.edu/glosstag.shtml

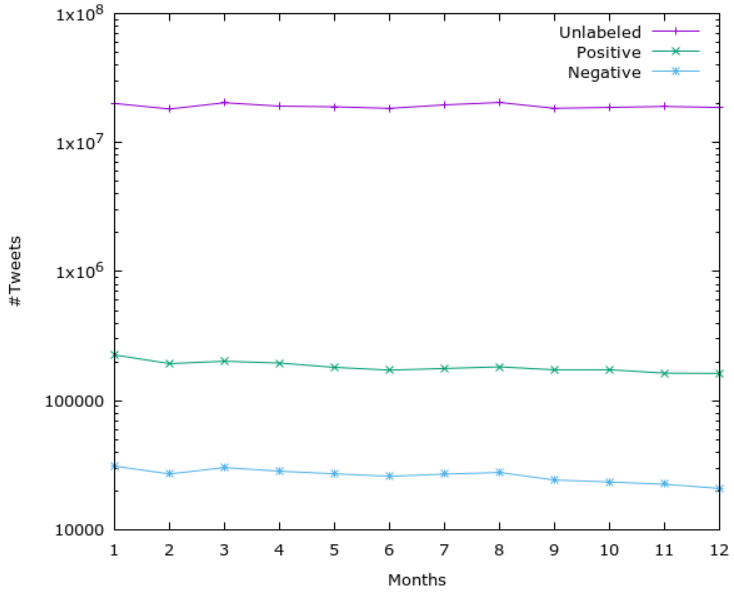


Figure 3.3: Dataset distribution on a monthly basis

the distant supervision approach that relies on emoticons as proxies for sentiment and therefore it is prone to errors.

3.3.3 Temporal characteristics

The temporal distribution of our dataset is depicted in Figure 3.3 including both ground-truth tweets (c.f., Section 3.3.2) and unlabeled ones. As we can see, the amount of unlabeled tweets is vast compared to the amount of labeled ones and this holds across the timeline. On monthly average, the unlabeled set is 82 times larger than the labeled set. For the labeled tweets, the negative class is miss-represented comparing to the positive class; the average ratio of positive to negative tweets per month is 3. Finally, there are no gaps in the monitoring period, i.e., we have both labeled and unlabeled tweets for each month.

3.3.4 Comparing ground truth to SentiStrength and TreeBank

For our derived ground truth, we also provide a comparison with state-of-the-art sentiment analysis methods such as SentiStrength [KCWF12] and Sentiment TreeBank [SPW+13].

SentiStrength [KCWF12] has been trained on social web data and can predict the sentiment (positive/negative) of short texts. In details, SentiStrength has been trained upon 2,600 human-classified comments from the social network website “mys-

		Ground truth	
		Positive	Negative
SentiStrength	Positive	**81.18** %	22.43 %
	Negative	1.39 %	**53.97** %
	Neutral	17.44 %	23.61 %

Table 3.2: SentiStrength vs Ground Truth labellings. Agreements marked in boldface.

		Ground truth	
		Positive	Negative
TreeBank	Positive	**28.26%**	5.69%
	Negative	45.70%	**75.48%**
	Neutral	26.04%	18.83%

Table 3.3: TreeBank vs Ground Truth labellings. Agreements marked in boldface.

pace.com", which were extracted in December 2008. This method assigns both positive and negative scores in the range +1 to +5 for the positive class and −1 to −5 for the negative class. For our comparison, we consider as neutral those tweets where both positive and negative scores are the same (in absolute values).

Sentiment TreeBank [SPW+13] has been trained upon 11,855 sentences which were extracted from movie reviews in 2005 from the "rottentomatoes.com" website [PL05]. TreeBank is able to classify sentences of arbitrary lengths into five classes: "negative", "somewhat negative", "neutral", "positive" and "somewhat positive". For our comparison, we merge "negative" and "somewhat negative" classes into the negative class as well as "positive" and "somewhat positive" classes into the positive class.

In Tables 3.2 and 3.3, we compare SentiStrength and TreeBank to our ground truth. SentiStrength has a higher agreement w.r.t the positive class (81.18%) - the agreement w.r.t. the negative class is much smaller (53.97%) though it still comprises the majority agreement class. For TreeBank, the situation is inverse; the agreement on the negative class is the strongest (75.48%), whereas the agreement in the positive class is much smaller (28,26%) and even worse; most of the positive ground truth was labeled as negative in TreeBank.

As we can see, both methods exhibit high disagreement w.r.t our ground truth labels. This is caused due to the different training data employed by the state-of-the-art methods, the different media/application domains from which the data are collected as well as the different spanning periods of each dataset. Moreover, we do not consider the neutral class, in contrast to TreeBank and SentiStrength.

3.4 Sentiment learning with limited labels

In our learning setup, we have a small number of labeled instances, denoted by L, and a huge number of unlabeled instances, denoted by U. This is a typical set-up in many real life applications as despite the huge amounts of data nowadays, only a small fraction of these data are labeled and can be therefore directly used for (supervised) learning. To deal with this issue, semi-supervised learning approaches exploit both labeled and unlabeled data for training - the underlying assumption is that having access also to the unlabeled data allow us to better understand the underlying class distribution [ZGBD09]. In this work, we investigate three popular semi-supervised learning approaches: Self-Learning, Co-Training and EM described hereafter.

3.4.1 Self-Learning

The main idea of Self-Learning [Fra67] is to use the labeled set L to build an initial classifier, then iteratively apply the model to the unlabeled corpus U and in each iteration, expand the training set L with instances from the unlabeled corpus which were predicted with high confidence by the classifier; the confidence is evaluated according to a user-defined threshold δ. The pseudocode of the Self-Learning algorithm is shown in Algorithm 2.

The initial training set T is the labeled set L (line 1). In each iteration, the training set is expanded by including confident predictions from U (lines 5–8); the expanded training set is used for building a new classifier (line 3). The procedure continues until the stopping criterion is met e.g., when U is empty or after a certain number of iterations or, if no further expansion is possible due to the threshold δ.

Algorithm 2: Pseudocode of Self-Learning

```
Input: L: labeled set, U: unlabeled set, δ: confidence threshold
Result: T: labeled set
T ⟵ L
while (stopping criterion) do
    Φ ⟵ train classifier on T;
    for i=1 to |U| do
        if (confidence of Φ.classify(U_i) ≥ δ) then
            T ⟵ T ∪ U_i, where U_i is the i-th instance in U
            Mark U_i as labeled;
    Update U by removing labeled instances;
return T;
```

The intuition behind Self-Learning is that we can use the confident decisions of the classifier to expand the training set, in some sort of exploitation of what the

classifier already knows sufficiently well. However, since some of the these predictions might be erroneous, Self-Learning might cause error propagation as at the end the training set is a mix of original labels and predictions which are taken equally into account for learning [HZ11, ZNS14]. Moreover, since the classifier mainly exploits what it already knows and expands the training set through similar instances, it is more difficult to learn new concepts.

3.4.2 Co-Training

Co-Training [BM98] assumes that the feature space, lets denote it by X, can be split into two parts, $X = (X^1, X^2)$, the so-called "views". The Co-Training algorithm trains two classifiers Φ^1, Φ^2, each working exclusively on one view, X^1, X^2, respectively. Initially, both classifiers are trained over the initial labeled set L, but each on its own view, $X_i, i \in \{1, 2\}$. In the original co-training approach [BM98], the unlabeled data are used to expand the joint training set as follows: At each iteration, Φ^1 classifies a few unlabeled instances for which it is more confident about and appends them to the joint training set. Similarly for Φ^2. The updated training set is then used for building the two new classifiers, again, each classifier is trained on its own view.

We follow a slightly different version, c.f., Algorithm 3, by maintaining a different training set for each classifier. We initialize Co-Training as above, with Φ^1, Φ^2 classifiers built upon the initial labeled set L but each exclusively on one view, X^1, X^2, respectively. At each iteration of Co-Training, the most confident predictions of each classifier are used for the expansion of the training set of the other classifier. That is, the most confident predictions of Φ^1 are used to expand the training set of Φ^2 and vice versa. Therefore, although both classifiers start with the same training set L and unlabeled set U (lines 1–2), over the iterations and as one learns from the other, the training sets of the two classifiers are different (lines 20–22). The procedure stops when the stopping criterion is met e.g., when U^1 or U^2 are empty or after a certain number of iterations or, if no further expansion is possible due to the threshold δ.

The intuition behind Co-Training is that each classifier provides labeled data to the other classifier, which the latter can use for learning. In contrast to Self-Learning, in Co-Training a classifier does not learn by its predictions rather by the confident predictions of the other learner (the first classifier might be non-confident for those predictions). Thus, the two views (classifiers) working together manage to progress learning while preventing each classifier to propagate its own error, as in Self-Learning. Two assumptions are proposed for Co-Training to work well: first, that each view (classifier) is able to learn the target concept given enough data, that is, each view is *sufficient* for learning and second, that the views are conditionally independent, that is the two views are independent given the class. Theoretical results have shown that if the sufficiency and independence assumptions are satisfied, co-training is guaranteed to work, however verifying that these assumptions hold in real datasets is not straightforward [DLZ11]. Luckily, however, Co-Training has been

Algorithm 3: Pseudocode of Co-Training

Input: L: labeled set, X^1, X^2: the two feature views, U: unlabeled set, δ: confidence threshold
Result: T_1, T_2: labeled sets
$T_1, T_2 \longleftarrow L$;
$U^1, U^2 \longleftarrow U$;
while *(stopping criterion)* **do**
  $\Phi_1 \longleftarrow$ train classifier on T_1 (using X_1 view);
  $\Phi_2 \longleftarrow$ train classifier on T_2 (using X_2 view);
  $TempSet_1 = \emptyset$;
  $TempSet_2 = \emptyset$;
  for *i=1* **to** $|U^1|$ **do**
    if *(confidence of* Φ_1*.classify(*U_i^1*)* $\geq \delta$*)* **then**
      $TempSet_1 \longleftarrow TempSet_1 \cup U_i^1$, where U_i^1 is the i-th instance in U^1
      Mark U_i^1 as labeled
  for *i=1* **to** $|U^2|$ **do**
    if *(confidence of* Φ_2*.classify(*U_i^2*)* $\geq \delta$*)* **then**
      $TempSet_2 \longleftarrow TempSet_2 \cup U_i^2$, where U_i^2 is the i-th instance in U^2
      Mark U_i^2 as labeled
  Update U^1 and U^2 by removing labeled instances;
  $T_1 \longleftarrow T_1 \cup TempSet_2$;
  $T_2 \longleftarrow T_2 \cup TempSet_1$;
return T_1, T_2;

successful in many real-world tasks even if the aforementioned conditions could not be ensured, e.g.,[NG00].

3.4.3 Expectation-Maximization (EM)

Expectation-Maximization (EM) belongs to a class of algorithms that iteratively estimate the maximum a posteriori probabilities in statistical models by exploiting the structure of the data (labeled and unlabeled) [DLR77]. The algorithms consist of two steps: the Expectation- or E-step and the Maximization- or M-step. EM starts by initializing model's parameters based on the labeled data (expected distribution, E-step). Afterwards, new instances are revealed to the model for which it tries to fit the current probability distribution to include the new data (M-step). These two steps are repeated until the stopping criterion is met e.g., maximum number of iterations is achieved or the current distribution does not change from the E-step to the M-step (convergence).

The EM pseudocode is shown in Algorithm 4. A classifier is trained upon the initial labeled data L (line 2). Afterwards, the classifier is employed to assign probabilistically-weighted class labels to the unlabeled data U (lines 4–7). Then, the classifier is rebuilt upon the labeled data and the unlabeled data which have been labeled by the previous classifier (line 8). The procedure is repeated until the stopping criterion is met (lines 3–9). The final classifier Φ is used for final labeling of the unlabeled data U (lines 10–12).

Algorithm 4: Pseudocode of EM

```
Input: L: labeled set, U: unlabeled set
Result: T: labeled set
T = ∅;
Φ ⟵ train classifier on L;
while (stopping criterion) do
    for i=1 to |U| do
        Φ.classify(U_i), where U_i is the i-th instance in U
        Mark U_i as labeled;
    Φ ⟵ train classifier on L ∪ U;
for i=1 to |U| do
    Φ.classify(U_i);
    T ⟵ T ∪ U_i;
return T;
```

The algorithm guarantees improvement of a parameter's estimation through each iteration if the mixture model assumption holds [DLR77]. If the model is wrong though, the unlabeled data may actually hurt the performance [CCC03]. Moreover,

EM is prone to local maxima and if a local maximum is far from the global maximum, unlabeled data might again hurt the performance of the learner [Nig01].

3.4.4 Discussion

We employ three well known semi-supervised methods to exploit the unlabeled data U together with the labeled data L in order to obtain more accurate predictions. All methods use unlabeled data to modify hypotheses obtained from labeled data alone. However each method comes with its own advantages and limitations. Self-Learning is an easy-to-use algorithm but it can propagate errors in the predictions which affect the next iterations and eventually the final result. Co-Training can overcome to some extent this problem by relying on two different learners that train each other by providing confident predicted labels. However, Co-Training works well under the sufficiency and independence assumption which do not always hold for real-world datasets. Finally, EM with generative mixture models tries to maximize the likelihood estimates of a model's parameters based on the data. However, it may perform poorly if the assumption on the correlation between classes and model components is violated and it might get stack to some local maximum. Moreover, EM can be extremely slow when data are multi-dimensional. Finally, both Self-Learning and Co-Training make hard assignments, whereas EM is the only method that probabilistically assigns an instance to all classes (soft assignment).

3.5 Overcoming class imbalance via data augmentation

Except for label scarcity, our learning setup is also characterized by class imbalance. In particular, the positive class is constantly overrepresented over the stream comparing to the negative sentiment class (c.f., Figure 3.3). Models trained upon imbalanced data learn mainly the majority class while ignoring the minority [HM13]. In our case, the problem is aggravated due to the propagation of the predicted labels in the next rounds of the semi-supervised learning process. As a result, the tendency of the models towards the majority class is much higher in the final models, as we also show in our experiments (c.f., Section 3.6.6).

Traditionally, class imbalance is handled through oversampling (from the minority class) and/or undersampling (from the majority class). Both approaches, however, come with limitations [DH+03, EJJ04]; in undersampling, one cannot control what information about the majority class is thrown away, whereas in oversampling, no new information is added to the training set, rather some instances from the minority class are replicated thus having a stronger effect on the classifier. In this work, except for oversampling and undersampling, we also employ data-augmentation techniques for generating plausible pseudo-instances for the minority class in order to correct for

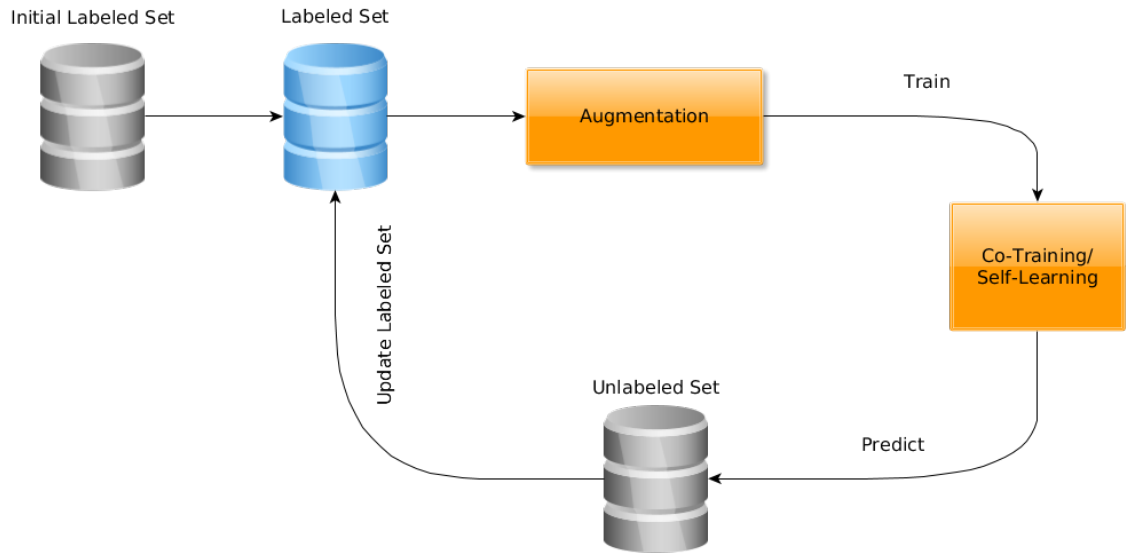


Figure 3.4: Augmentation-assisted semi-supervised learning

the class imbalance.

Augmentation is integrated into the semi-supervised learning process in an attempt to control the class imbalance problem over the training iterations. An overview of our approach is depicted in Figure 3.4. The training set is re-balanced using data augmentation (Section 3.5.1) and the semi-supervised learning process takes place as before but over the balanced data. We consider the labeled set balanced when the difference between positive (P_+) and negative (P_-) instances is smaller than a user defined class-balance threshold ϵ (in our experiments we set $\epsilon = 20\%$).

3.5.1 The Data Augmentation Process

Given an original binary classification dataset D with class imbalance (without loss of generality, let us assume that the negative class is the minority, i.e., $|P_-| << |P_+|$ for which $|P_-|$ is the amount of negative instances and $|P_+|$ the amount of positive instances), our goal is to build a balanced dataset D' via data augmentation, i.e., to generate new instances out of the original instances by applying domain-specific, label-preserving transformations. We refer to the generated instances as *pseudo-instances* and to D' as *augmented dataset*. Note that $D \subset D'$.

We propose two augmentation techniques: semantic augmentation (Section 3.5.1) and blankout corruption (Section 3.5.1) for balancing the classes, beyond the well-known under-sampling and over-sampling approaches (Section 3.5.1). The former employs semantic similarity between words (via word-embeddings) in order to create semantically similar instances out of the original instances. The latter corrupts the training instances by removing information (words) from the original instances and

thus, creates corrupted versions of the original instances. Both techniques are *feature transformation* techniques, i.e., they change the individual features/words. In order to ensure that the augmented instances will preserve their labels, we "transform" only non-sentimental words.

The words with a score higher than zero are considered as sentimental words, the rest as non-sentimental. For our dataset 38,222 words were found to be sentimental while the remaining terms (1,132,729) were filtered out.

Data augmentation via semantic similarity

To generate pseudo-instances of the same class, we employ the semantic similarity of words as captured through word embeddings [MSC+13]. Our idea is to generate pseudo-instances by replacing words in the original document with semantic similar words. In particular, for a selected word w occurring in an original document d, we generate its similar words based on their embedding vectors and we select randomly one of the top-k similar words w' to replace the original w in the pseudo-instance d'. We only consider words which are sentimental. As an example, the text "I love this car very much" could generate "I like this car very much".

There exist different word-embedding versions based on the data used for their training. In this work, we employ Glove word embeddings [PSM14] which have been generated by 2B tweets. For each sentimental word in our corpus we generate the top-k most similar words based on Glove embeddings (in our experiments, we set $k = 10$). A list of top-k similar words, called *similarity-list*, is generated from the aforementioned process which contains 33,037 terms (the remaining 5 thousand words were not included in Glove embeddings).

Given an instance d, the procedure for generating a pseudo-instance d' out of it is as follows: for the sentimental words $w \in d$, check if w exist in the similarity-list. If not, ignore w. Otherwise, replace w with a randomly selected similar word from its top-k semantic similar words according to Glove.

Data augmentation via corruption

Corruption of images via noise is a common transformation in the image domain and aims at building more robust machine learning models [GR06]. We follow a similar idea for text: we generate pseudo-instances by deleting a (randomly selected) word from the original document. To ensure that the class label is preserved, we do not remove negations or sentimental words (c.f. Section 3.5.1 on sentimental words).

To ensure that the resulting document is still plausible, we apply corruption to sufficiently long documents, namely to documents of at least four words as suggested by [TV14]. As an example, the text "I don't like the morning traffic" could be transformed into "I don't like the traffic". Since our target is class imbalance, we generate pseudo-instances only for the minority class.

Oversampling and Undersampling

Oversampling (shortly, Over.) and undersampling (shortly, Under.) do not operate on the feature level but rather on the instance level. Oversampling is repeatedly applied on the minority instances, by randomly duplicating instances, until the difference between positive and negative instances is less or equal to the class-balance user defined threshold ϵ. Undersampling is applied on the majority class by randomly removing instances from the majority class until the threshold ϵ is met.

Moreover, we used a combination of (random) oversampling and undersampling (shortly, Over. & Under.) that works as follows: (i) first undersampling is applied by removing half of the majority's instances (b) if there is still class imbalance (according to the user defined class-balance threshold), oversampling is applied on the minority instances until the threshold is met, otherwise the process terminates.

Discussion

Augmentation techniques like semantic similarity and corruption have a similar goal to oversampling/undersampling, namely to balance the population of the two classes. There are however fundamental differences between the different approaches and each one comes with its own assumptions and limitations. On the one hand, oversampling does not add any new information to the process and can also amplify existing noise by duplicating noisy instances. On the other hand, undersampling may result in removing valuable information from the dataset which can degrade the overall performance of the model. Semantic augmentation depends on the quality of the pre-trained word embeddings. If the employed word embeddings come from a different corpus than the applied then the dictionary intersection between word embeddings and the corpus will be limited. In addition, polysemous words may totally change the context of a sentence; for example, “I like apple products" which refers to the famous company can be converted to “I like vegetable products". Corruption can also change the sentiment of a sentence; for example, “I support banning smoking in public areas" can be converted to “I support smoking in public areas". Finally, a common pitfall in all augmentation methods is that by augmenting already noisy and/or biased instances the reinforcement of noise and mistakes is inevitable and therefore the overall data quality is degraded.

3.6 Experiments

We report on the Twitter dataset introduced in Section 3.3. We employ Multinomial Naive Bayes (MNB) as our basic classifier for Self-Learning, Co-Training and EM. We choose Naive Bayes as our base learner due to its ability to handle huge amounts of features, while being tolerant to irrelevant features. Moreover, due to its simplicity, it can be trained upon vast amounts of training data extremely fast [NMTM00, NMM06],

compared to e.g., neural networks such as [SPW+13] require up to 5h for training for small scale datasets (less than 10K instances). Finally, it can cope with dynamic feature spaces and errorless model update, both important properties for our stream evaluation. For the implementation[10], we have used Spark's distributed environment (version 1.6) and its Machine Learning library MLlib [MBY+16].

Co-Training requires two different feature spaces. Therefore, except for the unigrams we also experimented with two different feature sets: (i) bigrams: the feature space was extracted from the pre-processed text, as described in Section 3.3, (ii) language features: we extracted also POS tags using Stanford's POS tagger [TKMS03] as well as syntactic features like number of words in capital, words with repeated characters, links and mentions. Moreover, we employed a dictionary which contained sentimental hashtags [MKZ13] and counted the occurrences of positive and negative hashtags in our tweets, if any (the extraction was done over the original tweets, not the preprocessed ones). We refer to this feature space as "SpecialF".

Therefore we have two alternative feature setups for Co-Training:

- Co-Training$^{1}_{\text{[unigrams-bigrams]}}$
- Co-Training$^{2}_{\text{[unigrams-SpecialF]}}$

These *views* are not conditionally independent. Nonetheless, recent works indicate that relaxation of the independence criteria does not have much impact on the performance [BCM+13, LZX+13, ZTZX14, BBY05]. For the EM and Self-Learning approaches, we used unigrams (after pre-processing).

Our evaluation examines the performance of the different semi-supervised learning approaches w.r.t. the following aspects:

- batch vs stream annotation setup (Section 3.6.1, 3.6.2, respectively)
- effect of augmentation on class imbalance (Section 3.5)
- effect of redundancy (re-tweets) on performance (Section 3.6.5)
- a qualitative evaluation of the derived labels based on crowd-sourcing (Section 3.6.3).

3.6.1 Performance of batch annotation

We split our ground truth (2.5 million tweets) in 10 folds, 1 of which is used for testing and the rest, together with unlabeled data predictions, are used for training. In each iteration of the 10-cross-validation process, the test set is fixed, the training set

[10]Source code and data are available at: https://iosifidisvasileios.github.io/Semi-Supervised-Learning/

however is expanded through the addition of (unlabeled) tweets that were predicted with high confidence by the classifier. We report the averaged results of 10-fold cross-validation in terms of classification accuracy for Self-Learning and Co-Training, under different confidence thresholds δ that determines which of the classifier predictions are incorporated in the training set. For EM, we do not set any threshold since it is the algorithm's property to maximize the data likelihood based on the predictions.

Self-Learning-based batch annotation

In Figure 3.5 (top) we display the accuracy of the Self-Learning approach under different confidence thresholds δ, in the range [65%-100%] and how, the accuracy changes as the algorithm iterates through the remaining unlabeled tweets. We stop at 5 iterations as the algorithm manages to annotate almost all unlabeled tweets within those iterations. We also show the accuracy in the initial training set, i.e., before expansion.

The accuracy of Self-Learning drops comparing to the accuracy of the initial model (trained in the initial labeled set L); this is to be expected as the training set is expanded through predictions. The drop is more drastic in the first iteration. The reason is that the 1st iteration results in the largest expansion of the training set as a large number of predicted instances is added to the training set therefore affecting the extracted models. The expansion depends on the threshold δ, as higher values are more selective and therefore result in smaller expansion. The training set expansion under different δ thresholds and over the iterations is shown in Figure 3.5 (bottom). At δ=65%, for example, the expanded training set is about 8,100% larger than the original training set L.

The accuracy drops with δ. The decrease is directly related to the amount of expansion of the training set. For the low δ values, the decrease is very small after the first two iterations; the reason is that the bulk of predictions was already added to the training set in the first two iterations and therefore the addition of the ne predictions does not influence the classifiers. For larger δ values (90%-95%) though, the accuracy drops faster as the corresponding training set expands more gradually. The only exception is $\delta = 100\%$; the accuracy does not change because the training set is hardly influenced, as this threshold is very selective and therefore only a few predictions can satisfy it.

The annotated dataset is depicted in Table 3.4: for different δ we report the amount of positive, negative and unlabeled tweets, i.e., tweets that remained unlabeled after the fifth iteration. As we can see the more selective δ is the more tweets remain unlabeled, with the extreme case of $\delta = 100\%$ where almost all tweets (99.71%) remained unlabeled. We report the percentage of positive and negative annotations over the labeled set and not over the complete dataset, in order to highlight the class distribution of the predicted labels. In the last row of the table we also report the class distribution in the original training set, i.e., before expansion. The major-

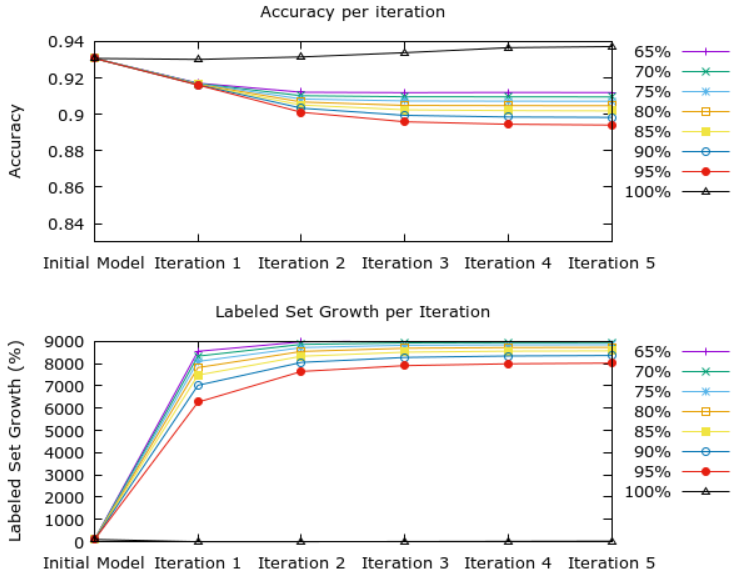


Figure 3.5: Batch annotation with Self-Learning: accuracy and labeled set growth under different δ values while the algorithm iterates through the remaining unlabeled tweets.

ity of the predictions refers to the positive class, on average 88% of the predictions are positive and 11% negative. As the confidence threshold increases, the positive class percentage in the predictions also increases. The higher percentage of positive class predictions (99,86%) is manifested with a threshold of 100%, implying that the classifier is more confident about the positive class and therefore the training set is expanded with more examples of the positive class.

