

**LIBERAL OR CONSERVATIVE: EVALUATION AND
CLASSIFICATION WITH DISTRIBUTION AS GROUND TRUTH**

by

Daniel Xiaodan Zhou

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Information)
in the University of Michigan
2013

Doctoral Committee:

Professor Paul Resnick, Chair
Assistant Professor Eytan Adar
Assistant Professor Jowei Chen
Assistant Professor Qiaozhu Mei
Associate Professor Rahul Sami

Dedication

To my family and my wife Allison

Acknowledgements

I am deeply appreciative of the many individuals who have supported my work and continually encouraged me through the writing of this dissertation. Without their time, attention, encouragement, thoughtful feedback, and patience, I would not have been able to see it through.

Above all, I would like to thank my advisor, Paul Resnick, for his inspirational and timely advice and constant encouragement (as well as occasional references to "Dr. Zhou") over the last several years. I have learned a great deal from his unique perspective on research, his sharp insight on almost any issue, and his personal integrity and expectations of excellence. He has been a great advisor for me outside of the academic world as well. He has always been patient when explaining the nuances of English words in writing, and has shared with me many witty jokes, metaphors, and lessons on how to strive for work-life balance. I really appreciate his support of my unconventional decision to choose an entrepreneurial career path after graduation.

I am very fortunate to have had Rahul Sami, Qiaozhu Mei, Eytan Adar and Jowei Chen serve on my committee. Rahul has been like a second advisor to me, ready with brilliant ideas, honest advice and encouraging words whenever I needed them; I enjoy working with him on recommender systems and other areas. Qiaozhu is both a mentor and a good friend, from whom I have learned many exciting machine learning models and techniques. Eytan and Jowei are great sources of inspiration regarding US politics and crowdsourcing. I'm looking forward to working with them more in the future.

I am indebted to other faculty members in the School of Information: Yan Chen and Jeff MacKie-Mason for introducing me to the fascinating field of information economics and incentive-centered design; Victor Rosenberg for teaching me information entrepreneurship; Steve Jackson for teaching me information policy analysis; Lada Adamic for teaching me network analysis; Chuck Severance for instructing me how to teach; Michael Cohen, Mark Ackerman, and Margaret Hedstrom for encouraging me to think about Information Science as a field and teaching me a broad range of research methodologies and topics at the beginning of the doctoral program. Also, I want to thank Xiao-Wen Zou, Sue Schuon, Jen Todd and other SI staff for their

warm-hearted support and encouragement, and Michael Hess, Mike Doa, Dennis Hogan and John Lockard from SI Computing for their support with computer equipment.

I am grateful to other faculty members and mentors outside of SI. In particular, I want to thank Mary Gallagher from the department of Political Science for introducing me to China studies from Western perspectives. Even though that part of my work was not incorporated into this dissertation, the learning experience was tremendously fulfilling. My appreciation also goes to the mentors and staff from U of M TechTransfer, Center for Entrepreneurship, Zell Institute for Entrepreneurial Studies, Ann Arbor SPARK, and NSF I-Corps program. Particularly, I want to thank Wesley Huffstutter for his constant support and for believing in me and in my venture, Doug Dockard for helping me understand issues related to intellectual property, and Norm Rapino for helping me with i-corps. I would like to thank Marty Byle, Will Stone, and Jack Miner for being my business mentors. Thanks to Kieran Lal, Gerhard Killesreiter and the Drupal.org infrastructure team for helping with the part of my work on recommender systems.

My fellow students' support has been continuous fuel during my long journey in finishing this doctoral program. Particularly, I'd like to thank my research group members, all under the supervision of Paul Resnick: Sean Munson, Will Riley, Tapan Khopkar, Sidharth Chhabra, Souneil Park, Nate Oostendorp and Erica Willar. I'd like to thank Qiaozhu's research group with whom I regularly attend to meetings and parties: Lei Yang, Yang Liu, Xin Rong, Jian Tang, Zhe Zhao, and Danny Wu, among others; and the MISC group: Tao Dong and Jessica Hullman, among others. I'd like to thank my cohort for staying by my side: Ayse Buyuktur, Eytan Bakshy, David Lee, Morgan Daniels, Sean Munson (again) and Xingxing Yao. Thanks to SI students who graduated before me and their advice on completing the PhD program: John Lin, Kevin Nam, Jiang Yang, Xiaomu Zhou, Ben Congleton, Benjamin Chiao, Jian Lian, and Tracy liu. Thanks to the 2008 MSI students cohort and other SI students I have worked with: Pae Chongthammakun, Grace Young-Joo Jeon, Yuanqing You, Xiaomin Zhang, Lingyun Xu, Sarvagya Kochak, and Adam Pierce. Thanks to Abe Gong, Isis Li and Gang Su for sharing my passion for entrepreneurship. Special thanks go to my good friend and collaborator Ya-Wen Lei, who has worked with me extensively on China related studies.

Life would not have been as colorful without the many good friends I met in Ann Arbor. I would like to extend my thanks to Zhichen Zhao, Zhou Du, Youjian Chi, Xi Chen, Fuyuan Wang, Kai Zhao, Ming Xu, Beilei Zhang, Yuxing Yun, and my companions of the 2007-2008 Chinese Students and Scholars Association working team. Our friendship is built not only on the many social gatherings we attended together, but also on the many values we share. I especially enjoy the various "salon discussions" organized by Lei Zhong, Benjamin Chiao (again), and Yizi Zhang of the AA Chinese Intellectuals, Lijing Yang and Xu Li of the Michigan Chinese Fellow, Jingchen Wu, Shuai Niu and Zhichen Zhao (again) of the AA Saturday Salon, and Huang Wei and Shinan Li of the Chinese News Club.

Many people helped me and offered their friendship before I joined graduate school. I want to thank my former colleagues in IBM China: Ning Wang, Mingliang Guo, Xiucheng Wu, Guangxin Xu, and Christine Smith; the mentors of my undergraduate research project at Tsinghua University: Lizhu Zhou and Chunxiao Xing; my mentors and friends at Shandong University: Haiyang Wang, Qian Zhu, Wei Zhang, Chengcheng Wang and Chunguang Qu; childhood friends from middle school who grew up with me: Wenhao Xu, Han Chen, Shan Cong, Feng Yue, Gang Wang and Tian Yang. Even though the ocean separates us, thinking about them always brings a warm smile.

Last but not least, I want to thank my loving and caring family. Thanks to my father Jianqing Zhou and my mother Ping Dong for teaching me to be curious and sincere to life, and for always being there for me. Thanks to my in-laws, Greg Richards and Mary Anne Santoro, for giving me constant love and encouragement and for treating me as their own son. Thanks to my sister-in-law, Bridget, for being such a sweet and uplifting little sister. Words cannot express my appreciation and love for my children. Thanks to Lan-Lan for being cute, making me laugh and being the great big sister that you are. Thanks to Nino for being healthy and eating well. To my wife, Allison, thank you for patiently editing this dissertation, for always supporting me in my academic pursuits and for the wonderful life that we share together.

Table of Contents

Dedication	ii
Acknowledgements	iii
List of Tables	viii
List of Figures	ix
List of Equations	xi
Abstract	xii
Chapter 1. Introduction	1
1.1 Motivation.....	1
1.2 Political Leaning Classification.....	4
1.3 Outline.....	9
Chapter 2. Distribution as Ground Truth	12
2.1 Introduction	12
2.2 Related Work.....	15
2.3 Objective Classification Problems and Label as Ground Truth.....	23
2.4 Subjective Classification Problems and Distribution as Ground Truth	26
2.5 Mapping Distributions into Labels	37
2.6 Discussion and Limitation.....	48
Chapter 3. Annotation Elicitation	53
3.1 Related Work.....	54
3.2 Data Collection	61
3.3 Results	65
3.4 Conclusions and Future Work	69
Chapter 4. Classifier Evaluation with Distributions	72
4.1 Related Work.....	74
4.2 Classifier Evaluation Schemes	78
4.2.1 Ψ_H : Direct Evaluation with Distributions	78
4.2.2 Ψ_1 : Evaluation with Labels Mapped From Distributions	80
4.2.3 Ψ_C : Evaluation under Label as Ground Truth	82
4.3 Problem Formulation	83
4.3.1 Ranking of Classifiers	84
4.3.2 Evaluating Evaluation Schemes	86

4.3.3	<i>Simulation Design</i>	87
4.4	Simulation Results	92
4.5	Discussion	97
4.6	Conclusions and Future Work	102
Chapter 5. LabelPropagator: A Semi-Supervised Classifier		107
5.1	Related Work	108
5.2	Semi-Supervised Learning Algorithms	110
5.2.1	<i>Problem Formulation</i>	110
5.2.2	<i>Random Walk with Restart (RWR)</i>	111
5.2.3	<i>Local Consistency Global Consistency (LCGC)</i>	111
5.2.4	<i>Absorbing Random Walk (ARW)</i>	112
5.3	Original Study	114
5.3.1	<i>Datasets</i>	114
5.3.2	<i>Evaluation</i>	117
5.3.3	<i>Discussion</i>	121
5.4	Follow-up Study	126
5.4.1	<i>Datasets</i>	127
5.4.2	<i>Evaluation</i>	129
5.4.3	<i>Discussion</i>	134
5.5	Conclusions and Future Work	136
Chapter 6. Closing Remarks		140
6.1	Summary	140
6.2	Future Work	142
6.3	Broader Impact	146
References		148

List of Tables

Table 1. Political leaning measurement	5
Table 2. Comparisons between the objective and subjective classification problems	35
Table 3. Cost function.....	38
Table 4. Comparison of cost functions and partitioning	44
Table 5. Heuristics that associate labels to distributions	47
Table 6. Cost functions and heuristics.....	47
Table 7. Differences among evaluation schemes	83
Table 8. Simulated datasets	88
Table 9. Comparisons of classifiers' ranking under different evaluation scheme	93
Table 10. Minimum n required to get $\tau \geq 0.9$ for C^{main}	96
Table 11. Probability of correctly discriminating the first classifier as better	99
Table 12. High recall for red; high precision for blue	118
Table 13. Blog sources are useful; not blog links.....	119
Table 14. Algorithms comparison.....	121
Table 15. Result of SVM.....	123
Table 16. Result of 1-level propagation on , with RWR	124
Table 17. AMT annotations in 2010, 2011, and 2012.....	128
Table 18. Confusion matrix: classification outcome changes.....	132
Table 20. Confusion matrix: classification results from different classifiers	133
Table 21. Accuracy evaluated against $L_{\text{mturk-2012}}$ using Ψ_1	134
Table 22. Comparison between LabelPropagator and text analysis classifiers	139

List of Figures

Figure 1. Illustration of displaying classification results in a news aggregator application	7
Figure 2. Outline of the thesis in chronicle order	11
Figure 3. Relationship of chapters 2, 3, and 4	14
Figure 4. Simplex notation for H_i	28
Figure 5. Bar chart notation for H_i	28
Figure 6. Ground truth estimation errors	29
Figure 7. A simple codebook example on political leaning classification	32
Figure 8. Define ground truth from the population	35
Figure 9. Simplex partitioning for the absolute scoring rule	42
Figure 11. Partitioning disputed region, highlighted in yellow: (a) between 0-1 and quadratic; (b) between absolute and quadratic	48
Figure 12. Using the wrong ground truth model	48
Figure 13. Screenshots of AMT task	63
Figure 14. AMT qualification test	64
Figure 15. Number of articles labeled by workers	66
Figure 16. Articles plotted on the simplex according to their distributions	67
Figure 17. Number of items (y-axis) according the number of disagreement labels (x-axis)	67
Figure 18. Partitioning difference between majority vote and quadratic	68
Figure 19. Illustration of partitioning when the number of labels per item is small: (a) $m = 4$, (b) $m = 1$	68
Figure 20. Intuition about the values of τ .	87
Figure 21. Two simulated classifier families	90
Figure 22. C_1 degrades to C_0 for C^{main} and C^{coder} : (a) Ψ_H , (b) Ψ_1	92
Figure 23. τ changes according to n , with fixed m	94
Figure 24. τ changes according to m , with fixed $n = 1000$	95

Figure 25. Minimum $k = m \times n$ required for C^{main} on I_{main}	97
Figure 26. Intuition of LabelPropagator	108
Figure 27. Three versions of the semi-supervised learning algorithms.....	113
Figure 28. (a) optimal $\alpha = 0.3$ (b) $V_{\text{source}}/E_{\text{source}}$ helpful, with optimal weight = 50 (c) E_{user} not helpful (d) E_{story} not helpful.....	120
Figure 29. Labeled datasets overlap.....	129
Figure 30. Co-SVM illustration.....	139

List of Equations

Equation 1. Definition of $P_1(i)$	27
Equation 2. Definition of H_1	27
Equation 3. Estimation: $H_i, P_1(i)$	29
Equation 4. Definition of cost.....	38
Equation 5. Total expected cost	39
Equation 6. Total expected cost over sample ground truth.....	39
Equation 7. Cost-minimizing label for item i under the absolute cost function.....	40
Equation 8. Cost-minimizing label for item i under the quadratic cost function	41
Equation 9. Linearity.....	42
Equation 10. Generalized rules of partitioning.....	43
Equation 11. Normalized accuracy under Ψ_H	79
Equation 12. Normalized accuracy defined as regret.....	80
Equation 13. Normalized accuracy under Ψ_1	81
Equation 15. Ψ as a function	84
Equation 16. Evaluating Ψ	86
Equation 17. Two steps evaluation of Ψ	86

Abstract

The ability to classify the political leaning of a large number of articles and items is valuable to both academic research and practical applications. The challenge, though, is not only about developing innovative classification algorithms, which constitutes a “classifier” theme in this thesis, but also about how to define the “ground truth” of items’ political leaning, how to elicit labels when labelers do not agree, and how to evaluate classifiers with unreliable labeled data, which constitutes a “ground truth” theme in the thesis.

The “ground truth” theme argues for the use of distributions (e.g., 0.6 conservative, 0.4 liberal) instead of labels (e.g, conservative, liberal) as the underlying ground truth of items’ political leaning, where disagreements among labelers are not human errors but rather useful information reflecting the distribution of people’s subjective opinions. Empirical data demonstrate that distributions are dispersed: there are many items upon which labelers simply do not agree. Therefore, mapping distributions into single labels requires more than just majority vote. Also, one can no longer assume the labels from a few labelers are reliable because a different small sample of labelers might yield a very different picture.

However, even though individual labeled items are not reliable, simulation suggests that we may still reliably evaluate and rank classifiers, as long as we have a large number of labeled items for evaluation. The optimal way is to obtain one label per item with many items (e.g., 1000~3000) for evaluation.

The “classifier” theme proposes the LabelPropagator algorithm that propagates the political leaning of known articles and users to the target nodes in order to classify them. LabelPropagator achieves higher accuracy than the alternative classifiers based on text analysis, suggesting that a relatively small number of labeled people and stories, together with a large number of people to item votes, can be used to classify the other people and items. An article’s source is useful as an input for propagation, while text similarities, users’ friendship, and “href” links to articles are not.

Chapter 1. Introduction

1.1 Motivation

Observers are concerned about the increasing political polarization of our society, with opposing groups unable to engage in civil dialogue to find common ground or solutions. Sunstein (2011) and others have argued that, as people have more choices about their news sources, they will live in echo chambers. Republicans and Democrats read different newspapers and political books (Krebs, 2008), watch different TV news stations, and even live in different places (Bishop, 2008). If people prefer to avoid hearing challenging views, we may see even greater political fragmentation as people get better tools to filter the media they consume based on their own reactions and the reactions of other people like them.

In recent years, a burst of research has studied how to mitigate media bias and encourage citizens to consume a balanced mixture of political opinions. For example, Park et al (2008, 2009, 2011) proposed the NewsCube system to mitigate media bias by showing multiple aspects of news stories. Matlin et al (2010) developed a system to track Slate Magazine readers' liberal and conservative news consumption. Munson, Zhou and Resnick (2009) proposed a simple news aggregator that displays a balanced list of liberal and conservative articles. Munson and Resnick (2010) studied how to present diverse political opinions to readers that are challenging to their existing opinions but not too challenging to upset them.

One key to studies regarding media bias and polarization is the ability to classify the political leaning of a large number of articles. Other studies also require political leaning classification on various types of items. One line of work is about user experience with

political leaning annotations. For example, Gamon et al (2008) built the BLEWS system and argued that displaying the political leaning of articles and other contextual information would improve users' reading experience. Oh, Lee and Kim (2009) studied users' experience by showing the political leaning of articles in search engine results. Another line of work is about using the political leaning of articles, tweets and people to study political and social phenomena. For example, Levine et al (2011) used politicians' political affiliation to study the structure and cohesiveness of their twitter messages, and predicted candidate victory with high accuracy. Conover et al (2012) used the political leaning of twitter users and messages to study the differences of conservative and liberal users' online behavior.

All in all, the ability to classify the political leaning of a large number of articles and items is critical to many studies. It is desirable to design computer algorithms to classify the political leaning of items automatically. Many algorithms have been proposed over the past two decades, but they have two major limitations.

The foremost and fundamental limitation is the lack of clear definitions of "conservative" and "liberal" to serve as the "ground truth" of articles' political leaning. Most studies never discussed the meaning of "conservative" and "liberal" and simply adopted existing labeled datasets from multiple third party sources to train and evaluate classifiers without checking the validity, consistency and reliability of the datasets (e.g. Jiang and Argamon 2008). Other studies gave vague, arbitrary definitions of "conservative" and "liberal", and assumed that they could rely on these unclear definitions and usually poorly labeled data to optimize their classifiers (e.g. Oh, Lee and Kim 2009). As I will discuss later, political leaning is inherently subjective in that different people have their own interpretations. It is questionable to draw conclusions about the classifiers when the labeled data to train and evaluate the classifiers are unreliable.

The second limitation is that most existing political leaning classifiers only used text analysis approaches (e.g., Oh et al, 2009). Text-based approaches might work well when

different categories are associated with drastically different sets of keywords, or when there is lots of training data. As I will discuss later, these are not usually true for political leaning classification on individual articles. In addition, text-based approaches are not able to classify the political leaning of non-textual items such as people, images, or video.

This thesis aims to tackle these two limitations in order to develop and correctly evaluate political leaning classifiers with high accuracy. It has two main themes in response to the two limitations in prior literature. The first theme is to study how to define the “ground truth” of articles’ political leaning and how to evaluate classifiers with inaccurate labeled data. This theme is mainly discussed in chapters 2, 3 and 4, and I will call it the “ground truth” theme. The second theme is to develop a political leaning classifier that does not rely on text analysis. It is mainly discussed in chapter 5, and I will call it the “classifier” theme. I will discuss more about the two themes in sections 1.3 and 1.4.

The intended target audience of the thesis are media bias researchers, political and social scientists, human computer interaction designers, and machine learning developers and practitioners, who want to develop and evaluate algorithms that classify the political leaning of a large number of items. Machine learning researchers and practitioners might also be interested in the “ground truth” theme about how to obtain annotations properly in practice for classifier evaluation.

The rest of this introductory chapter is organized as follows. In section 1.2, I will discuss the problem of political leaning classification, and introduce the more general class, subjective classification problems. In section 1.3, I will summarize the content of each chapter, and discuss the relationships between the chapters.

Throughout the entire thesis, I will use the following terminology. According to the current convention in U.S. media, I will color-code *conservative*, *liberal* and *others* into

red, blue and *gray*¹ respectively, and use the colors and political labels inter-changeably. These labels are also called *classes* or *categories*. The political articles and artifacts to be labeled or classified are also called *items, instances, objects* or *examples*. I will use the terms *coders, raters, labelers, annotators, or assessors* inter-changeably to refer to those people who label the political leaning of items. The results produced by the human coders are called *annotations, ratings, labels, codes, or assessments* interchangeably.

1.2 Political Leaning Classification

Conservative versus Liberal Dichotomy

In U.S. politics, opinions on a variety of issues involving taxes, the role of government, domestic policy, and international relations are substantially though imperfectly correlated with each other and with party affiliation and with an overall self-identification as liberal or conservative. Thus, classifying people, media outlets, and opinions expressed in individual articles as liberal or conservative conveys meaning to most people.

The liberal versus conservative classification scheme has its critics (e.g., Klein and Stern 2008). One reason is that the single dimension cannot capture cases such as libertarians or populists who align with liberals on some issues and conservatives on others. Another is that definitions of conservative and liberal are vague and inconsistently applied.

Moreover, many political news and opinion articles express a mixture of conservative and liberal ideology. Thus, not everyone will agree about the correct classification of particular items, or even the correct classification of their own stance.

Despite these fuzzy boundaries, however, the one-dimensional classification scheme persists in our discourse, and many people, articles, and news sources fit clearly into one category or the other. The ability to classify blogs as liberal or conservative enabled

¹ Purple is the conventional color for independent ideology, but gray is used to denote anything other than red or blue.

Adamic and Glance (2005) to analyze patterns of inter-linking between them. It also served as the basis for investigating people’s preferences for difference mixtures of reinforcing and challenging articles in a news aggregator (Munson and Resnick 2010) and for sorted or annotated displays (Gamon et al 2008; Oh et al 2009; Munson and Resnick 2010). Just like the other studies, this thesis is also based on the conservative versus liberal dichotomy.

Political Leaning Measurement

Researchers could apply several types of metrics to measure political leaning based on the conservative versus liberal dichotomy. Table 1 lists four possibilities. The columns specify whether political leaning should be measured by a range of continuous scores or by a limited set of labels. The rows specify whether to measure political leaning in two dimensions, treating “red-ness” and “blue-ness” as two orthogonal dimensions of articles, or in a single dimension, treating political leaning as a single spectrum.

Table 1. Political leaning measurement

	Continuous scores	Discrete labels
Two dimensions	E.g., <red=0.8, blue=0.2>, <red=0, blue=0.9>	E.g., red, blue, both, neither
Single dimension	E.g., -0.6 (red), 0.05 (gray), 0.9 (quite blue)	E.g., red, gray, blue

Political leaning measurement are used in three situations: 1) representing the true political leaning of items, 2) having human coders annotate the political leaning of items, and 3) having classifiers classify the political leaning of items. Researchers usually use the same measurement across all three situations, but it is not necessary. For example, we could treat the underlying political leaning of items as continuous scores (such as 0.9, -0.6, etc.), yet classify them into a small number of categories (such as red, blue, etc.) rather than assigning scores to them. Or, we could have human coders annotate items in two dimensions, and then collapse the annotations into a signal dimension to represent the true political leaning of items.

In this thesis, I will design political leaning classifiers to label items as one of red, gray, and blue, which is a “single dimension, discrete labels” measurement. The reason to use this particular metrics as classification output is discussed in the next sub-section. I will also have human coders annotate the political leaning of items as red, gray and blue. Conventionally, the underlying political leaning of items should be conveniently treated as red, gray and blue as well. But this thesis proposes the “distribution as ground truth” model, which treats the underlying true political leaning of an item as a distribution over red, gray and blue dimensions. I will discuss this model in chapter 2 and explain why it is better than simply treating the underlying model as discrete labels of red, gray and blue.

Classification Output

As mentioned earlier, political leaning classifiers in this thesis are designed to classify items into red, gray and blue. This sub-section explains why it is preferable not to use other measurements in Table 1 for classification output. Recall that the very motivation to design a political leaning classifier is to be able to use the classification results in applications such as media bias study or user interface design. The particular requirement of an application will determine the right measurement of political leaning for classification output.

This thesis is born out of the need to generate political leaning annotations to display together with articles in a news aggregator, such as the one illustrated in Figure 1. The end users will directly view the classification results while they read political articles. Therefore, the measurement of political leaning should be intuitive to users and not cause confusion.



Figure 1. Illustration of displaying classification results in a news aggregator application

I'd like to argue that the two dimensions measurement (first row of Table 1) is not optimal for this typical application scenario. First, it is quite confusing to the end users about having both red and blue ideological dimensions: for example, how does one distinguish between "both" and "neither"? In addition, the red and blue dimensions are not usually orthogonal, but are often negatively correlated. In other words, an article that has a high score in the red dimension usually has a low score in the blue dimension, and vice versa. So it might be natural to reduce it into a single dimension. Finally, the two dimensions measurement is not commonly known in existing literature, and thus is inconvenient to use.

Also I'd like to argue that the continuous scores measurement (first column of Table 1) is not optimal for the application scenario because it could cause confusion. Suppose a user views a 0.9 article and a 0.8 article (where 1 means total blue and -1 means total red), how does the user interpret the 0.1 unit of difference between the two articles? Furthermore, the scores -0.9 and -0.7 has 0.2 unit of difference, which has the same difference as -0.1 and 0.1. But to the end users, the former case might be less different than the latter case because the latter case changes from slightly red to slightly blue. In

short, using continuous scores as classifiers' outputs is less straightforward and could cause problems.

Finally, I'd like to argue that using three labels – red, gray and blue – is more suitable to the application scenario than using other sets of labels. Using only two labels, red and blue, would be too limiting and doesn't cover many cases that are not clearly red or blue. Using more than three labels – for example, “strong-red, weak-red, gray, weak-blue, strong-blue” or 7 points Likert scale – would introduce too many categories than necessary, which demands more “cognitive effort” (see chapter 2 “related work” in page 22) from users who will read the classification results.

Note that the argument above – that is, to show red, gray and blue labels as items' political leaning to end users – is based on discussions with a few researchers and end users. Future work should practice a user-centered approach to study whether it is indeed optimal to display red, gray and blue labels to end users, which will then determine how to design the classification output.

Subject Classification Problems

The political leaning classification problem as I have discussed above is a typical case of a more general set of classification problems, subjective classification problems. For a subjective classification problem, people often have their own different subjective opinions on the correct labels of items, and therefore it is not clear how to have human annotators label items correctly. If the labeled items are not reliable, then it is questionable to design and evaluate classification algorithms correctly with the unreliable labeled dataset. Such subjective classification problems include word sense disambiguation, twitter message classification, spam detection, image labeling, and many more.

The “ground truth” theme of this thesis is to use the political leaning classification problem as an example to study how to define the underlying ground truth model, how to elicit annotations, and how to evaluate classifiers for subjective classification

problems. I will discuss subjective classification problems as opposed to the traditional objective classification problems in more detail in chapter 2.

1.3 Outline

The thesis has four main chapters. The “ground truth” theme is discussed in chapters 2, 3 and 4, and the “classifier” theme is discussed in chapter 5. I will briefly introduce the main arguments and contributions of each of the four chapters as follows.

Chapter 2 lays out the conceptual foundation for the rest of the thesis. It characterizes the subjective classification problems as opposed to the objective classification problems, proposes the distribution as ground truth model as opposed to the label as ground truth model, and discusses a principled approach to map distributions into labels for practical purposes.

Chapter 3 focuses on the practical annotation process to elicit ground truth dataset in distributions. It uses empirical data to show that subjective classification problems do exist and using the distribution as ground truth model does make a difference. The annotation process is exemplary to other subjective classification problems. The dataset obtained will also serve as ground truth to train and evaluate the political leaning classifiers in chapter 5.

Chapter 4 focuses on classifier evaluation with the distribution as ground truth model. It uses computer simulation to show that using the traditional “label as ground truth” model for a subjective classification problem will run the risk of incorrectly ranking the accuracy of classifiers. Computer simulations also illustrate the optimal way of evaluating classifiers with the distribution as ground truth model, and conclude that even though individual items in the labeled dataset are unreliable, classifier evaluation (in terms of ranking according to accuracy) with distribution as ground truth can still be reliable with a large number of labeled items.

Chapter 5 proposes a semi-supervised political leaning classifier called LabelPropagator that automatically classifies people and items as liberal or conservative. The inspiration

is that a few manually coded labels on articles and people might be propagated to other people and articles, since liberal people are likely to endorse liberal articles, and similarly for conservative people and articles. This chapter has two parts. The first part is the original study of the algorithm in the year 2010 with problematic ground truth data. The second part is to re-evaluate the algorithm in the year 2012 with new ground truth data obtained from chapter 3. I will show that using different ground truth models and evaluation schemes indeed leads to different results about the classifier.

Chronologically, the thesis started in 2010 with the first part of chapter 5 that proposed the LabelPropagator algorithm. However, the original study was faced with the problem of an unreliable labeled dataset, and the question was raised of whether the unreliable labeled dataset could affect the reliability of classifier evaluation. Therefore, I started to work on chapters 2 and 4 to study how to evaluate classifiers correctly with the “distribution as ground truth” model. Based on the findings of chapters 2 and 4, I then started to work on chapter 3 and obtained a labeled dataset that was reliable and powerful enough to evaluate classifiers correctly. With the new labeled dataset as ground truth, I finally worked on the second part of chapter 5 and showed that using a different labeled dataset indeed led to different results. This process as well as the organization of the thesis is illustrated in Figure 2 below.

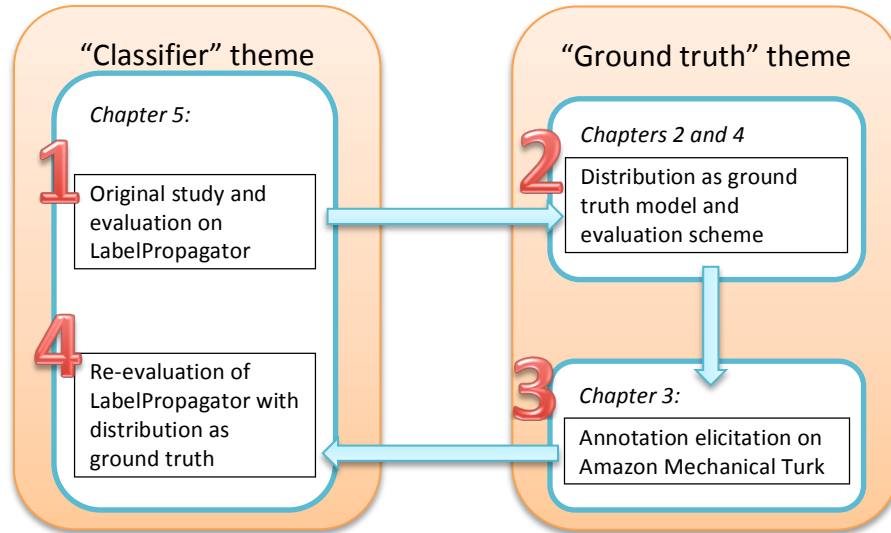


Figure 2. Outline of the thesis in chronicle order

Chapter 2. Distribution as Ground Truth

2.1 Introduction

In machine learning theoretical and algorithmic research, labeled examples for training and evaluation purposes are assumed to be 100% accurate, because “a well-defined learning problem requires a well-specified task, performance metric, and source of training experience” (Mitchell, 1997). Therefore, the practical process of acquiring labeled data is *not* a concern of machine learning theoretical and algorithmic research. In fact, none of the five widely used and most influential textbooks on machine learning discusses how to obtain labeled data as ground truth for practical use (Mitchell, 1997; Witten & Frank, 2005; Hastie et al, 2005; Bishop, 2006; Alpaydin, 2010).

However, one should not confuse theoretical and algorithmic machine learning research with applications of machine learning theories and algorithms. For many real world machine learning applications, the first step is always to obtain labeled data as ground truth for training and evaluation purposes. Due to the lack of systematic guidance, the actual annotation process to obtain ground truth data is usually rather chaotic. One common practice is to use existing labeled datasets from multiple third party sources, without checking their validity and reliability (e.g. Jiang and Argamon 2008). Another common practice is to have one human coder (usually one of the researchers) label a small set of items, assuming there are no errors or only small errors in the labeled dataset (e.g. Oh, Lee and Kim 2009). Another more sophisticated approach is to define a codebook, hire multiple human coders to label items, and use the majority vote or consensus vote as the ground truth label of items, assuming human errors are corrected with multiple coders (e.g., Zhou, Resnick and Mei 2011).

Those approaches all assume that the obtained labels are reliable. That is, if the labeling process is to be repeated either by the same or by other raters, the labels would remain the same. The assumption is certainly true in some cases where the correct labels for items are clear beyond doubt (e.g., Sharma et al 2002). But it might not be true for many other subjective classification problems such as political leaning classification. As I will show later in chapter 5, when labeling the same 1000 political articles again in year 2012 compared to the first time labeling them in 2010, 27% of the articles received different labels.

When labels are not reliable, it is important to ask these questions: If we label a set of items for multiple times, and each time certain items are labeled differently, then which labels are correct? For those items that have the same labels across multiple rounds of annotating, can we still expect to see the same labels if we label them again? If we are to use these unreliable labels to train and evaluate classifiers, can we expect to obtain reliable results about the classifiers?

I'd like to give a hypothetical example to illustrate the potential danger of evaluating classifiers with unreliable labels. Suppose we have two binary classifiers, one is able to classify any item 100% correctly and the other 100% incorrectly. If we have 10 labeled examples to evaluate these two classifiers but all of them are incorrectly labeled by the human raters, then according to these 10 incorrectly labeled examples, the bad classifier would be 100% accurate, while the perfect classifier would be 0% accurate. One would then draw the wrong conclusion that the bad classifier is better than the perfect classifier. Although in practice we would not expect to see all examples labeled wrongly, it is still questionable that we can always correctly rank classifiers using erroneous "ground truth" that happens to favor the bad classifier.

This and the next two chapters are closely related under the "ground truth" theme of the thesis, which mainly concern the problem of unreliable labels dataset and how it affects classifier evaluation. In particular, this chapter (chapter 2) lays out the conceptual foundation for the rest of the thesis. It characterizes the "subjective

classification problems”, proposes the “distribution as ground truth” model, and discusses a principled way to map distributions into labels for practical purposes. Chapter 3 focuses on practical annotation process, and uses empirical data to show that subjective classification problems (where raters do not agree on many items) do exist and using the distribution as ground truth model does make a difference. Chapter 4 focuses on classifier evaluation with distributions, and uses computer simulations to show that even though individual items in the labeled dataset are unreliable, evaluation of classifiers would still be reliable if we have a large number of labeled items to evaluate. The relationship of the three chapters is shown in Figure 3.

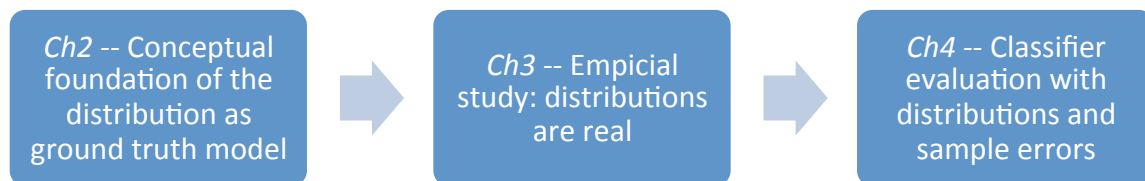


Figure 3. Relationship of chapters 2, 3, and 4

The rest of this chapter is organized as follows. Section 2.2 discusses related work, which is also relevant to the next two chapters. Sections 2.3 and 2.4 distinguish the objective and subjective classification problems and their corresponding ground truth model. I will argue that, for subjective classification problems, disagreements among human coders are not due to observation errors as implied by the common annotation process, but are due to substantive difference among people’s subjective assessments. And therefore, in order to keep the disagreement as useful information rather than eliminating them as errors, the underlying ground truth model should be distributions over labels rather than single labels. Section 2.5 discusses a principled approach to map distributions into labels using cost functions. Section 2.6 discusses caveats and future work of using the distribution as ground truth model. This chapter is mainly conceptual, and practical work will be discussed in next two chapters.

2.2 Related Work

Theories of Truth

This chapter is about ground truth. What is “truth”? The answer to this question is central to this chapter, and perhaps to the entire endeavor of scientific inquiry.

According to Glanzberg (2009), there are a few different theories of truth.

One of the most important and popular theories of truth is the neo-classical corresponding theory that says: “what we believe or say is true if it corresponds to the way things actually are – to the facts” (Glanzberg 2009). This theory presupposes the existence of an objective world. To apply the corresponding theory of truth to the political leaning classification problem requires objective, factual definitions of “conservative”, “liberal” and “political”, which do not exist. As I have discussed in the first chapter, people don’t agree on the exact definitions of conservative and liberal, and even the validity of the dichotomy itself is fiercely challenged (Klein & Stern 2008). The interpretation of “conservative” and “liberal” is mostly subjective, and therefore it is impossible to define the true political leaning labels for items according to the neo-classical corresponding theory of truth.