Co-Training-based batch annotation

Figure 3.6 demonstrates the accuracy of Co-Training for two confidence levels (δ = 65% and δ = 95%) and how the accuracy changes as the algorithm iterates through the remaining unlabeled tweets set. We stopped at four iterations since after the 3rd iteration the number of unlabeled tweets is very small.

The best performance is achieved when we learn from unigrams (1st classifier) and bigrams (2nd classifier), i.e., by the Co-Training$^{1}{}_{\text{[unigrams-bigrams]}}$ model. Hereafter, we use this classifier for the comparison, and we refer to it as Co-Training. We show the accuracy of this model under different thresholds δ in Table 3.5. We also show the accuracy of the initial model, i.e., before the training set expansion; the expansion results in accuracy loss (around 2%). As we increase δ, there is a small improvement for values in the range 65%-80%, but the performance slightly drops with higher values in the range 85%-95%. The only exception (which outperforms the initial model) is δ = 100% which is not affected that much as the training set is not much

δ	positive predictions	negative predictions	unlabeled
65%	201,860,127 (88.46%)	26,315,605 (11.53%)	1.13%
70%	200,212,418 (88.49%)	26,033,446 (11.50%)	1.97%
75%	198,296,101 (88.59%)	25,525,791 (11.40%)	3.02%
80%	196,017,401 (88.78%)	24,757,934 (11.21%)	4.34%
85%	193,134,363 (89.06%)	23,720,362 (10.93%)	6.03%
90%	189,271,805 (89.49%)	22,217,878 (10.50%)	8.36%
95%	183,012,328 (90.21%)	19,843,802 (9.78%)	12.10%
100%	650,450 (99.86%)	877 (0.13%)	99.71%
Initial Model	2.211.091 (87,47%)	316.662(12,52%)	

Table 3.4: Batch annotations with Self-Learning: Annotated results per class for different confidence values δ.

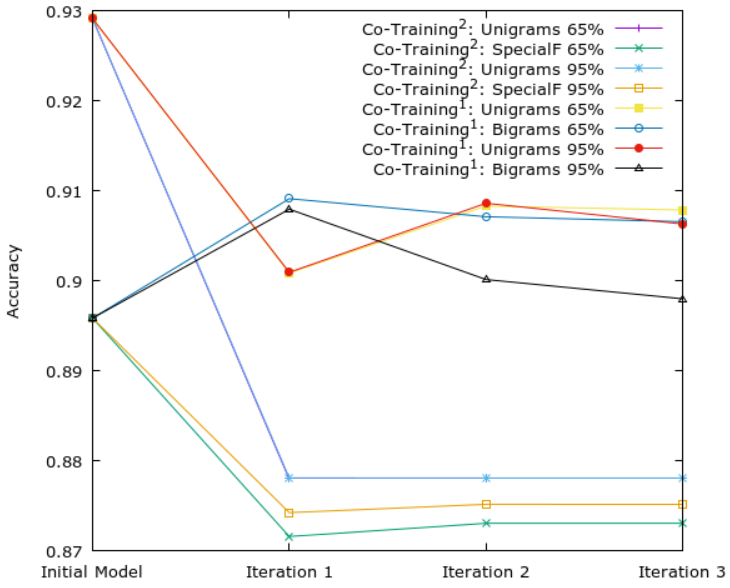


Figure 3.6: Batch annotation with Co-Training: accuracy for $\delta = 65\%$, $\delta = 95\%$ while the algorithm iterates through the remaining unlabeled tweets. The accuracy is displayed for each Co-Training classifier-member.

δ	Unigrams	Bigrams
65%	90.65%	90.71%
70%	90.76%	90.66%
75%	90.81%	90.57%
80%	90.82%	90.51%
85%	90.78%	90.41%
90%	90.69%	90.28%
95%	90.50%	90.02%
100%	93.16%	89.03%
Initial Model	93.07%	88.52%

Table 3.5: Average accuracy of Co-Training$^{1}{}_{\text{[unigrams-bigrams]}}$ members under different δ.

δ	positive predictions	negative predictions	unlabeled
65%	175,704,567 (76.64%)	53,547,361 (23.35%)	0.66%
70%	178,361,861 (78.26%)	49,544,295 (21.73%)	1.25%
75%	180,646,395 (79.90%)	45,419,649 (20.09%)	2.04%
80%	182,180,488 (81.52%)	41,287,186 (18.47%)	3.17%
85%	182,758,504 (83.04%)	37,300,375 (16.95%)	4.65%
90%	182,707,849 (85.06%)	32,069,200 (14.93%)	6.93%
95%	179,527,239 (87.43%)	25,810,993 (12.56%)	11.02%
100%	1,281,748 (99.60%)	5,116 (0.39%)	99.44%
Initial Model	2.211.091 (87,47%)	316.662(12,52%)	

Table 3.6: Batch annotation with Co-Training: Annotated results per class for different confidence values δ.

expanded due to the very selective δ.

How the performance varies over the different iterations and for different thresholds δ and what degree of original training set expansion is achieved is shown in Figure 3.7 (top). The picture is similar to Self-Learning, the first two iterations produce the largest amount of confident predictions, especially for lower δ values.

The annotated dataset is depicted in Table 3.6. Similarly to what we have observed for Self-Learning, the more selective δ is the more tweets remain unlabeled, at $\delta = 100\%$ almost all tweets (99.44%) remained unlabeled. Moreover, the amount of unlabeled tweets is smaller comparing to the Self-Learning in Table 3.4. Regarding the class distribution of the predictions, the positive class is still predicted more often. However and on the contrary to Self-Learning, the negative class is better represented in this setting. The explanation lies on the fact that in Co-Training classifiers learn from each other, rather than only from their predictions.

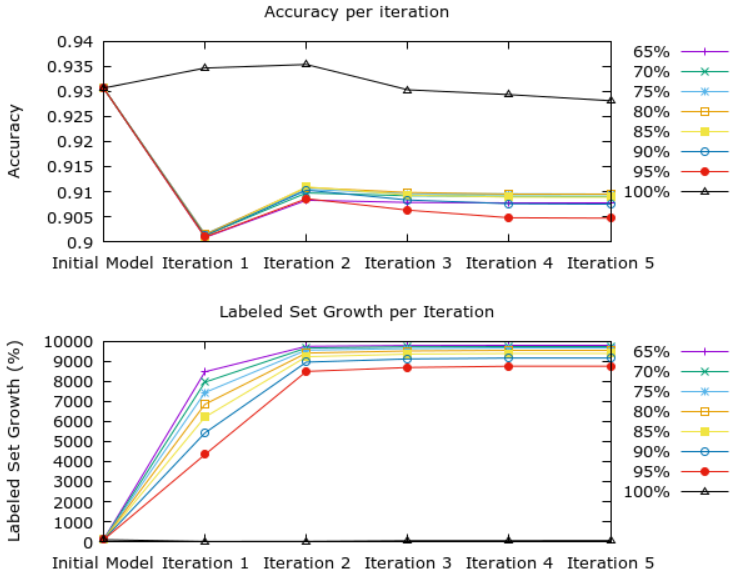


Figure 3.7: Batch annotation with Co-Training: accuracy and labeled set growth under different δ values while the algorithm iterates through the remaining unlabeled tweets.

EM-based batch annotation

In Table 3.7 we report the accuracy of EM per iteration. We stop at 5 iterations as after that the log likelihood does not change too much. The accuracy of EM drops significantly as the number of iterations increases. In particular, starting with a good initial model (93.52% in the first iteration trained on only labeled data) it drops to 83.53% in the fifth iteration trained upon all data. A possible reason is that the unlabeled data alter the model in a way that hurts the overall performance. Note here that in contrast to Self-Learning and Co-Training we do not use only qualified instances for the dataset expansion (based on some threshold δ), rather we accept all instances.

We also observe that the classifier predictions become more balanced over the iterations with the amount of predicted negative instances growing by 210K from the first to second iteration and by 860K instances from second to third iteraton.

Comparison of EM, Self-Learning and Co-Training

In Table 3.8 we report the average (over all iterations) accuracy of Co-Training and Self-Learning for different δ values (for the Co-Training, we report here on the performance of the best classifier member, i.e., the one built upon unigrams). As we can see, Self-Learning accuracy decreases (in absolute numbers) with δ much faster than the accuracy of Co-Training. As we can see from Tables 3.4, 3.6, Co-Training

Iter.	Accuracy (%)	positive predictions	negative predictions
1	93.52	177,770,034 (77.03%)	53,023,155 (22.97%)
2	90.87	177,559,316 (76.93%)	53,233,873 (23.07%)
3	87.83	176,702,013 (76.56%)	54,091,176 (23.44%)
4	85.31	175,936,532 (76.23%)	54,856,657 (23.77%)
5	83.53	175,162,259 (75.90%)	55,630,930 (24.10%)

Table 3.7: Batch annotation with EM: annotated results per class over the iterations

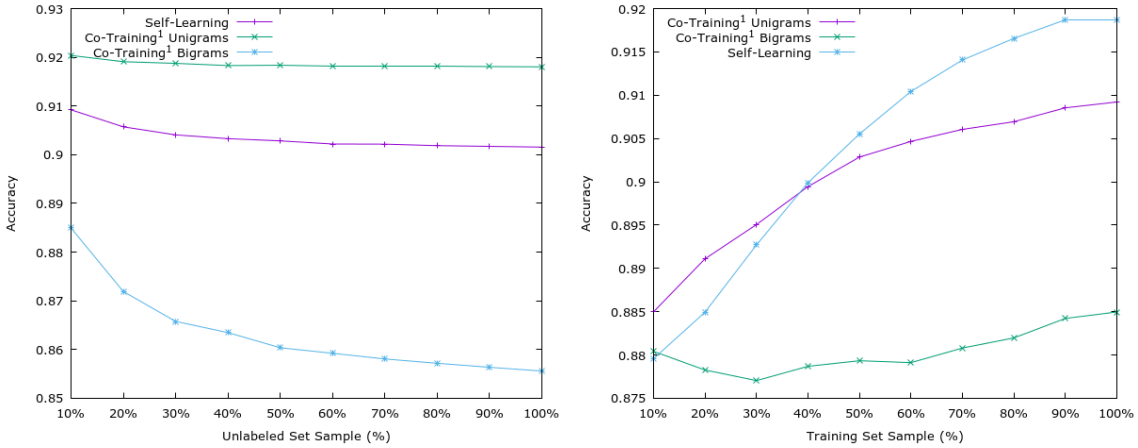
δ	Self-Learning	Co-Training (Unigrams)
65%	91.30%	90.65%
70%	91.11%	90.76%
75%	90.93%	90.81%
80%	90.75%	90.82%
85%	90.49%	90.78%
90%	90.31%	90.69%
95%	90.03%	90.50%
100%	93.38%	93.16%
Initial Model	93.07%	93.07%

Table 3.8: Batch: Self-Learning vs Co-Training, average accuracy for different δ

produces more labels than Self-Learning and results in better class balance.

By comparing Tables 3.4, 3.6 and 3.7, we observe that Self Learning produces more imbalanced outcomes as δ increases compared to Co-Training and EM. Since the initial ground truth is already highly imbalanced, Self Learning propagates this behavior stronger than Co-Training and EM, as it is the most sensitive to its own errors method. As δ increases, Self Learning trains progressively upon its most confident predictions, which in the majority are positive instances. As we show in a later section (c.f., Section 3.6.6), Self Learning propagates more positive errors compared to Co-Training, as δ increases. Co-Training also produces imbalanced outcomes as δ increases, however at a much lower rate compared to Self Learning. In Co-Training, the models may not propagate their own errors, however they are still subject to errors; in our case, such errors might occur due to e.g., the fact that unigrams and bigrams do not provide completely independent feature spaces and therefore they might fall in the same pitfall, as Self Learning, for high δ values. EM on the other hand, is not bound to threshold (δ) scores (in this scenario), thus it maintains a similar class ratio to the initial ground truth dataset.

Thus far we expand the training set based on the confidence threshold δ, which however results on an uncontrolled expansion (in terms of number of annotated instances) of the training set. To evaluate the effect of the magnitude of dataset expansion, we performed a controlled experiment where we gradually expand the training



(a) Unlabeled set expansion

(b) Labeled set expansion

Figure 3.8: Batch: Effect of predicted instances used for dataset expansion (left) and effect of labeled set (right) using Co-Training and Self-Learning, per iteration

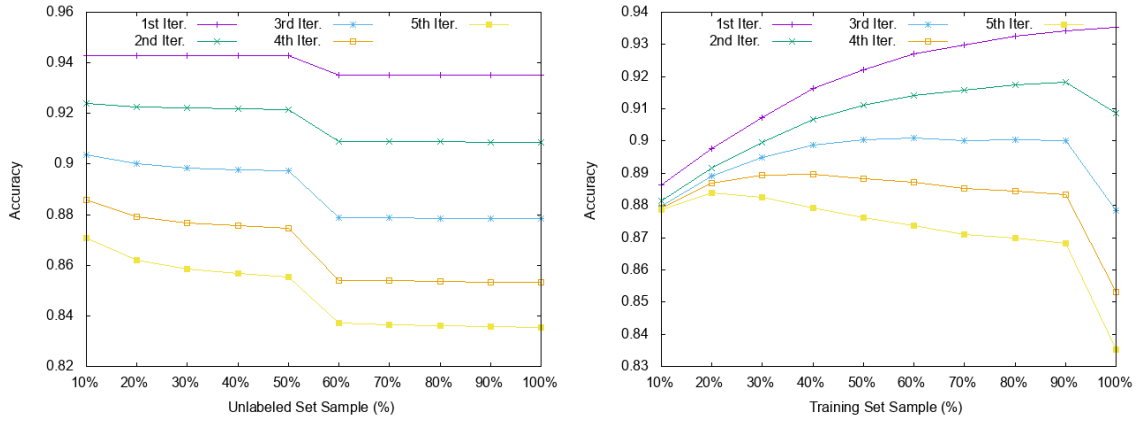
set by adding classifier predictions. In particular, we built an initial model in the original training set which we then used to annotate the unlabeled set. From the annotated set we randomly select [10%-100%] predictions/instances which we then use for dataset expansion. The results are depicted in Figure 3.8a. As we can see the accuracy drops as we further expand the dataset, for both Co-Training and Self-Learning. As the ratio of annotated instances increases, Co-Training (bigram's model) experiences a faster drop in its performance.

Same behavior is exhibited by EM, in Figure 3.9a. As the unlabeled amount of instances is increasing we observe that the accuracy is declining, faster for iterations four and five. When the unlabeled set reaches 50% of its original volume size then EM algorithm starts making more and more errors which indicates noise in the unlabeled data. Note that such a drop is observed also when adding 10% of unlabeled data; at 10% however, the unlabeled data are still around 10 times larger than the labeled ones.

We also evaluated the effect of labeled data, by varying the amount of labeled data and using a 10% sample of the unlabeled set for predictions (which scores the best performance in Figure 3.8a). The results are depicted in Figure 3.8b. As we can see, when the number of labels is small Co-Training performs better than Self-Learning. With 40% of labels or more Self-Learning is marginally better.

The results of EM for the same experiment are depicted in Figure 3.9b. 10% of the unlabeled data are employed by EM to find the maximum likelihood. One interesting observation is that the first three iterations are improving and the training set expands, while in the last two iterations accuracy declines. When the whole labeled set is used, only the first iteration is improving while all the rest are decreasing.

Due to the bad performance of EM in the batch case and for efficiency reasons



(a) EM: Unlabeled set expansion

(b) EM: Labeled set expansion

Figure 3.9: Batch: Effect of predicted instances used for dataset expansion (left) and effect of labeled set (right) using EM, per iteration

(EM was significantly slower than the other two methods), we report hereafter only on Self-Learning and Co-Training. We plan to further investigate the EM performance in future research.

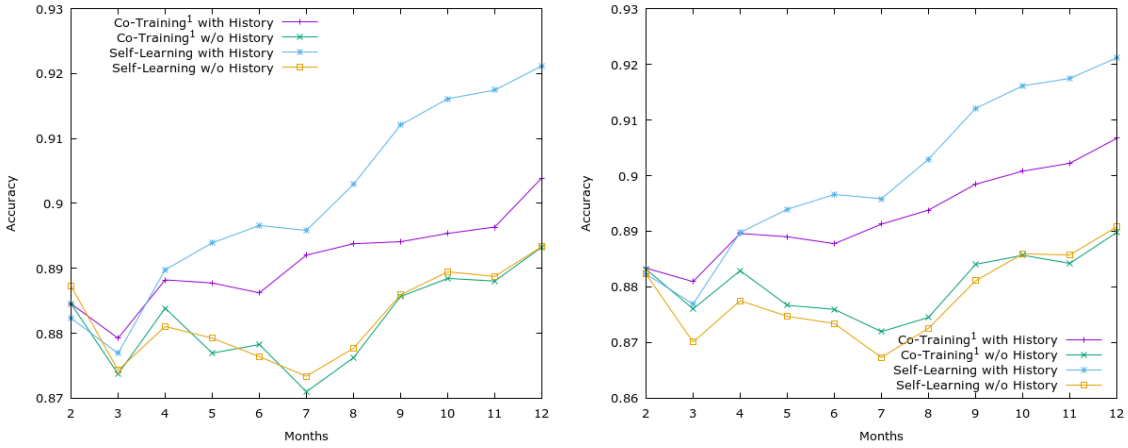
3.6.2 Performance of stream annotation

For the stream approach we process the data on a monthly basis, and we evaluate how the temporal processing affects our methods. Let L_i be the labeled data (ground truth) for month i and let U_i be the corresponding unlabeled set. Our complete dataset therefore is a sequence of the form: $((L_1, U_1), (L_2, U_2), \cdots (L_{12}, U_{12}))$ covering the whole year 2015.

We evaluate two variants:

- *without history*: we learn our models (Self-Learning, Co-Training) on each month i based on the labeled data of that month L_i and also by including confident model predictions from the corresponding unlabeled dataset U_i. We evaluate those models with the ground truth for the next month L_{i+1}.
- *with history*: for a month i, the labeled set upon which we build our model consists of all labeled instances up to month i, i.e., $\sum_{i=1}^{i} L_i$. Similarly, for the expansion we consider all unlabeled instances up to month i, i.e., $\sum_{i=1}^{i} U_i$, and we add to the training set those that were predicted with high confidence by the model.

That is, we differentiate on whether we use historical data from the stream to build our models, or we just use data from the current time point (month).



(a) Stream: Holdout

(b) Stream: Prequential

Figure 3.10: Stream comparison between Co-Training and Self-Learning

In the above scenario all labeled data are used for training and testing, as each month is tested with the labeled data of the next month. We refer to this as *prequential evaluation*. We also consider a *holdout evaluation*: we split the original dataset into a training and a testing set spanning the whole period. The evaluation procedure is similar to prequential evaluation, the only difference is that we use for training (testing) only data from the training (testing, accordingly) set of the given month(s). That is not all labeled data are used for training/testing, rather a sample of them according to the initial split.

Self-Learning vs Co-Training

The holdout evaluation is depicted in Figure 3.10a, the prequential in Figure 3.10b; the performance is similar for both evaluations. For both Co-Training and Self-Learning, history improves the performance. For the models with history, Co-Training is better in the beginning but as the history grows its performance decreases and Self-Learning results in the best performance. So Co-Training is more effective with fewer labels; this is also evident in the non-history models, where we see that Co-Training outperforms Self-Learning for almost all months.

From the above experiment it is clear that history improves performance. To evaluate the effect of history's length, we run the same experiment with a sliding window of three months; in particular, we used the labeled instances of months [1-3] for building an initial model, we expand the training set including predictions for unlabeled instances in months [1-3] and we use the derived model to score the next month i.e., month 4. The results are depicted in Figure 3.11. As we can see, Self-Learning is better for almost all months. Again, we denote that Co-Training works better with limited labels.

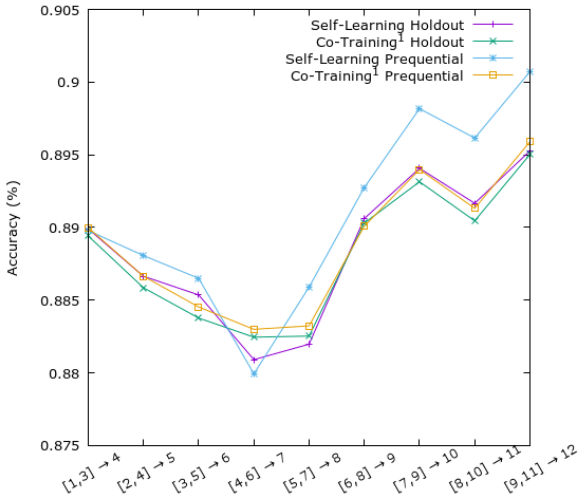


Figure 3.11: Stream: Sliding (3 months)

Comparing to the full-history case, in the sliding window approach we have a small decrease in the performance (less than 2.0%) but on the other hand much lighter models and therefore better efficiency (time, memory). The amount of data for each approach is depicted in Figure 3.12: labeled set is the original labeled data, training set is the expanded dataset of labeled instances and confident classifier predictions that was used for training.

As we can see, when we consider historical data, the amount of labeled and training instances is increasing over time, whereas for the non-history version these amounts are not changing that much over time. A similar behavior occurs for the sliding window version.

The class distribution of the predictions is shown in Figure 3.13, for all different window models: without history, with full history and with a sliding history of 3 months. For all window models, most of the predictions for both Co-Training and Self-Learning refer to the positive class. Co-Training produces on average less positive instances than Self-Learning. This is evident in the with-history and sliding-window approach: Self-Learning produces more positive predictions than Co-Training. This is due to the fact that Self-Learning is biased towards what it knows best (the positive class in this case). On the contrary, Co-Training is less biased as the two classifiers learn by each other.

To conclude the stream approach, Co-Training achieves the best performance with limited labels; as the amount of labeled data increase, Self-Learning surpasses its performance. Self-Learning is more biased to its own predictions comparing to Co-Training and therefore it results in more positive predictions.

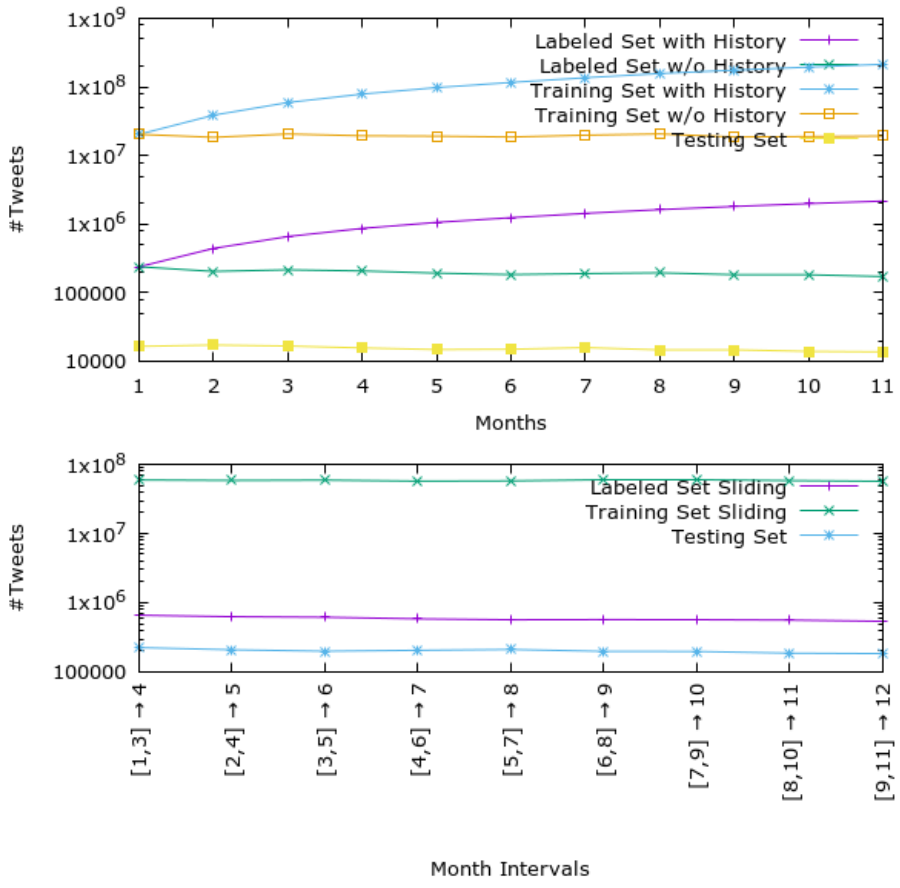


Figure 3.12: Stream: Prequential evaluation - cardinality of labeled, training, testing sets.