Another theory, the consensus theory of truth, holds that “truth is whatever is agreed upon”, or “that which is universal among men carries the weight of truth” (Bohman & Rehg, 2011). According to this theory, truth does not necessarily correspond to facts, but is collectively determined by the population. This theory inspires the distribution as ground truth model, which will be discussed later in this chapter, because the emphasis is not on the existence of objective “facts” but on people’s subjective opinions. For the political leaning classification problem, however, the theory requires consensus among people, which still doesn’t exist for many, if not all, items. Therefore, this theory cannot directly apply here.

The last theory of truth I’ll discuss here is the pragmatic theory of truth, whose slogan is “truth is satisfactory to believe”, or “true beliefs are guaranteed not to conflict with

subsequent experience” (Glanzberg, 2009). According to the pragmatic view, truth depends on how one defines truth for practical purposes. And thus the pragmatic theory is not incompatible with other theories of truth, as long as they are useful in practice. Hookway (2010) discussed an example to illustrate the pragmatic view on truth:

... This human witness tries to get sight of the squirrel by moving rapidly round the tree, but no matter how fast he goes, the squirrel moves as fast in the opposite direction, and always keeps the tree between himself and the man, so that never a glimpse of him is caught. The resultant metaphysical problem now is this: Does the man go round the squirrel or not? James proposed to solve the problem by pointing out that which answer is correct depends on what you ‘practically mean’ by ‘going round’. If you mean passing from north of him to east, then south, then west, then the answer to the question is ‘yes’. If, on the other hand, you mean first in front of him, then to his right, then behind him, and then to his left, before returning to being in front of him again, then the answer is ‘no’. Pragmatic clarification disambiguates the question, and once that is done, all dispute comes to an end. The ‘pragmatic method’ promises to eliminate all apparently irresolvable metaphysical disputes. (Hookway 2010)

Ground truth, when used in machine learning classification problems, takes the pragmatic view of truth. The annotation process that assigns “true” labels to items is not about the metaphysical “true labels”, but about how researchers define the true labels of items in order to train and evaluate classification algorithms to solve real problems. For the political leaning classification problem and other subjective classification problems, I propose to use distributions as the “truth” of items. Other people might not agree with this particular definition of “truth” on items, but as long as the definition satisfies my reasons and purposes, I can still treat it as the “truth” according to the pragmatic view.

For the rest of the thesis, I will take the pragmatic view on truth to define the true political leaning of items. I will then not discuss what is “conservative”, “liberal” or “political” on the metaphysics level.

Inferring Correct Labels from Noisy Input

In order to increase the reliability of the labeled dataset, one obvious attempt is to use more advanced techniques to infer correct labels from multiple human coders’ noisy input. This line of work assumes that each item to be labeled does have an objective, correct true label, even though it is hard, if possible at all, to observe. Due to the difficulty of observing the true label, most human coders will make considerable errors, but expert coders are less erroneous than inexperienced coders. Therefore, using

advanced techniques other than majority vote that account for coders' quality (or expertise level) and then aggregating multiple coders' input will lead to better quality labeled dataset.

For example, Dawid and Skene (1979) proposed a latent class model and used the EM approach to infer coders' quality, and then use the quality score of each coder to discount or increase the weight of their labels. Built on that work, Smyth et al (1995) studied the problem of labeling volcano images of Venus, and showed that using the latent class model to aggregate multiple coders' input made a significant improvement over simply using majority vote. This type of work is quite popular in Amazon Mechanical Turk research (e.g., Ipeirotis, Provost and Wang 2010) where coders are quite noisy. I will discuss more about it in the "related work" section of the next chapter.

This line of work is conceptually different from the work of this chapter in that it assumes that each item to be labeled has an objective, correct true label. Disagreement among coders is simply due to errors, and thus the main goal of this line of work is to reduce or eliminate the errors. But this chapter assumes that disagreement among coders is not due to errors, but is due to subjective differences, and therefore coders' disagreement should be kept in the labeled dataset. Furthermore, this line of work argues that using the latent class model will reduce errors. But again, it assumes that the labeled dataset thus obtained is reliable despite the fact that there are still considerable errors in the labeled dataset. The researchers haven't studied the consequence of having inevitable errors in the labeled dataset for classifier evaluation.

Empirical Loss Minimization and Decision Theory

My work in this chapter resembles "empirical loss minimization" or "empirical risk minimization" (ERM) in decision theory (Berger 1985, Haussler 1992, Vapnik 1999, Bishop 2006). Haussler (1992) described the general framework of ERM as follows:

In this general framework, we assume the learner receives randomly drawn training examples, each example consisting of an instance $x \in X$ and an outcome $y \in Y$, X and Y are arbitrary sets called instance and outcome spaces, respectively. These examples are generated according to a joint distribution on $X \times Y$, unknown to the learner. This distribution comes from a (known) class \mathcal{P} of joint distributions on $X \times Y$ representing possible "states of nature."

After training, the learner receives further random examples drawn from this same joint distribution. For each example (x, y) , the learner will be shown only the instance x . Then he will be asked to choose an action a from a set of possible actions A , called the decision space. Following this, the outcome y will be revealed to the learner. In the case that we examine here, the outcome y depends only on the instance x and not on the action a chosen by the learner. For each action a and outcome y , the learner will suffer a loss, which is measured by a fixed real-valued loss function l on $Y \times A$. We assume that the loss function is known to the learner. The learner tries to choose his actions so as to minimize his loss.

Here we look at the case in which, based on the training examples, the learner develops a deterministic strategy that specifies what he believes is the appropriate action a for each instance x in X . He then uses this strategy on all future examples The learner's strategy, which is a function from the instance space X into the decision space A , is called a decision rule. We assume that the decision rule is chosen from a fixed decision rule space \mathcal{H} of functions from X into A . For example, instances in X may be encoded as inputs to a neural network, and outputs of the network may be interpreted as actions in A . In this case the network represents a decision rule, and the decision rule space \mathcal{H} may be all functions represented by networks obtained by varying the parameters of a fixed underlying network. The goal of learning is to find a decision rule in \mathcal{H} that minimizes the expected loss, when examples are drawn at random from the unknown joint distribution on $X \times Y$.

This general framework enables researchers to answer theoretical questions such as how many training examples are needed, how to choose the hypothesis design space \mathcal{H} , what is the computational complexity to find the optimal hypothesis, what are the theoretical error bounds of learning algorithms, and so on (Haussler 1992). Practically, this general framework can apply to a variety of problems, including regression, betting, decision making and classification (Haussler 1992).

This chapter builds on many ideas from the ERM literature, as can be seen later.

However, this chapter still differs significantly from the ERM literature. First, in the ERM framework, the outcome y for each instance x is the true state of Nature, and will be revealed after decisions are made. For political leaning classification, applying the ERM framework would assume that each article (x) in the labeled dataset has a perfectly accurate label (y). My work in this chapter, however, assumes that an article's political leaning is a distribution and does not have one perfectly accurate label.

Second, according to the problem formulation of the ERM framework, even though $P(y|x)$ is a distribution over the $X \times Y$ space, the outcome y is still a single label for each individual instance x . In other words, the underlying ground truth model uses labels. On the contrary, this chapter proposes the underlying ground truth model uses distributions over labels instead of single labels.

Last but most importantly, the ERM literature has a different research problem, which is to find a hypothesis h^* among a fixed class of functions \mathcal{H} for which the risk is minimal. The research problem of this and the next two chapters is not about finding the best hypothesis h^* , (or in my case, a classifier), but about finding the best evaluation scheme that is able to discriminate the best hypothesis h^* from the inaccurate ones when the labels from human coders (y) are not reliable. Thus, the conclusions drawn from this thesis would be in a different category from the ERM literature.

Multi-class, Multi-label, Multiple-label, and Multiple-rater Problems

Existing literature discusses the multi-class, multi-label, multiple-label and multiple-rater problems. Here I'd like to make some clarifications on these rather confusing terms. A multi-class classification problem is distinguished from a binary classification problem by having more than two classification labels (Tsoumakas and Katakis, 2007). A multi-label classification problem is to assign multiple labels to one item, and all label assignments are considered correct (Tsoumakas and Katakis, 2007). A few examples of multi-label classifications are, for examples, Zinovev et al (2011), Vannoorenberghe and Denoeux (2002), Li, Zhang and Zhu (2006), and Quost and Denoeux (2009), among others. A multiple-label classification problem is different from a multi-label problem in that although multiple labels are associated with a single item, only one label is the correct one (Jin and Ghahramani, 2003). A multiple-rater problem directly uses labels from multiple raters for classifier training, and at the same time computes the quality of coders and discounts erroneous input in the training process. Literature about the multiple-rater problem will be discussed in the final chapter.

What distinguish this thesis (in particular, this and the next two chapters) from these previous studies are as follows. First, the thesis assumes the ground truth of an item is a distribution rather than a single label or multiple labels. Second, the thesis focuses on classifier evaluation with distribution as ground truth rather than classifier training with distributions. Third, the thesis evaluates classification outcomes as labels against ground truth as distributions rather than "labels against labels" or "distributions against

distributions”. And finally, the thesis assumes the ground truth dataset for classifier evaluation is unreliable and studies whether we can still draw reliable conclusions.

Probabilistic Forecasting and Scoring Rules

In a probabilistic forecasting problem, an expert makes a prediction with a probability distribution over possible outcomes, and the true state will be revealed after the prediction and the prediction is evaluated against the true state (Dawid 1984). Weather forecasting is a typical probabilistic forecasting problem, where weather experts make probabilistic predictions on a few possible outcomes of today’s weather. Then Nature reveals today’s weather, and the experts will get rewarded based on the quality of their predictions evaluated against the true state (Brier 1950).

Scoring rules play an important role in probabilistic forecasting problems that evaluate predictions as distributions against the true states as discrete labels, as discussed below:

Scoring rules provide summary measures for the evaluation of probabilistic forecasts, by assigning a numerical score based on the predictive distribution and on the event or value that materializes. In terms of elicitation, the role of scoring rules is to encourage the assessor to make careful assessments and to be honest. In terms of evaluation, scoring rules measure the quality of the probabilistic forecast, reward probability assessors for forecasting jobs, and rank competing forecast procedures. (Gneiting and Raftery 2007)

The incentivizing role of scoring rules is *ex ante*, while the evaluation role is *ex post* (Winkler and Jose 2010). Scholars have studied both roles of scoring rules. For example, Bickel (2007) compared the difference between quadratic, spherical, and logarithmic scoring rules in terms of both of their *ex ante* and *ex post* roles.

This line of research is relevant to the work of this chapter and chapter 4 (about classifier evaluation) because here evaluation is also between labels and distributions (as opposed to “labels against labels” or “distributions against distributions”). However, in a probabilistic forecasting problem, the predictions are distributions whereas the ground truth outcomes are labels; in this chapter, the predictions (or classification outcomes) are labels while the ground truth data are distributions. In addition, this chapter particularly focuses on sample errors in the ground truth data, whereas in a probabilistic forecasting problem, ground truth outcomes from Nature are indeed 100%

accurate. Finally, studies about “proper” scoring rules (Gneiting and Raftery 2007) focus on the scoring rules’ *ex ante* role on incentivizing forecasters. To some extent, the process of classifier evaluation also “incentivizes” researchers to design a classifier algorithm according to a chosen scoring rule in order to optimize for evaluation outcome. However, it is not the focus of this chapter to study the *ex ante* role of scoring rules used in classifier evaluation. As can be seen later, this chapter only uses one simple form of a scoring rule, and I will not discuss its properness either.

Measurement Error in Statistics

Measurement error in the statistics literature is relevant to this chapter and chapter 4 because both chapters are about sample error in the ground truth data for subjective classification problems. Fuller (2008) studied how to assess the effects of measurement errors in regression models and other univariate and multivariate models, and then how to correct for measurement errors when estimating the parameters for those models. Hyslop and Imbens (2001) argued against the classical measurement error (CME) model, which assumes measurement error is independent of the true value. The paper argues that in addition to CME, there are optimal prediction errors (OPE) that are independent of the reported values but could be dependent on the true values. The paper then discusses how to use the concept of OPE to correct errors in linear regression model and generate better predictions. To my knowledge, there is no existing work in the measurement error literature studying how errors in the ground truth affect classifier evaluation. My work could add to this line of work.

Classification Systems and Theories

According to Bowker and Star (1999), “a classification is a spatial, temporal, or spatial-temporal segmentation of the world”, and an ideal classification system should have three properties: 1) clearly defined, 2) mutually exclusive, and 3) collectively exhaustive. Political leaning classification seems to be far from those properties, and one might ask whether such a classification problem is useful or valid at all.

To explain why any classification – even though as “messy” as political leaning classification, for example – is useful and valid, Rosch (1999) discussed two principles of classification. The first principle is about why a classification is useful or desirable: “the task of category systems is to provide maximum information with the least cognitive effort”. In the extreme case where people don’t classify at all, each individual instance would be considered a category by itself, resulting in tremendous cognitive effort. Therefore, people want to classify and keep the number of categories small. The second principle explains why a classification is workable or valid: “the perceived world comes as structured information rather than as arbitrary or unpredictable attributes”. If the target problem does not have any structural information, then it is not a valid classification problem at all. I’d like to argue that political leaning classification satisfies both principles, and thus is a useful and valid classification problem.

In this chapter, I will discuss objective versus subjective classification problems, which should not be confused with the Aristotelian versus the prototype classification systems (Bowker and Star, 1999). An Aristotelian classification “works according to a set of binary characteristics that the object being classified either presents or does not present. At each level of classification, enough binary features are adduced to place any member of a given population into one and only one class (Bowker and Star, 1999).” A prototype classification makes use of prototypes or exemplars “rooted in people’s experience” to categorize objects without consulting any Aristotelian-style classification rules, even though “conceptual categories are not identical for different cultures, or indeed, for every individual in the same culture (Rosch 1999).” Aristotelian and prototype systems differ in how to classify items into different categories, whereas objective and subjective classification problems differ in whether there exist many items where human raters do not agree.

Furthermore, the study of the subjective classification problems should not be confused with the study of folksonomy. Folksonomy is a term coined by Vander Wal (2007) to contrast with formal taxonomy. My work is similar to folksonomy in the “folk” part: it is

the users who define which items should be labeled into which categories. However, my work is different from folksonomy in the “-sonomy” part: my work pre-defines the categories of red, blue and gray, and people are not allowed to contribute new categories (or labels) to the categorization scheme.

2.3 Objective Classification Problems and Label as Ground Truth

Before discussing objective classification problems and the label as ground truth model, I'd like to summarize what researchers usually do to annotate items. One common practice is to use only one human coder (usually one of the researchers) to annotate a small set of items. The validity of this approach rests on the assumption that the coder follows objective rules (usually implicitly in the coder's mind) during annotation, and makes only small, negligible errors, if at all, during the process so that the resultant labeled dataset is accurate and reliable. That assumption might be too strong in practice because it is easy for human beings to make mistakes. So another common but more sophisticated approach is to use more than one coder to annotate items in order to assess the quality of the annotations and correct for human errors. Examples are seen in word sense disambiguation (Kilgarriff 1998, Veronis 1998), subjectivity classification (Wiebe et al 1999), and many other studies in the machine learning literature.

The multiple coders approach is actually borrowed from the qualitative coding process widely seen in social sciences. Krippendorff (2004) summarized a full-fledged qualitative coding process that involves multiple coders and the use of a codebook. In such a coding process, the researchers first define a sophisticated codebook instructing how items should be coded. For example, MacQueen et al (1998) described a codebook structure with six components: 1) the code (or categories), 2) a brief definition, 3) a full definition, 4) guidelines for when to use the code, 5) guidelines for when not to use the code, and 6) examples. The researchers then hire and train multiple coders to annotate a small set of items according to the codebook. Then, the researchers compute the inter-rater reliability (IRR) using Cohen's kappa or Fleiss's kappa. If IRR is high, then the researchers can have the coders annotate the rest of the items and conclude that the resultant

labeled data is reliable. But if IRR is low (which is usually the case), which implies either the codebook is not well-defined or the coders are not well trained, then it will trigger several iterations to improve the codebook, re-train the coders, and re-code the items. The process is repeated until IRR is high. For items upon which coders don't agree, the coders and researchers should discuss them and reach an agreement, and perhaps revise the codebook too. MacQueen et al (1998) described a typical coding and recoding process as follows:

The codebook is reviewed to determine whether the inconsistencies are due to coder error, e.g., misunderstanding of terminology or guidelines. We view these as training errors and they are generally handled by the coders and team leader(s). Other inconsistencies are due to problems with the code definitions, e.g., overlapping or ambiguous inclusion criteria that make it difficult to distinguish between two codes. These types of problems are generally discussed by the whole team, as they have implications for the interpretation of the text. Once the problems are identified and the codebook clarified, all previously coded text is reviewed and, if necessary, recoded so that it is consistent with the revised definitions. Intercoder agreement is again checked, to ensure that the new guidelines have resolved the problem. This iterative coding process continues until all text has been satisfactorily coded. ... Schedule regular meetings where the coding team reviews each code and definition in the codebook. It is easy for a coder to develop a set of implicit rules without realizing that the codebook no longer reflects his or her actual coding process; in addition, this evolving process may or may not be shared by other members of the coding team. (MacQueen et al 1998)

When applying the qualitative coding process to label items for machine learning classification problems, researchers usually simplify this complicated process to various degrees, as can be seen from many machine learning papers. For example, researchers may not take the time and effort to define a sophisticated codebook or train the coders well to such a degree like what has been described in MacQueen et al (1998). In addition, disagreements on items may be resolved simply by majority votes rather than discussing them collectively (e.g., Zhou et al, 2011). Furthermore, even though IRR is only "fair" (Cohan's kappa 0.2 ~ 0.4), which means coders don't agree on many items, researchers may just treat the labeled dataset as reliable instead of going through iterations to improve IRR (e.g., Zhou et al, 2011). Finally, researchers may sometimes hire multiple coders to label a few items to check IRR, and proceed with having one coder label the rest of the items even though IRR does not indicate high agreement.

It is understandable to see these simplifications in a machine learning annotation process, because the researchers are more interested in the computational aspect of their research (such as algorithm development), rather than spending all of their

resources and efforts on a complicated annotation process. But more importantly, such a simplified coding process (including the one coder coding process) could turn out to be sufficient for a set of classification problems where the classification boundaries are clear and straightforward. For example, an annotation process for detecting the presence of faces in images (Shakhnarovich et al 2002) and gender classification of passport photos (Sharma et al 2002) are very straightforward, even though admittedly there might be a small, negligible number of ambiguous cases.

I'd like to define as "**objective classification problems**" or **OCPs** those where all items (perhaps with few negligible exceptions) can be clearly classified into categories according to some objective, well-defined criteria. These objective criteria could be implicitly held by coders, or explicitly codified in a codebook. Following these objective criteria, multiple coders would all agree on the correct label of items and repeated annotations will result in the same labels. The labeled dataset would be reliable with no error or with only a small number of errors due to random human error, which are negligible. Clearly, for this type of classification problem, it is valid to use either one coder or multiple coders (to correct for human errors with majority votes) for the annotation process. Many classification problems in the machine learning literature do satisfy this characterization (e.g., Shakhnarovich et al 2002, Sharma et al 2002). However, even though some classification problems are not objective such as the political leaning classification problem, researchers usually still treat them as objective, and, as I will argue later, it is a defective model.

For OCPs, the underlying ground truth model is indeed on the label space. That is, each instance has one and only one label as its true label. If coders disagree on the correct label of an item, it is due to human errors and such errors should be resolved. Under this model, error rate is assumed to be fixed for all items (whereas variable error rates for different items imply that coders' disagreement are "subjective", which will be discussed later). For any valid classification problem (see "related work" at page 22 for discussions about a valid classification system), it is safe to assume that the correct label

is more likely to be observed than any other label. According to the principle of maximum likelihood, disagreement among coders should be resolved by majority vote in order to get the correct label. I'd like to call this model the **"label as ground truth"** model for the objective classification problems. This is to contrast with the "distribution as ground truth" model for the subjective classification problems, which will be discussed next.

2.4 Subjective Classification Problems and Distribution as Ground Truth

As I have briefly introduced in chapter 1, there is a set of classification problems different from the objective classification problems, and I'd like to call them the **"subjective classification problems" or SCPs**. The fundamental difference here is that we cannot find objective, well-defined classification criteria to put items into categories. When presented with an item, people have to use their subjective judgments to label it, and the subjective judgments on many items may vary among different people. As a result, there are a non-negligible number of items whose labels people simply do not agree upon. The disagreement is not simply due to random observation errors that could be avoided, but rather substantive subjective differences among people. The distributions of coders' different labels indeed reflect the true nature of an item. Therefore, I'd like to call the underlying ground truth model the **"distribution as ground truth"** model for subjective classification problems, as opposed to the "label as ground truth" model for objective classification problems. Here, the ground truth for an item is a distribution over labels, not a single label anymore. Next, I'll describe distribution as ground truth in detail.

Defining Ground Truth Using Distributions

I'd like to start by introducing notations. Here I will only study the political leaning classification problem as a representative example of SCPs, without making explicit effort to generalize it to other SCPs.

Let I^* be the entire target political articles corpus, and i be an item or article, $i \in I^*$. Let $I \subseteq I^*$ be a subset of articles to be labeled as ground truth, and I_{train} and I_{test} are for training and testing purposes respectively, where $I_{train} \cup I_{test} = I$. Let $I' = I^* - I$, which are the unlabeled items to be classified by a classifier. Let U^* be the target population and u as a person from the population, $u \in U^*$. Let U_i be a random sample of the population, $U_i \subseteq U^*$, where U_i denotes a random sample of the population (or, in the case of $U_i = U^*$, the entire population) who will label item i . Let U be the set of coders sampled from the population to label I , that is, $U = U_1 \cup U_2 \cup \dots \cup U_i \dots, \forall i \in I$, and $U \subseteq U^*$. Let $\mathcal{L} = \{red, gray, blue\}$ be the political leaning label space, and $\ell \in \mathcal{L}$ be a label. For convenience purposes, I'll sometimes use $\{r, g, b\}$ as an abbreviation for $\{red, gray, blue\}$.

Let s_{ui} be the personal assessment of user $u \in U_i$ on item $i \in I$, $s_{ui} \in \mathcal{L}$. Here I assume u reports s_{ui} truthfully without errors: s_{ui} is not a random variable. I will discuss this assumption and extensions to the simplified s_{ui} model in the last section. Let $\theta_\ell(s_{ui}) = 1$ if $s_{ui} = \ell$, or 0 otherwise. Then, define the empirical frequency of labels for item i over the entire population (i.e., $U_i = U^*$) as:

$$P_\ell(i) = \frac{\sum_{u \in U^*} \theta_\ell(s_{ui})}{|U^*|}, \quad \ell \in \mathcal{L} = \{r, g, b\}, \forall i \in I$$

Equation 1. Definition of $P_\ell(i)$

For convenience purposes, I'll use R_i , G_i and B_i to denote $P_r(i)$, $P_g(i)$, and $P_b(i)$ respectively. Intuitively, R_i , G_i and B_i are the proportions of the entire population who would label item i as red, gray and blue respectively, and $R_i + G_i + B_i = 1$. Now, I'd like to define H_i as the ground truth for the political leaning of article i as:

$$H_i = \langle P_r(i), P_g(i), P_b(i) \rangle = \langle R_i, G_i, B_i \rangle, \quad \forall i \in I$$

Equation 2. Definition of H_i

Since H_i is a distribution, this ground truth model is thus called distribution as ground truth. Let $H_I = \{H_1, H_2, \dots, H_i, \dots\}, \forall i \in I$, which denotes the annotated labeled ground

truth data for I . The goal of the annotation process is to obtain H_I , and researchers will use H_I to train and evaluate classifiers.

Simplex Notation

Any item $i \in I$ with $H_i = \langle R_i, G_i, B_i \rangle$ can be represented as a point in a simplex (Elte, 1912), as illustrated in Figure 4. For example, $H_i = \langle 1, 0, 0 \rangle$ is the point on the simplex's left corner; $H_i = \langle 0.33, 0.33, 0.33 \rangle$ is the point in the middle of the simplex. If H_i is higher in one dimension (such as R_i), on the simplex the point will be closer to the corresponding corner (such as the left corner). Similarly, any point in a simplex could map to a H_i distribution. In short, there is a one-to-one correspondence between H_i and a point on the simplex. For the rest of the dissertation, I'll use the distribution H_i and its simplex notation interchangeably to represent the political leaning of article i . More information about the simplex can be found in (Elte, 1912).

Sometimes I use the bar chart graphical notation to represent H_i for item i , where the red, gray and blue bars correspond to $R_i, G_i,$ and B_i (see Figure 5). A bar chart maps to a point in the simplex.

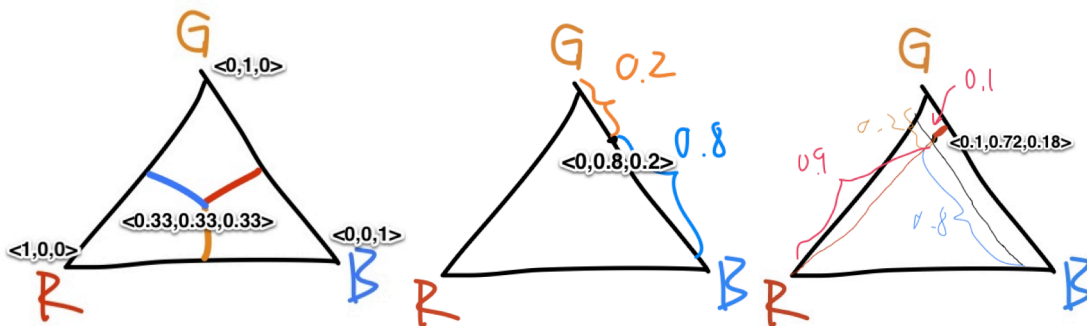


Figure 4. Simplex notation for H_i

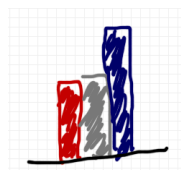


Figure 5. Bar chart notation for H_i

Sample Errors

Ideally, I would use $H_i = \langle R_i, G_i, B_i \rangle$ as the ground truth for item i , but H_i is defined from the entire population U^* and is not directly observable. I will call H_i “**population ground truth**”, or the “true” ground truth. Practically, one can only get an estimate of H_i from a subset of the population, U_i , where $|U_i| \ll |U^*|$. I will define \hat{H}_i as the “**sample ground truth**”, or the “observed” ground truth for i , as follows.

$$\hat{P}_\ell(i) = \frac{\sum_{\forall u \in U_i} \theta_\ell(s_{ui})}{|U_i|}, \quad \ell \in \mathcal{L} = \{r, g, b\}, \forall i \in I$$

$$\hat{H}_i = \langle \hat{P}_r(i), \hat{P}_g(i), \hat{P}_b(i) \rangle = \langle \hat{R}_i, \hat{G}_i, \hat{B}_i \rangle, \quad \forall i \in I$$

Equation 3. Estimation: $\hat{H}_i, \hat{P}_\ell(i)$

In Equation 3, U_i is a random sample of users drawn from U^* to label item i . The ground truth of i is then collectively estimated by the sample of users U_i rather than by the entire population U^* . Note that for any two items i and j , U_i and U_j do not have to be the same. Also, according to statistical theories, \hat{H}_i is an unbiased estimator of H_i , that is, $E(\hat{H}_i) = H_i$, even if $\hat{H}_i \neq H_i$. One can avoid sample bias by randomly sampling U_i , and reduce sample error by increasing $|U_i|$. With a small $|U_i|$, inevitably the sample ground truth dataset \hat{H}_I will have errors, as shown in Figure 6.

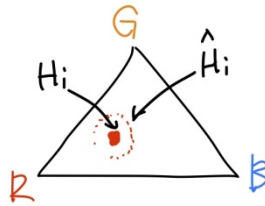


Figure 6. Ground truth estimation errors

Here, I'd like to introduce a few more notations. Let $m = |U_i| \ll |U^*|$. That is, m is the number of human coders to label item $i \in I$. Here I'd like to assume m remains the same for all $i \in I$. In the “active learning” literature, m could differ for different items, but it's not the concern of this chapter. Let $n = |I| < |I^*|$. That is, n is the number of items to label in the labeled dataset I .

The total number of labels to elicit from human coders is denoted as k , where $k = m \times n$. Assuming the cost per label is a constant, then under limited budget and a constant k , one can either have a small m but a large n , or a large m but a small n . For example, given the same budget, one can either annotate 1000 articles with 4 labels per article, or annotate 4 articles with 1000 labels per article. It is important to find the optimal m and n given the presence of sample errors in the ground truth dataset. I will discuss this in chapter 4.

Roles of Codebooks and Coders

For an OCP that uses label as ground truth, a codebook explicitly codifies the objective criteria instructing how to put items into categories, and thus defines the ground truth of items. Human coders are merely an “instrument” to annotate items following the rules in the codebook. As with any other instruments, the coders could make errors, but the errors should be eliminated.

For an SCP where classification boundaries are fuzzy, it is hard for researchers to define a good codebook that clearly instructs how coders should label items. Researchers have two alternatives here. The first is to apply the qualitative coding process and turn the subjective classification problem into an objective classification problem by developing a comprehensive codebook through several iterations. Through this process, subjectivity is eliminated by forcing the coders to follow the same comprehensive rules in the codebook. Consequently, coders would all agree on most items and the resultant labeled items are reliable. Note that the inherent subjectivity of the classification problem at hand is not eliminated: it is just embedded in the making of the codebook.

For the political leaning classification problem, following this approach (that is, making it an objective) means that we have to clearly give our own definitions of “conservative”, “liberal”, “populist”, “libertarian”, “political”, etc, and then define a comprehensive codebook including definitions, examples, guidelines and so on. We have to revise the codebook several rounds and work with the coders to resolve ambiguous cases.

However, for machine learning application studies, I'd like to argue against this approach of turning a subjective classification problem into objective through the qualitative coding process. As I have argued earlier, this process is too costly for machine learning researchers who are more interested in developing classification models and algorithms than developing a sophisticated codebook. Furthermore, even though researchers actually do go through several iterations of improving the codebook, it doesn't necessarily mean they will get good results due to extreme subjectivity in the labeling process. For example, despite several rounds of iterations, researchers never came to acceptable inter-rater agreement on labeling "humorous tweets" (Munson, Rosengren and Resnick 2011) or "related Drupal modules" (Zhou and Resnick 2009).

More importantly, a comprehensive codebook means high specificity and low generalizability, whereas machine learning researchers want low specificity and high generalizability. For example, if I define a comprehensive codebook for political leaning classification, then my own specific interpretation of "conservative" and "liberal", out of many other different possible interpretations, would be embedded in the codebook. As a result, my research results would be limited to this specific interpretation (hence high specificity) and not generalizable to a wider audience who has different interpretations of "conservative" and "liberal" (hence low generalizability).

Instead of turning a subjective classification problem into objective, another choice is to acknowledge the inherent subjectivity of the classification problem. In this case, researchers should use the distribution as ground truth model, and have the coders use their own subjective assessments to label items. In this case, human coders are not merely an instrument; rather, the coders, who represent the population if randomly sampled, collectively define the ground truth of items as distributions by aggregating subjective opinions. The codebook, if given, only provides some general classification principles and leaves several undefined areas. Therefore, the codebook has some educational value, but does not define the ground truth of items. Clearly, having the

population define the ground truth of items implies low specificity (i.e., not limited to any particular definition of the labels or categories) and high generalizability (i.e., the definition of the labels or categories are acceptable by a wide range of audience) of the labeled data, and thus is desirable for machine learning studies.

For example, Figure 7 shows a codebook example we used in an early study to label articles into red, blue and gray (Zhou, Resnick, and Mei, 2011). To many coders who already know about US politics, this codebook doesn't teach them anything new in terms of how to label articles into red, blue or gray. When labeling ambiguous articles (such as liberals criticizing Obama, or Republican candidate Ron Paul arguing for more civil liberties), those coders are not able to find clear guidelines in the codebook, and have to use their own subjective assessment. The codebook is indeed helpful to inexperienced coders who don't know much about US politics. They can apply some general principles found in the codebook such as "'inequality is bad' means liberal" to label items, but still, they have to use their own subjective assessment on many ambiguous articles.

Liberal, if an article
<ul style="list-style-type: none">■ argues from a liberal perspective, and/or uses liberal ideological arguments (e.g., inequality is bad, social safety net is important, government should not interfere with personal life style choices, etc.)■ presents facts selectively to promote liberal perspective■ attacks conservative positions, perspectives, or people from a liberal perspective
Conservative, if an article
<ul style="list-style-type: none">■ argues from a conservative perspective, and/or uses conservative ideological arguments (e.g., big government is bad, taxes should be low, government should not interfere with markets, etc.)■ presents facts selectively to promote conservative perspective■ attacks liberal positions, perspectives, or people from a conservative perspective
Not-sure, if an article
<ul style="list-style-type: none">■ presents simple facts with no hint of political opinions■ does not satisfy criteria of either "liberal" or "conservative"■ presents well balanced opinions from both "liberal" and "conservative"

Figure 7. A simple codebook example on political leaning classification

Sometimes it is even hard to compile a set of very generalized classification guidelines in a codebook, or the instructions in a codebook are too generalized to the point of being trivial (such as the example of Figure 7), then it is not necessary to provide a codebook under the distribution as ground truth model. For example, when asked to define “pornography”, a Supreme Court judge responded with “you know it when you see it”². Another example is to label “humorous tweets”: everyone knows whether a tweet is humorous or not when they see it (although people might disagree), but it is just hard to define what a “humorous tweet” really is. I’d like to call this type of cases as “you-know-it-when-you-see-it” or YKIWYSI³. The YKIWYSI classification problems are all subjective, and do not require a codebook.

Note that regardless of whether we provide a codebook or not, there is an implicit assumption for the SCPs that people would still agree on the labels of many items due to shared tacit knowledge despite their subjective differences on many other items. Otherwise, if people simply don’t agree on any item, then there would be no valid classification system at all.

Inter-rater Reliability

For an objective classification problem, we would expect to see high inter-rater reliability (IRR) among multiple coders (or if there is only one coder, we would expect to see high IRR if the annotation process is repeated again by either the same or another coder). This is because all coders follow the same objective instructions in the same codebook. We don’t expect, though, to see a perfect IRR score because human coders make errors. But still, low IRR indicates that there’s something wrong in the annotation process and the researchers should correct it (by revising the codebook, for example). Only when IRR is high can the researchers be confident on the reliability of the labeled dataset.

² Further reading see: http://en.wikipedia.org/wiki/I_know_it_when_I_see_it

³ Acronym follows the same fashion as WYSIWYG: what-you-see-is-what-you-get.

For a subjective classification problem, however, disagreement among coders is expected and thus it is not required to have a high IRR. But we should not expect to see a very low IRR either: too low an IRR score means people don't agree more than they would randomly agree, which perhaps implies that the classification problem is not valid at all. IRR score for a typical SCP is low, but not too low. For example, as I will show later in chapter 3, Fleiss' kappa for the political leaning classification is 0.39 to 0.44 (which varies across multiple annotation tasks).

The implication about low IRR for SCPs is that we cannot fully trust the reliability of the labeled dataset anymore due to large sample errors with a only few coders. For example, suppose there's an item with the true distribution $H_i = \langle 0.5, 0.2, 0.3 \rangle$, then there is still 12.5% chance that three coders could all label it as *red*, even though we know the true distribution is far from the observed $\hat{H}_i = \langle 1, 0, 0 \rangle$. If we only have one coder label it as *red*, then there is 50% chance that it does not get labeled again as *red*. And there are many items like this for a typical SCP. In contrast, suppose an average coder for an objective classification problem has 10% error rate (which means 90% of the time a coder is able to label an item correctly), then there is 97.2% chance an item can be correctly labeled with majority votes and three coders, which means the resultant labeled dataset is indeed reliable.

Finally, by assumption the error rate under the label as ground truth model for an OCP is fixed and is small for all items. In practice, when we observe different "error rates" for different items, it is usually an indicator that we should use the distribution as ground truth model and treat the "errors" as distributions of subjective opinions.