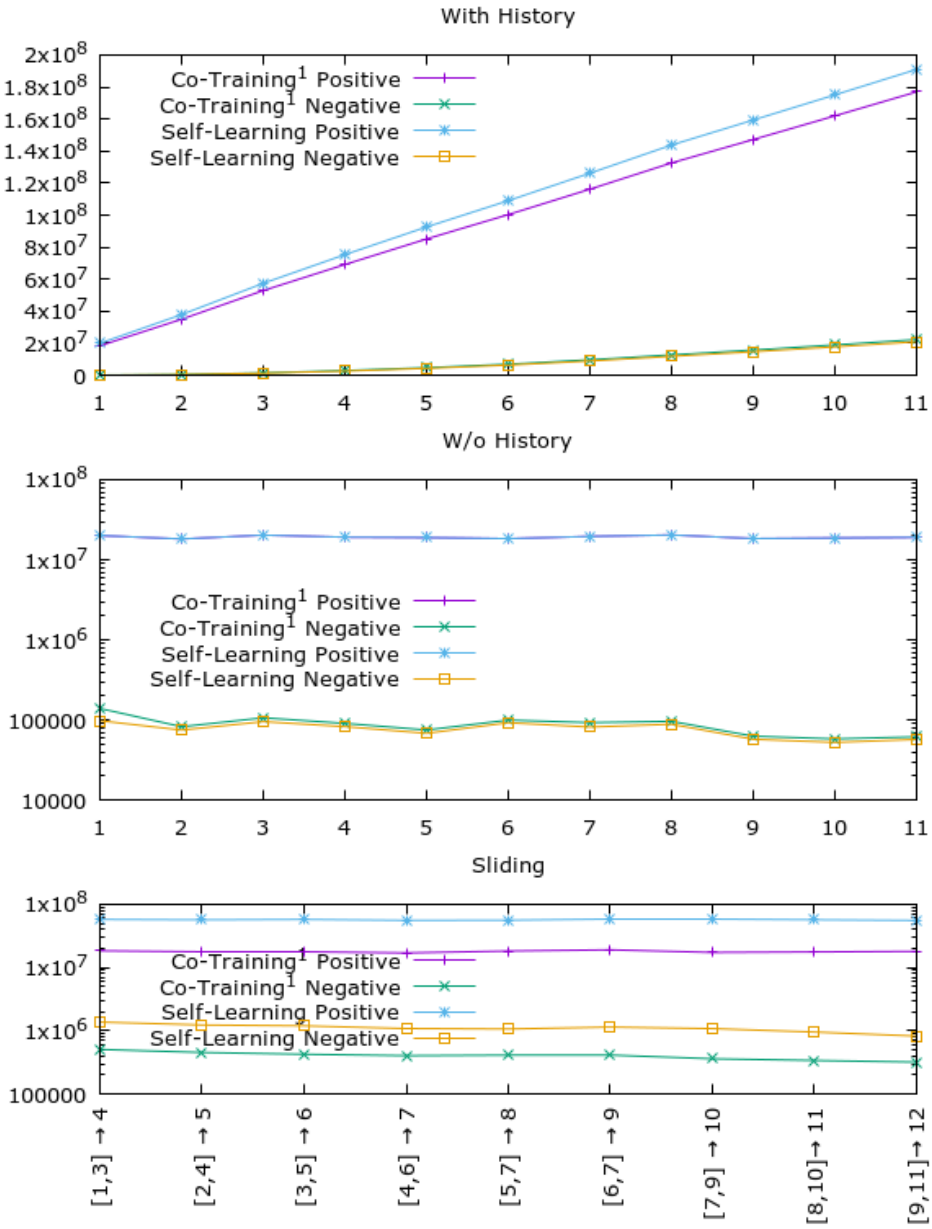


Figure 3.13: Stream: Class distribution of the annotated tweets over time

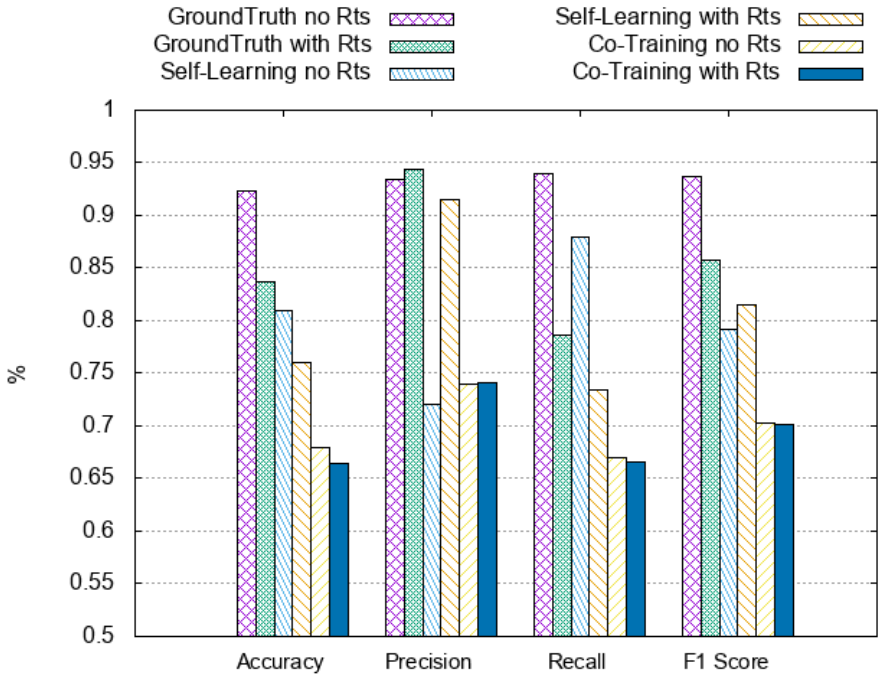


Figure 3.14: Evaluation using Crowdflower platform

3.6.3 Crowd-Sourcing Evaluation

Except for the quantitative evaluation, we have also performed a qualitative evaluation of the derived labels by crowdsourcing. There is a plethora of crowd-sourcing platforms such as MTurk, CrowdFlower, CloudCrowd, ShortTask, MicroWorkers and so on [VL13]. We chose CrowdFlower[11], a platform in which annotators work on tasks such as data research tasks, transcription, categorization, text production for product descriptions, etc. Almost 5 million contributors have completed more than 1 billion tasks so far.

In total 6,000 tweets were annotated by the crowd, selected as follows: we randomly extracted 1,000 tweets from each corpus: GroundTruth with retweets, GroundTruth without retweets, Self-Learning with retweets, Self-Learning without retweets, Co-Training with retweets and Co-Training without retweets. We requested three votes per tweet and filtered tweets which their average confidence score was less than 80%. The average label confidence is computed based on the confidence score given by each worker (in the range of $[0, 100]$). After the filtering, we ended up with 3,928 crowd annotated tweets. Figure 3.14 depicts the evaluation over our datasets.

Datasets which do not contain retweets have better accuracy than datasets with retweets. We make these human-annotated tweets (3,928) available to the community by providing the tweet id and the human-annotated label.

[11] http://www.crowdflower.com/

		No Retweets Self-Learning		Retweets Self-Learning	
		Positive	Negative	Positive	Negative
Senti Stren.	Positive	**37.84%**	15.25%	**43.72%**	16.47%
	Negative	11.00%	**48.96%**	8.00%	**31.69%**
	Neutral	51.16%	35.79%	48.29%	51.85%

Table 3.9: SentiStrength vs Self-Learning. Agreements among predictions marked in boldface.

		Unigrams				Bigrams			
		No Retweets CoTraining		Retweets CoTraining		No Retweets CoTraining		Retweets CoTraining	
		Positive	Negative	Positive	Negative	Positive	Negative	Positive	Negative
Senti Stren.	Positive	**39.37%**	21.04%	**42.30%**	21.72%	**38.09%**	13.62%	**44.37%**	16.47%
	Negative	10.93%	**30.62%**	9.86%	**25.79%**	10.91%	**48.89%**	7.78%	**31.10%**
	Neutral	49.70%	48.35%	47.84%	52.49%	51.00%	37.49%	47.85%	52.43%

Table 3.10: SentiStrength vs Co-Training. Agreements among predictions marked in boldface.

3.6.4 Comparing Self-Learnig and Co-Training to SentiStrength and TreeBank

In this section, we compare our predictions to those of SentiStrength and TreeBank.

In Table 3.9, we compare SentiStrength to Self-Learning for each corpus: with retweets and without retweets. We see that for the corpus which does not contain retweets the negative percentage agreement is higher while the positive percentage agreement is lower in contrast to the corpus which contains retweets. The same behavior is observed in Table 3.10, where we compare SentiStrength to Co-Training. In both comparisons and in contrast to TreeBank (Self-Learning and Co-Training versus SentiStrength), we observe that SentiStrength tends to classify more tweets

		No Retweets Self-Learning		Retweets Self-Learning	
		Positive	Negative	Positive	Negative
Tree Bank	Positive	**15.95%**	4.40%	**18.32%**	4.60%
	Negative	40.50%	**75.95%**	37.42%	**60.64%**
	Neutral	43.55%	19.66%	44.26%	34.76%

Table 3.11: TreeBank vs Self-Learning. Agreements among predictions marked in boldface.

		Unigrams				Bigrams			
		No Retweets CoTraining		Retweets CoTraining		No Retweets CoTraining		Retweets CoTraining	
		Positive	Negative	Positive	Negative	Positive	Negative	Positive	Negative
Tree Bank	Positive	**16.37%**	6.10%	**18.00%**	6.73%	**15.41%**	3.64%	**18.66%**	4.55%
	Negative	38.77%	**62.41%**	37.33%	**57.44%**	41.51%	**65.41%**	36.95%	**61.07%**
	Neutral	44.86%	31.49%	44.67%	35.83%	43.07%	30.95%	44.39%	34.38%

Table 3.12: TreeBank vs Co-Training. Agreements among predictions marked in boldface.

as neutral. This occurs due to our SentiStrength setup, in which we consider as neutral instances tweets that have same positive and negative score (absolute values). In many cases, positive and negative scores are equally high, however SentiStrength cannot determine which sentiment is stronger than the other.

For Sentiment TreeBank, we observe the same behavior as in SentiStrength comparisons, where positive agreement increases and negative agreement decreases between the corpus without retweets and corpus with retweets (Tables 3.11 and 3.12). Negative agreement in both comparisons is much higher than positive agreement. Moreover, we observe that the positive agreement, in both Self-Learning and Co-Training comparisons, is much lower than SentiStrength while the negative agreement is higher.

For the positively annotated tweets (by Self-Learning or Co-Training), TreeBank classifies most of them as negative compared to SentiStrength e.g., SentiStrength compared to Self-Learning has 11% disagreement, in the corpus without retweets, while TreeBank has 40% disagreement.

In conclusion, comparing the performance of methods trained upon different datasets is not easy, as already discussed in Section 3.3.4 regarding the ground truth comparison. Tweets are far different from comments or reviews due to the character limitations, therefore the structure of a tweet can vary significantly [KK10] compared to other texts. Moreover, the different spanning periods may lead to feature drifts, i.e., changes in the features/words or their relevance to the different classes [MSN18, HLJ16].

3.6.5 Performance under Redundancy (retweets)

Thus far, we reported on English tweets without redundancy (retweets). To evaluate the impact of redundancy on the aforementioned methods we repeat all our experiments with retweets. We perform holdout evaluation (67% training and 33% testing split) and report here on the most interesting findings. In Table 3.13, same setup as Table 3.8, we report on the performance of the redundant dataset. As we can see, the accuracy values are lower comparing to the non-retweets case. Moreover, the drop in the performance as δ increases is higher.

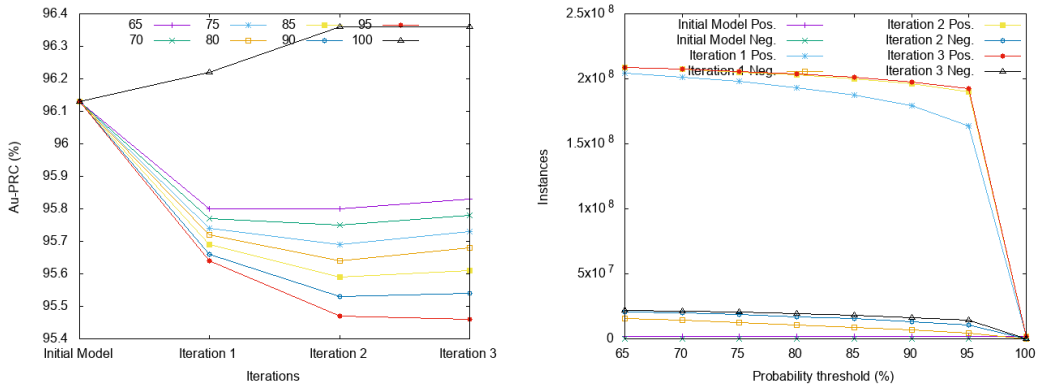
δ	Self-Learning	Co-Training (Unigrams)
65%	87.09%	87.24%
70%	86.73%	87.09%
75%	86.42%	86.91%
80%	86.09%	86.68%
85%	85.75%	86.39%
90%	85.31%	86.04%
95%	84.71%	85.43%
100%	92.79%	91.25%
Initial Model	92.92%	92.92%

Table 3.13: Batch: Self-Learning vs Co-Training, average accuracy for different δ (Retweets version)

δ	Self-Learning noRTs	Co-Training noRTs	Self-Learning with RTs	Co-Training with RTs
65%	1:8	1:4	1:2	1:2
70%	1:8	1:4	1:2	1:3
75%	1:8	1:4	1:2	1:2
80%	1:8	1:4	1:2	1:2
85%	1:8	1:5	1:2	1:2
90%	1:9	1:6	1:2	1:2
95%	1:9	1:7	1:2	1:2
100%	1:741	1:248	1:2	1:14

Table 3.14: Class Ratio (negative:positive) of the predictions for different datasets and methods

The effect of redundancy on the class imbalance of the predicted labels is shown in Table 3.14, where we display the negative: positive class ratio for different δ values, for Self-Learning and Co-Training for the dataset with retweets and the dataset without retweets. It is clear that the class imbalance is significantly affected by duplicates. For both methods, the predictions are more balanced for the dataset with retweets. A possible explanation is that by eliminating the retweets, the negative class which was already a minority (75%-25% in the dataset with retweets) became even more underrepresented (87%-13% in the dataset without retweets). In the retweets version, on the other hand, the redundancy acted to some extent as over-sampling (though in our case, retweets come from both classes) and this helped the classifier to make more balanced predictions.



(a) Au-PRC (b) Number of instances per iteration

Figure 3.15: Self-Learning original

3.6.6 Performance of augmentation techniques

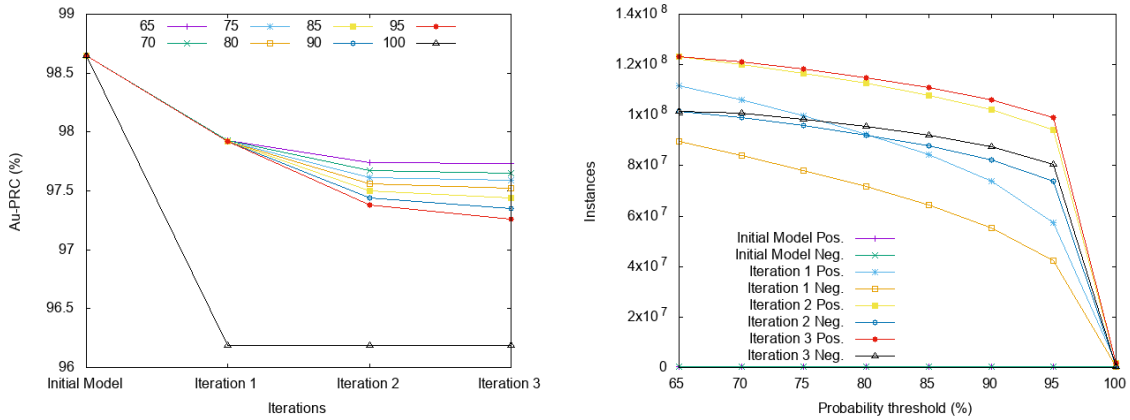
The performance of Co-Training and Self-Learning is evaluated using the area under precision-recall curve (AU-PRC) which better reflects classifier's performance, comparing to e.g., accuracy, in case of class imbalance [SR15] and also provides more informative representations than AUC [HG09]. We perform holdout evaluation (67% training and 33% testing split) and report AU-PRC and the number of predicted instances per iteration. In addition, we show the ratio of the annotated corpus (including ground truth while in Table 3.14 we show only the ratio of final predictions).

Self-Learning-based augmented batch annotation

Figure 3.15, demonstrates the performance of the original unmodified data same as in Section 3.6.1. We observe the same behavior as before, e.g., performance is degrading while δ is increased, however for $\delta = 100\%$ performance is maximized while the labeled instances are significantly reduced compared to other thresholds.

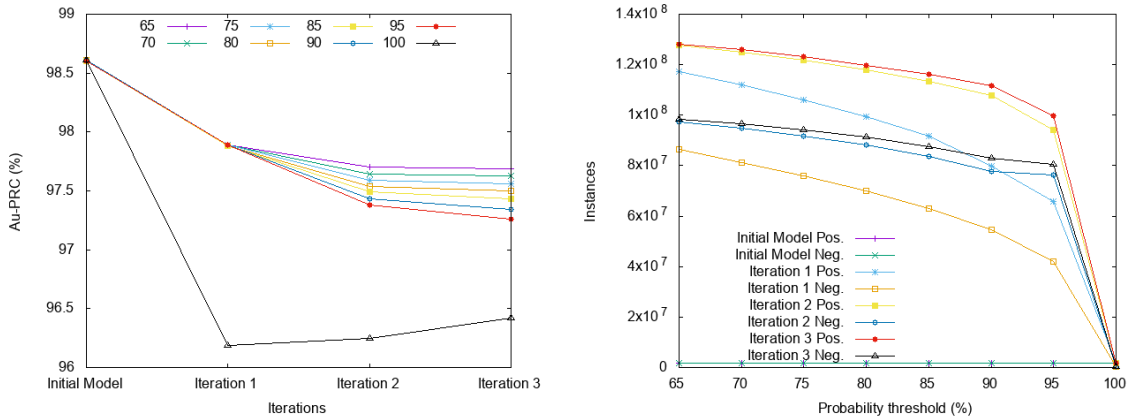
On the other hand, undersampling, oversampling, their combination and blank-out methods have better performance for low δ values (Figures 3.16, 3.17, 3.18 and 3.19). Moreover, by comparing these methods to the original Self-Learning procedure in Figure 3.15, we observe that the performance is better when augmentation is employed. Also, the class imbalance problem is tackled as in each iteration the minority (negative) class receives more instances compared to the original Self-Learning.

However, word embeddings do not perform equally good as the other augmentation methods. In Figure 3.20, we see that word embeddings have same behavior as original Self-Learning. Also, the negative instances are fewer in all the other methods. A probable reason of this behavior could be that the augmentation process generates pseudo-instances of the opposite class. Swapping terms with other semantically similar terms does not guarantee that the sentimentality of the terms will be the same.



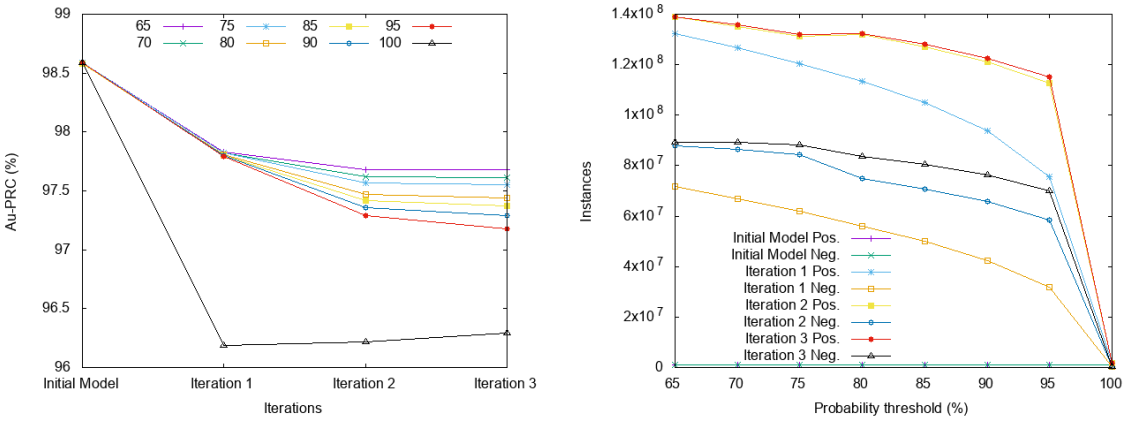
(a) Au-PRC (b) Number of instances per iteration

Figure 3.16: Self-Learning combined with under-sampling



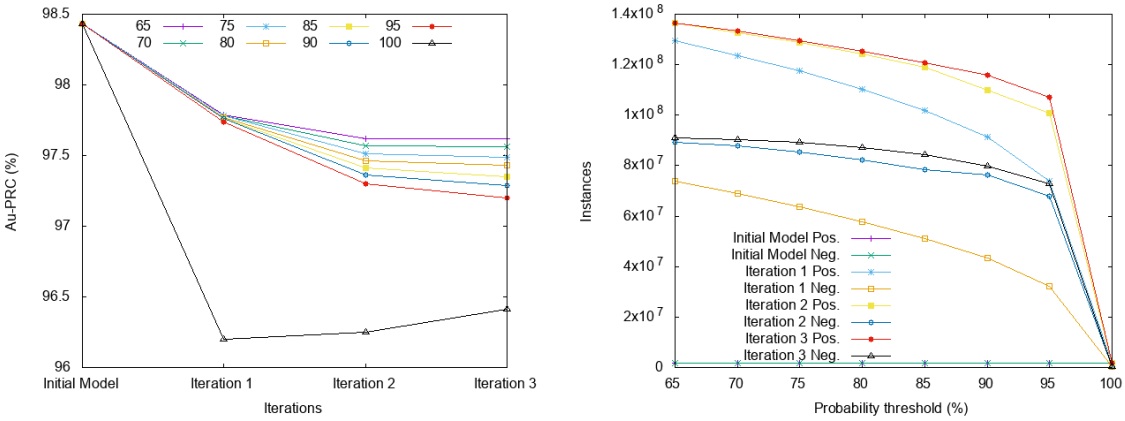
(a) Au-PRC (b) Number of instances per iteration

Figure 3.17: Self-Learning combined with over-sampling



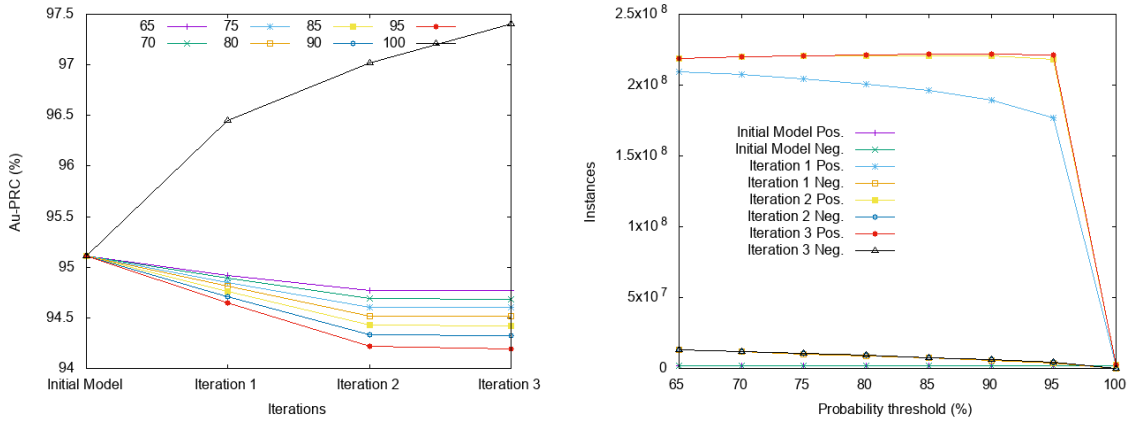
(a) Au-PRC (b) Number of instances per iteration

Figure 3.18: Self-Learning combined with under-sampling and over-sampling



(a) Au-PRC (b) Number of instances per iteration

Figure 3.19: Self-Learning combined with blankout



(a) Au-PRC (b) Number of instances per iteration

Figure 3.20: Self-Learning combined with word embeddings

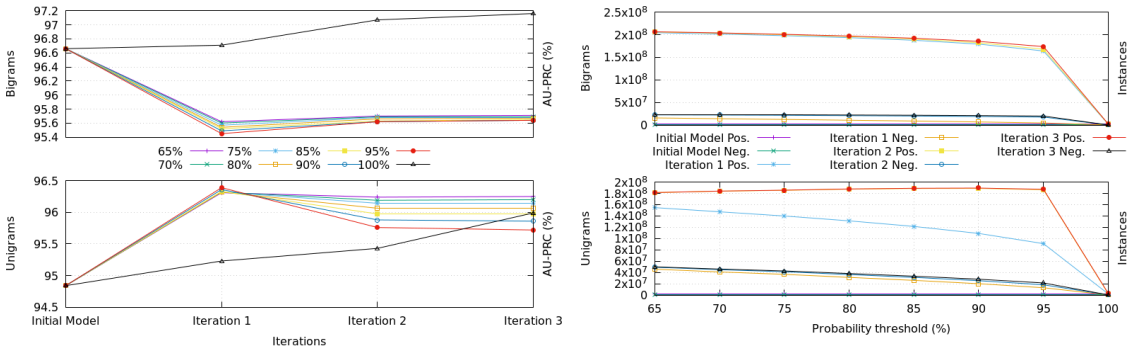
δ	Original	Blankout	Glove	Over.	Under.	Over.& Under.
65%	1:9	1:2	1:16	1:1	1:1	1:2
70%	1:9	1:1	1:18	1:1	1:1	1:2
75%	1:9	1:1	1:21	1:1	1:1	1:2
80%	1:9	1:1	1:24	1:1	1:1	1:2
85%	1:9	1:1	1:29	1:1	1:1	1:2
90%	1:10	1:1	1:36	1:1	1:1	1:2
95%	1:11	1:1	1:52	1:1	1:1	1:2
100%	1:10	1:7	1:11	1:7	1:7	1:7

Table 3.15: Self-Learning: Class Ratio (negative:positive) of the predictions for different methods

Even though we employ SentiWordNet to tackle this problem, it is not enough due to the grammatical and syntactical structure of a tweet, therefore the performance is degrading.

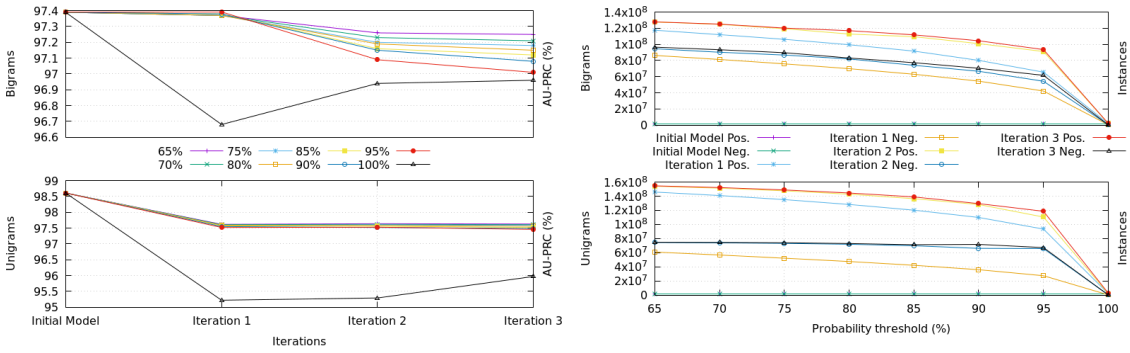
In Table 3.15, we show the ratio of negative, positive predicted instances per method. In contrast to Table 3.14, this table includes the ground truth while the latter shows the overall predicted instances, thus when $\delta = 100\%$ the difference is significantly reduced. Blankout, oversampling (Over.), undersampling (Under.) and the combination of the last two (Over. & Under.) methods tackle the problem of class imbalance. Word embeddings on the other hand, are enhancing the gap between the two classes.

Furthermore, we have performed two significant tests: paired t-test and McNemar's test, to compare original Self-Learning with the other augmentation methods. For every method, we obtained highly significant results in both tests (for $\alpha = 0.01$).