Annotation Process

I will briefly discuss the annotation process under the distribution as ground truth model for political leaning classification problem. Figure 8 illustrates the procedure.

- Step 1: Define the target population U^* to consist of any US person who understands the basics of US politics. This population will collectively define the ground truth for the political leaning of any article $i \in I$.

- Step 2: Survey an unbiased random sample of the population⁴, and ask for their personal assessments on an article. This gives $s_{ui}, \forall u \in U_i$. In practice, this can be done on a crowdsourcing platform such as Amazon Mechanical Turk.
- Step 3: Compute $\hat{H}_i = \langle \hat{R}_i, \hat{G}_i, \hat{B}_i \rangle$ using Equation 3.
- Step 4: Repeat Step 2 and 3 to compute \hat{H}_i for all $i \in I$ to form the observed sample ground truth dataset \hat{H}_I .

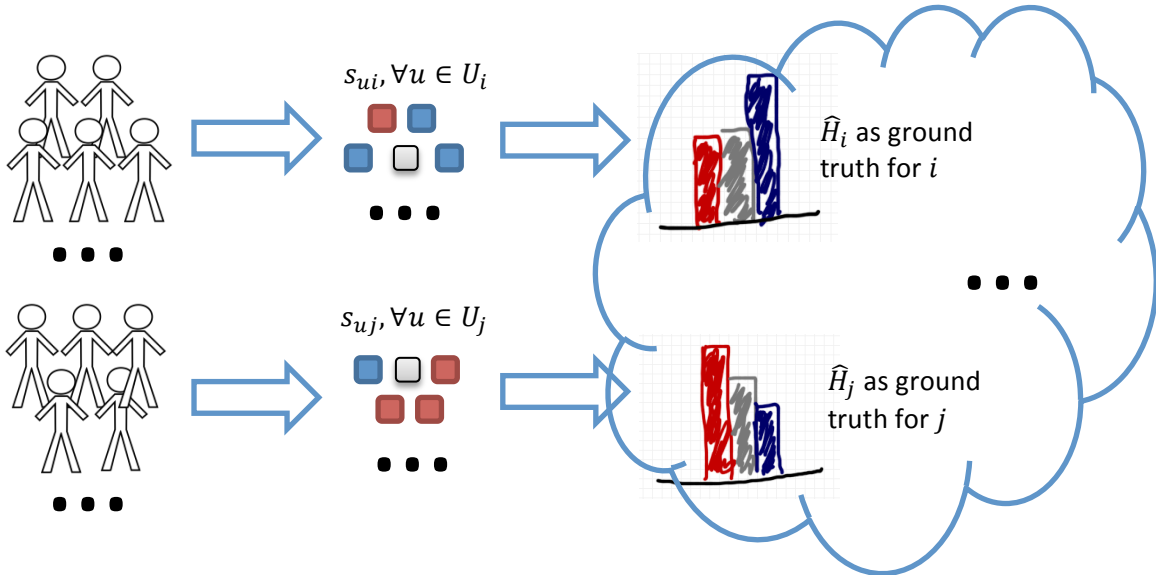


Figure 8. Define ground truth from the population

Summary: OCP versus SCP

Table 2 summarizes the main differences between the subjective and objective classification problems and their corresponding ground truth model.

Table 2. Comparisons between the objective and subjective classification problems

		Objective classification problems	Subjective classification problems
Problem characteristics	# of high agreement items	Almost all	Many
	# of low agreement items	Few	Many

⁴ Chapter 3 will show that it is advisable to sample the population from crowdsourcing platforms such as Amazon Mechanical Turk.

	Level of agreement among coders	High	Medium
	Existence of objective classification rules	Yes	No
Ground truth model	Underlying ground truth model	Labels	Distributions
	Items' ground truth defined by	Objective classification rules explicitly codified in the codebook by researchers	Coders who represent the population's collective opinions
Annotation process	Requirement of a codebook	Yes	No (optional)
	Role of coders	Instrument	Creator of the ground truth
	Coders' agreement due to	Following the same objective rules in the codebook	Shared tacit knowledge
	Coders' disagreement due to	Human errors	Subjective differences
	How to handle coders' disagreements	Eliminate them because errors are distractions	Keep them because subjective differences are useful information
	Challenges to reliability	Rules in the codebook not well-defined; coders not well trained	Sample bias and small sample size (large variance) when sampling coders
	Quality measurement of labeled items	IRR	Difference between \hat{H}_i and H_i
	Quality improvement by reducing coders' errors	Yes	No (coders' subjective differences are not errors, but it's a problem if they are sloppy and don't report what they think)
	Quality improvement by using larger # of coders	No (only small # of coders are needed if error rate is low, but it does help for any error rate)	Yes

2.5 Mapping Distributions into Labels

Defining the ground truth as distributions instead of labels does not necessarily mean that we will always use distributions in practice. Sometimes we might want to map distributions into labels. For example, we might want to display the political leaning of an article in a news aggregator as a red, gray or blue label instead of a distribution (see Figure 1 in chapter 1). Or, as I will argue later in chapter 4, we might want to map distributions into labels and then use the labels to train and evaluate political leaning classifiers that classify items into discrete categories.

Majority vote seems to be the obvious choice to map distributions into labels, which has been commonly used in previous studies. But it has some problems. For example, suppose we have 6 labels for an item i , including 3 red labels and 3 blue labels. Using majority vote, we would either label i as red or blue, not gray. But clearly i is quite disputed, and perhaps we should best label it as gray. Other studies used arbitrary criteria to map distributions into labels. For example, Zhou et al (2011) defined *red* as having $>2/3$ *red* ratings from 6 ratings, *blue* as having $>2/3$ *blue* ratings, and *gray* for the rest. Thus, 3 red and 3 blue would be labeled as gray. But one might ask why the mapping should be done in this particular way and not the other.

In this section, I will discuss a principled approach to map a distribution over the label space $\mathcal{L} = \{r, g, b\}$ into a single label $\ell \in \mathcal{L}$.

Cost Functions

Let $w_i \in \mathcal{L} = \{r, g, b\}$ be an arbitrary label for item i which will be shown to users as the political leaning of i . Let w_I be a set of $w_i, \forall i \in I$. For convenient purpose, consider w_i as the classification outcome of i from a political leaning classifier. I will use the cost function $L(w_i, s_{ui})$ to measure the cost of a person u , whose subjective opinion on i is s_{ui} , who sees a classification outcome on i as w_i . Here, “cost” could be interpreted as how much cognitive discomfort or resistance the person will experience if she sees a classification result different from her own opinion. Here, I’ll make a simplified

assumption that $L(w_i, s_{ui})$ does not depend on any other characteristics of the particular item beyond the mismatch between the user’s perceived label and the classifier’s. That is:

$$L(w_i, s_{ui}) = L(w, s), \quad \forall u \in U^*, \forall i \in I; w, s \in \mathcal{L}$$

Equation 4. Definition of cost

Therefore, a cost function here is defined only by nine values, as shown in Table 3. For example, $L(r, r)$ specifies the cost of a person, whose personal opinion to an article is red ($s_{ui} = r$), seeing the article labeled as red too ($w_i = r$): presumably the cost is 0. If a person’s own assessment to an article is blue ($s_{ui} = b$), then seeing it labeled as gray ($w_i = g$) might incur 1 unit of cost, that is $L(g, b)=1$, or seeing it labeled as red ($w_i = r$) might incur 2 units of cost, that is $L(r, b)=2$. Clearly this is a very simple model. A more sophisticated cost model will be discussed in the last section.

Table 3. Cost function

w \ s	red	gray	blue
red	$L(r, r)$	$L(r, g)$	$L(r, b)$
gray	$L(g, r)$	$L(g, g)$	$L(g, b)$
blue	$L(b, r)$	$L(b, g)$	$L(b, b)$

There are three types of cost in the table. First, $L(r, r)$, $L(g, g)$ and $L(b, b)$ are the “hit” cases where the classification result w_i matches a person’s personal opinion s_{ui} . Second, $L(r, b)$ and $L(b, r)$ are the “far miss” cases where the classification result w_i is the opposite of a person’s opinion s_{ui} . Third, $L(r, g)$, $L(b, g)$, $L(g, r)$ and $L(g, b)$ are the “near miss” cases where the classification result w_i does not match a person’s opinion s_{ui} , and either one of them is gray.

I’d like to assign numerical scores to the labels: red as “-1”, gray as “0”, and blue as “1”. There are other ways to assign the score, but I’d like to argue that in many cases, this is the most succinct form. Using the numerical scores, it is possible to define a cost

function using a simple equation form instead of a full 3 by 3 matrix like Table 3. For example:

- Absolute cost function: $L(w, s) = |w - s|$.
- Quadratic cost function: $L(w, s) = (w - s)^2$.
- 0-1 cost function: $L(w, s) = 1$ if $w \neq s$, or 0 otherwise.

Regardless of which cost function is used, the total expected cost of showing the classification results w_i for all items $i \in I$ to the entire population would be:

$$\begin{aligned} \mathbb{L} &= \sum_{i \in I} \mathbb{L}_i, \text{ where } \mathbb{L}_i = \sum_{\ell \in \mathcal{L}} P_\ell(i) \cdot L(w_i, \ell) \\ &= R_i \cdot L(w_i, r) + G_i \cdot L(w_i, g) + B_i \cdot L(w_i, b) \end{aligned}$$

Equation 5. Total expected cost

Recall that $P_\ell(i)$ and $\langle R_i, G_i, B_i \rangle$ were defined in Equation 1 as the “distribution as ground truth”, H_i , for item i . Since we are only able to observe the sample ground truth \hat{H}_i , we can only get an estimate of \mathbb{L} , defined in Equation 6. Note that after we observe \hat{H}_I , the total expected cost $\hat{\mathbb{L}}$ is solely determined by the cost function L and the classification outcomes w_I .

$$\begin{aligned} \hat{\mathbb{L}} &= \sum_{i \in I} \hat{\mathbb{L}}_i, \text{ where } \hat{\mathbb{L}}_i = \sum_{\ell \in \mathcal{L}} \hat{P}_\ell(i) \cdot L(w_i, \ell) \\ &= \hat{R}_i \cdot L(w_i, r) + \hat{G}_i \cdot L(w_i, g) + \hat{B}_i \cdot L(w_i, b) \end{aligned}$$

Equation 6. Total expected cost over sample ground truth

I’d like to use an example to illustrate the intuition of Equation 5 and Equation 6. Suppose we have an article i with $H_i = \langle 0.7, 0.2, 0.1 \rangle$. If we label it as red, then 70% of the population would be satisfied with the result, while the other 20% and 10% of the population who consider it as gray and blue would incur some cost. Suppose I use the quadratic cost function, then $\mathbb{L}_i = .7 \times 0 + .2 \times 1 + .1 \times 4 = 1.3$. That is, showing the article as red would have an expected cost of 1.3.

Cost-minimizing Label and Partitioning

For any item i with H_i , there is an optimal label $l_i \in \mathcal{L}$ that minimizes \mathbb{L}_i . I'd like to call l_i the “**cost-minimizing label**” of i . I'd like to argue that the principled way to map a distribution into a label is to map H_i into its cost-minimizing label $l_i, \forall i \in I$.

Take the absolute cost function for example, the expected cost of classifying i as red is:

$$\mathbb{L}_{i,r} = R_i \cdot 0 + G_i \cdot 1 + B_i \cdot 2 = G_i + 2B_i$$

Here I use the notation $\mathbb{L}_{i,r}$ to denote the cost of classifying i as red given H_i . Similarly, we have:

$$\mathbb{L}_{i,g} = R_i \cdot 1 + G_i \cdot 0 + B_i \cdot 1 = R_i + B_i$$

$$\mathbb{L}_{i,b} = R_i \cdot 2 + G_i \cdot 1 + B_i \cdot 0 = G_i + 2R_i$$

In order for red to be the cost-minimizing label of i , $\mathbb{L}_{i,r}$ has to be lower than both $\mathbb{L}_{i,g}$ and $\mathbb{L}_{i,b}$. That is, $R_i > 0.5$, because:

$$\begin{cases} \mathbb{L}_{i,r} < \mathbb{L}_{i,g} \Rightarrow G_i + 2B_i < R_i + B_i \xrightarrow{R_i+G_i+B_i=1} R_i > 0.5 \Rightarrow R_i > 0.5 \\ \mathbb{L}_{i,r} < \mathbb{L}_{i,b} \Rightarrow G_i + 2B_i < G_i + 2R_i \Rightarrow R_i > B_i \end{cases}$$

Similarly, in order for blue to be the cost-minimizing label of i , we need to have $B_i > 0.5$. Therefore, under the absolute cost function, the cost minimizing label for i would be:

$$l_i = \begin{cases} r & \text{if } R_i > 0.5 \\ b & \text{if } B_i > 0.5 \\ g & \text{otherwise} \end{cases}$$

Equation 7. Cost-minimizing label for item i under the absolute cost function

Take the quadratic cost function for another example. The cost of classifying item i as red, gray, or blue would be:

$$\mathbb{L}_{i,r} = R_i \cdot 0 + G_i \cdot 1 + B_i \cdot 4 = G_i + 4B_i$$

$$\mathbb{L}_{i,g} = R_i \cdot 1 + G_i \cdot 0 + B_i \cdot 1 = R_i + B_i$$

$$\mathbb{L}_{i,b} = R_i \cdot 4 + G_i \cdot 1 + B_i \cdot 0 = G_i + 4R_i$$

In order for red to be the cost-minimizing label for i , we need:

$$\begin{cases} \mathbb{L}_{i,r} < \mathbb{L}_{i,g} \implies G_i + 4B_i < R_i + B_i \xrightarrow{R_i+G_i+B_i=1} R_i > 0.5 + B_i \implies R_i > 0.5 + B_i \\ \mathbb{L}_{i,r} < \mathbb{L}_{i,b} \implies G_i + 2B_i < G_i + 2R_i \implies R_i > B_i \end{cases}$$

Therefore, under the quadratic cost function, the cost minimizing label for i would be:

$$l_i = \begin{cases} r & \text{if } R_i > 0.5 + B_i \\ b & \text{if } B_i > 0.5 + R_i \\ g & \text{otherwise} \end{cases}$$

Equation 8. Cost-minimizing label for item i under the quadratic cost function

Suppose there is an item i with $H_i = \langle 0.51, 0, 0.49 \rangle$. Using the absolute cost function, l_i should be red because $R_i = 0.51 > 0.5$. Intuitively, it means that showing i labeled as red to the readers would incur less cost than showing it as either gray or blue, given that the readers' cost is modeled under the absolute cost function. Using the quadratic cost function however, l_i should be gray according to Equation 8. Note that the quadratic cost function punishes "far misses" more than the absolute cost function, because a far miss incurs 4 units of cost instead of 2. Since i has 49% of the population looking at it as blue, thus l_i should be gray in order to avoid the excessive penalty for a far miss under the quadratic cost function.

Intuitively, equations such as Equation 7 and Equation 8 are rules that determine how to partition a set of items with distributions H_i into cost-minimizing labels l_i , which I'd like to call "rules partitioning" or partitioning in the form of rules. Recall that an item's political leaning can be represented either as a distribution H_i or as a point on the simplex: an item's simplex notation is equivalent to its distribution notation. Similarly, the rules partitioning also maps to a particular partitioning on the simplex, and vice versa. For example, Figure 9 shows the partitioning on the simplex according to Equation 7 for the absolute scoring rule. A partitioning on the simplex cuts a simplex into three partitions, where the items covered in each partition all have the same cost-

minimizing label, which I'd like to call "simplex partitioning" or partitioning in the simplex form. Partitioning in the form of rules or in the form of simplex are for use with items' distribution notation or simplex notation respectively, and thus both forms are equivalent. I will use both forms interchangeably.

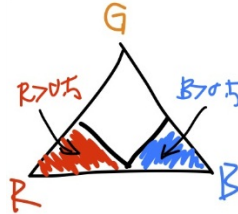


Figure 9. Simplex partitioning for the absolute scoring rule

Equivalence of Cost Function and Partitioning

Next, I'll prove that a cost function and a partitioning have a one-to-one correspondence. The proof requires some axioms about cost functions and partitioning, which are also used for choosing the right cost function (to be discussed in the next sub-section). I will introduce the axioms here, and return to them in the next sub-section.