(a) Au-PRC (b) Number of instances per iteration

Figure 3.21: Original Co-Training



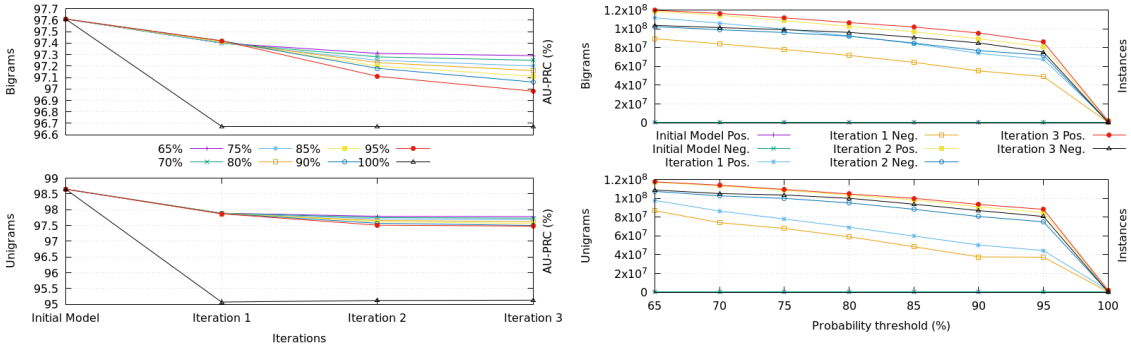
(a) Au-PRC (b) Number of instances per iteration

Figure 3.22: Co-Training combined with oversampling

Co-Training-based augmented batch annotation

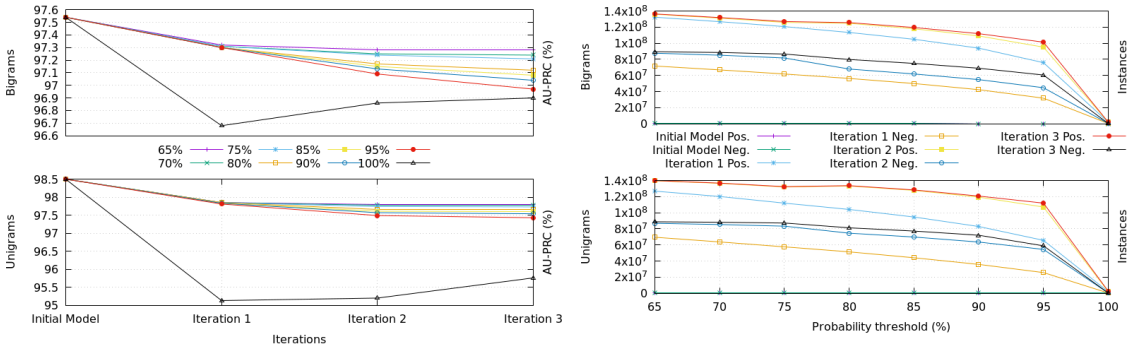
For Co-Training, we report on both models: bigrams and unigrams in Figures 3.21, 3.22, 3.23, 3.24, 3.25 and 3.26. In Figure 3.21, the original Co-Training method is shown for which unigrams are slightly better than bigrams. Nonetheless, by comparing the class labels and performance of original Co-Training with the augmented methods, we observe significant differences. However, same as in Self-Learning the word embeddings do not exhibit good performance compared to the other augmentation methods.

Nonetheless, by comparing Tables 3.16 and 3.17, we see that word embeddings reduce the gap between the two classes in the unigram model while the opposite is happening in the bigrams model. The latter model however is trained based on the predictions of the first model which implies that unigrams as feature space are affected more than bigrams. Other augmentation methods such as oversampling and undersampling deal with class imbalance efficiently by balancing the two classes while maintaining high performance. In addition, same as in Self-Learning we performed the two significant tests, namely McNemar and pair t-test for which we compared the



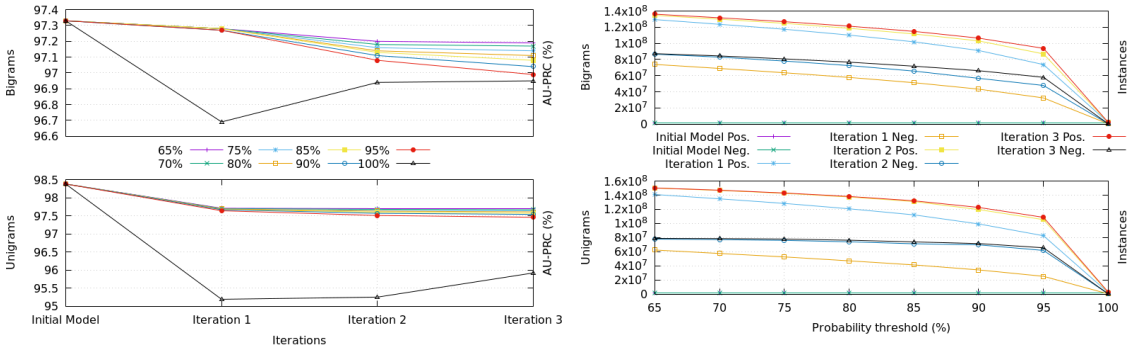
(a) Au-PRC (b) Number of instances per iteration

Figure 3.23: Co-Training combined with undersampling



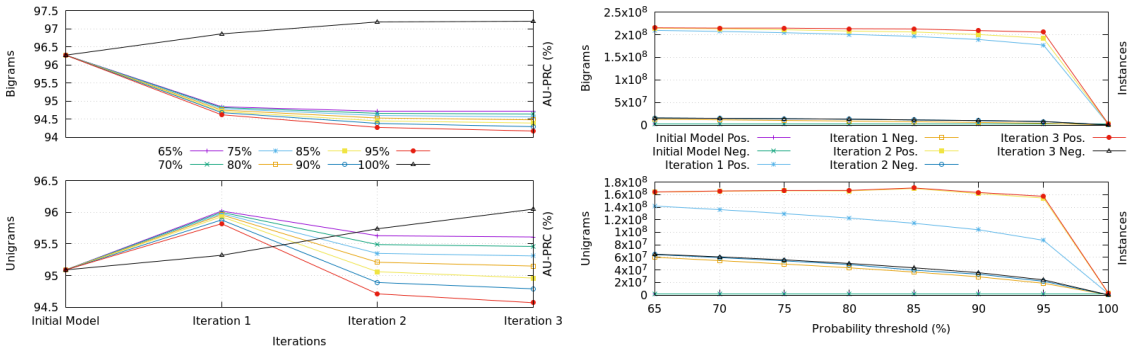
(a) Au-PRC (b) Number of instances per iteration

Figure 3.24: Co-Training combined with over and under sampling



(a) Au-PRC (b) Number of instances per iteration

Figure 3.25: Co-Training combined with blankout



(a) Au-PRC (b) Number of instances per iteration

Figure 3.26: Co-Training combined with embeddings

δ	Original	Blankout	Glove	Over.	Under.	Over.& Under.
65%	1:4	1:2	1:3	1:2	1:1	1:1
70%	1:4	1:2	1:3	1:2	1:1	1:1
75%	1:4	1:2	1:3	1:2	1:1	1:2
80%	1:5	1:2	1:3	1:2	1:1	1:2
85%	1:6	1:2	1:4	1:2	1:1	1:2
90%	1:7	1:2	1:5	1:2	1:1	1:2
95%	1:9	1:2	1:6	1:2	1:1	1:2
100%	1:12	1:10	1:13	1:10	1:7	1:10

Table 3.16: Co-Training Unigrams: Class Ratio (negative:positive) of the predictions for different methods

δ	Original	Blankout	Glove	Over.	Under.	Over.& Under.
65%	1:8	1:1	1:14	1:1	1:1	1:1
70%	1:9	1:2	1:14	1:1	1:1	1:1
75%	1:9	1:2	1:15	1:1	1:1	1:1
80%	1:9	1:2	1:16	1:1	1:1	1:2
85%	1:9	1:2	1:19	1:1	1:1	1:2
90%	1:9	1:2	1:21	1:1	1:1	1:2
95%	1:9	1:2	1:26	1:2	1:1	1:2
100%	1:9	1:8	1:11	1:8	1:7	1:8

Table 3.17: Co-Training Bigrams: Class Ratio (negative:positive) of the predictions for different methods

augmentation methods with the original Co-Training. For all the methods and the δ values we obtained highly significant results (for $\tau = 0.01$).

3.7 Chapter Summary

In this chapter, we presented how to annotate large textual collections with sentiment labels using distant supervision and semi-supervised learning. Our case study is TSentiment15, a 228 million tweets dataset with no retweets and 275 million tweets with retweets, spanning the whole year 2015. The motivation for this work is the lack of large scale labeled datasets that span large periods of time, especially important for stream mining research [SNZ16].

Except for the annotated datasets (with and without retweets) which we make available to the community, our analysis resulted in interesting insights:
Co-Training performs better than Self-Learning with limited labels. Although both Self-Learning and Co-Training benefit from more labeled data, after a certain point (40% labeled data, c.f., Figure 3.8b) Self-Learning improves faster than Co-Training. Both approaches result in more positive predictions (c.f., Tables 3.4 and 3.6), thus favoring the majority class. Self-Learning moreover propagates the original class imbalance to the successive iterations (c.f., Table 3.4). This is not the case for Co-Training (c.f., Table 3.6). Surprisingly the performance of EM does not improve over the iterations, probably due to huge volume of unlabeled data ($|U|$ is almost 91 times larger than $|L|$) affecting the learner. A possible solution would be to select only instances that are labeled with a high probability for some class for the expansion; we leave this as part of future research. However, the predictions of the EM algorithm over the iterations became more balanced.

For streaming, (full) history helps with the performance (c.f., Figures 3.10a and 3.10b). However, comparing to a sliding window approach, a sliding window of three months history (c.f., Figure 3.11) performs almost equally well, while employing fewer data, thus offering a good trade-off. The batch approach is better than streaming in terms of accuracy, however the latter is much more efficient.

In our learning setup we deal, except for the label scarcity problem, also with class imbalance with the positive sentiment class being highly over-represented comparing to the negative class. To tackle the imbalance, we exploited data augmentation, namely semantic augmentation through word-embeddings and corruption, as well as traditional over-sampling and under-sampling techniques. The goal of the augmentation process was to create more training data out of the existing training data by adding variation through domain-meaningful and sound transformation. In both cases, it was our intention to preserve the class labels while keeping the tweets plausible; of course, this is not guaranteed as discussed already in the text and therefore, augmentation might cause further degradation of the data quality. Based on the experiments in Section 3.6.6, we see that augmentation techniques tackle the problem

of class imbalance while maintaining very high performance. Compared to original Self-Learning and Co-Training, methods achieved highly significant difference when equipped with augmentation methods (according to paired test and McNemar test).

In the augmentation direction, we also investigated the impact of dataset redundancy on performance. In particular, Twitter is characterized by high redundancy through retweets. Having such a redundant training set helped class-imbalance as, to some extent, it resembles over-sampling (though in our case, retweets come from both classes). By comparing the two versions (c.f., Tables 3.8 and 3.13), we observe that the retweet version has lower performance compared to the version which does not contain retweets. However, there is a huge difference w.r.t class imbalance between these two versions which can be observed in Table 3.14. We carefully removed duplicates from the test set to ensure that our evaluation is not affected by the redundancy (a similar effect has been studied for recommendation systems in [BNZ17]).

In our future work, we will investigate further data augmentation techniques as a tool for expanding a training set in meaningful ways and tackling problems like class imbalance. In particular, we want to focus on how to identify meaningful augmentations for a given domain and also on what parts of the population should be augmented to avoid noise generation and degradation of data quality.

Moreover, augmentation typically refers to label-preserving transformations; however, in certain applications including text, instances of the "opposite" class could be generated. In text, for example one can turn a positive text into a negative one using negations e.g., like "I like summer" $\rightarrow$ "I dont like summer" or "clever" $\rightarrow$ "stupid". Finally, we will investigate the impact of refined word-embeddings w.r.t the sentiment task [YWLZ17] for our semantic augmentation procedure.

4

Dealing with Class- and Within-Class Imbalance to Prevent Unfair Outcomes

Automated decision making based on big data and machine learning (ML) algorithms can result in discriminatory decisions against certain sensitive groups defined upon personal data like gender, race, sexual orientation, etc. Such algorithms designed to discover patterns in big data might not only pick up any encoded societal biases in the training data but even worse, they might reinforce such biases resulting in more severe discrimination. The majority of thus far proposed fairness-aware machine learning approaches focus solely on the pre-, in- or post-processing steps of the machine learning process to remove the encoded societal biases or prevent their propagation from the induced models.

However, the fairness problem cannot be isolated to a single step of the ML process. Rather, discrimination is often an artefact of complex interactions between big complex data and algorithms, and therefore, a more holistic approach is required. As we show in this chapter, both class- and within-class (rare cases) imbalance have an impact on the discriminatory behavior of a classifier. To this end, we propose FAE (Fairness-Aware Ensemble) framework that combines fairness-related interventions at both pre- and post-processing steps. At the pre-processing step, we tackle the problem of under-representation of the sensitive group, referred to as group imbalance hereafter, as well as the problem of class-imbalance with the target class being the minority class. At the post-processing step, we tackle the problem of class overlap in the feature space by tweaking the decision boundary in the direction of fairness.

4.1 Introduction

Machine Learning powered by big data offers incredible opportunities for effective decision making and automation. However, several recent incidents have raised concerns about the implications of such systems in terms of fairness [otPP14]. Amazon's

models, to name but one example, that decide which regions of a city are eligible for the prime service, excluded predominantly black ZIP codes in several US cities, like Bronx [IS16]. According to Amazon, the *protected attribute* race was not used as a predictor. Nonetheless, there might exist *proxy-attributes* to race which lead to discriminatory decisions. Protected attributes and proxies are not the only causes of the problem [CŽ13]. Training data often reflect *societal biases* and are not representative of the population (*sample bias*). Moreover, *system bias* might lead into generation of biased data which result into biased models that further reinforce such discriminatory policies, like in predictive policing [LI16].

Despite extensive research work in the area of fairness-aware learning, most of the approaches isolate the problem and its solutions to a single step of the ML process, namely, input data, algorithms or resulting models. While, we share the view on the importance of working on the main source of bias, i.e., the training data as pointed out by recent work, e.g., [ZVGRG17, CWV+17, KXPK18], we believe that this in itself is insufficient, and that in- and post-processing adjustments are necessary to deal with discrimination. In addition, previous methods do not consider the ***class-imbalance*** problem i.e., these approaches aim to minimize standard error error while mitigating discriminatory outcomes. However, they end up rejecting more instances from the minority class due to their inability to learn effectively the minority class.

To this end, we propose the Fairness-Aware Ensemble (FAE) framework, a holistic approach that combines pre- and post-processing fairness-enhancing interventions to deal with different bias factors and real-world data complexities, namely group imbalance, class imbalance and class overlap. At pre-processing, we learn an ensemble of ensembles through a combination of bagging and boosting; the bags are carefully selected via stratified cluster sampling to ensure a balanced group- and class-representation, whereas boosting on each bag forces the classifier to focus on the hard-to-classify examples. At post-processing, the decision boundary of the learner is shifted so that the target fairness criterion is fulfilled. Our experiments show that such a joint consideration ensures better fairness- and predictive-performance.

4.2 Related Work

Pre-processing methods aim to tackle discrimination by "correcting" the training data to eliminate any biases. Bias can be inherited from the input data, e.g., there might exist proxies to sensitive attributes, or under-represented groups or biased class labels. Among the most popular methods in this category are class-label swapping, instance re-weighting, sampling, and instance transformation [KC12, IN18, CWV+17]. *In-processing* methods modify the learning algorithm to eliminate discriminatory behavior. These interventions are typically learner-specific [ZVGRG17, KXPK18, KCP10, DIKL18, IN19a]. For instance, Zafar et al. [ZVGRG17] add fairness-related constraints in the objective function of a logistic regression model to account for fairness.

Post-processing methods try to modify the model's predictions or decision boundary in order to ensure fairness [FKL16, KCP10, PRT09a]. Kamiran et al. [KCP10] propose a fair decision tree learner that combines a fairness-aware splitting criterion with post-processing leaf-relabeling. Fish et al. [FKL16] adjust the decision boundary of a boosting model based on the confidence scores of the misclassified instances. Finally, *class-imbalance* methods aim to deal with skewed class distributions. Over the years, many methods have been proposed such as over-sampling [EJJ04], under-sampling [DH+03], synthetic data generation like SMOTE [HBGL08] and boosting [LWZ09].

4.3 Basic concepts

We consider binary classification with $A = \{A_1, \ldots, A_m\}$ being the attribute space and $Y = \{y^+, y^-\}$ the class attribute. Let $dom(A_i)$ be the domain of A_i, and y^+ is the *target class*, for example, "receive a benefit". Let $SA \in A$ be a *protected attribute* with $dom(SA) = \{s, \bar{s}\}$; s is the discriminated group (referred to as *protected group*), and $\bar{s}$ is the non-discriminated group (referred to as *non-protected group*). For instance, $SA =$'*gender*' could be the protected attribute with $s =$'*female*' being the protected group and $\bar{s} =$'*male*' the non-protected. By combining sensitive attribute SA and class Y values, we define four sub-groups: s^-, s^+,$\bar{s}^-$,$\bar{s}^+$; e.g., s^- denotes the protected negative group, $\bar{s}^+$ denotes the non-protected positive group etc. We assume the following learning challenges: *class imbalance*, that is $|s^+| + |\bar{s}^+| \ll |s^-| + |\bar{s}^-|$; *group imbalance*, that is $|s^+| + |s^-| \ll |\bar{s} + | + |\bar{s}^-|$ as well as class overlap, i.e, the positive class y^+ overlaps with the negative class y^-.

The goal of a fairness-aware classifier is to learn a function $f(\cdot) : dom(A_i) \times \cdots \times dom(A_m) \rightarrow Y$, s.t. $f(\cdot)$ can generalize well to unseen instances and does not discriminate against the protected group for the target class y^+.

Discrimination measure: We adopt the *equal opportunity measure* (EQOP) [HPS+16] that compares the probability of being predicted as positive while belonging to the positive class (TPR) between protected s and non-protected $\bar{s}$ groups:

$$EQOP : P\Big(f(d) = y^+ | \bar{s}^+\Big) - P\Big(f(d) = y^+ | s^+\Big) \tag{4.1}$$

$EQOP \in [-1, 1]$: a value close to 0 means *fair outcomes*, and is desirable, whereas a value close to 1 indicates *discriminatory* behavior towards the protected group. A value close to -1 indicates *reverse discrimination* towards the non-protected group. A classifier $f(\cdot)$ is said to *not discriminate* if: $\mid EQOP \mid \leq \epsilon$. The user-defined threshold ϵ controls how much discrepancy between the two groups is tolerated.

Predictive performance measure: The vast majority of existing works minimize the standard error rate, e.g., [ZVGRG17, KC12, CWV+17, KCP10, FKL16], which is not useful in case of *class-imbalance* as it mainly reflects the performance of

the model in the majority class. Moreover, EQOP measure, (c.f., Equation4.1) which relies on the TPR difference, is oblivious to the problem of class imbalance. As an extreme case, if a classifier totally rejects the minority (positive) class and correctly classifies the majority (negative) class then, based on EQOP, the classifier is both fair (in terms of EQOP) and accurate (in terms of error rate). Recent methods fall in this pitfall and their low reported discrimination scores are mainly due to low TPR values (c.f., Section 4.6). Hence, we use balanced accuracy [BOSB10]:

$$B.ACC = \frac{1}{2} \cdot \left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right) = \frac{(TPR+TNR)}{2} \tag{4.2}$$

Our approach resembles the EasyEnsemble approach [LWZ09], which we adapt for group as well as class imbalance. Specifically, we combine *bagging* and *boosting*; thus, the final model is an ensemble of ensembles. *Bagging* reduces model variance by generating multiple models from bootstrap samples drawn from the training data. *Boosting* reduces both (model) bias and variance by combining many weak learners, each focusing on missclassified examples from previous learners [Sch99].

4.4 FAE - A Fairness-Aware Ensemble Framework

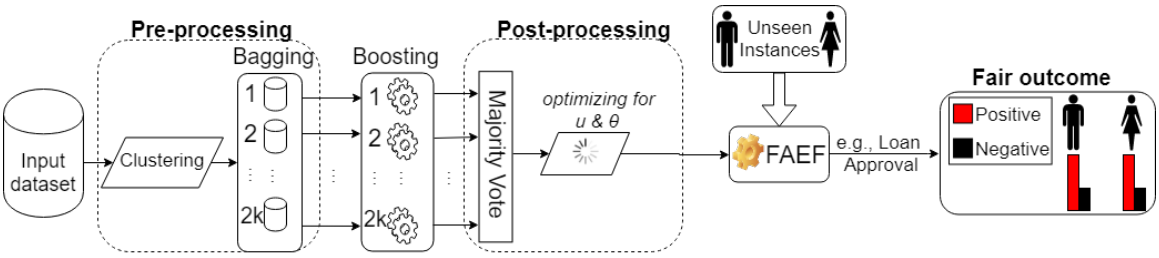


Figure 4.1: An overview of our holistic pre- and post-processing FAE framework

Figure 4.1 shows an overview of FAE, from training (left side) to prediction of new instances (right side). FAE combines pre- and post-processing fairness-related interventions, as follows:

- **Fairness-aware ensemble learning**
  In *pre-processing*, we tackle the problems of group- and class-imbalance. In particular, we employ *bagging* to *balance* the groups in each bag by taking into account the protected positive group, and a representative sample from the other groups (Section 4.4.1). Afterwards, *boosting* [Sch99] is employed on each bag, so at the end, an ensemble of ensembles is learned.

- **Fairness-aware decision boundary shift**
  In the *post-processing*, we shift the decision boundary of the learner in the

direction of fairness based on a tunable parameter θ, until the $EQOP$ score satisfies the user-defined threshold ϵ (Section 4.4.2).

- **Selecting the shortest hypothesis** Finally, we select the optimal number of boosting models $u \in [k, 2k]$ that exhibits the best performance in terms of both fairness and balanced error (Section 4.4.3).

4.4.1 Learning equal representations

In the pre-processing step, we tackle discrimination in the training data caused by group and class imbalance ensuring that the protected positive group will also be learned by the model. For that, we propose a fair and representative sample generation process. Each sample is created s.t it contains the whole *protected positive group* s^+ and a *representative equisized* sample from each of the other groups (i.e., from $s^-, \bar{s}^+, \bar{s}^-$).

Algorithm 5 shows the different steps in the ensemble's training phase. Clustering is applied in the beginning for each group $s^-, \bar{s}^+, \bar{s}^-$ (line 2). We employ *stratified sampling* to ensure a balanced representation, where the *strata* correspond to clusters[1] extracted through some clustering algorithm from the other groups $s^-, \bar{s}^+, \bar{s}^-$. The bags are created (lines 6-7) by combining s^+ and a stratified sample from the generated clusters for each group. In each bag, an AdaBoost classifier is trained (line 8) and added to the ensemble (line 9). The output model is an ensemble of ensembles E (line 12):

$$E(x) = \sum_{i=1}^{2k} \left(\sum_{j=1}^{z} \left(a_{i,j} h_{i,j}(x) \right) \right) \tag{4.3}$$

where k is the number of bags (c.f., Eq. 4.4), z the number of boosting rounds and $a_{i,j}$ is the weight of the weak learner $h_{i,j}$ (a and h are obtained through AdaBoost).

Stratified sampling

The goal is to generate the different bags $s^{-'}, \bar{s}^{+'}, \bar{s}^{-'}$ from the majority groups $s^-, \bar{s}^+, \bar{s}^-$, respectively, such that: $|s^+| = |s^{-'}| = |\bar{s}^{+'}| = |\bar{s}^{-'}|$. To ensure representative samples from each group, we cluster each group (i.e., each of $s^-, \bar{s}^+, \bar{s}^-$) and use the resulting clusters for bag generation. Note that clusters are generated only once in the beginning of the training process (line 2, Algorithm 5) and re-used afterwards.

[1] Clustering better approximates the underlying data distributions, accounting for sub-groups, and thus ensuring representative samples from each group.

Algorithm 5: Pre-processing step

Input: Training set D, target class y^+, SA, k
Output: Ensemble E

Extract groups $s^+, s^-, \bar{s}^+, \bar{s}^-$ based on y^+ and SA from D;
Generate clusterings $C_{s^-}, C_{\bar{s}^+}, C_{\bar{s}^-}$ from $s^-, \bar{s}^+, \bar{s}^-$, respectively;
Ensemble $E \leftarrow \{\emptyset\}$;
$i \leftarrow 1$;
For $i = 1 : 2k$ **do;**
    Stratified sample $s_i^{-'}, \bar{s}_i^{+'}, \bar{s}_i^{-'}$ from $C_{s^-}, C_{\bar{s}^+}, C_{\bar{s}^-}$;
    Bag $B_i = s^+ \cup s_i^{-'} \cup \bar{s}_i^{+'} \cup \bar{s}_i^{-'}$;
    Train an AdaBoost classifier H_i upon B_i;
    $E \leftarrow E \cup H_i$;
    $i \leftarrow i + 1$;
EndFor
return ensemble E;

Estimating the initial number of bags

The number of bags k must be sufficient to overcome the drawback of potential loss of useful information due to under-sampling (i.e., each bag is a sample of the training data). We overcome this drawback by estimating the number of bags k s.t. we insure that the clustered instances are at least in one of the bags. We calculate the number of bags k as following:

$$k = \left\lceil \frac{max\{|s^-|, |\bar{s}^+|, |\bar{s}^-|\}}{|s^+|} \right\rceil + 1 \tag{4.4}$$

In other words, k provides an estimation that an instance from the most populated group will be at least in one bag, thus, avoiding the under-sampling drawback. In practice, we train the ensemble with twice the amount of bags ($2k$ bags); at the post-processing step, we select the best set of learners for the ensemble (Section 4.4.3).

4.4.2 Fairness-aware decision boundary tuning

Despite the pre-processing interventions, the resulting model E might not fulfill the discrimination threshold ϵ. In FAE, if EQOP $> \epsilon$, a post-processing procedure is invoked that shifts the decision boundary based on a parameter θ s.t. EQOP $\leq \epsilon$.

As we show in Section 4.6, by employing only the pre-processing step, the discrimination is significantly reduced. However, a post-processing step is necessary given that discrimination can stem from other factors including class overlap and the accuracy-oriented objective function of Adaboost.

Parameter tuning. For a SA (e.g. SA ='*gender*') our goal is to find the optimal threshold parameter θ_s or $\theta_{\bar{s}}$ (for the different attribute values $dom(SA) = \{s, \bar{s}\}$) to minimize $EQOP$. Furthermore, at any given time our ensemble learner E can discriminate against only one of the group s or $\bar{s}$.

Algorithm 6 shows the detailed steps for tuning the optimal θ_s and $\theta_{\bar{s}}$. To begin with, we compute the $EQOP$ score, which represents the difference between true positive ratios between s and $\bar{s}$ (line 6). Next, we sort the misclassified instances from s^+ and $\bar{s}^+$ groups (lines 7 – 8) in a descending order (w.r.t the target class) based on their ensemble classification score from Equation 4.3. In case $EQOP$ score is below the discrimination threshold ϵ, then $\theta_{\bar{s}} = \theta_s = 0.5$ (lines 9 – 10). Setting the threshold parameter to 0.5 has no implication in classifying test instances in Equation 4.7. For $|EQOP| > \epsilon$, we distinguish between *discrimination* and *reverse discrimination* (lines 11 – 17). That is, for $EQOP > 0$ the model discriminates against instances with $SA = s$, otherwise against instances with $SA = \bar{s}$. The threshold parameter θ_s or $\theta_{\bar{s}}$ represents the $E(d)$ score of the last instance from the top_k necessary instances from MC_{s^+} or $MC_{\bar{s}^+}$ (lines 12 and 15) that need to be classified correctly to fulfill the criteria $|EQOP| \leq \epsilon$. The top_k instances needed for minimizing the discrimination are obtained as following:

$$\frac{top_k + TP_s}{TP_s + FN_s} = \frac{TP_{\bar{s}}}{TP_{\bar{s}} + FN_{\bar{s}}} \Rightarrow top_k = \left\lceil \frac{TP_{\bar{s}}(TP_s + FN_s)}{TP_{\bar{s}} + FN_{\bar{s}}} - TP_s \right\rceil \tag{4.5}$$

where TP and FN stand for true positive and false negative instances of protected and non-protected group respectively.