- *Locality.* The cost-minimizing label is red for an item i with $H_i = \langle 1, 0, 0 \rangle$. And there exists $\varepsilon > 0$ such that the cost minimizing label for another item j with $H_j = \langle 1 - 2\varepsilon, \varepsilon, \varepsilon \rangle$ is also red. Similarly for blue and gray.
- *Monotonicity.* If the cost-minimizing label for an item i with $H_i = \langle R, G, B \rangle$ is red, then the item j with $H_j = \langle R + k, G - k_1, B - k_2 \rangle$ is also red, where $k, k_1, k_2 > 0$, and $k = k_1 + k_2$. Similarly for blue. Note the similar property need not hold for gray.
- *Canonicity.* In a cost function, $L(w, s) = 0$ if $w = s$, and $L(w, s) > 0$ if $w \neq s$. The smallest non-zero unit of $L(w, s)$ is 1.
- *Symmetry.* Red and blue are symmetric in that $L(g, r) = L(g, b) = k_1$, $L(b, g) = L(r, g) = k_2$, and $L(r, b) = L(b, r) = k_3$. Or, on the simplex notation, the red and blue partitions are symmetric about the bisecting line from gray to red and blue.
- *Linearity.* Any rules partitioning take the form of Equation 9. Or, any simplex partitioning is defined by straight lines instead of curves.

$$\begin{cases} r \text{ if } R_i > \beta_1 + \beta_2 \cdot B_i \\ b \text{ if } B_i > \beta_1 + \beta_2 \cdot R_i, \\ g \text{ otherwise} \end{cases} \quad \text{where } \beta_1, \beta_2 \in (-\infty, \infty)$$

Equation 9. Linearity

Now, I will generalize the process that derives Equation 7 and Equation 8, and show that given a cost function, there will be one and only one corresponding partitioning. The following equations show \mathbb{L}_i if i is classified as red, gray, or blue respectively:

$$\mathbb{L}_{i,r} = R_i \cdot L(r, r) + G_i \cdot L(r, g) + B_i \cdot L(r, b)$$

$$\mathbb{L}_{i,g} = R_i \cdot L(g, r) + G_i \cdot L(g, g) + B_i \cdot L(g, b)$$

$$\mathbb{L}_{i,b} = R_i \cdot L(b, r) + G_i \cdot L(b, g) + B_i \cdot L(b, b)$$

In order for red to be the cost-minimizing label for i , we need to have 1) $\mathbb{L}_{i,r} < \mathbb{L}_{i,g}$, and 2) $\mathbb{L}_{i,r} < \mathbb{L}_{i,b}$. Note that according to the “canonicity” axiom, $L(r, r) = L(g, g) = L(b, b) = 0$. And according to the definition of H_i , we have $G_i = 1 - R_i - B_i$. Therefore, we have:

$$1) \mathbb{L}_{i,r} < \mathbb{L}_{i,g}$$

$$\Rightarrow G_i \cdot L(r, g) + B_i \cdot L(r, b) < R_i \cdot L(g, r) + B_i \cdot L(g, b)$$

$$\xrightarrow{R_i + G_i + B_i = 1} R_i \cdot L(g, r) + B_i \cdot L(g, b) > (1 - R_i - B_i) \cdot L(r, g) + B_i \cdot L(r, b)$$

$$\Rightarrow R_i > \frac{L(r, g)}{L(g, r) + L(r, g)} + \frac{L(r, b) - L(r, g) - L(g, b)}{L(g, r) + L(r, g)} \cdot B_i$$

$$\Rightarrow R_i > \frac{k_2}{k_1 + k_2} + \frac{k_3 - k_2 - k_1}{k_1 + k_2} \cdot B_i$$

$$2) \mathbb{L}_{i,r} < \mathbb{L}_{i,b}$$

$$\Rightarrow G_i \cdot L(r, g) + B_i \cdot L(r, b) < R_i \cdot L(b, r) + G_i \cdot L(b, g)$$

$$\Rightarrow R_i > B_i$$

Equation 10. Generalized rules of partitioning

According to the “symmetry” axiom, we will have similar results for blue to be the cost-minimizing label for item i . In short, given a cost function $L(w, s)$, we can always derive

rules partitioning according to the procedure listed in Equation 10. Therefore, I have concluded that a cost function will determine a partitioning.

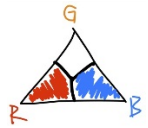
On the other hand, according to the “linearity” axiom, any partitioning in the form of rules would look like Equation 9. Let:

$$\begin{cases} \beta_1 = \frac{k_2}{k_1 + k_2} \\ \beta_2 = \frac{k_3 - k_2 - k_1}{k_1 + k_2} \end{cases}$$

Note that according to the “canonicity” axiom, either k_1 or k_2 would be the smallest unit of cost, which is 1. So we can easily solve the first equation to get the positive values of k_1 and k_2 , and then solve the second equation to get the positive value of k_3 as well. In short, given a partitioning (which takes the form of Equation 9 under the “linearity” axiom), we can always derive a corresponding cost function. All in all, that concludes that a cost function and a partitioning have a one-to-one correspondence, given the axioms.

Table 4 summarizes a few cost functions and their equivalent partitioning. For the rest of the thesis, I will use “cost function” and “partitioning” interchangeably. Note that the 0-1 cost function maps to the “majority vote” partitioning, as can be seen in both its rules and simplex notation. Intuitively, since the 0-1 cost function only distinguishes between a hit and a miss regardless of whether the miss is a far miss or a near miss, the label with the most votes would naturally be the cost-minimizing label.

Table 4. Comparison of cost functions and partitioning

Mnemonics	Cost function	Rules partitioning	Simplex partitioning
0-1, or majority vote	$\begin{matrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{matrix}$	$\begin{cases} r \text{ if } R > 0.5 - 0.5B \\ b \text{ if } B > 0.5 - 0.5R \\ g \text{ otherwise} \end{cases}$ <p>or equivalently,</p> $\begin{cases} r \text{ if } R > B \text{ and } R > G \\ b \text{ if } B > R \text{ and } B > G \\ g \text{ if } G \geq R \text{ and } G \geq B \end{cases}$	

Absolute	$\begin{matrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{matrix}$ or $ w-s $	$\begin{cases} r \text{ if } R > 0.5 \\ b \text{ if } B > 0.5 \\ g \text{ otherwise} \end{cases}$	
Quadratic	$\begin{matrix} 0 & 1 & 4 \\ 1 & 0 & 1 \\ 4 & 1 & 0 \end{matrix}$ or $(w-s)^2$	$\begin{cases} r \text{ if } R > 0.5 + B \\ b \text{ if } B > 0.5 + R \\ g \text{ otherwise} \end{cases}$	
S_3	$\begin{matrix} 0 & 1 & 3 \\ 1 & 0 & 1 \\ 3 & 1 & 0 \end{matrix}$ or $(w-s)^{\log_2 3}$	$\begin{cases} r \text{ if } R > 0.5 + 0.5B \\ b \text{ if } B > 0.5 + 0.5R \\ g \text{ otherwise} \end{cases}$	
$S_{1/3}$	$\begin{matrix} 0 & 1 & 6 \\ 2 & 0 & 2 \\ 6 & 1 & 0 \end{matrix}$	$\begin{cases} r \text{ if } R > 1/3 + B \\ b \text{ if } B > 1/3 + R \\ g \text{ otherwise} \end{cases}$	
$S_{2/3}$	$\begin{matrix} 0 & 2 & 6 \\ 1 & 0 & 1 \\ 6 & 2 & 0 \end{matrix}$	$\begin{cases} r \text{ if } R > 2/3 + B \\ b \text{ if } B > 2/3 + R \\ g \text{ otherwise} \end{cases}$	
$S_{2/5}$	$\begin{matrix} 0 & 1 & 2.5 \\ 1 & 0 & 1 \\ 2.5 & 1 & 0 \end{matrix}$	$\begin{cases} r \text{ if } R > 0.5 + 0.25B \\ b \text{ if } B > 0.5 + 0.25R \\ g \text{ otherwise} \end{cases}$	
$S_{3/5}$	$\begin{matrix} 0 & 1 & 5 \\ 3 & 0 & 3 \\ 5 & 1 & 0 \end{matrix}$	$\begin{cases} r \text{ if } R > 0.25 + 0.5B \\ b \text{ if } B > 0.25 + 0.5R \\ g \text{ otherwise} \end{cases}$	

Choosing a Cost Function

Clearly, we should choose the cost function that best describes the reality of cost. Perhaps the best approach to choose the right cost function is to directly inquire of the target population by modeling the cost when a person views an article labeled differently from her own personal opinion. However, due to time and scope constraints, I'm not able to do that in this thesis, and will discuss it in future work. Instead, I will use two steps to choose the cost function for political leaning classification.

The first step is to apply the axioms discussed earlier. Note that the “canonicity” axiom greatly simplifies the model by ruling out “negative costs” which can be interpreted as “gains” for cases where a classification matches a person’s own opinion. It also defines the smallest unit of cost as 1, which normalizes a cost function not to take any arbitrary number. Also, the “symmetry” axiom implies that whatever applies to red also applies to blue, but gray could be different from both red and blue, which can be seen in the “monotonicity” axiom as well. Finally, note that the axioms rule out many irregular cases of cost functions and partitioning. But we still need to choose the right cost function from many possible candidates, such as those listed in Table 4, which all satisfy the axioms.

The second step is to use heuristics that specify how to associate labels to distributions, and then infer the best partitioning and cost function that matches the heuristics. For example, suppose a heuristic says a distribution $\langle 0.51, 0, 0.49 \rangle$ should be labeled as gray instead of red. Then it implies that we should choose the quadratic partitioning over the absolute partitioning because the former matches the heuristic while the latter does not. As illustrated in Figure 10, the coloring of each point on the simplex represents a heuristic, and the quadratic partitioning (Figure 10a) is preferred over the absolute partitioning (Figure 10b) because it better matches the heuristics.

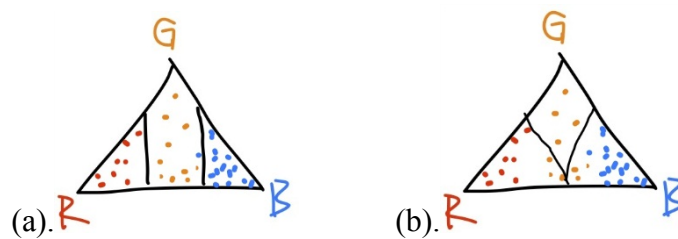


Figure 10. The quadratic partitioning (a) matches the heuristics better than the absolute partitioning (b)

Table 5 lists a few heuristics. I’d like to point out that gray is special in that low agreement cases such as $\langle 0.51, 0, 0.49 \rangle$ or $\langle 0.4, 0.3, 0.3 \rangle$ are labeled as gray even though the majority of the population don’t consider them gray. I will discuss more about high agreement gray and low agreement gray in the last section.

Table 6 lists a few candidate cost functions and how they respond to the heuristics. Overall, I'd like to choose the quadratic cost function because it matches the heuristics well. **In this thesis, I will use the quadratic cost function and partitioning** to model the cost for political leaning classification, where a “far miss” incurs 4 times more costs than a “near miss”. The choice of the quadratic cost function is for the political leaning classification problem where the heuristics are drawn. Other problems might find other cost functions more appropriate. The structure of the distribution as ground truth model allows using any cost function to map distributions into labels.

Table 5. Heuristics that associate labels to distributions

Label l	Distribution H
Red	<ul style="list-style-type: none"> • Strong red: $\langle 0.9, 0, 0.1 \rangle, \langle 0.8, 0.1, 0.1 \rangle$ • Weak red: $\langle 0.7, 0.2, 0.1 \rangle$
Blue	<ul style="list-style-type: none"> • Strong blue: $\langle 0.1, 0, 0.9 \rangle$ • Weak blue: $\langle 0.1, 0.3, 0.6 \rangle, \langle 0.2, 0.2, 0.6 \rangle$
Gray	<ul style="list-style-type: none"> • Clean gray: $\langle 0.1, 0.8, 0.1 \rangle$ • Weak gray: $\langle 0.2, 0.7, 0.1 \rangle$ • Polarized gray (could be rare): $\langle 0.4, 0, 0.6 \rangle, \langle 0.51, 0, 0.49 \rangle$ • Mixed gray (could be rare): $\langle 0.4, 0.3, 0.3 \rangle$

Table 6. Cost functions and heuristics

Heuristics		R>G	R>0.5	R>0.5+B	R>0.5+0.5B	R>1/3+B	R>2/3+B
$\langle 0.51, 0, 0.49 \rangle$	gray	red	red	gray	gray	gray	gray
$\langle 0.2, 0.2, 0.6 \rangle$	blue	blue	blue	gray	gray	blue	gray
$\langle 0.19, 0.2, 0.61 \rangle$	blue	blue	blue	gray	blue	blue	gray
$\langle 0.7, 0.2, 0.1 \rangle$	red	red	red	red	red	red	gray
$\langle 0.9, 0.05, 0.05 \rangle$	red	red	red	red	red	red	red
$\langle 0.09, 0.4, 0.51 \rangle$	gray	blue	blue	gray	gray	blue	gray
$\langle 0.51, 0.2, 0.29 \rangle$	gray	red	red	gray	gray	gray	gray
$\langle 0.4, 0.5, 0.1 \rangle$	gray	gray	gray	gray	gray	gray	gray
$\langle 0.31, 0.69, 0 \rangle$	gray	gray	gray	gray	gray	red	gray

Finally, note that according to the “locality” and “monotonicity” axioms, the small regions by the three corners of the simplex would always remain the same color as red, gray and blue respectively regardless of which cost function to choose. Different cost functions would only affect the partitioning of items in the middle of the simplex where

people don't always agree. I'd like to call the region on the simplex where items are assigned different colors between two partitioning as the "partitioning disputed region" or simply "disputed region". For example, Figure 11(a) illustrates the disputed region between the 0-1 cost function and quadratic cost function, and Figure 11(b) illustrates the disputed region between the quadratic cost function and the absolute cost function. In chapter 3, I will show that many items are covered in the disputed region in the real dataset, which implies that choosing different cost functions does matter.

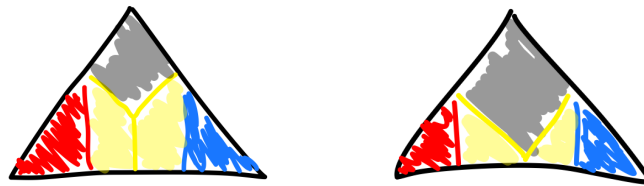


Figure 11. Partitioning disputed region, highlighted in yellow: (a) between 0-1 and quadratic; (b) between absolute and quadratic

2.6 Discussion and Limitation

Using the Wrong Ground Truth Model

I have introduced both the objective and subjective classification problems, and have argued to use the label as ground truth for OCPs, and the distribution as ground truth for SCPs. One mistake is to use label as ground truth on SCPs, as illustrated in Figure 12. This sub-section discusses the problem of using the wrong ground truth model.

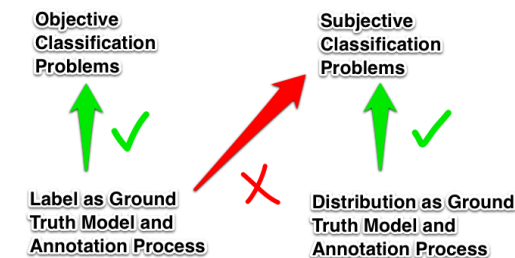


Figure 12. Using the wrong ground truth model

To begin with, if we treat a classification problem as objective and use the label as ground truth model, then the focus is on how to decrease or eliminate rater's error rate

in order to get the true label. However, with the distribution as ground truth model, we will not try to decrease rater's error rate because the ratings are subjective, and we will not assume the labeled dataset is reliable with only a few labels per item. It opens up new possibilities (such as this chapter) to explore the phenomenon of low agreement among coders and study how that affects the reliability of labeled dataset and classifier evaluation results.

In addition, the distribution as ground truth model allows various partitioning (quadratic in particular in this thesis) to map distributions into single labels, while the label as ground truth model only allows majority vote, resulting in many items in the partitioning disputed region get assigned different labels (see Figure 11). In chapter 3, I will use real empirical dataset to show that the political leaning classification, as a typical SCP, has 19.4% items in the disputed region which will get labeled differently if we treat it as an OCP (with majority vote) instead of an SCP (with quadratic partitioning).

Furthermore, the labeled dataset for an SCP is much less reliable than the researchers would have expected if the classification problem had been objective. Using the unreliable labeled dataset, researchers are more likely to draw unreliable conclusions about classifier evaluation. I will discuss it in more detail in chapter 4 with computer simulations.

It might seem that we do not have distributions if we only get one label per item. However, I'd like to point out that the distribution as ground truth model is defined regardless of the number of labels to get for each item. Even if we only get one label per item, the one label is an estimate to a true distribution under the distribution as ground truth model (although with high sample error), instead of an estimate to a true label under the label as ground truth model.

Finally, I'd like to point out that there is no clear line to cut between OCPs and SCPs. Even for a clearly objective classification problem, such as to label passport images as male or female, there could still be a small number of ambiguous cases where human coders have to consult their own subjective opinions. One might reasonably argue that a

“pure” objective classification problem does not exist in the real world. However, the concern of this chapter is not about a typical OCP: even though it might have a few ambiguous items, most items are still clearly labeled. The bottom line is that there exist a few problems such as the political leaning classification problem where subjectivity level is quite high and coders don’t agree on *many* items. For such problems, the traditional view of label as ground truth label simply does not suffice anymore, both conceptually and practically (which will be discussed in the next two chapters), and thus it is necessary to use the distributions as ground truth model.

Eliciting s_{ui}

Recall that s_{ui} is the personal assessment from user u on item i , and H_i and \hat{H}_i are defined simply by aggregating s_{ui} . I’d like to address some common concerns about this simple process and point out directions for future research.

To begin with, s_{ui} is drawn randomly from the population regardless of whether the coder is an expert or not. Being an expert coder implies that there exists an objective ground truth on which to judge how accurate a coder can label items correctly. However, I have argued that for a subjective classification problem, people have their own opinions on items and there is no objective “correct” label against which to judge a person’s expertise. Thus it is not fair to distinguish a coder as expert or inexperienced based on their subjective assessment. One might confuse eliciting people’s personal opinions with eliciting people’s predictions on other people’s opinions, where the latter is affected by expertise and will be discussed in chapter 3 “related work”.

Let r_{ui} be the report of a rater u ’s personal assessment of an item i . So far in this chapter, I have assumed $r_{ui} = s_{ui}$. In reality, however, there might be a few cases where $r_{ui} \neq s_{ui}$. For example, some raters might want to game the system by altering r_{ui} in order to earn more bonus. Or some coders simply do not put forth enough effort to label items. In chapter 3, I will address some practical concerns of eliciting truthful reports from raters. More sophisticated incentive-compatible models of eliciting s_{ui} or assuming s_{ui} to be a random variable would be left to future work. Note that there

could be small random errors when eliciting s_{ui} , but these small random errors are accounted for in the definition of H_i .

People from different sub-populations might have systematically different s_{ui} . For example, liberals are more likely to view liberal articles as neutral while conservatives view conservative articles as neutral (Garrett 2005). Politically savvy people might be reluctant to say "gray", and less involved people might be more willing to say "I don't know". Future work should address the effects of bias from sub-populations, and perhaps define distributions of the distributions H_i .

Finally, note that $s_{ui} \in \mathcal{L}$ is a label while the ground truth of items are defined as distributions over \mathcal{L} . In this chapter, I assume each individual person's assessment of an article is still a red, gray or blue label, not a distribution – only when aggregating individual assessments from multiple people can we get the distributions.

Cost Model

Previously, I have made a simplified assumption that cost remains the same across items and users regardless of their individual differences (see Equation 4 and Table 3). Next, I will discuss some possible extensions to the simplified cost model, which will be left to future work.

Cost might depend on many other factors. For example, misclassification may occur due to biases or assumptions regarding the source of an article. People might find it acceptable that an article from the New York Times is labeled as blue, even though without knowing the source, they may have otherwise considered it red. Or people from different sub-populations, such as politically savvy users and apolitical people, might have different cost models. Future work might need to consider these issues too.