4.4.3 Hypothesis selection for class-imbalance

Out of the $2k$ learners, we select the shortest hypothesis (in terms of number of bags) that optimizes the following objective function:

$$\underset{u}{\text{argmin}} \ (B.ERR_u + 2 \cdot |EQOP_u|) \tag{4.6}$$

where $B.ERR$ is the balanced error rate and $u \in [k, 2k]$ is a set of AdaBoost models (each AdaBoost is trained upon a different bag). The objective function is applied after the decision boundary adjustment i.e., Algorithm 6 is taking place after the pre-processing step, and afterwards the set of learners that minimize Equation 4.6 is selected. Since class imbalance is tackled in the pre-processing step, more emphasis is given to the ensemble's fairness in the objective function. The final model (FAE) is:

$$E(x) = \sum_{i=1}^{u} \Big(\sum_{j=1}^{z} \Big(a_{i,j} h_{i,j}(x) \Big) \Big)$$

.

Algorithm 6: Post-processing step

Input: D, E, $s, \bar{s}$, ϵ
Output: $\theta_s, \theta_{\bar{s}}$

$\theta_s = \theta_{\bar{s}} = 0.5$
MC_{s^+}, $MC_{\bar{s}^+} \leftarrow \{\emptyset\}$
True positive rate TPR_s and $TPR_{\bar{s}}$ for s and $\bar{s}$
CC_{s^+} = #correctly classified instances in s^+
$CC_{\bar{s}^+}$ = #correctly classified instances in $\bar{s}^+$
$EQOP = TPR_{\bar{s}} - TPR_s$
Misclassified instances MC_{s^+} and $MC_{\bar{s}^+}$ for s^+ and $\bar{s}^+$
Sort MC_{s^+}, $MC_{\bar{s}^+}$ in descending order based on $E(d)$
IF $|EQOP| \leq \epsilon$ // no discrimination
    $\theta_s = \theta_{\bar{s}} = 0.5$
ELSE IF $EQOP > 0$ // discrimination
    $top_k = \frac{CC_{\bar{s}^+}}{|\bar{s}^+|}|s^+| - CC_{s^+}$
    $\theta_{\bar{s}} = MC_{s^+}[top_k]$
ELSE IF $EQOP < 0$ // reverse discrimination
    $top_k = \frac{CC_{s^+}}{|s^+|}|\bar{s}^+| - CC_{\bar{s}^+}$
    $\theta_s = MC_{\bar{s}^+}[top_k]$
ENDIF
return $\theta_s, \theta_{\bar{s}}$

4.4.4 FAE Classification

In classifying instances with *FAE*, we distinguish two cases. If $|EQOP| \leq \epsilon$, the classification is done solely through the majority voting scheme in $E(d)$ (c.f., Equation 4.3). This is the case, where no post-processing tuning is required, rather pre-processing interventions are adequate in fulfilling the $EQOP$ threshold. For $EQOP < 0$ and $|EQOP| > \epsilon$, our model discriminates against $SA = \bar{s}$ in the training set, hence, instances will be classified based on Equation 4.7.

$$f(d) = \begin{cases} y^+ & \text{if d(SA)} = \bar{s} \text{ and } E^+(d) \geq \theta_{\bar{s}} \\ \text{E(d)} & \text{otherwise.} \end{cases} \tag{4.7}$$

where E^+ is the probability of d assigned to y^+. Similar is the case for $EQOP > 0$ and $|EQOP| > \epsilon$; in this case, Equation 4.7 is altered by replacing $d(SA) = \bar{s}$ to $d(SA) = s$ and $\theta_{\bar{s}}$ to θ_s.

4.5 Experimental Setup

Our framework[2] has been instantiated with Logistic Regression as base learners. Each dataset is randomly split into train (2/3) and test set (1/3) (holdout evaluation, similar to [ZVGRG17]). We report on the average of 10 random splits. We set $\epsilon = 0$ as a threshold for EQOP (no discrimination). For AdaBoost, the maximum number of boosting rounds z is set to 25. We evaluate the following aspects: (i) classification performance based on balanced accuracy (B.ACC, Equation 4.2) and (ii) discriminative performance based on *EQOP* (Equation 4.1) .

4.5.1 Datasets

We evaluate our approach with two well known datasets: Adult census income and Bank. **Adult census income** dataset [BL13] contains demographic data from the U.S. The task is to determine if a person receives more than 50K dollars annually. We use as the target (positive) class, people who receive more than 50K per year. We remove duplicate instances and instances containing missing values which results to 45,175 instances. The class ratio of adult census dataset is 1 : 3 (positive:negative). We consider as protected attribute $SA = Gender$ with $s = female$. **Bank** dataset [BL13] is related to direct marketing campaigns of a Portuguese banking institution and contains 40,004 instances. The task is to determine if a person subscribes to the product (bank term deposit). As target (positive) class we consider people who subscribed to a term deposit. The class ratio of bank dataset is 1 : 2 (positive:negative). We consider as $SA = maritial\ status$ with $s = married$.

4.5.2 Baselines and FAE Ablations

Baselines

Shifted Decision Boundary (SDB) [FKL16]: SDB uses a set of base classifiers in an AdaBoost classifier. Instead of majority voting (i.e., $\sum_{i=1}^{T} a_i h_i(x)$), SDB employs confidence scores (i.e., $\frac{\sum_{i=1}^{T} a_i h_i(x)}{\sum_{i=1}^{T} a_i}$) for predictions. The best threshold value for a specific protected group is established to minimize statistical parity. The shift in the boundary takes place after the training phase, thus, making it a post-processing method and suitable for comparison. To have a fair comparison, we find the best threshold estimation of SDB for EQOP, instead of statistical parity as in the original paper.

Disparate Mistreatment (DM): Zafar et al. [ZVGRG17] formulate the fairness problem as a set of constraints, for which they optimize a logistic regression (LR)

[2] https://iosifidisvasileios.github.io/Fairness-Aware-Ensemble-Framework/

model. They consider three sets of constrains: (i) minimize difference in FPR (false positive rate), (ii) minimize difference in FNR (false negative rate), and (iii) a combination of both. For our comparison, we employ only (ii) since $TPR = 1 - FNR$. We employ the method's default parameters.

AdaBoost: here we consider an ensemble learner (equipped with LR as a weak learner) without any pre- or post-processing fairness-related interventions. The goal is to show the ability of these ensembles to classify under group and class imbalance and its impact on discrimination scores like EQOP.

EasyEnsemble: EasyEnsemble [LWZ09] is an ensemble that employs bagging and AdaBoost to tackle class imbalance, with LR as a weak learner. We employ EasyEnsemble to compare our approach with a method that directly tackles class imbalance. We set as number of bags to $N = 20$.

FAE Model Ablation

FAE is a joint framework of pre-and post-processing interventions. We consider the following ablations, to evaluate the individual effect of the pre- and post-processing interventions:

Only Bagging (OB) is the pre-processing step in FAE (c.f. Section 4.4.1). We use OB to show the behavior of the ensemble that is trained upon fair and representative groups, without further tuning its decision boundary.

Simple Majority Threshold (SMT) refers to the post-processing part in FAE (c.f. Section 4.4.2). This method is similar to SDB [FKL16], however, instead of using confidence scores, we use the default majority vote of an AdaBoost classifier. That is, after training, we compute the best parameter θ for a specific protected group to minimize EQOP (Algorithm 6). We use SMT to show how individual post-processing tuning affects the performance of the models.

We use EM and K-means clustering algorithms to compare the impact of clustering in the bagging step in FAE and its pre-processing step OB, which we indicate with FAE (EM) and OB (EM), and FAE (K-means) and OB (K-means), respectively. For EM, the optimal number of clusters for each group is estimated via cross validation (100 iterations) while for K-means we use the elbow metric (least squares), where the number of clusters ranges in $[2, 25]$.

4.6 Evaluation Results and Discussion

We report on: (i) *classification performance* w.r.t *B.ACC* and (ii) *fairness performance* w.r.t. EQOP.

Table 4.1 shows the scores for the B.ACC metric for both datasets and approaches

	Adult Cen.		*Bank*	
Approach	**B.ACC. (%)**	**EQOP (%)**	**B.ACC. (%)**	**EQOP (%)**
AdaBoost	76.56	11.92	66.32	-6.25
EasyEnsemble	80.58	15.72	83.24	-4.52
DM	70.96	-11.83	65.69	-0.97
SDB	77.02	-2.72	66.23	-5.88
SMT	76.86	-2.99	73.26	30.58
OB (EM)	80.91	-4.31	83.10	2.21
OB (K-means)	80.92	-4.70	83.10	1.89
FAE (EM)	**81.09**	**1.52**	**83.29**	**-0.12**
FAE (K-means)	81.01	1.67	83.24	0.24

Table 4.1: Evaluation results for $B.ACC.$ and $EQOP$. EQOP is in the range of [-1,1], in this case we show the percentage points. The best results are marked in boldface.

under comparison. Our approach FAE achieves some of the highest $B.ACC$ scores, with an average score of $\overline{B.ACC} = 82.19\%$ across all datasets for FAE (EM). Similar is the score of EasyEnsemble with $\overline{B.ACC} = 81.91\%$. Yet, in terms of EQOP EasyEnsemble produces highly discriminatory results, since it focuses solely on predictive performance.

A detailed inspection across the competing approaches reveals that the differences between non-bagging and non-ensemble approaches are highly significant. An even representation of all groups is important for classification performance. For models like AdaBoost, SMT, SDB, DM that do not account for the group imbalance, we see a huge drop in B.ACC scores. FAE (EM) has a 20% relative increase when compared against DM, and 15% relative increase against the other models.

Ensemble Learners: The case of AdaBoost shows that using solely ensemble learners is not sufficient to ensure a non-discriminatory classification. It has the second lowest performance with $\overline{B.ACC} = 71.44\%$. EasyEnsemble which focus on class imbalance has very good predictive performance with $\overline{B.ACC} = 81.91\%$; however, this is not sufficient to tackle discrimination. Same behavior can be observed for OB. This confirms our assumption, that such discriminatory behaviors are a result of other factors such as class overlap.

Bagging: Bagging ensures even representations of the different groups, thus, it enables models that achieve better B.ACC. Models that employ bagging achieve similar B.ACC scores. Comparing against other non-bagging approaches, such as AdaBoost, DM and SMT, we note a significant drop in terms of B.ACC. However, it is important to note that a high B.ACC score is not sufficient for non-discriminatory classification behavior because, discrimination is often manifested in terms of uneven probabilities for granting a benefit to different groups (c.f. Section 4.3).

Regarding discrimination, we observe that high B.ACC scores do not necessarily correlate with low EQOP scores, that is, discrimination free classification behavior. In our choice of competitors, it is evident that such strategies are often insufficient in minimize discrimination.

From the competitors, only AdaBoost and EasyEnsemble have low EQOP scores. EasyEnsemble is particularly interesting; its B.ACC score is on average close to FAE (EM), however, it exhibits a high discrimination score with $\overline{EQOP} = 10.12\%$. This highlights that optimizing only for classification performance is subject to pitfalls of uneven distributions of groups. Whereas our models, the pre-processing stage OB, and FAE, achieve the lowest discrimination results. FAE (EM) has the lowest score with $\overline{EQOP} = 0.82\%$ with nearly an ideal EQOP score.

Contrary, for models that optimize for discrimination free classification, we note a significant decrease of EQOP scores compared to AdaBoost and EasyEnsemble. For example, DM in its optimization function minimizes for the EQOP score, leading to $\overline{EQOP} = 8.18\%$. Yet, its B.ACC score is severely impacted. This is mostly due to the fact that it learns a logistic regression model under high group imbalance.

An important comparison is between FAE and DM. FAE provides a high relative decrease of 90% in terms of EQOP. This shows, that despite the fact that DM optimizes the training objective to reduce discrimination, the impact of fair and balanced representations of all groups in training supervised models is highly important.

4.7 Chapter Summary

In this chapter, we addressed the problem of discrimination against marginal groups in classification models caused by group imbalance, class imbalance and societal encoded biases manifested as class overlap esp. for the protected group. We presented the FAE framework, a holistic approach to fairness-aware classification that combines pre-processing balancing strategies with post-processing decision boundary adjustment. The pre-processing stage, which computes the number of bags and determines the different groups and clusters to ensure fair representation allows the models to learn representative classifiers that significantly increase the performance and at the same time reduce the discrimination. Due to the encoded societal biases (*class overlap*) in the data, even representations among groups are insufficient in addressing discrimination. Hence, we shift the decision boundary and additionally select hypotheses from the ensemble learners for nearly ideal EQOP scores. Such steps ensure that a reduction in terms of EQOP does not come at the cost of the ability of the model to correctly classify instances into their corresponding classes.

Our experiments show that discrimination free models are feasible, and for a given *feature space*, we can achieve maximal classification performance, and account for important factors like discrimination for a given target measure, e.g., EQOP. In our current version of FAE, we employ pre- and post-processing fairness-enhancing interventions. Furthermore, improvements are possible by including in-processing interventions at the algorithm level, thus targeting the whole ML process from data to algorithms and models.

5

Tackling Class-Imbalance and Unfair Outcomes in Boosting

The widespread use of ML-based decision making in domains with high societal impact such as recidivism, job hiring, and loan credit has raised a lot of concerns regarding potential discrimination. In particular, in certain cases, it has been observed that ML algorithms can provide different decisions based on sensitive attributes such as gender or race and, therefore, can lead to discrimination. Although, several fairness-aware ML approaches have been proposed, their focus has been largely on preserving the *overall* classification accuracy while improving fairness in predictions for both protected and non-protected groups (defined based on the sensitive attribute(s)). The overall accuracy however is not a good indicator of performance in case of class imbalance, as it is biased towards the majority class. As we will see in our experiments, many of the fairness-related datasets suffer from class imbalance, and therefore, tackling fairness requires also tackling the imbalance problem.

In this chapter, we propose AdaFair, a fairness-aware classifier based on AdaBoost that further updates the weights of the instances in each boosting round taking into account a cumulative notion of fairness based upon all current ensemble members, while explicitly tackling class-imbalance by optimizing the number of ensemble members for balanced classification error.

5.1 Introduction

AI-based decision making systems are employed nowadays in an ever growing number of application areas such as finance industry, autonomous driving, health-care, risk assessment, etc. The impressive performance of such systems - in several tasks, has reached or even outperformed human performance - leads to anticipation about automating human tasks by AI in the foreseeable future [GSD+18]. Such a demand for automation is also driven by the ever growing amount and complexity of data.

However, in parallel to the growing demand for automation using AI, an increased number of concerns regarding accountability, fairness and transparency of such systems, esp. for domains of high societal impact, has been raised over the recent years [otPP14]. There is already a plethora of observed incidents regarding discrimination caused by AI-based decision making systems [IS16, DTD15, EL14, Swe13, LMKA16]. In Amazon's Prime case [IS16], for example, in which AI algorithm's task was to decide which areas of a city are eligible to advanced services, areas mostly inhabited by black people were ignored (*racial-bias*), even though the algorithm did not consider race as a feature. In another case, it was discovered that Google's AdFisher tool displayed significantly more advertisements of highly paid jobs to men than women [DTD15] (*gender-bias*). Such incidents point out the urge to consider fairness in AI algorithms in order to exploit the amazing potential of the technology for societal advance.

A growing body of works has been proposed over the last year to address the problem of fairness and algorithmic discrimination. Such methods can be broadly categorized into pre-, in- and post-processing approaches based on whether they "correct" for fairness at the input training data, at the algorithm itself or at the output model. To the best of our knowledge, the vast majority of these methods, e.g., [KXPK18, ZVGRG17, CKP09, CWV+17, KC12, HPS+16, FKL16, KC09, KCP10], focus on optimizing for fairness while maintaining an overall high classification performance. Considering overall classification error is problematic in case of class-imbalance as the performance of the model on the minority class is typically ignored and therefore, such methods end up with low TPR and high TNR rates. Such approaches might achieve fairness, as typically fairness is evaluated in terms of performance parity between the protected and non-protected groups, however the predictive performance of the model on the minority group is insufficient (low TPRs). Recent state-of-the-art approaches do not consider this problem, as we show in Section 5.5; these methods achieve fairness by lowering TPRs for both groups.

Class-imbalance is an inherited problem of fairness - as we will see in our experiments, many of the datasets exhibit high class imbalance (c.f., Table 5.1) and therefore, tackling fairness requires also tackling imbalance[GFB+12]. Our proposed approach, AdaFair , overcomes this issue and achieves fairness while preserving high TPRs and high TNRs for both groups. AdaFair is based on AdaBoost and extends its instance weighting strategy for each round based on the thus far observed ensemble fairness. This way, in each round, the weak learner focuses on both hard classification examples (as in traditional boosting) and on the discriminated group per class. The discriminated group, per class, is identified dynamically at each boosting round and its effect on the instance weighting is evaluated based on a cumulative notion of fairness that considers the fairness behavior of the thus far built ensemble model in the particular round. We further prioritize the instances for training, by taking into account not only the error of the weak learner but also its confidence in the predictions per instance so that harder instances that lie further away from the boundary, are

further boosted. Finally, at the end of the training phase, we select the best sequence of weak learners which achieves high performance and fairness.

Our contributions are summarized as follows: i) we propose AdaFair, a fairness-aware boosting method that achieves parity between the two groups (thus achieving fairness) while maintaining high TPRs and TNRs (thus tackling class imbalance). ii) We define the notion of cumulative fairness for the ensemble and a dynamic group weighting schema for the per round discriminated group. iii) AdaFair outperforms current state-of-the-art methods in terms of performance in a value range from 8% to 25% (depending on the severity of class-imbalance). iv) We show the superiority of our cumulative notion of fairness vs a non-cumulative alternative. v) We show that including confidence in the weighting, results in more confident predictions for AdaFair.

The rest of the chapter is organized as follows: in Section 2, we review the related work. In Section 3, we define the problem and describe basic concepts. Our AdaFair is presented in Section 4, whereas the experimental evaluation is described in Section 5. Conclusions and outlook are summarized in Section 6.

5.2 Related Work

Fairness notions. A growing body of fairness notions have been proposed over the recent years. Up to now, more than twenty fairness notions exist for fairness in classification [RR14, VR18], however there is no specific arrangement on which fairness notion is universally suitable. One of the earliest measures of discrimination, the so-called *statistical* or *demographic parity* [KC09, KC12], measures the percentage difference between non-protected and protected group which are assigned to a target class i.e., grant a loan application, etc. However, this definition only requires a balanced representation of both groups to a target class without controlling if the selected instances are qualified or not [DHP+12]. On the other hand, a recent measure, called equal opportunity [HPS+16, VR18], eliviates this pitfall by measuring percentage difference between true positive percentages of two groups. Although equal opportunity has potential, it does not capture the full picture therefore *equalized odds* [HPS+16] (also called *disparate mistreatment* [ZVGRG17]) has been proposed. Equalized odds extends equal opportunity by considering the difference of true classified instances between protected and non-protected group in all classes. Although a plethora of fairness notions exist, in this work we employ equalized odds since it seems the most promising notion and also has been adopted by recent state-of-the-art methods [ZVGRG17, HPS+16, KXPK18, PRW+17].

Pre-processing approaches. One of the most common causes of machine learning discrimination is arising from discrimination which resides in historical data. Pre-processing methods aim to deal with this issue by performing data transformations, perturbations or augmentation to eliminate underlying discrimination. These

methods are typically model-agnostic and therefore, any classifier is applicable after the pre-processing phase. *Massaging* [KC09], *re-weighting* [CKP09], *uniform- or preferential-sampling* [KC12] and *data augmentation* [IN18] are pre-processing methods which change data distributions directly; they aim to restore balance among protected and non-protected group within data by changing instance labels, assigning different weights, under- or over-sampling instances, and generating pseudo-instances, respectively. *Massaging* and *re-weighting* have also been extended for stream classification [ITN19]. Another interesting approach by [CWV+17], performs data transformations to eliminate existing dependence of instances' labels.

In-processing approaches. Approaches which operate during learner's training phase aim to mitigate discrimination as part of the objective function by employing set of constraints or using regularization. In [KCP10, ZN19], for example, they embed the statistical parity fairness notion into the splitting criterion of a decision tree. A regularization approach [KAAS12] scales down the correlation (mutual information) of the sensitive features and the class label in order to avoid outcomes based on these features. [DHP+12] introduced the notion of "individual fairness-constraint" where similar instances should be treated similarly and then minimize a loss function subject to this notion. In [ZVGRG17], they insert a set of convex-concave constraints, which aim to minimize equalized odds, into logistic regression classifiers. Finally, [KXPK18] assume that data do not contain robust and fair labels and proceed to estimate the true labels by re-adjusting iteratively the instance weights while minimizing equalized odds.

Post-processing approaches. Post-processing approaches can be divided into two sub categories: the ones that change the decision boundary of a model (white-box approaches) and the ones that directly change the prediction labels (black-box approaches). In the first category, for example [FKL16] shift the decision boundary of AdaBoost to minimize discrimination while [PRT09b] alter the confidence of CPAR classification rules and in [CV10] authors change Naïve Bayes probabilities w.r.t. fairness. Approaches in the latter category consider only the outcome of a classifier. For example, [HPS+16] set thresholds to predictions in order to achieve same error rates for protected and non-protected group. An extension of this work [PRW+17] analyze how to obtain calibrated classifiers under same error rates among groups. Finally, in [ABD+18] they build a fair classifier out of the predictions of another black-box classifier.

Since our approach can be considered as in-processing approach, we select methods from this category as competitors. We do not consider [KCP10, KAAS12, DHP+12] as competitors, while they are built upon fairness notions which seem incomplete or outdated and alternating them to optimize for another fairness notion may degrade their performance. Therefore, we select [ZVGRG17, KXPK18] as our competitors while equalized odds is our common fairness measure. In contrast to our approach, our competitors only consider error rate in their loss function which fails to tackle the problem of class-imbalance.

5.3 Basic concepts and definitions

We assume a dataset D consisting of n i.i.d. samples drawn from a joint distribution $P(F, S, y)$: S denotes *sensitive attributes* such as gender and race, F denotes other *non-sensitive attributes* and y is the class label. For simplicity, we consider that the classification problem is binary, that is, $y \in \{+, -\}$ and that there exist a single sensitive attribute S, also binary. That is, $S \in \{s, \bar{s}\}$ with s and $\bar{s}$ denoting the *protected* and *non-protected group*, respectively. We use the notation s_+ (s_-), $\bar{s}_+$ ($\bar{s}_-$) to denote the protected and non-protected group for the positive (negative, respectively) class. The goal of classification is to find a mapping function $f : (F, S) \rightarrow y$ to predict the class labels of future unseen instance.

Fairness definition: As already discussed in Section 5.2, we adopt ***Eq**ualized **Odds*** (shortly *Eq.Odds*) [HPS+16] as our fairness measure. Eq.Odds, measures the difference in prediction errors between the protected and non-protected group. In particular, let δFPR (δFNR) be the difference in false positive rates (false negative rates, respectively) between the protected and non-protected group, defined as follows ($\hat{y}$ denotes the predicted label):

$$\delta FPR = P(y \neq \hat{y}|\bar{s}_-) - P(y \neq \hat{y}|s_-)$$
$$\delta FNR = P(y \neq \hat{y}|\bar{s}_+) - P(y \neq \hat{y}|s_+) \tag{5.1}$$

The goal is to minimize both differences, the so-called *Eq.Odds*:

$$Eq.Odds = |\delta FPR| + |\delta FNR| \tag{5.2}$$

The value range for each of $|\delta FPR|$, $|\delta FNR|$ is [0,1], where 0 stands for no discrimination and 1 stands for maximum discrimination. Eq.Odds values lie in [0-2] range.

The goal of fairness-aware classification is finding a mapping function $f(\cdot)$ that minimizes Eq.Odds discrimination while maintaining good predictive performance. Typically, the predictive performance in the context of fairness is evaluated in terms of the *error rate*, e.g., [KXPK18, ZVGRG17, CKP09, CWV+17, KC12, HPS+16, FKL16, KC09, KCP10], defined as:

$$ER = \frac{FN + FP}{TP + TN + FN + FP} \tag{5.3}$$

However, optimizing for error rate is problematic in cases of class imbalance. A possible outcome in such a case is that the classifier will misclassify most (in the extreme case all) of the minority instances while correctly classifying the majority - the error rate ER will still be low despite the poor performance in the minority class. W.r.t. fairness such a classifier might still be fair, i.e., $Eq.Odds \approx 0$, as the difference between the FPRs, FNRs for each group will be low (c.f., Equations 5.3,5.2).

However, the non-discriminative behavior of the classifier would be achieved by just reducing drastically the correct predictions for the minority class, with the extreme case of misclassifying everything from the minority. Our goal in this work is to achieve *Eq.Odds* while maintaining low FPRs (equivalently, high TNRs) and low FNRs (equivalently, high TPRs) for both groups.

To this end, we propose (c.f., Section 5.4.3) to replace the error rate (which is not a good performance indicator in case of imbalance, as we also show in Section 5.5) with the *balanced error rate* defined as follows [BOSB10]:

$$BER = 1 - \frac{1}{2} \cdot (\frac{TP}{TP + FN} + \frac{TN}{TN + FP}) = 1 - \frac{1}{2} \cdot (TPR + TNR) \tag{5.4}$$

AdaBoost: Boosting is an ensemble technique that combines weak learners to create a strong learner. AdaBoost [Sch99] calls a weak learner iteratively by adjusting the instance weights in each iteration (boosting round) based on misclassified instances. We believe boosting is a promising technique for fairness-aware classification as it divides the learning problem into multiple sub-problems and then combines their solutions (sub-models) into an overall (global) model. Intuitively, it is easier to tackle the fairness problem in the simpler sub-models rather than in a global complex model. In order to apply AdaBoost for fairness one has to carefully change the underlying data distribution between consecutive rounds so that both predictive performance aspects and fairness-related aspects are considered (Section 5.4).

5.4 Cumulative Fairness Adaptive Boosting

In this work, we tailor AdaBoost for fairness by adjusting its re-weighting process. In particular: i) we directly consider the fairness behavior of the model in the weighting process by introducing the notion of cumulative fairness that assesses the *Eq.Odds* related behavior of the model up to the current boosting round (c.f., Equation 5.4.1). Moreover, differently from vanilla AdaBoost, ii) we employ confidence scores in the re-weighting process to allow for differentiation in instance weighting based on how confident is the model regarding their class. The fairness-aware adjustments in the re-weighting process are described in Section 5.4.2. Finally, we optimize the number of weak learners in the final ensemble by taking into account the balanced error rate and thus directly considering class imbalance in the best model selection (Section 5.4.3).