In section 2.5, I have introduced some axioms that claim a cost function does not have negative costs or gains, and needs to be symmetric and linear. These limitations introduced by the axioms could be released in future work too.

Low Agreement Items

A subjective classification problem is special because it has many low agreement items. In this chapter, I have used the quadratic partitioning to map the low agreement items into the “gray” category. These low agreement gray items might be conceptually different from the high agreement gray items, because in the former category an item could be gray even though no one considers it as gray. Further work is needed to study whether low agreement items should be put into a separate category from items that everyone agrees are gray.

Generalization

The political leaning classification problem has two symmetric, opposing polarized categories – red and blue – and a middle ground gray category. Many other subjective classification problems only have one pole and two categories. For example, to label whether a tweet is humorous or not has only one pole. Of course, we can add an extra “gray” category to such problems. But it is not clear whether we can still assign -1, 0, and 1 to the three categories and assume symmetry between the -1 and 1 categories. More importantly, perhaps we want to keep the two categories without adding a new “gray” category. Future work should study how to define distribution as ground truth model for such problems.

Chapter 3. Annotation Elicitation

This chapter fills a gap of chapter 2 by showing that the real political leaning dataset indeed contains many low agreement items whose ground truth should be modeled as distributions instead of labels. It is also an example to demonstrate how to practically obtain labeled data for a subjective classification problem. The labeled items obtained in this chapter will be used as ground truth to train and evaluate political leaning classifiers in chapter 5.

One goal of annotation elicitation for a subjective classification problem is to be able to obtain raters' truthful personal assessments. It should not be confused with eliciting raters' predictions on other people's assessments. Researchers sometimes evaluate raters' ratings against each other and reward those raters whose ratings match the majority (e.g., Ipeirotis et al 2010). However, this will create incentives for raters to report their predictions on other people's assessments instead of truthfully report their own. For example, if a rater personally considers an article red, but believes that most people would label it as blue, then she would report blue instead of red if she knew her annotation would be evaluated against others'. Nevertheless, eliciting predictions could be combined with eliciting personal assessments as a quality control mechanism. I will discuss more about it in the "related work" sections.

Another goal of annotation elicitation is to randomly sample raters from the population in order to avoid bias. Hiring raters who have similar background and who live in the same geological location (such as undergraduate students in a liberal college town) could be problematic: their assessments on the political leaning of items might not represent how the population views them. Therefore, it is natural to hire raters from a

platform such as Amazon Mechanical Turk (AMT) that has a large and diverse pool of raters. I will discuss more about it in the “data collection” section.

The organization of the chapter is as follows. I will discuss related work in section 3.1. Section 3.2 discusses data collection on Amazon Mechanical Turk. Section 3.3 reports results. And finally section 3.4 discusses conclusions and future work.

3.1 Related Work

Quality Control for Crowdsourcing

The term “crowdsourcing” was introduced by Howe (2006), and systematically studied by von Ahn (2006), Howe (2008), and Quinn and Bederson (2011). Crowdsourcing is a process that involves outsourcing tasks to a distributed group of people, in which humans and computers can work together to solve problems. This chapter uses the paid micro-task crowdsourcing platform, Amazon Mechanical Turk (AMT), to collect human annotations on political leaning of articles.

Adar (2011) distinguishes “the science of crowdsourcing” from “crowd-sourced science”, where the former refers to the systematic study of crowdsourcing as a scientific methodology, and the latter refers to using crowdsourcing to do science. This chapter belongs to the “crowd-sourced science” category, and does not contribute to the science of crowdsourcing. Here, I will discuss related work about the validity of crowd-sourced science and quality control for crowdsourcing. I will directly apply findings from prior research.

One line of work is to study the validity of using crowdsourcing to do scientific research. For example, Paolacci et al (2010) duplicated social science experiments on AMT and concluded that the results from AMT are as good as results from traditional studies. Snow et al (2008) found that experts hired offline are better labelers than non-experts on AMT, but aggregating multiple labels from non-experts to train supervised learning algorithms is better than using only one expert to train the system. They also found that four non-experts were as good as one expert for their task. Mason and Suri (2011)

summarized prior research about the validity of using AMT to do behavioral experiments and discussed guidelines on how to do them properly. All in all, prior research has convincingly demonstrated that using crowdsourcing platforms such as AMT is valid to do scientific research.

However, one challenge to the validity of crowd-sourced science is the concern of unreliable human laborers on the crowdsourcing platform. Next, I will discuss different approaches in terms of quality control, some of which will be applied in the annotation process in the “data collection” section.

Monetary incentives: One line of work about quality control is to study how to design monetary incentives to increase work quality on AMT. Rogstadius et al (2011) studied incentives with AMT and found that intrinsic motivations (e.g., non-profit work) increase quality but not quantity when the price is low, and that extrinsic motivations (e.g., monetary reward) increase quantity but not quality. Mason and Watts (2009) found that: 1) increased financial incentives increase the quantity but not the quality, 2) all workers felt like they were being paid less than they deserved, thus were no more motivated to perform better no matter how much they were actually paid, 3) the number of completed tasks decreased with increasing task difficulty, and 4) paying workers a low rate led them to perceive their work as less valuable than not paying them at all. Harris (2011) compared workers’ accuracy against a pre-labeled gold standard and then used both bonus and punishment as incentive to improve work quality. Kazai (2010) found that 1) higher pay results in more usable labels and less spam, 2) workers with qualifications are not affected by pay, 3) over-confident workers produce less accuracy, and 4) hard tasks (which require more effort) reduce accuracy. Shaw et al (2011) studied 14 incentives categorized as social, financial and hybrid, and found that only two incentives worked: 1) Bayesian Truth Serum, where the workers were asked to predict other workers’ responses and got rewarded accordingly, and 2) punishment by reducing the payment if a particular worker’s response doesn’t agree with the others’.

Qualification based approaches: Kittur et al (2008) found that it is important to have explicitly verifiable questions as part of the task and that it is advantageous to design the task such that completing it accurately and in good faith requires as much or less effort than non-obvious random or malicious completion. They also found that it is useful to have multiple ways to detect suspect responses. Chen and Dolan (2011) argued for maintaining long-term relationship with AMT workers and using a multi-tiered system to pay workers based on their performance.

Excluding poor quality workers: Dekel and Shamir (2009) used AMT to collect labels, and argued that the label qualities would increase by removing low quality workers. Donmez et al (2009) proposed an algorithm to compute the quality of workers in AMT and then used the IETHresh algorithm to obtain labels only from the good workers for active learning. They also proposed an extension of the algorithm that learns the quality of workers assuming it changes over time (Donmez et al 2010)

Better user interface design: Chandler and Horton (2011) argued that researchers should pay attention to HIT interface design to put the incentive descriptions at the “focal point” on the page. The paper also argued that \$0.01 bonus is almost as good as \$0.05 bonus, and monetary incentive works better than cheap-talk. Chilton et al (2010) argued that AMT tasks that are at the top of the most recently updated list would get processed faster.

Iterative vs. parallel design: Instead of workers working in parallel, the iterative HIT design asks each worker’s input as incrementally built upon previous inputs (Little et al 2009). Little et al (2010) discussed the difference between iterative and parallel design. Bernstein et al (2010) proposed the “Find-Fix-Verify” iterative design to break down a big task into smaller subtasks and used it to revise research papers on AMT.

Minimum working time requirement: Kapelner and Chandler (2010) discussed preventing “satisficing” in AMT online surveys when it is impossible to directly evaluate the results. The paper suggested using the “trip door questions” as an indirect measure of survey quality. They found that forcing workers to spend a certain amount of time on

HITs increases quality and does not decrease quantity with the “Kapcha” technique. Downs et al (2010) found that the time workers spent on a survey cannot predict the quality of their work, and argued that verification questions, if used, should require mental effort in order to be effective.

Instant feedback of “gold”: Sorokin and Forsyth (2008) and Le et al (2010) argued that displaying gold standard results as feedback while workers are working on the HITs will increase work quality. Dow et al (2011) also argued that showing feedback to workers would be useful. Horton (2010) explored whether having peer workers review each other’s work would improve bad workers’ performance, and argued against using peer reviews in the real workplace.

Effects of task difficulty: Horton and Chilton (2010) studied how work difficulty level would affect output, and found that workers would work on the same amount of tasks regardless of whether the tasks are easy (spending less time) or hard (spending more time). They also found that within the same difficulty level, reducing price would reduce output. The paper then argued that there might be “target earners”, who want to earn a certain amount of money regardless of how much time they spend. For “target earners”, increased price means reduced output and decreased price means increased output.

Effects of demographics: Chandler and Kapelner (2010) found that “US, but not Indian, workers are induced to work at a higher proportion when given cues that their task was meaningful. However, conditional on working, whether a task was framed as meaningful does not induce greater or higher quality output in either the US or in India.”

Eliciting Truthful Signals

The goal of eliciting signals is to have the coders report signals that match their true beliefs. The challenge is how to evaluate the subjective signals when there is no ground truth to tell whether the signals are truthful or not.

Prelec (2004) proposed the Bayesian Truth Serum (BTS) approach to elicit both the truthful subjective judgments and predictions on distributions at the same time. The BTS

approach is derived from Bayesian theory that says by observing X , the estimate of $P(X)$ as the posterior would be higher than the prior $P(X)$. And therefore, this approach “assigns high scores not to the most common answers but to the answers that are more common than collectively predicted”. The BTS approach only works when one can obtain a large m (number of labels for each item). Witkowski and Parkes (2012) proposed the “robust” BTS approach that can work even with a relatively small m as long as $m \geq 3$. Besides the line of research on BTS, Miller, Resnick, and Zeckhauser (2005) proposed the “peer prediction” method that elicits truthful signals from raters without having to ask for predictions on the distributions.

One limitation of the BTS approach or the peer prediction approach, when applying it in practice, is that it is hard to explain and convince human coders to believe that it works. In reality, people are not typically rational enough to follow what has been suggested from theories, especially when the theories are complicated. Another limitation is the lack of empirical studies to support that those approaches really work in practice. For these reasons, this chapter will take a much simpler approach to elicit truthful signals, which will be discussed in the “data collection” section.

Another related line of work in terms of eliciting truthful signals is found in the web survey literature. For example, Reips (2000) discussed 18 advantages and 7 disadvantages of web surveys, among which one of the disadvantages is the difficulty of evaluating the truthfulness of respondents. The bottom line of this work is to ask surveyors to follow best practice guidelines when conducting web surveys, and then proceed assuming the obtained survey result is reliable. This chapter takes a similar stance: after applying quality control approaches, I assume the personal signals obtained are truthful.

Eliciting Predictions

Rothschild and Wolfers (2011) suggested that there could be two alternative types of reports from human coders: report your own assessment, and predict other people's assessments. They studied presidential election polling based on projections from

questions of voter intentions (or signals) versus questions probing voters' expectations (or predictions), where the former typically asks "If the election were held today, who would you vote for?" versus the other typically asks "Regardless of who you plan to vote for, who do you think will win the upcoming election?" They found that "polls of voter expectations consistently yield more accurate forecasts than polls of voter intentions", because "we are polling from a broader information set, and voters respond as if they had polled twenty of their friends". They proposed a small-scale structural model as a rational interpretation for why respondents' forecasts are correlated with their expectations, and showed that they could "use expectations polls to extract accurate election forecasts even from extremely skewed samples".

Another work along the same line by Krupka and Weber (2012) studied a pure matching coordination game to elicit people's ratings on social norms, in which the researchers "provide respondents with incentives not to reveal their own personal preferences but instead to match the responses of others". The coordination game approach traces back to earlier studies where researchers found that common culture and shared experiences can create focal points that makes one equilibrium favorable over others (Schelling 1960; Mehta et al, 1994; Sugden, 1995), and coordination games can be used with monetary incentives to reveal shared understanding (Camerer & Fehr, 2004). The work uses game theory to explain why eliciting predictions might work better than eliciting signals.

Both work suggest a new way of annotation elicitation: people might have more information about an article's political leaning than their own personal assessments, and therefore asking them to predict the article's distribution or its cost-minimizing label might return better results compared to simply eliciting their personal signals. I will discuss more about this approach in the "future work" section.

Data collection in this chapter indeed asked coders to report their predictions as well as their personal signals. But eliciting predictions was used primarily as a quality control approach, in the similar fashion as the Bayesian Truth Serum approach discussed earlier. I will not use the predictions data as ground truth in the following chapters.

Latent Model on Worker Quality

The annotation process of this chapter rewards bonus to workers who have contributed good efforts and good quality work. This sub-section discusses related work about computing worker's quality based on evaluating their ratings – in my case, the predictions – against others'.

The seminal paper by Dawid and Skene (1979) proposed the Expectation-Maximization algorithm to estimate the competence of experts as a latent variable and then compute the latent “true state” of items from those experts' annotations discounted with their competence. This work has many derivative works. For example, Smyth et al (1995) applied the latent model approach to infer ground truth of Venus images labels. Ipeirotis et al (2010) extended the approach by correcting systematic biases and applied it to clean labels from Amazon Mechanical Turk workers. Raykar et al (2009) extended the approach by learning not only the accuracy of the workers, but also the best logit classifier at the same time. Welinder et al (2010) assumed that coders had different competence on multiple dimensions of items, and then proposed the “multidimensional” technique to estimate the quality of workers on each dimension and used that to infer the true label. Welinder and Perona (2010) proposed an "online" algorithm that can compute the latent variables right away rather than computing them only after a complete set of labels is found.

There are a few other examples along the same line of inferring the latent variable of coders' quality. For example, Volkmer et al (2007) used the “latent class modeling” technique to estimate a true label from multiple noisy labels for image labeling. Snow et al (2008) used confusion matrices and Bayesian methods to estimate the true labels. Carpenter (2008) used Multilevel Bayesian models to estimate the true label of items, competence of coders, and difficulty level of items at the same time. Whitehill et al (2009) used the EM method to achieve the same goals as in Carpenter (2008).

In this chapter, I will directly apply the Get-Another-Label approach proposed by Ipeirotis et al (2010) to compute the quality of workers as the basis to offer bonus.

3.2 Data Collection

Data collection is done through the Amazon Mechanical Turk (AMT) platform. I applied the best practices suggested from prior literature (see “related work” section) to the data collection process. The dataset to be labeled consists 1911 articles, where each article is to be labeled by 20 AMT workers. This section mainly discusses the article selection, experiment setup and quality control.

The dataset to be labeled consists of two parts crawled from the “Politics” section on Digg.com. Digg.com is a news aggregator listing popular stories “dugg” by its users. The first part of the dataset consists of the same 1000 political articles from (Zhou, Resnick & Mei, 2011), which were randomly selected from 2009-5-25 to 2010-8-11 on Digg.com from those that received at least 10 diggs. More details about this dataset are discussed in chapter 5. The reason to reuse the same 1000 articles is to compare the first round of annotation process in year 2010 to the new annotation process in year 2012.

The second part of the dataset consists of 1500 randomly selected articles crawled from Digg.com between 2011-05-30 and 2011-11-30. More details about this dataset will be discussed in chapter 5. Each of the articles received 4 labels from AMT in year 2011. Digg.com is dominantly liberal. In order to prevent a dominant blue dataset, where coders on AMT might simply label articles as blue by default, I have stratified the samples to have 500 red, 500 blue, and 500 gray articles according to the annotations in year 2011, where the labels of articles are mapped from the distributions (based on the four labels per article) using the quadratic cost function.

Then I used Diffbot.com⁵ to extract pure textual content from the articles, stripping out ads, comments, authors, sidebars, urls, and other information not relevant to the articles. This also prevented workers from labeling articles simply by looking at their sources. I removed 589 articles that had zero length (which were mainly images and videos) or broken links, resulting in 1911 valid articles, including 746 from the first part

⁵ <http://diffbot.com>

and 1165 from the second part. These 1911 articles are the dataset that was posted to AMT, denoted as *I*.

As introduced in the “related work” section, I asked workers to answer both the “prediction” question and the “personal signal” question. A worker first read the instructions, the bonus information, and then read the political article with only the text content. Then she first answered the “prediction” question, which asked her to “Predict the majority of worker’s opinion on the article”, and then she answered the “personal signal” question, which asked her to “Tell us your own opinion on the article”. The “prediction” question was used for quality control in this particular study, and is not required for other subjective annotation elicitation process. Figure 13 shows the screenshots of a typical article posted on AMT.

The articles were posted on 2012-08-31, and all labeling was completed by 2012-09-10. Each label earned \$0.05. I did not upload all of the articles at once. Instead, I divided the articles into several 100-articles batches, and gradually posted the articles over several days. This was to prevent the same coders labeling all the items, and also to update HITs regularly to keep it at the top of the list. Note that 128 of the 1911 articles were labeled between 2012-08-25 and 2012-08-29 as a test with \$0.02 per label and slightly different bonus structure and task description. These articles were not treated differently from the other labeled articles for the rest of the thesis.

The figure consists of two screenshots of the Amazon Mechanical Turk (AMT) interface. The top screenshot shows the HIT details page for a task titled "Classify Political Articles (easy, quick, & bonus!)". The requester is "Openturk.org" and the qualifications required are "HIT approval rate (%) is greater than 90, Location is US, Understanding of USA Politics is greater than 80". The reward is \$0.05 per HIT, and there are 500 HITs available with a 10-minute duration. The page includes a search bar, a timer (00:00:00 of 10 minutes), and buttons to "Accept HIT" and "Skip HIT". The main content area is divided into "Instructions" and "About Bonus" sections. The "Instructions" section explains the task: to label the political leaning of an article into Red (conservative, republican, right-wing), Blue (liberal, democratic, left-wing), or Gray (not sure, independent, other). The first question asks to predict the majority of worker's opinion, and the second asks for the worker's own opinion. The "About Bonus" section details a merit-based bonus of \$2 to the worker with the highest quality and an extra bonus of \$2 to a random qualified worker. The article text is titled "More On Congress Taxing Cadillac Health Plans" and is dated Friday, January 8, 2010 - 15:36. The bottom screenshot shows the same HIT with a survey form. The survey has two main questions: "Predict the majority of worker's opinion on the article: *" and "Tell us your own opinion on the article: *". Both questions have three radio button options: Red (conservative, right leaning), Blue (liberal, left leaning), and Gray (not sure, independent, other). Below the survey is a "Feedback (Optional)" section with a text area and a "Submit" button. The bottom of the page includes a "Report this HIT: violates the Amazon Mechanical Turk policies or broken" link and a "Why?" link.