5.4.1 Cumulative boosting fairness and fairness costs

Let $j \in [1, T]$ be the current boosting round, where T is a user defined parameter indicating the number of boosting rounds. Let $H_{1:j}(x) = \sum_{i=1}^{j} a_i h_i(x)$ be the ensemble model up to round j.

The *cumulative boosting fairness* of the model $H_{1:j}(x)$ at round j is defined in terms of $|\delta FPR|$, $|\delta FNR|$ for both protected and non-protected groups, as follows:

$$\delta FNR^{1:j} = \frac{\sum_{i=1}^{|\bar{s}_+|} 1 \cdot \mathbb{I}[\sum_{k=1}^{j} a_k h_k(x_i^{\bar{s}_+}) \neq y_i]}{|\bar{s}_+|} - \frac{\sum_{i=1}^{|s_+|} 1 \cdot \mathbb{I}[\sum_{k=1}^{j} a_k h_k(x_i^{s_+}) \neq y_i]}{|s_+|}$$

$$\delta FPR^{1:j} = \frac{\sum_{i=1}^{|\bar{s}_-|} 1 \cdot \mathbb{I}[\sum_{k=1}^{j} a_k h_k(x_i^{\bar{s}_-}) \neq y_i]}{|\bar{s}_-|} - \frac{\sum_{i=1}^{|s_-|} 1 \cdot \mathbb{I}[\sum_{k=1}^{j} a_k h_k(x_i^{s_-}) \neq y_i]}{|s_-|} \quad (5.5)$$

where function $\mathbb{I}(\cdot)$ returns 1 iff the expression within is true, otherwise 0. In other words, the cumulative fairness at round j evaluates current ensemble's parity among protected and non-protected groups for both positive and negative class.

If there is no parity, we change the weights of the instances so that discriminated groups are boosted extra in the next round $j+1$. Note that vanilla AdaBoost already boosts misclassified instances for the next round. Our weighting therefore aims at achieving parity across the groups and the classes. To avoid confusion we use the term *costs* for our fairness-related extra weights. The fairness related costs are dynamically estimated in each round. In particular, the fairness related cost u_i for an instance x_i in the boosting round j is computed as follows:

$$u_i = \begin{cases} |\delta FNR^{1:j}|, & if\ \mathbb{I}((y_i \neq h_j(x_i)) \wedge |\delta FNR^{1:j}| > \epsilon), x_i \in s_+, \delta FNR^{1:j} > 0 \\ |\delta FNR^{1:j}|, & if\ \mathbb{I}((y_i \neq h_j(x_i)) \wedge |\delta FNR^{1:j}| > \epsilon), x_i \in \bar{s}_+, \delta FNR^{1:j} < 0 \\ |\delta FPR^{1:j}|, & if\ \mathbb{I}((y_i \neq h_j(x_i)) \wedge |\delta FPR^{1:j}| > \epsilon), x_i \in s_-, \delta FPR^{1:j} > 0 \\ |\delta FPR^{1:j}|, & if\ \mathbb{I}((y_i \neq h_j(x_i)) \wedge |\delta FPR^{1:j}| > \epsilon), x_i \in \bar{s}_-, \delta FPR^{1:j} < 0 \\ 0, & otherwise \end{cases} \quad (5.6)$$

where $u_i \in [0, 1]$ and parameter $\epsilon \in \mathbf{R}$ reflects the tolerance to fairness and is typically set to zero or to a very small value.

At each round, instances which belong to groups that are treated unfairly, receive fairness-related cost. E.g., if in round j group s_+ is discriminated, which means $\delta FNR^{1:j} > 0$ and $\delta FNR^{1:j} > \epsilon$, then misclassified instances in this group will receive fairness related costs for the next round. The signs (+/-) of $\delta FNR^{1:j}$ and $\delta FNR^{1:j}$ denote which group will receive the costs in each class (only one group per class may receive fairness related costs), while ϵ is a condition for the necessity of fairness-related costs in the upcoming round $j+1$. Note that costs u are global, i.e., all instances of a group and class combination will receive the same cost in a round j.

Confidence scores: AdaBoost weighting a_j relies on the classification error of the weak learner h_j built in round j. This is a global weighting, so all instances are weighted with the same a_j. Moreover, this weighting does not take into account how reliable the decision of the classifier was. To this end, we propose to also use the confidence of the predictions $\hat{h}_j(x)$ so that misclassified instances for which the learner is confident in its predictions receive more weight comparing to other instances. This differentiation in the weighting allows the classifier to focus on harder cases.

Algorithm 7: Training phase

Input: $D = (x_i, y_i)_1^N, T, \epsilon$
Output: Ensemble H

1. Initialize $w_i = 1/N$ and $u_i = 0$, for $i = 1, 2, \ldots, N$
2. For $j = 1$ to T:
   (a) Train a classifier h_j to the training data using weights w_i.
   (b) Compute the error rate $\text{err}_j = \frac{\sum_{i=1}^{N} w_i I(y_i \neq h_j(x_i))}{\sum_{i=1}^{N} w_i}$
   (c) Compute the weight $\alpha_j = \frac{1}{2} \cdot \ln(\frac{1-\text{err}_j}{\text{err}_j})$
   (d) Compute fairness-related $\delta FNR^{1:j}$
   (e) Compute fairness-related $\delta FPR^{1:j}$
   (f) Compute fairness-related costs u_i
   (g) Update the distribution as
   $w_i \leftarrow \frac{1}{Z_j} w_i \cdot e^{\alpha_j \cdot \hat{h}_j(x) \cdot \mathbb{I}(y_i \neq h_j(x_i))} \cdot (1 + u_i)$
   // Z_j is normalization factor; $\hat{h}_j$ is the confidence score
3. Output $H(x) = \sum_{j=1}^{T} \alpha_i h_j(x)$

5.4.2 The AdaFair Algorithm

Algorithm 7 shows AdaFair's training phase. In the beginning, instance weights w and costs u are initialized (line 1). Next, a weak learner is trained upon a given weight distribution (line 2a) while the error rate, α_j, $\delta FNR^{1:j}$, $\delta FPR^{1:j}$ and u_i are computed for the current round (lines 2b, 2c, 2d, 2e and 2f, respectively). After u is computed, instance weights are estimated and normalized by a factor Z (line 2g). In contrast to original AdaBoost, we employ the confidence score (line 2g, $\hat{h}$) to contribute to the re-weighting process. Based on the confidence score, we assign higher weights to misclassified instances which are harder to learn compared to instances which are near the decision boundary. The algorithm converges in 3 cases: a) $error = 0.5$, b) $Eq.Odds \leq \epsilon$ and $balanced\ error \neq 0.5$ or c) maximum number of rounds is reached.

5.4.3 Optimizing for class-imbalance and unfair outcomes

AdaBoost requires as input the number of training rounds T. We propose to select the optimal number of weak learners $1 \ldots \theta, \theta \leq T$ that minimizes the error of the model. We propose to optimize for the balanced error rate (Equation 5.4) instead

of the error rate ER (Equation 5.3) in order to tackle the imbalance problem and thus select a model that depicts good performance in both classes, i.e., high TPRs, high TNRs. In case of balanced data, BER corresponds to ER. To allow for different combinations of ER and BER in the θ computation, we consider both ER and BER in the objective function as follows:

$$\arg\min_{\theta} \ (c \cdot BER_{\theta} + (1 - c) \cdot ER_{\theta} + Eq.Odds_{\theta}) \tag{5.7}$$

The parameter c controls the impact of BER and ER in the computation. A detailed evaluation of its impact in the performance of AdaFair is presented in Section 5.5.5. As our analysis show, AdaFair achieves fairness (i.e., $Eq.Odds \approx 0$) for all different values of c, i.e., for all different combinations of ER and BER. However with $c = 1$, that is when optimizing for BER, our method achieves the highest TPRs and only slight decreases in TNRs for both groups. Therefore, we recommend to use our method optimized for BER, that is, with $c = 1$. The result of this optimization step is a final ensemble model with $Eq.Odds$ fairness: $H(x) = \sum_{i=1}^{\theta} a_i h_i(x)$.

5.5 Evaluation

The first goal of our experiments is to evaluate the predictive performance and fairness behavior of AdaFair vs other related approaches (Section 5.5.2). Regarding predictive behavior, we report on both accuracy and balanced accuracy (Equation 5.4), whereas for fairness we report on $Eq.Odds$ (c.f., Section 5.3). Since $Eq.Odds$ (Equation 5.2) is an aggregated measure, we also report on TPR and TNR values for both protected and non-protected groups to shed more light on methods' performance. The second goal of our experiments is to understand the behavior of AdaFair . To this end, we investigate the effect of cumulative vs non-cumulative fairness (Section 5.5.3) and the impact of adopting balanced error rate vs error rate (Section 5.5.5). Moreover, we compare the cumulative distribution of margins to demonstrate the impact of confidence scores in the instance weight estimation (Section 5.5.4). Details on the datasets, baselines, parameter selection and evaluation are provided in Section 5.5.1.

5.5.1 Experimental setup

Datasets

We evaluate our approach on four real-world datasets whose characteristics are summarized in Table 5.1. As we can see, they vary w.r.t. cardinality, dimensionality and class imbalance and therefore provide an interesting benchmark for evaluation.

Adult census income [BL13] dataset contains demographic data from the U.S. and the task is to predict whether the annual income of a person will exceed 50K dollars. The sensitive attribute is $S = Gender$ with $s = female$ being the protected group;

	Adult Census	Bank	Compass	KDD Census
#Instances	45,175	40,004	5,278	299,285
#Attributes	14	16	9	41
Sen.Attr.	Gender	Marit. Status	Gender	Gender
Class ratio (+:−)	1:3.03	1:7.57	1:1.12	1:15.11
Positive class	*>50K*	*subscription*	*recidivism*	*>50K*

Table 5.1: An overview of the datasets.

the positive class is people receiving more than 50K. We remove duplicate instances and instances containing missing values. The positive to negative class ratio is 1:3 (exact ratio 24%:76%).

Bank dataset [BL13] is related to direct marketing campaigns of a Portuguese banking institution. The task is to determine if a person subscribes to the product (bank term deposit). As positive class we consider people who subscribed to a term deposit. We consider as $S = maritial\ status$ with $s = married$ being the protected group. The dataset suffers from severe class imbalance, with a positive to negative ratio of 1:8 (exact ratio 11%:89%).

Compass dataset [LMKA16] contains information on prisoners in Broward County such as the number of juvenile felonies. The task is to determine if a person will be re-arrested within two years *(recidivism)*. We consider *recidivism* as the positive class and $S = Gender$ with $s = female$ as the protected group. For this dataset, we followed the pre-processing steps of [ZVGRG17]. The dataset is almost balanced, the exact positive to negative ratio is 46%:54%.

KDD census income [BL13] has the same prediction task as the **adult census** dataset. However in KDD census "the class labels were drawn from the total person income field rather than the adjusted gross income" [BL13]. We consider $S = Gender$ with $s = female$ as the protected group, and as positive class people receiving more than 50K annually. This is the most skewed dataset in our benchmark with a positive to negative ratio of 1:15 (exact ratio 6%:94%).

Baselines

We evaluate our approach against recently proposed state-of-the-art in-processing approaches that also aim to minimize *Eq.Odds*, namely the methods of Krasanakis et al. and Zafar et al. In addition, we compare against two fairness-agnostic boosting versions: vanilla AdaBoost [Sch99] and SMOTEBoost [CLHB03] (an AdaBoost approach that tackles class-imbalance). Finally, to study the behavior of our approach we also compare AdaFair against different variations described hereafter:

Zafar et al.[ZVGRG17]: The authors formulate the fairness problem as a set of convex-concave constraints to minimize *Eq.Odds*, for which they optimize a logistic

regression model.

Krasanakis et al. [KXPK18]: The authors assume the existence of latent fair classes and propose an iterative training approach towards those classes which alters in-training the instance weights. Fairness is assessed via *Eq.Odds*.

AdaBoost [Sch99]: This is the vanilla AdaBoost that does not consider fairness nor confidence scores.

SMOTEBoost [CLHB03]: This is an extension of AdaBoost for imbalanced data. At each boosting round, new synthetic instances of the minority class are generated via SMOTE [CBHK02], to compensate for the imbalance. SMOTEBoost does not consider fairness nor confidence scores. The goal of this baseline is to see whether by only tackling imbalance the fairness problem can be addressed.

AdaFair: Our proposed approach that combines cumulative fairness, balanced error rate and confidence scores.

AdaFair NoCumul: Similar to AdaFair , however the *Eq.Odds* is computed per round rather than over all previous rounds, therefore impacting the instance weighting. The goal of this baseline is to help clarifying the impact of cumulative vs non-cumulative fairness (Section 5.5.3).

AdaFair NoConf: Similar to AdaFair , but is does not employ confidence scores for weight estimation. We employ this baseline to depict the impact of confidence scores in the cumulative distribution of margins of the training instances (Section 5.5.4).

Parameter selection and evaluation

We follow the same evaluation setup as in [ZVGRG17, KXPK18] by splitting each dataset randomly into train (50%) and test set (50%) and report on the average of 10 random splits. We set $\epsilon = 0$ as a threshold for *Eq.Odds*, which means zero tolerance to discrimination. Moreover, we set the number of boosting rounds for AdaFair to $T = 200$ (same for the other ensemble approaches, c.f., Section 5.5.1). For Krasanakis et al. and Zafar et al. methods, we employ their default parameters. For SMOTEBoost, we set N (the number of synthetic instances generated per round) to 2, 100, 100 and 500 for datasets compass, adult census, bank and kdd census, respectively. Furthermore, for experiments in Section 5.5.2, 5.5.3 and 5.5.4, we set parameter $c = 1$ (c.f., Equation 5.7), that is the proposed AdaFair optimized for balanced error rate; the effect of c is studied in Section 5.5.5.

Our method[1] has been instantiated with Decision Stumps as weak learners.

[1] https://iosifidisvasileios.github.io/AdaFair

5.5.2 Predictive and fairness performance

Adult census income

In Figure 5.1, we show the performance of the different approaches on Adult census dataset. Regarding predictive performance, we report on accuracy and balanced accuracy (*Bal.Acc.* for short) whereas regarding fairness, we report on *Eq.Odds* and TPR, TNR values for both protected and non-protected groups (*Prot.* and $Non-Prot.$, respectively for short).

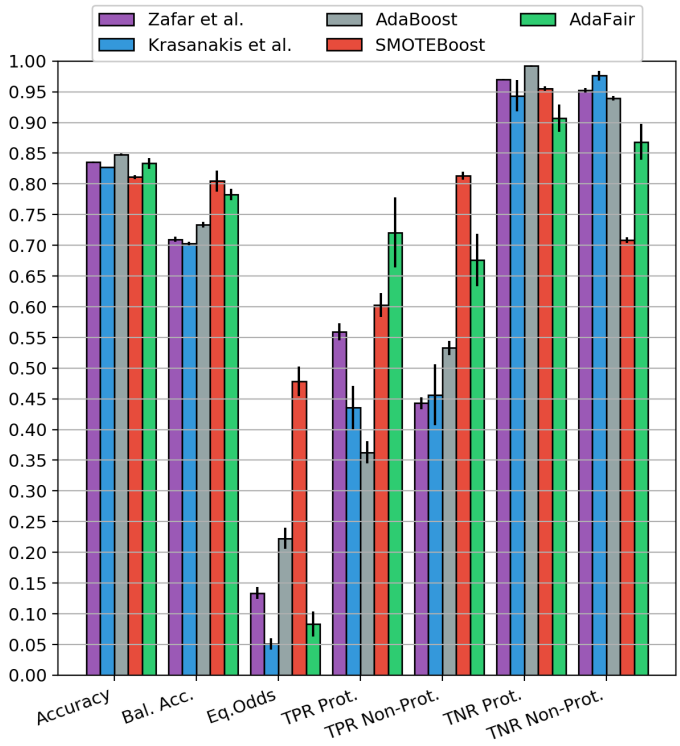


Figure 5.1: Adult census: Predictive and fairness performance - higher values are better; for Eq.Odds, lower values are better.

The best balanced accuracy is achieved by SMOTEBoost followed by AdaFair (2% ↓); both methods target class imbalance, the latter however also considers fairness. AdaBoost, Krasanakis et al. and Zafar et al. that do not consider class imbalance have a 5%↓, 8%↓ and 8%↓, respectively drop in their balanced accuracy comparing to AdaFair .

Regarding *Eq.Odds*, as expected AdaBoost and SMOTEBoost perform worse as they do not consider fairness. The best overall *Eq.Odds* score is achieved by the method of Krasanakis et al., followed by our AdaFair (3% ↑, recall that lower values are better). A closer look, however, at the actual TPRs and TNRs values per group shows that our method achieves the highest TPRs values for both protected

and non-protected groups compared to the other two fairness aware approaches. In particular, for the protected (non-protected) group our TPR is 29%↑ (22%↑, respectively) higher that of the second best method of Krasanakis et al. So, it seems that the methods of Krasanakis et al. and Zafar et al. produce low TPRs and high TNRs, i.e., these methods "reject" more instances of the positive class in order to minimize *Eq.Odds* (this explains their high TNRs, low TPRs values). On the contrary, our AdaFair achieves good performance for both classes (high TPRs, high TNRs) while maintaining a good *Eq.Odds* (i.e., low difference in TPRs, TNRs for both protected and non-protected group).

Bank

The results are shown in Figure 5.2. All approaches, except for AdaBoost and SMOTEBoost, achieve low *Eq.Odds*. Interestingly, AdaFair achieves better balanced accuracy than SMOTEBoost, while it outperforms the other approaches.

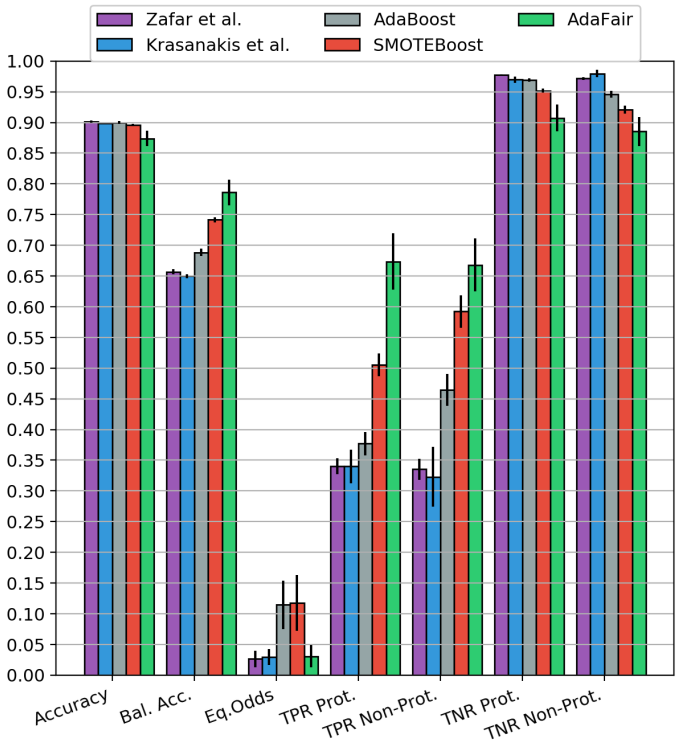


Figure 5.2: Bank: Predictive and fairness performance - higher values are better; for Eq.Odds, lower values are better.

A closer look at *Eq.Odds* and namely at TPRs and TNRs shows significant differences between the approaches. Namely, w.r.t. TPRs our method outperforms the second best (Zafar et al.) by almost 33%↑ for each group. Interestingly, AdaBoost and SMOTEBoost maintain higher TPRs than the methods of Krasanakis et al. and

Zafar et al., even though they do not consider fairness. The methods of Krasanakis et al. and Zafar et al. have very similar behavior and it seems that both of them focus on the majority class (therefore high TNRs, low TPRs). Regarding TNRs, our method has a small drop of 6%↓ and 7%↓ drop for the protected and non-protected group compared to the second best approach of Zafar et al.; this is expected as we optimize for balanced error rather than overall error.

Compass

The results are shown in Figure 5.3. Regarding balanced accuracy, AdaBoost performs best and Zafar et al worse. However, the differences between the approaches are not that high. The similar performance of the different approaches is to be expected as the dataset is balanced (c.f., Table 5.1) and therefore, imbalance treatment has no strong effect.

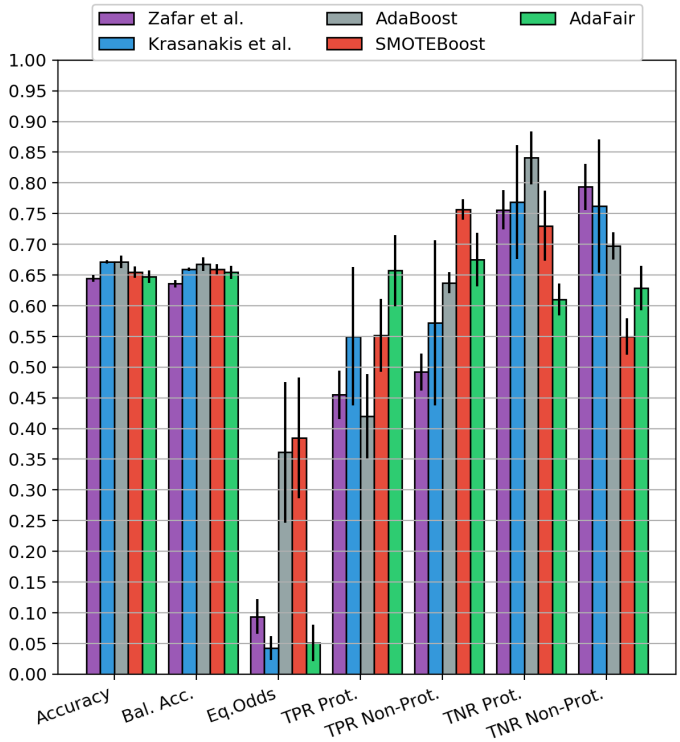


Figure 5.3: Compass: Predictive and fairness performance - higher values are better; for Eq.Odds, lower values are better.

Regarding fairness, the method of Krasanakis et al. achieves slightly better performance in terms of *Eq.Odds* (0.6%↓) and balanced accuracy (0.2%↑) compared to the second best AdaFair . Zafar et al. has the worst *Eq.Odds* (almost twice the value of Krasanakis et al.), recall that its *Bal.Acc.* was the worse among the approaches.

By examining the TPRs and TNRs of both protected and non-protected groups, we observe that the performance of Krasanakis et al. is not stable (highest standard deviation among the methods). Our AdaFair has better TPR values for both groups. Our TNRs are the lowest among the approaches ($61\% - 65\%$) as we optimize for balanced error and the negative class represents 54% of the population.

KDD census income

The dataset suffers from extremely high class imbalance, with a ratio of 1:15 (c.f., Table 5.1). Zafar et al. approach could not be applied to this dataset due to its inability to estimate the optimal parameters, therefore we omit it.

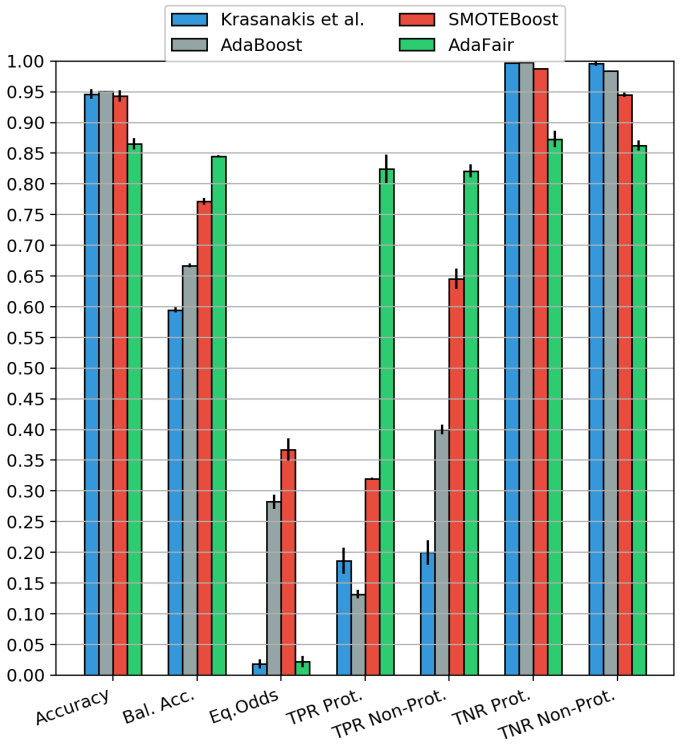


Figure 5.4: KDD census: Predictive and fairness performance - higher values are better; for Eq.Odds, lower values are better.

The results of KDD census dataset are presented in Figure 5.4. In terms of balanced accuracy AdaFair performs 25%↑ than Krasanakis et al., 18%↑ than AdaBoost and 8%↑ than SMOTEBoost. AdaBoost and Krasanakis et al. classify almost perfectly the negative class (i.e., TNRs close to 100%), which comprises 94% of the population. SMOTEBoost has 2%↓ to 4%↓ drop in TNRs of protected and non-protected group respectively, compared to AdaBoost. In terms of TPR, AdaFair achieves the highest TPR scores for both groups (above 80%) while the method of Krasanakis et al. results in values below 20%. Both fairness-aware approaches, AdaFair and

Krasanakis et al., minimize discrimination to 2% while AdaBoost and SMOTEBoost result in 28% and 36% *Eq.Odds*, respectively.

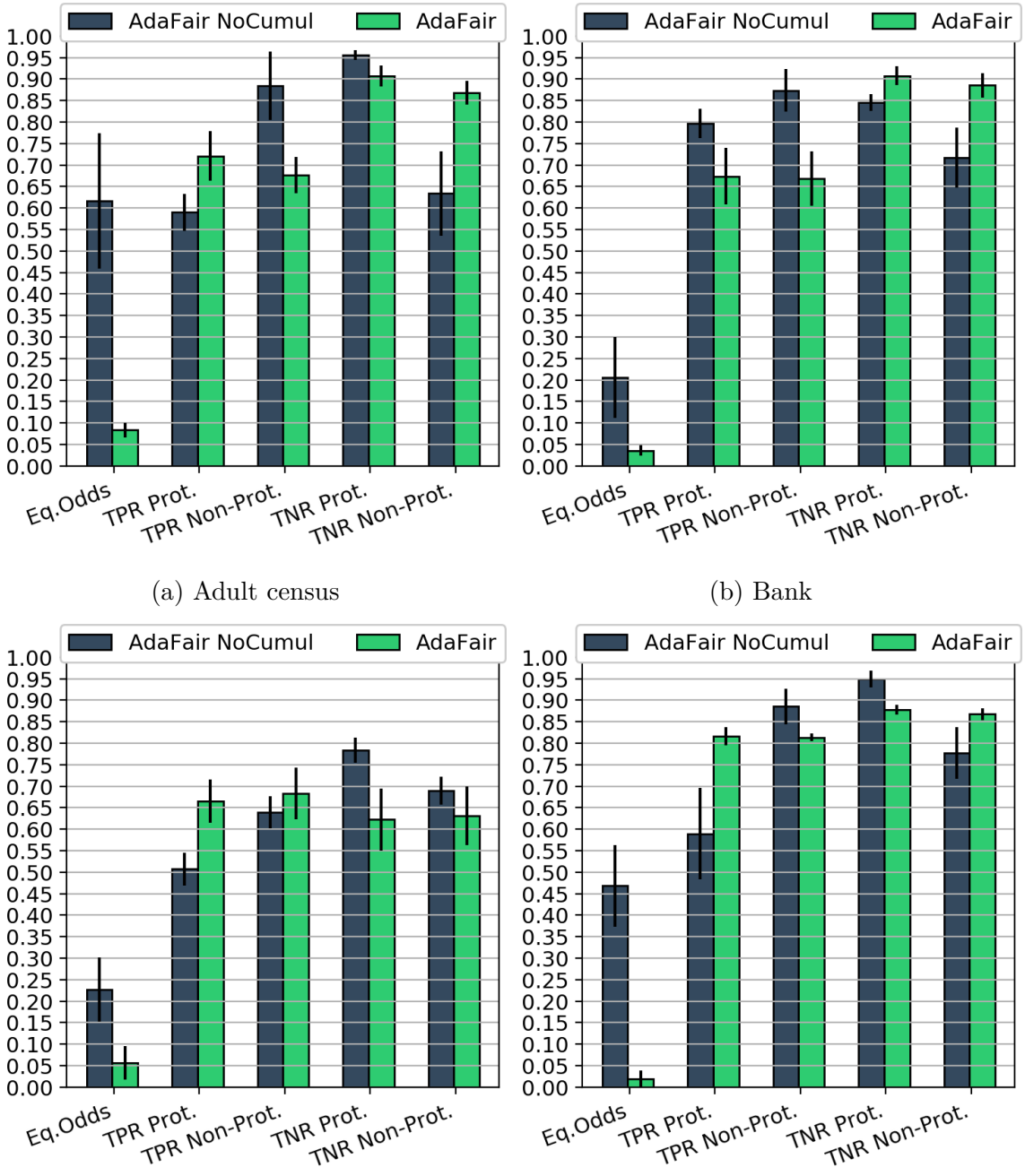
Conclusion: To conclude, our AdaFair is able to achieve high balanced accuracy and low discrimination by maintaining high TPRs and only slightly worse TNRs for both groups. On the contrary, the other fairness-aware approaches, namely Zafar et al. and Krasanakis et al, eliminate discrimination by reducing TPRs, that is by rejecting more instances of the positive class to achieve parity among the protected and non-protected groups. Zafar et al. is unable to handle multi-dimensional datasets while it can not estimate the optimal parameters. Furthermore, we perform paired t-test between AdaFair and each baseline for all datasets. The results show highly significant difference ($p < 0.001$) for all datasets between AdaFair and each baseline (it is also visible from the reported TPRs and TNRs of each dataset).

5.5.3 Cumulative vs non-cumulative fairness

The notion of cumulative fairness (c.f. Equation 5.4.1), is crucial for AdaFair ' ability to mitigate discrimination. To investigate its impact we compare our AdaFair (with cumulative fairness of models $1 : j$, where j is the current boosting round) with a version that considers only the fairness of the individual weak learner at round j (refereed to as AdaFair NoCumul).

Their predictive and fairness performance for the different datasets is shown in Figure 5.5. Overall, AdaFair NoCumul method results in poor fairness performance with very high *Eq.Odds* values compared to AdaFair . In particular, we observe an increase of 52%↑ for the adult dataset, an increase of 15%↑ for bank and compass datasets and an increase of 45%↑ for the kdd census dataset. A closer look at the individual TPR, TNR scores shows that regarding TPR, the scores of the protected group are lower whereas w.r.t. TNR, the scores of the protected group are higher. That is, more protected instances are rejected (low TPR, high TNR). Moreover, standard deviation for the non-cumulative version is higher than AdaFair , indicating AdaFair NoCumul is not stable. It appears that a cumulative notion of fairness based on the current ensemble composition is better than a non-cumulative approach that considers solely the fairness behavior of the current/last weak learner.

Except for the overall ensemble behavior, we also show the behavior per round. In particular, in Figure 5.6 we compare the per round $\delta FNRs$ and $\delta FPRs$ of the two approaches. Recall that δFNR and δFPR define the extra weighting/cost u related to fairness that affects the weighting of the instances for the next round (see Section 5.4.2). Costs of AdaFair NoCumul exhibit a high fluctuation. On the contrary, the costs for our AdaFair are smoother and converge after a sufficient number of rounds to a certain range $[-0.05, 0.05]$. That means that our method mitigates discrimination over the early rounds. This results further confirms that the cumulative definition of fairness is superior to a non-cumulative approach.



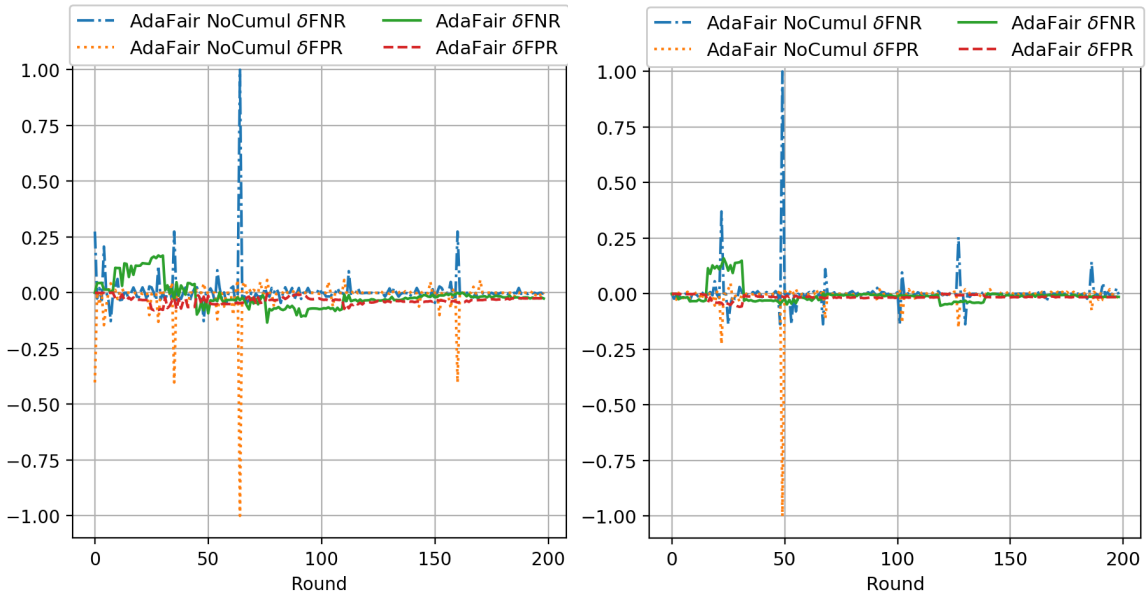
(a) Adult census

(b) Bank

(c) Compass

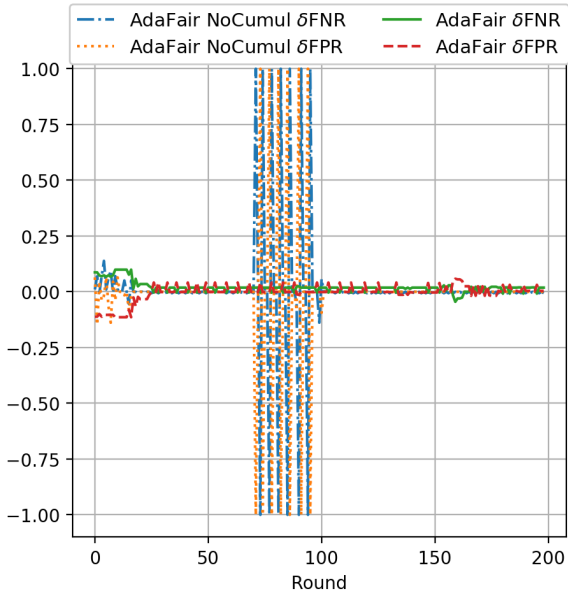
(d) KDD census

Figure 5.5: Effect of cumulative fairness: AdaFair vs AdaFair NoCumul

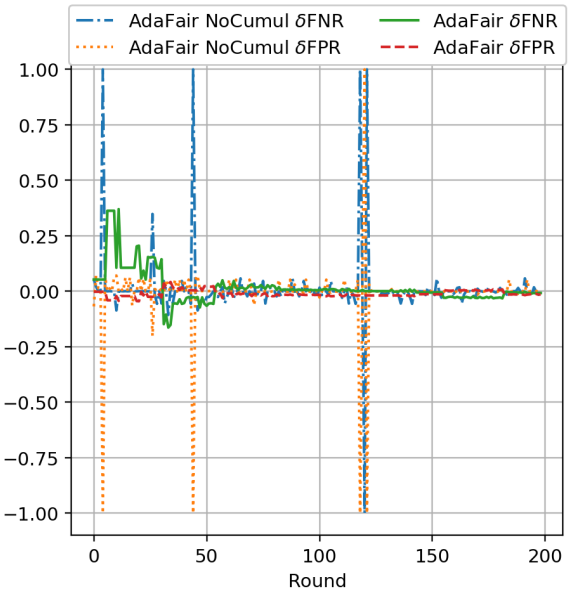


(a) Adult census

(b) Bank



(c) Compass



(d) KDD census

Figure 5.6: Costs per boosting round: AdaFair vs AdaFair NoCumul

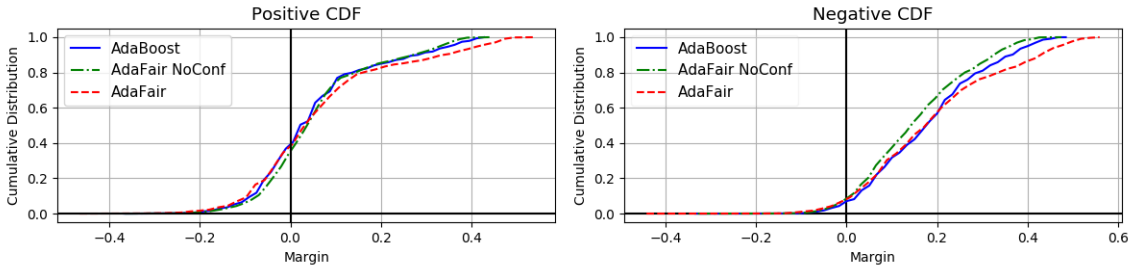
5.5.4 The effect of confidence scores

Our AdaFair utilizes confidence scores in the instance weight estimation (c.f., Section 5.4.2) in contrast to vanilla AdaBoost. To demonstrate the impact of confidence scores, we compare AdaFair to a variation that does not use confidence scores (Algorithm 7 line 2g, confidence score $\hat{h}$ is removed); we refer to this as AdaFair NoConf, hereafter. We also include vanilla AdaBoost in the comparison. We evaluate the three approaches w.r.t. their cumulative margins. Comparing to accuracy and balanced accuracy that rely only on correctly and incorrectly classifications, the margins also reveal information about how confident are the predictions of the different methods [Sch13]. The cumulative distribution of margins for each dataset and per class is provided in Figure 5.7. The margin values lie in the $[-1, 1]$ range; values close to -1 indicate misclassifications with high confidence whereas values close to 1 indicate correct classifications with high confidence. The bulk of the distribution is located where the curve increases abruptly.

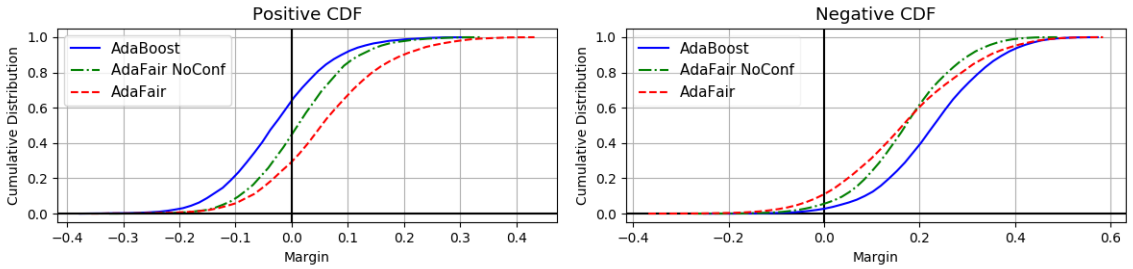
Based on the results, we observe that for the positive class our AdaFair is more confident comparing to the other approaches (its curve is shifted to the right) and this holds for all datasets except for compass where there is no clear winner between AdaFair and AdaFair NoConf - recall that this dataset is balanced. Also, we can observe that AdaFair has the lowest errors in the positive class and the misclassifications in this class have lower confidence values comparing to the other methods. Regarding the negative class, we can observe that AdaBoost achieves the best performance compared to AdaFair and AdaFair NoConf, since it focuses solely on overall accuracy, thus it learns better the majority class.

The differences in the performances of the different methods increase with the dataset imbalance; for example for the *Bank dataset*, AdaFair has significantly less misclassifications in the positive class comparing to AdaFair NoConf (15% $\downarrow$) and AdaBoost (36% $\downarrow$), and only small increase in the misclassifications of the negative class, 5% $\downarrow$ and 7% $\downarrow$, respectively. Similarly, for the *KDD dataset*, our AdaFair has 23% $\downarrow$ and 46% $\downarrow$ less misclassifications than AdaFair NoConf and AdaBoost, respectively, while the increase in error at the negative class is 12% $\downarrow$ and 14% $\downarrow$, respectively. In this case, AdaBoost classifies correctly almost all the negative (majority) instances, but its performance on the minority class is the worst (it has the leftmost curve).

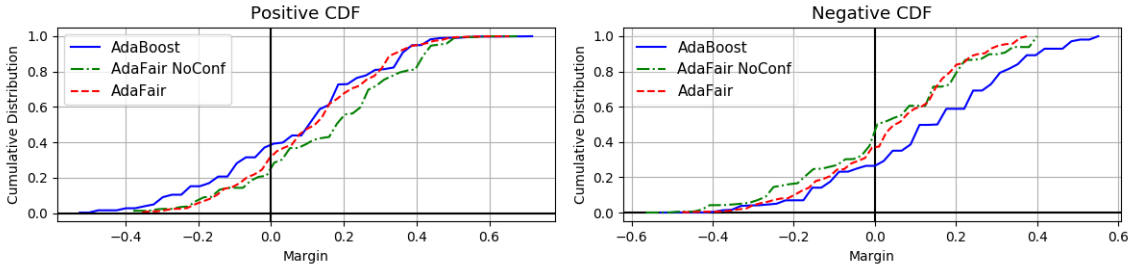
To conclude, considering confidence scores for instance weight estimation, the ensemble's margins increase especially in case of class imbalance. Instead of "boosting" all misclassified instances equally, this approach differentiates based on the confidence score of the misclassification, s.t. misclassified instances close to the decision boundary are weighted less than those that are further away.



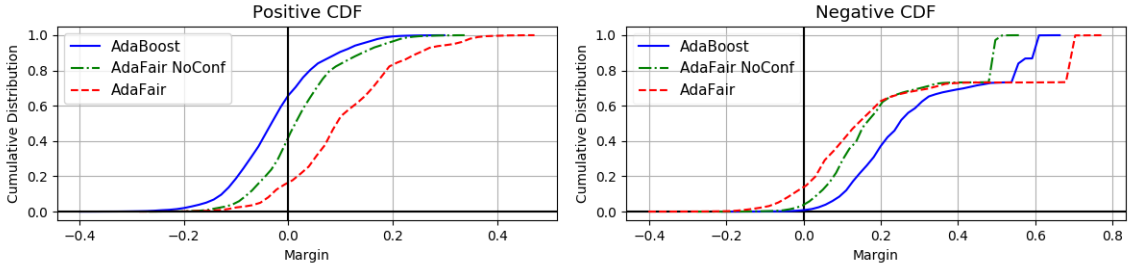
(a) Adult census



(b) Bank



(c) Compass

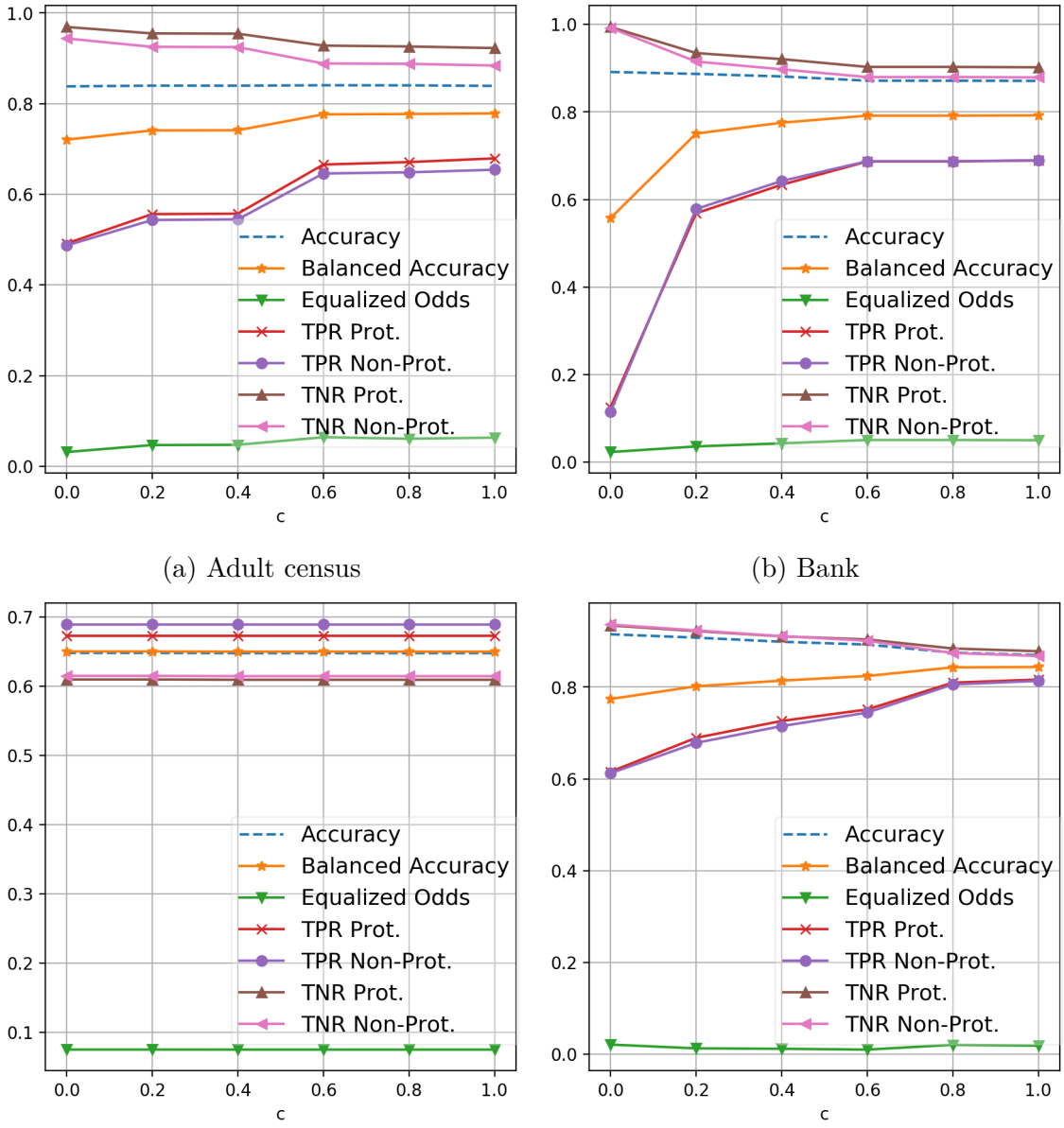


(d) KDD census

Figure 5.7: Effect of confidence scores per dataset

5.5.5 The effect of balanced error

Our AdaFair is able to achieve parity even when it does not optimize for balanced error rate because of the cumulative fairness and confidence scores that alter the data distribution during training in the direction of fairness. In this section, we show that varying parameter c, which alternates the objective goal (Equation 5.7), has a strong positive effect on TPRs while only slightly affecting TNRs.



(a) Adult census

(b) Bank

(c) Compass

(d) KDD census

Figure 5.8: The effect of balanced error (here we report on accuracy and balanced accuracy instead)

In particular, in Figure 5.8, we plot accuracy and fairness related measures for different values of $c \in [0, 1]$. For $c = 1$, the balanced error is optimized (our proposed AdaFair). For $c = 0$, the error rate is optimized. Values in-between (we use a step of 0.2 for c) correspond to different combinations of balanced error and error rate. Values are averaged over 10 random splits. Note that different values of c result into different AdaFair sizes, i.e., the sub-sequence of weak learners in the ensemble varies with c.

As we can see, for all imbalanced datasets (adult census, bank, kdd census) the balanced accuracy increases with c. For $c = 0$ (only error rate is considered) the TPRs for both groups are very low. The TPRs increase with c, reaching their best values at $c = 1$, i.e., when balanced accuracy is considered. TNRs decrease with c, though their decrease is lower than the increase of TPRs. This again supports our previous findings that AdaFair achieves parity between the two groups for both TPRs and TNRs while achieving high TPRs, on the contrary to Zafar et al. and Krasanakis et al. (c.f., Section 5.5). For Compass, accuracy and balanced accuracy are very close and therefore no significant differences are observed by varying c.

More precisely, a comparison of the results for $c = 0$ (error rate) and $c = 1$ (balanced error rate) shows i) for the adult dataset, 18%↑ and 15%↑ increase in TPRs (for s, $\bar{s}$ groups respectively) and only 5%↓ reduction in TNRs (for both groups); ii) for the bank dataset, 56%↑ increase in TPRs (for both groups) and only 8%↓ and 9%↓ decrease in TNRs (for s and $\bar{s}$,respectively)and iii) for the kdd dataset, 20%↑ growth in TPR (for both groups) and only 5%↓ and 6%↓ reduction in TNRs (for s and $\bar{s}$, respectively).

To conclude, directly tackling the imbalance via the balanced error is more effective for fairness-aware classification.

5.6 Chapter Summary

In this chapter, we have proposed AdaFair , a fairness-aware boosting approach that adapts AdaBoost to fairness by changing the data distribution at each round based on the notion of cumulative fairness that evaluates per round the fairness-related behavior of the thus far built ensemble and adjusts accordingly the weight/cost of the per round discriminated group.

Our experiments, on four real-world datasets, show a substantial difference in TPRs compared to current fairness aware state-of-the-art approaches. We have seen, in cases of severe and extreme class-imbalance, that AdaFair can achieve parity among protected and non-protected group and at the same time maintain significantly better classification performance compared to other approaches. Moreover, in cases of extreme class-imbalance, AdaFair is able to outperform methods which focus solely on class-imbalance. In cases of class-balanced datasets, such as compass, AdaFair achieves parity similar to other fairness aware methods.

Moreover, AdaFair is able to achieve parity even when it does not optimize for balanced error, however this impacts its performance in terms of balanced error rate i.e., low TPRs, high TNRs and low Eq.Odds. In addition, from our comparisons of cumulative versus non-cumulative fairness, we come to the conclusion that cumulative fairness is without doubt superior to non-cumulative fairness. Furthermore, adapting confidence scores in the weight estimation process makes AdaFair more confident compared to vanilla AdaBoost.

A possible extension of our method is the online selection of the optimal number of rounds θ for the ensemble, to replace our post-training search for the optimal θ parameter. Moreover, we plan to work on the theoretical aspects of our method, in particular regarding its convergence properties.

6 Conclusions and Future Work

In this thesis, we have focused on class-imbalance and its implications into two popular areas: i) semi-supervised learning, where label scarcity exists, and ii) fairness-aware learning, where population segments are treated unequally. In this section, we conclude our main findings and discuss future directions for each chapter.

In **Chapter 3**, we have explored semi-supervised methods, such as Co-Training and Self-Learning, under the prism of class-imbalance. We have shown that semi-supervised methods propagate errors and class-imbalance in each iteration. To deal with class-imbalance, we have coupled semi-supervised methods with various augmentation methods such as: over-sampling, under-sampling, distortion, and semantic-similarity (via word-embeddings). By employing augmentation methods, we have created more training data, and also added more variation through domain-meaningful and sound transformations. Our experiments indicate that such a combination of methods (semi-supervised with augmentation methods) tackle the class-imbalance propagation. In addition, we have annotated large amounts of unlabeled textual data. Our large scale dataset, called *TSentiment15*, contains more than 200 million English short texts, and is publicly available to the community.

Future work: Although augmentation methods are essential to effectively deal with class-imbalance, they can amplify existing errors if they are not used with caution. When coupled with semi-supervised methods, these errors are propagated to the overall predictions, which makes the combination of such methods not trivial. A filtering mechanism should be studied in order to filter pseudo-instances which are not semantically similar to the original instances (for the task of sentiment classification). In addition, for the semantic-similarity augmentation, the impact of standard versus refined word-embeddings [YWLZ17] should be explored since in [YWLZ17] they construct domain-specific (sentiment analysis) word-embedding projections.

In **Chapter 4**, we have studied the problem of unfair outcomes in supervised learning models through the prism of class- and within-class imbalance. We have shown that skewed data distributions affect the supervised models w.r.t discrimina-

tory behaviour. We have seen that *within class-imbalance* (group-imbalance) forces the models to misclassify minority segments disproportionately compared to other segments since they are not able to learn all population segments effectively. We presented a **f**airness-**a**ware **e**nsemble framework (FAE) which mitigates unfair outcomes and deals with class-imbalance at the same time. FAE combines pre-processing and post-processing fairness-aware interventions to deal with class- and within-class imbalance, as well as efficiently mitigate unfair outcomes. Our experiments show that our approach significantly outperforms state-of-the-art methods, and show that class- and within-class imbalance have an impact in a model's discriminatory behavior.

Future work: In the chapter, we have shown that jointly pre- and post-processing fairness enhancing interventions are able to tackle class-imbalance as well as mitigate unfair outcomes. A future direction for FAE, is to study the selection of the boosting models since we applied a grid search to select the appropriate boosting models. A heuristic selection may lead to better performance and non-discriminatory behavior. Another interesting direction would be to reduce the computational complexity of our approach by removing the post-processing step and embed it into the training phase.

Finally, in **Chapter 5**, we have studied fairness-aware learning in sequential models such as AdaBoost, and we also introduced the notion of cumulative fairness. Our approach, called AdaFair, is equipped with an objective function that minimizes the balanced error as well as unfair outcomes. We have shown that cumulative fairness, combined with the AdaBoost, can efficiently mitigate discriminatory outcomes. By assigning fairness related weights to the misclassified instances during the training phase based on cumulative fairness, our model is able to obtain a sequence of weak learners which are able to produce fair outcomes. Therefore, AdaFair is able to tackle class-imbalance and mitigate unfair outcomes at the same time. Our experiments show that our model outperforms state-of-the-art methods in terms of performance and fairness i.e., AdaFair minimizes unfair outcomes, and at the same time, preserves high TPR scores for protected and non-protected group, in contrast to recent state-of-the-art methods that produce low TPR scores.

Future work: A natural avenue for future work is to incorporate balanced error during the weight assignment of the training phase to avoid the grid search of θ. An initial investigation in this direction, in which we have converted AdaFair into cost-sensitive boosting method, showed that by assigning fixed misclassification costs to the training procedure makes the model unable to mitigate unfair outcomes. Therefore, misclassification costs and fairness-related costs should not be independent of each other, rather properly combined to deal with the joint problem of class-imbalance and unfair outcomes. Another solution to this problem, which is in the same line of thought as [DHP+12], would be to introduce a new fairness notion which accounts not only for parity across groups but also across classes and incorporate into training phase. Furthermore, the theoretical properties of AdaFair should be extensively studied, such as bounding the upper training error.

In this thesis, we have studied the impact of class-imbalance in two different

areas, such as semi-supervised learning and fairness-aware learning; however, class-imbalance affects various fields in machine learning. Although standard class-imbalance techniques are powerful methods to deal with the standalone class-imbalance problem (e.g., classification performance), they may become ineffective when facing a joint problem. Therefore, domain-specific or heuristic approaches are necessary in order to tackle combined problems.

Bibliography

[ABD+18] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna M. Wallach. A reductions approach to fair classification. In *ICML*, volume 80, pages 60–69, 2018.

[AG05] Anthony Aue and Michael Gamon. Customizing sentiment classifiers to new domains: A case study. In *Proceedings of recent advances in natural language processing (RANLP)*, volume 1, pages 2–1, 2005.

[ASB+19] Muhammad Ali, Piotr Sapiezynski, Miranda Bogen, Aleksandra Korolova, Alan Mislove, and Aaron Rieke. Discrimination through optimization: How facebook's ad delivery can lead to skewed outcomes. *arXiv preprint arXiv:1904.02095*, 2019.

[BBY05] Maria-Florina Balcan, Avrim Blum, and Ke Yang. Co-training and expansion: Towards bridging theory and practice. In *Advances in neural information processing systems*, pages 89–96, 2005.

[BCM+13] Prakhar Biyani, Cornelia Caragea, Prasenjit Mitra, Chong Zhou, John Yen, Greta E Greer, and Kenneth Portier. Co-training over domain-independent and domain-dependent features for sentiment analysis of an online cancer support community. In *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*, pages 413–417. IEEE, 2013.

[BD99] Kristin P Bennett and Ayhan Demiriz. Semi-supervised support vector machines. In *Advances in Neural Information processing systems*, pages 368–374, 1999.

[BES10] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, volume 10, pages 2200–2204, 2010.

[BESS13] Giacomo Berardi, Andrea Esuli, Fabrizio Sebastiani, and Fabrizio Silvestri. Endorsements and rebuttals in blog distillation. *Information Sciences*, 249:38–47, 2013.

[BF10] Albert Bifet and Eibe Frank. Sentiment knowledge discovery in twitter streaming data. In *International Conference on Discovery Science*, pages 1–15. Springer, 2010.

[BKK+98] Jeffrey P Bradford, Clayton Kunz, Ron Kohavi, Cliff Brunk, and Carla E Brodley. Pruning decision trees with misclassification costs. In *European Conference on Machine Learning*, pages 131–136. Springer, 1998.

[BL13] Kevin Bache and Moshe Lichman. Uci machine learning repository, 2013.

[BLRR04] Avrim Blum, John Lafferty, Mugizi Robert Rwebangira, and Rajashekar Reddy. Semi-supervised learning using randomized mincuts. In *Proceedings of the twenty-first international conference on Machine learning*, page 13, 2004.

[BM98] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

[BNZ17] Daniel Basaran, Eirini Ntoutsi, and Arthur Zimek. Redundancies in data and their effect on the evaluation of recommendation systems: A case study on the amazon reviews datasets. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 390–398. SIAM, 2017.

[BOSB10] Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M Buhmann. The balanced accuracy and its posterior distribution. In *2010 20th International Conference on Pattern Recognition*, pages 3121–3124. IEEE, 2010.

[Bre96] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[CBHK02] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[CCC03] Fabio G Cozman, Ira Cohen, and Marcelo C Cirelo. Semi-supervised learning of mixture models. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 99–106, 2003.

[CGK15] Xiaodong Cui, Vaibhava Goel, and Brian Kingsbury. Data augmentation for deep neural network acoustic modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(9):1469–1477, 2015.

[CKP09] Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. Building classifiers with independency constraints. In *2009 IEEE ICDM Workshops*, pages 13–18. IEEE, 2009.

[CLHB03] Nitesh V Chawla, Aleksandar Lazarevic, Lawrence O Hall, and Kevin W Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *ECML PKDD*, pages 107–119. Springer, 2003.

[CSZ09] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[CV10] Toon Calders and Sicco Verwer. Three naive bayes approaches for discrimination-free classification. *DMKD*, 21(2):277–292, 2010.

[CWV+17] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. In *NIPS*, pages 3992–4001, 2017.

[CZ05] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *AISTATS*, volume 2005, pages 57–64. Citeseer, 2005.

[CŽ13] Toon Calders and Indrė Žliobaitė. Why unbiased computational processes can lead to discriminative decision procedures. In *Discrimination and privacy in the information society*, pages 43–57. Springer, 2013.

[DARW+19] Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. Bias in bios: A case study of semantic representation bias in a high-stakes setting. *arXiv preprint arXiv:1901.09451*, 2019.

[DB01] Ayhan Demiriz and Kristin P Bennett. Optimization approaches to semi-supervised learning. In *Complementarity: Applications, Algorithms and Extensions*, pages 121–141. Springer, 2001.

[DH+03] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, volume 11, pages 1–8. Citeseer, 2003.

[DHP+12] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM, 2012.

[DIKL18] Cynthia Dwork, Nicole Immorlica, Adam Tauman Kalai, and Mark DM Leiserson. Decoupled classifiers for group-fair and efficient machine learning. In *Conference on Fairness, Accountability and Transparency*, pages 119–133, 2018.

[DIR98] HAS ADOPTED THIS DIRECTIVE. Council directive 97/80/ec of 15 december 1997 on the burden of proof in cases of discrimination based on sex. *Official Journal L*, 14(20/01):0006–0008, 1998.

[DLR77] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[DLZ11] Jun Du, Charles X Ling, and Zhi-Hua Zhou. When does cotraining work in real data? *IEEE Transactions on Knowledge and Data Engineering*, 23(5):788–799, 2011.

[DN09] Sajib Dasgupta and Vincent Ng. Mine the easy, classify the hard: a semi-supervised approach to automatic sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 701–709. Association for Computational Linguistics, 2009.

[Dom] P Domingos. A general method for making classifiers cost-sensitive. *Artificial Inelligence Group, Instituto Superior Técnico, Lisboa*, pages 1049–001.

[DTD15] Amit Datta, Michael Carl Tschantz, and Anupam Datta. Automated experiments on ad privacy settings. *Privacy Enhancing Technologies*, 2015(1):92–112, 2015.

[EJJ04] Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational intelligence*, 20(1):18–36, 2004.

[EL14] Benjamin G Edelman and Michael Luca. Digital discrimination: The case of airbnb. com. 2014.

[ES16] Harrison Edwards and Amos J. Storkey. Censoring representations with an adversary. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[FAKA+18] Maayan Frid-Adar, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Synthetic data augmentation using gan for improved liver lesion classification. In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 289–293. IEEE, 2018.

[FIND18] Pavlos Fafalios, Vasileios Iosifidis, Eirini Ntoutsi, and Stefan Dietze. Tweetskb: A public and large-scale rdf corpus of annotated tweets. In *European Semantic Web Conference*, pages 177–190. Springer, 2018.

[FISN17] Pavlos Fafalios, Vasileios Iosifidis, Kostas Stefanidis, and Eirini Ntoutsi. Multi-aspect entity-centric analysis of big social media archives. In *International Conference on Theory and Practice of Digital Libraries*, pages 261–273. Springer, 2017.

[FISN18] Pavlos Fafalios, Vasileios Iosifidis, Kostas Stefanidis, and Eirini Ntoutsi. Tracking the history and evolution of entities: entity-centric temporal analysis of large social media archives. *International Journal on Digital Libraries*, pages 1–13, 2018.

[FKL16] Benjamin Fish, Jeremy Kun, and Ádám D Lelkes. A confidence-based approach for balancing fairness and accuracy. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 144–152. SIAM, 2016.

[FM01] Glenn Fung and Olvi L Mangasarian. Semi-superyised support vector machines for unlabeled data classification. *Optimization methods and software*, 15(1):29–44, 2001.

[Fra67] S Fralick. Learning to recognize patterns without a teacher. *IEEE Transactions on Information Theory*, 13(1):57–64, January 1967.

[FSV+19] Sorelle A Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P Hamilton, and Derek Roth. A comparative study of fairness-enhancing interventions in machine learning. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 329–338. ACM, 2019.

[FSZC99] Wei Fan, Salvatore J Stolfo, Junxin Zhang, and Philip K Chan. Adacost: misclassification cost-sensitive boosting. In *Icml*, volume 99, pages 97–105, 1999.

[GBH09] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12):2009, 2009.

[GFB+12] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):463–484, 2012.

[GGT16] Lorenzo Gatti, Marco Guerini, and Marco Turchi. Sentiwords: Deriving a high precision and high coverage lexicon for sentiment analysis. *IEEE Transactions on Affective Computing*, 7(4):409–421, 2016.

[GR06] Amir Globerson and Sam Roweis. Nightmare at test time: robust learning by feature deletion. In *Proceedings of the 23rd international conference on Machine learning*, pages 353–360. ACM, 2006.

[GSD+18] Katja Grace, John Salvatier, Allan Dafoe, Baobao Zhang, and Owain Evans. When will ai exceed human performance? evidence from ai experts. *Journal of Artificial Intelligence Research*, 62:729–754, 2018.

[GTK+19] Simon Gottschalk, Nicolas Tempelmeier, Günter Kniesel, Vasileios Iosifidis, Besnik Fetahu, and Elena Demidova. Simple-ml: Towards a framework for semantic data analytics workflows. In *International Conference on Semantic Systems*, pages 359–366. Springer, 2019.

[GV04] Hongyu Guo and Herna L Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *ACM Sigkdd Explorations Newsletter*, 6(1):30–39, 2004.

[HBGL08] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 1322–1328. IEEE, 2008.

[HF19] Christoph Hube and Besnik Fetahu. Neural based statement classification for biased language. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 195–203. ACM, 2019.

[HG09] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 2009.

[HIL+20] Hongxin Hu, Vasileios Iosifidis, Wentong Liao, Hang Zhang, Michael YingYang, Eirini Ntoutsi, and Bodo Rosenhahn. Fairnn-conjoint learning of fair representations for fair decisions. *Discovery Science*, 2020.

[HLJ16] William L Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*, 2016.

[HM13] Haibo He and Yunqian Ma. *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons, 2013.

[HPS+16] Moritz Hardt, Eric Price, Nati Srebro, et al. Equality of opportunity in supervised learning. In *NIPS*, pages 3315–3323, 2016.

[HZ11] Yulan He and Deyu Zhou. Self-training from labeled features for sentiment analysis. *Information Processing & Management*, 47(4):606–616, 2011.

[IFN19] Vasileios Iosifidis, Besnik Fetahu, and Eirini Ntoutsi. Fae: A fairness-aware ensemble framework. In *IEEE International Conference on Big Data*, pages 1375–1380, 2019.

[IN17] Vasileios Iosifidis and Eirini Ntoutsi. Large scale sentiment learning with limited labels. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1823–1832. ACM, 2017.

[IN18] Vasileios Iosifidis and Eirini Ntoutsi. Dealing with bias via data augmentation in supervised learning scenarios. *Jo Bates Paul D. Clough Robert Jäschke*, page 24, 2018.

[IN19a] Vasileios Iosifidis and Eirini Ntoutsi. Adafair: Cumulative fairness adaptive boosting. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 781–790, 2019.

[IN19b] Vasileios Iosifidis and Eirini Ntoutsi. Sentiment analysis on big sparse data streams with limited labels. *Knowledge and Information Systems*, pages 1–40, 2019.

[IN20] Vasileios Iosifidis and Eirini Ntoutsi. Fabboo - online fairness-aware learning under class imbalance. *Discovery Science*, 2020.

[Ino18] Hiroshi Inoue. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.

[ION17] Vasileios Iosifidis, Annina Oelschlager, and Eirini Ntoutsi. Sentiment classification over opinionated data streams through informed model adaptation. In *International Conference on Theory and Practice of Digital Libraries*, pages 369–381. Springer, 2017.

[IS16] David Ingold and Spencer Soper. Amazon doesnt consider the race of its customers. should it. *Bloomberg, April*, 2016.

[ITN19] Vasileios Iosifidis, Thi Ngoc Han Tran, and Eirini Ntoutsi. Fairness-enhancing interventions in stream classification. In *International Conference on Database and Expert Systems Applications*, pages 261–276. Springer, 2019.

[JKA01] Mahesh V Joshi, Vipin Kumar, and Ramesh C Agarwal. Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 257–264. IEEE, 2001.

[Joa99] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Icml*, volume 99, pages 200–209, 1999.

[JS02] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[KAAS12] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *ECML PKDD*, pages 35–50. Springer, 2012.

[KC09] Faisal Kamiran and Toon Calders. Classifying without discriminating. In *Computer, Control and Communication*, pages 1–6. IEEE, 2009.

[KC12] Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *KAIS*, 33(1):1–33, 2012.

[KCP10] Faisal Kamiran, Toon Calders, and Mykola Pechenizkiy. Discrimination aware decision tree learning. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 869–874. IEEE, 2010.

[KCWF12] Onur Kucuktunc, B Barla Cambazoglu, Ingmar Weber, and Hakan Ferhatosmanoglu. A large-scale sentiment analysis for yahoo! answers. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 633–642. ACM, 2012.

[KJ13] Max Kuhn and Kjell Johnson. *Applied predictive modeling*, volume 26. Springer, 2013.

[KK10] Max Kaufmann and Jugal Kalita. Syntactic normalization of twitter messages. In *International conference on natural language processing, Kharagpur, India*, 2010.

[KPP+17] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L Seltzer, and Sanjeev Khudanpur. A study on data augmentation of reverberant speech for robust speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5220–5224. IEEE, 2017.

[KSXPK17] Emmanouil Krasanakis, Eleftherios Spyromitros-Xioufis, Symeon Papadopoulos, and Yiannis Kompatsiaris. Tunable plug-in rules with reduced posterior certainty loss in imbalanced datasets. In *First International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pages 116–128, 2017.

[KXPK18] Emmanouil Krasanakis, Eleftherios Spyromitros Xioufis, Symeon Papadopoulos, and Yiannis Kompatsiaris. Adaptive sensitive reweighting to mitigate bias in fairness-aware classification. In *WWW*, pages 853–862. ACM, 2018.

[LD13] Michael Lucas and Doug Downey. Scaling semi-supervised naive bayes with feature marginals. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 343–351, 2013.

[LI16] Kristian Lum and William Isaac. To predict and serve? *Significance*, 13(5):14–19, 2016.

[LL98] Charles X Ling and Chenghui Li. Data mining for direct marketing: Problems and solutions. In *Kdd*, volume 98, pages 73–79, 1998.

[LMKA16] Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. How we analyzed the compas recidivism algorithm. *ProPublica (5 2016)*, 9, 2016.

[LRT11] Binh Thanh Luong, Salvatore Ruggieri, and Franco Turini. k-nn as an implementation of situation testing for discrimination discovery and prevention. In *SIGKDD*, pages 502–510. ACM, 2011.

[LSL+16] Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard S. Zemel. The variational fair autoencoder. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[LWZ09] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.

[LWZL11] Shoushan Li, Zhongqing Wang, Guodong Zhou, and Sophia Yat Mei Lee. Semi-supervised learning for imbalanced sentiment classification. In *IJCAI proceedings-international joint conference on artificial intelligence*, volume 22, page 1826, 2011.

[LYAH13] Yang Liu, Xiaohui Yu, Aijun An, and Xiangji Huang. Riding the tide of sentiment change: Sentiment analysis with evolving online reviews. *World Wide Web*, 16(4):477–496, July 2013.

[LYWZ04] Charles X Ling, Qiang Yang, Jianning Wang, and Shichao Zhang. Decision trees with minimal costs. In *Proceedings of the twenty-first international conference on Machine learning*, page 69, 2004.

[LZX+13] Shenghua Liu, Wenjun Zhu, Ning Xu, Fangtao Li, Xue-qi Cheng, Yue Liu, and Yuanzhuo Wang. Co-training and visualizing sentiment evolvement for tweet events. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 105–106. ACM, 2013.

[M+67] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[Mar61] Melvin Earl Maron. Automatic indexing: an experimental inquiry. *Journal of the ACM (JACM)*, 8(3):404–417, 1961.

[MBY+16] Xiangrui Meng, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman, DB Tsai, Manish Amde, Sean Owen, et al. Mllib: Machine learning in apache spark. *Journal of Machine Learning Research*, 17(34):1–7, 2016.

[MCI+18] Damianos P Melidis, Alvaro Veizaga Campero, Vasileios Iosifidis, Eirini Ntoutsi, and Myra Spiliopoulou. Enriching lexicons with ephemeral words for sentiment analysis in social streams. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, page 38. ACM, 2018.

[MCPZ18] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. *arXiv preprint arXiv:1802.06309*, 2018.

[MGL09] Prem Melville, Wojciech Gryc, and Richard D Lawrence. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1275–1284. ACM, 2009.

[MIE+18] Nilamadhaba Mohapatra, Vasileios Iosifidis, Asif Ekbal, Stefan Dietze, and Pavlos Fafalios. Time-aware and corpus-specific entity relatedness. *arXiv preprint arXiv:1810.10004*, 2018.

[MKZ13] Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *arXiv preprint arXiv:1308.6242*, 2013.

[MLH04] Beatriz Maeireizo, Diane Litman, and Rebecca Hwa. Co-training for predicting emotions with spoken dialogue data. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 202–205, 2004.

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[MSN18] Damianos P. Melidis, Myra Spiliopoulou, and Eirini Ntoutsi. Learning under feature drifts in textual streams. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 527–536, New York, NY, USA, 2018. ACM.

[NEK+16] Nikolaos Nikolaou, Narayanan Edakunni, Meelis Kull, Peter Flach, and Gavin Brown. Cost-sensitive boosting algorithms: Do we really need them? *Machine Learning*, 104(2-3):359–384, 2016.

[NFG+20] Eirini Ntoutsi, Pavlos Fafalios, Ujwal Gadiraju, Vasileios Iosifidis, Wolfgang Nejdl, Maria-Esther Vidal, Salvatore Ruggieri, Franco Turini, Symeon Papadopoulos, Emmanouil Krasanakis, et al. Bias in data-driven artificial intelligence systemsan introductory survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, page e1356, 2020.

[NG00] Kamal Nigam and Rayid Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the ninth international conference on Information and knowledge management*, pages 86–93. ACM, 2000.

[Nig01] Kamal P Nigam. Using unlabeled data to improve text classification. Technical report, Carnegie-mellon univ Pittsburgh pa school of computer science, 2001.

[NMM06] Kamal Nigam, Andrew McCallum, and Tom Mitchell. Semi-supervised text classification using em. *Semi-Supervised Learning*, pages 33–56, 2006.

[NMTM00] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000.

[otPP14] United States. Executive Office of the President and John Podesta. *Big data: Seizing opportunities, preserving values*. White House, Executive Office of the President, 2014.

[Par14] R Parikh. Garbage in, garbage out: How anomalies can wreck your data, 2014.

[PL04] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.

[PL05] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics, 2005.

[PL+08] Bo Pang, Lillian Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, January 2008.

[PLV02] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, EMNLP, pages 79–86, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics, ACL.

[PNS+10] Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th international conference on World wide web*, pages 751–760. ACM, 2010.

[PRT09a] Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Measuring discrimination in socially-sensitive decision records. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pages 581–592. SIAM, 2009.

[PRT09b] Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Measuring discrimination in socially-sensitive decision records. In *SDM*, pages 581–592. SIAM, 2009.

[PRW+17] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. In *NIPS*, pages 5680–5689, 2017.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[PT10] Georgios Paltoglou and Mike Thelwall. A study of information retrieval weighting schemes for sentiment analysis. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 1386–1395. Association for Computational Linguistics, 2010.

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[Qui91] J. Ross Quinlan. Improved estimates for the accuracy of small disjuncts. *Machine Learning*, 6(1):93–98, 1991.

[Qui14] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[QWZZ13] Zhenxing Qin, Alan Tao Wang, Chengqi Zhang, and Shichao Zhang. Cost-sensitive classification with k-nearest neighbors. In *International Conference on Knowledge Science, Engineering and Management*, pages 112–131. Springer, 2013.

[RPT10] Salvatore Ruggieri, Dino Pedreschi, and Franco Turini. Data mining for discrimination discovery. *TKDD*, 4(2):9, 2010.

[RR14] Andrea Romei and Salvatore Ruggieri. A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review*, 29(5):582–638, 2014.

[RV95] Joel Ratsaby and Santosh S Venkatesh. Learning from a mixture of labeled and unlabeled examples with parametric side information. In *Proceedings of the eighth annual conference on Computational learning theory*, pages 412–417, 1995.

[Sch99] Robert E Schapire. A brief introduction to boosting. In *IJCAI*, volume 99, pages 1401–1406, 1999.

[Sch13] Robert E Schapire. Explaining adaboost. In *Empirical inference*, pages 37–52. Springer, 2013.

[SCH16] Nadia Felix F Da Silva, Luiz FS Coletta, and Eduardo R Hruschka. A survey and comparative study of tweet sentiment analysis via semi-supervised learning. *ACM Computing Surveys (CSUR)*, 49(1):15, 2016.

[SG15] Jan Schlüter and Thomas Grill. Exploring data augmentation for improved singing voice detection with neural networks. In *ISMIR*, pages 121–126, 2015.

[SKVHN09] Chris Seiffert, Taghi M Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 40(1):185–197, 2009.

[SKWW07] Yanmin Sun, Mohamed S Kamel, Andrew KC Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378, 2007.

[SNZ16] Myra Spiliopoulou, Eirini Ntoutsi, and Max Zimmermann. Opinion stream mining. *Encyclopedia of Machine Learning and Data Mining*, pages 1–10, 2016.

[SPW+13] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[SR15] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[SS99] Robert E Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine learning*, 37(3):297–336, 1999.

[SS15] Surendra Sedhai and Aixin Sun. Hspam14: A collection of 14 million tweets for hashtag-oriented spam research. In *SIGIR*, pages 223–232. ACM, 2015.

[SSM11] Jiang Su, Jelber S Shirab, and Stan Matwin. Large scale text classification using semi-supervised multinomial naive bayes. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 97–104. Citeseer, 2011.

[Ste13] Jerzy Stefanowski. Overlapping, rare examples and class decomposition in learning classifiers from imbalanced data. In *Emerging paradigms in machine learning*, pages 277–306. Springer, 2013.

[STM+18] Samira Samadi, Uthaipon Tantipongpipat, Jamie H Morgenstern, Mohit Singh, and Santosh Vempala. The price of fair pca: One extra dimension. In *NIPS*, pages 10976–10987, 2018.

[STR+18] Hoo-Chang Shin, Neil A Tenenholtz, Jameson K Rogers, Christopher G Schwarz, Matthew L Senjem, Jeffrey L Gunter, Katherine P Andriole, and Mark Michalski. Medical image synthesis for data augmentation and anonymization using generative adversarial networks. In *International workshop on simulation and synthesis in medical imaging*, pages 1–11. Springer, 2018.

[Swe13] Latanya Sweeney. Discrimination in online ad delivery. *arXiv preprint arXiv:1301.6822*, 2013.

[Tin00] Kai Ming Ting. A comparative study of cost-sensitive boosting algorithms. In *In Proceedings of the 17th International Conference on Machine Learning*. Citeseer, 2000.

[TKMS03] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.

[TV14] Pablo A Tapia and Juan D Velásquez. Twitter sentiment polarity analysis: A novel approach for improving the automated labeling in a text corpora. In *International Conference on Active Media Technology*, pages 274–285. Springer, 2014.

[Twa09] Bhekisipho Twala. An empirical comparison of techniques for handling incomplete data using decision trees. *Applied Artificial Intelligence*, 23(5):373–405, 2009.

[UBM+18] Vishnu Unnikrishnan, Christian Beyer, Pawel Matuszyk, Uli Niemann, Rüdiger Pryss, Winfried Schlee, Eirini Ntoutsi, and Myra Spiliopoulou. Entity-level stream classification: exploiting entity similarity to label the future observations referring to an entity. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 246–255. IEEE, 2018.

[Utg89] Paul E Utgoff. Incremental induction of decision trees. *Machine learning*, 4(2):161–186, 1989.

[Vap99] Vladimir N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

[VHLY15] Keyon Vafa, Christian Haigh, Alvin Leung, and Noah Yonack. Price discrimination in the princeton reviews online sat tutoring service. *JOTS Technology Science, Sep*, 1, 2015.

[VL13] Donna Vakharia and Matthew Lease. Beyond amt: An analysis of crowd work platforms. *arXiv preprint arXiv:1310.1672*, 2013.

[VR18] Sahil Verma and Julia Rubin. Fairness definitions explained. 2018.

[WC03] Gang Wu and Edward Y Chang. Class-boundary alignment for imbalanced dataset learning. In *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, pages 49–56, 2003.

[Wik] Wikipedia. Discrimination. https://en.wikipedia.org/wiki/Discrimination.

[WM12] Sida Wang and Christopher D Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Short papers-volume 2*, pages 90–94. Association for Computational Linguistics, 2012.

[WZNS15] Sebastian Wagner, Max Zimmermann, Eirini Ntoutsi, and Myra Spiliopoulou. Ageing-based multinomial naive bayes classifiers over opinionated data streams. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 401–416. Springer, 2015.

[XWDL15] Rui Xia, Cheng Wang, Xin-Yu Dai, and Tao Li. Co-training for semi-supervised sentiment classification based on dual-view bags-of-words representation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1054–1063, 2015.

[YWLZ17] Liang-Chih Yu, Jin Wang, K Robert Lai, and Xuejie Zhang. Refining word embeddings for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 534–539, 2017.

[YZL09] Qiang Ye, Ziqiong Zhang, and Rob Law. Sentiment classification of online reviews to travel destinations by supervised machine learning approaches. *Expert systems with applications*, 36(3):6527–6535, 2009.

[ZCP05] Richard S Zemel and Miguel Á Carreira-Perpiñán. Proximity graphs for clustering and manifold learning. In *Advances in neural information processing systems*, pages 225–232, 2005.

[ZE01] Bianca Zadrozny and Charles Elkan. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 204–213, 2001.

[ZGBD09] Xiaojin Zhu, Andrew B. Goldberg, Ronald Brachman, and Thomas Dietterich. *Introduction to Semi-Supervised Learning*. Morgan and Claypool Publishers, 2009.

[ZHY+16] Li Zhao, Minlie Huang, Ziyu Yao, Rongwei Su, Yingying Jiang, and Xiaoyan Zhu. Semi-supervised multinomial naive bayes for text classification by leveraging word-level statistical constraint. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[ZN19] Wenbin Zhang and Eirini Ntoutsi. Faht: An adaptive fairness-aware decision tree classifier. *To appear in International Joint Conferences on Artificial Intelligence (IJCAI) proceedings*, 2019.

[ZNS14] Max Zimmerann, Eirini Ntoutsi, and Myra Spiliopoulou. A semi-supervised self-adaptive classifier over opinionated streams. In *2014 IEEE International Conference on Data Mining Workshop*, pages 425–432. IEEE, 2014.

[ZTZX14] Mi Zhang, Jie Tang, Xuchen Zhang, and Xiangyang Xue. Addressing cold start in recommender systems: A semi-supervised co-training algorithm. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 73–82. ACM, 2014.

[ZVGRG17] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1171–1180. WWW, 2017.