Figure 13. Screenshots of AMT task

The annotation process mainly adopts two approaches for quality control. Other quality control approaches such as the ones discussed in the related work section were not used due to resource constraints.

The first approach was a qualification test (e.g., Kittur et al 2008). Workers had to satisfy three conditions in order to participate in the study as a coder. First, they had to be US residents: non-US residents are generally not likely to have a good understanding of US politics to qualify as competent coders. Second, they had to have a previous record of 90% approval rate, which is a general indicator of a worker's good quality. Also, anecdotal evidence⁶ shows that workers with 90% approval rate care about their reputation and tend to avoid sloppy work. Finally, they had to pass a simple qualification test that consists of 4 questions (see Figure 14). They had to be correct on all questions in order to qualify. If they failed, they had to wait at least two hours before they could re-take the test.

Understanding of US Politics

Who is the current president of the USA:

- George W. Bush
- George Washington
- Barack Obama
- Bill Clinton

What viewpoint would a liberal most likely approve:

- small government
- pro-life (abortion)
- withdrawal from Iraq war
- no national health care

What viewpoint would a conservative most likely approve:

- gay marriage
- gun control
- free market
- affirmative action

Who was the Republican candidate in the 2008 Presidential Election:

- Barack Obama
- John McCain
- Sarah Palin
- Hillary Clinton

Figure 14. AMT qualification test

The second quality control approach was to incentivize workers with a monetary bonus to prevent moral hazard. Before the workers started labeling, they read the instructions as follows:

⁶ Observations from Turk Nation discussion forum.

Your job is to label the political leaning of the following article into Red (conservative, republican, right-wing), Blue (liberal, democratic, left-wing), or Gray (not sure, independent, other). The first question asks you to predict the majority of worker's opinion on the article. The second question asks for your own opinion on the article. The 2 answers don't need to match. We will assess the quality of your work by comparing your answer of the first question to other workers' answers of the first question on the same article, using the algorithm described in Quality management on Amazon Mechanical Turk. This mechanism rewards you when you more accurately predict the most frequently given classification for the given article. Your best strategy is to try to guess the first answer correctly, and honestly report your second answer. See sidebar for bonus information.

The quality of workers was computed by comparing the workers' answers to the "prediction" question against each other using the Get-Another-Label algorithm proposed by Ipeirotis et al (2010), where the idea is that a worker will have a high quality score if his or her answers consistently agree with the majority of workers answers. I also have a sidebar block that explains the bonus information:

Merit-based bonus: \$2 to the worker with the highest quality. Another \$2 to a random qualified worker. You need to label 50+ articles to qualify. Bonus will be granted next Tuesday (Sept.4).

Extra bonus: Grant to workers who label lots of articles with good quality. The exact bonus amount and number of recipients depend on the results we get. Total budget is \$50.

By offering a bonus, I have created incentives for workers to spend more effort trying to answer the "prediction" question correctly (see Krupka and Weber 2012). Granted that workers do spend some effort reading the article to make good predictions for the bonuses, there is no reason for the workers not to report truthfully for the "personal signal" question about their opinions, which incurs no extra cost. Therefore, I assume the elicited personal assessments are truthful, based on which the ground truth distributions of articles' political leaning are defined.

3.3 Results

A total of 165 workers participated in the annotation process. Among them, 73 labeled more than 50 articles and they contributed most of the labels. Figure 15 shows the number of articles each worker labeled, and it roughly follows a power law curve. The workers spent an average of 68 seconds on each label, with a standard deviation of 87 seconds. The median was 34 seconds, and 90% took more than 12 seconds.

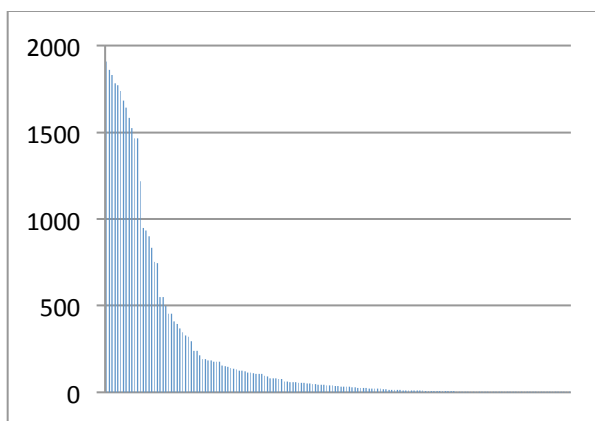


Figure 15. Number of articles labeled by workers

Figure 16 shows the 1911 labeled articles plotted on the simplex according to their distributions, $\hat{H}_i, \forall i \in I$. Define “disagreement labels” for an article as those labels that disagree with the majority label for the article. Figure 17 shows the number of articles (y-axis) according to the number of disagreement labels (x-axis): only 74 (3.9%) articles got 20 out of 20 unanimous agreement on articles’ political leaning; 139 (7.3%) and 207 (10.8%) articles got 1 and 2 disagreement labels respectively. More than half of the items (50.6%) got more than 5 disagreement labels. Fleiss’ Kappa⁷ (Gwet, 2010) inter-rater agreement is a low 0.393. In short, the empirical dataset shows that the political leaning classification problem is indeed subjective: there are some high agreement items and also quite a few low-agreement items too, which reflect the subjective disagreement among coders instead of errors. Therefore, we should use distributions instead of labels as the underlying ground truth in order to keep the disagreement information.

⁷ Cohen’s Kappa was not used here because each item was labeled by different coders.

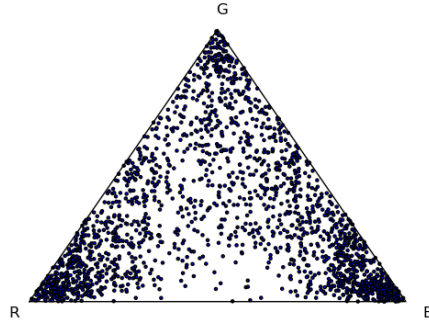


Figure 16. Articles plotted on the simplex according to their distributions

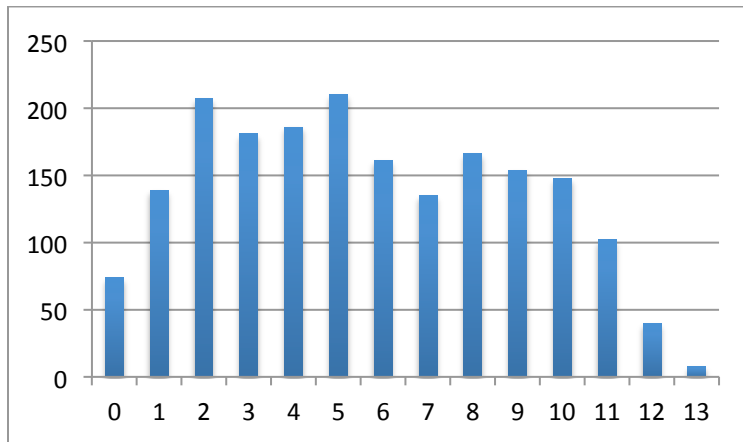


Figure 17. Number of items (y-axis) according the number of disagreement labels (x-axis)

The existence of many low agreement items in the center of the simplex raises two problems. The first problem is how to map the distributions of those items into single labels. Figure 18 illustrates the difference between the majority vote partitioning and quadratic partitioning. 19.4% of the articles are covered in the partitioning disputed region, which means they will have different cost-minimizing labels if we simply use majority vote instead of the quadratic cost function to map distributions (or multiple labels) into single labels.

There are also 15 articles, each of which got 8+ red labels as well as 8+ blue labels, but less than 4 gray labels, yet they are still in the gray partition according to the quadratic partitioning. For example, one article – “The Muslim Brotherhood Is Infiltrating The Boy

Scouts!”⁸ – received 9 red and 11 blue labels. Another article – “Obama moves to embrace OWS”⁹ – received 10 red, 9 blue, and 1 gray labels. These examples demonstrate the existence of articles where people have opposing opinions. More work is needed to understand why people have opposing opinions on those articles.

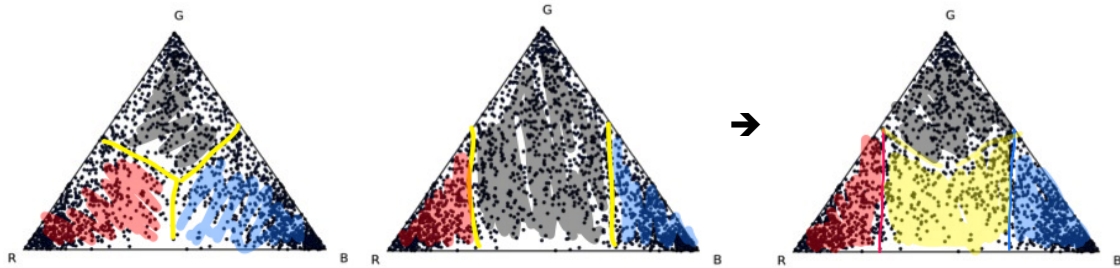


Figure 18. Partitioning difference between majority vote and quadratic

If we only have 4 labels per item instead of 20 labels per item, there are only 15 possible combinations of red, gray and blue, and \hat{H}_i are concentrated to 15 points on the simplex as shown in Figure 19(a). However, the partitioning disputed region (highlighted in yellow in Figure 19) still covers a few cases. When we only have one label per item, however, there are only three possible cases: 1 red, 1 gray, or 1 blue, as shown in Figure 19(b). The disputed region does not cover any case, which means the cost-minimizing labels would be the same regardless of which partitioning is used.

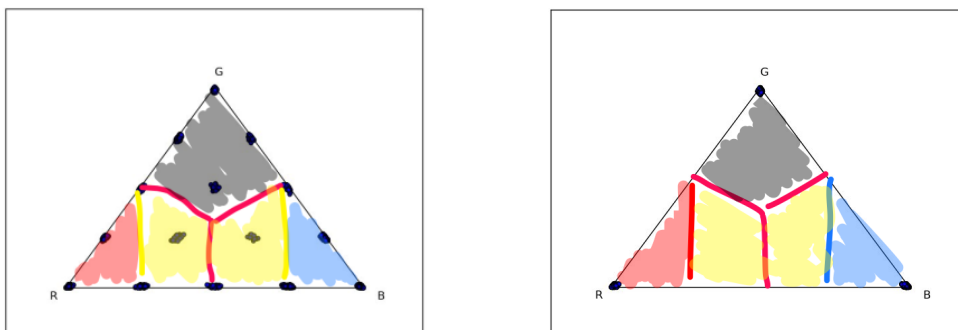


Figure 19. Illustration of partitioning when the number of labels per item is small: (a) $m = 4$, (b) $m = 1$

⁸ <http://redbluegray.com/node/72959>.

⁹ <http://redbluegray.com/node/73027>

The second problem when we have many low agreement items is that the labeled dataset is not reliable when we only have a small number of labels per item. For example, one article in the dataset has a distribution $\langle 0.5, 0.3, 0.2 \rangle$: if we only use one rater to label it in one study and label it again in another study, it is quite likely for the two labels to be different. And there are many items like that in a typical subjective classification problem.

I will discuss in chapter 5 that 733 of the 1911 labeled articles in this chapter were also labeled in 2010 with 6 labels per items. 24.8% of the 733 items received different cost-minimizing labels (with quadratic partitioning) between the first round of annotation in 2010 and the second round of annotation discussed in this chapter. Using computer simulation (which will be discussed in chapter 4), I have simulated having only one rater label an item. On average, 30.1% of the items, when labeled again for a second time, would get labeled differently.

Machine learning researchers and practitioners cannot simply ignore this large number of unreliably labeled items, pretending they are reliable to train and evaluate classifiers. One typical response is to make the dataset more reliable by revising the codebook, training the coders better, and turning the subjective classification problem into objective, like what I have discussed in chapter 2. However, I have also argued against this approach in chapter 2 because it is too costly and decreases classifiers' generalizability. Another response is to accept that it is inevitable to have many unreliably labeled items in the dataset. And the next question is to ask whether we can still use the unreliable dataset to draw reliable conclusions about classifier evaluation, which is the main topic of the next chapter.

3.4 Conclusions and Future Work

This chapter elicited political leaning annotations for a real dataset, and demonstrated that there were many articles raters could not agree on their political leaning. Therefore, ground truth should be modeled as distributions instead of labels. When there are many

low agreement items, we should be careful about using the right cost function or partitioning to map distributions into labels, which will affect many items in the partitioning disputed region, and we should be cautious about the many unreliably labeled items. This chapter also serves as an example to show how to practically elicit annotations for a typical subjective classification problem.

Eliciting people’s predictions instead of their personal assessments could be a promising direction for future research. The intuition is that people might have more information about an article’s political leaning than their own personal assessments, and therefore asking them to directly predict the article’s distribution or its cost-minimizing label might be more informative than eliciting their personal signals. In this chapter, I have also collected raters’ predictions on articles’ political leaning as a by-product of using the “prediction” question as a quality control approach. Next I will discuss some preliminary results about comparing predictions against personal signals.

Recall that $s_{ui} \in \mathcal{L} = \{red, gray, blue\}$ is a rater u ’s personal assessment on item i . Let $l_i \in \mathcal{L} = \{red, gray, blue\}$ be the 20 raters’ majority vote on item i ’s political leaning according to s_{ui} , $\forall u \in U_i$ (recall that U_i is defined as the subset of raters who have rated i). Let $t_{ui} \in \mathcal{L} = \{red, gray, blue\}$ be a rater u ’s prediction on item i ’s majority vote, l_i . As discussed earlier, I have asked raters to report both s_{ui} and t_{ui} , $\forall u \in U_i, \forall i \in I$, where the set of s_{ui} is used to define items’ ground truth distributions and the set of t_{ui} is used for quality control purpose. Define a “mismatch” as a pair of (s_{ui}, t_{ui}) where $s_{ui} \neq t_{ui}$. Of all the (s_{ui}, t_{ui}) pairs, 7.7% are mismatches. 74.5% of all raters and all those who have rated more than 50 articles have made at least one mismatch. 71.8% of all labeled items have received at least 1 mismatch. Define a “positive mismatch” as a mismatch where $t_{ui} = l_i$ and $s_{ui} \neq l_i$. Intuitively, a positive mismatch means that even though a rater u ’s personal opinion s_{ui} does not agree with l_i , she still possesses more information about i to make a correct prediction t_{ui} that matches l_i . Define a “negative mismatch” as a mismatch where $s_{ui} = l_i$ and $t_{ui} \neq l_i$, which means a rater u possesses incorrect information on i that misleads her to make

an incorrect prediction $t_{ui} \neq l_i$ even though her personal signal s_{ui} matches l_i . Of all the mismatches, 46.4% are positive mismatches and 39.9% are negative mismatches.

In short, these preliminary results show that raters do realize the difference between eliciting predictions and eliciting personal opinions, and would occasionally report differently given the chance to do so. Furthermore, when the predictions are different from personal opinions, the predictions are more likely to be positive than negative in that they are more likely to match the majority vote, implying that eliciting predictions is usually more informative than eliciting personal signals. A more thorough study of comparing the two approaches of annotation elicitation will be left to future work.

Chapter 4. Classifier Evaluation with Distributions

Using the distribution as ground truth model for political leaning classification (as a typical subjective classification problem) is unique in two ways. First, it proposes a particular way of defining *gray* items using the quadratic cost function: for example, a distribution such as $\langle 0.5, 0, 0.5 \rangle$ would map to *gray* even though no one personally assesses the item as *gray*. Second, it defines the ground truth of an item's political leaning as a distribution instead of a label, and thus using only a few human coders to annotate items will inevitably result in a dataset with sample errors. These unique features present two main problems: 1) how to train classifiers in the presence of distributions, and 2) how to evaluate the performance of those classifiers with distributions. This chapter discusses the second problem only, about classifier evaluation with distributions.

A few research works studied the first problem of classifier training with distributions (e.g., Rogers et al 2009). A full literature review will be discussed in the last chapter as future work. The major problem of that line of work is that researchers evaluated their classifiers based on the assumption that the labeled dataset for evaluation is reliable, which, as I have argued, is not true for subjective classification problems with only a few ratings per item. I'd like to argue that to do any classifier development work with distribution as ground truth, one should first consider whether the labeled dataset for evaluation is reliable, and if not, whether one can still draw reliable conclusions about classifier evaluation using an unreliable dataset. That is one main question this chapter is trying to answer.

A political leaning classifier in this thesis would classify items into red, gray and blue labels rather than distributions over the labels, although the ground truth of items is

distributions. One can directly evaluate the classifiers that output labels against distributions as ground truth; or, alternatively, one can map the distributions into labels first and then evaluate classifiers with the labels. Which approach is optimal? In addition, when preparing the ground truth dataset for evaluation, one can either have a small m (number of labels per item) but a large n (number of items), or a large m but a small n . What is the optimal m and n ? And above all, does classifier evaluation with distributions really make a difference to classifier evaluation with labels? That is another set of questions this chapter is trying to answer.

I'd like to call a particular way to evaluate classifiers as an “**evaluation scheme**”. The purpose of an evaluation scheme is to be able to correctly discriminate classifiers: the more accurate classifiers should be evaluated and ranked favorably compared to the less accurate classifiers. This chapter studies various evaluation schemes in terms of their ability to discriminate good classifiers from bad ones, not to evaluate various political leaning classifiers, which is the topic of chapter 5.

Using computer simulation, this chapter argues for four major findings about classifier evaluation with distributions. The first finding concerns the particular definition of *gray* items under the quadratic cost function: evaluating classifiers according to the conventional 0-1 cost function instead of the quadratic to define *gray* items will lead to incorrect classifier ranking – the classifier that mistakenly classifies “disputed” items into *red* or *blue* instead of *gray* would be discriminated as the better classifier, whose classifications are not cost-minimizing when showing to users because users’ cost model is quadratic. The other three findings are about how to properly evaluate classifiers using distributions as ground truth, not particularly depending on which cost function is used. The chapter’s second finding is that even though individual labeled items are not reliable due to sample errors, classifier ranking might still be reliable as long as we have many labeled items for evaluation. The third finding is that it is optimal for classifier evaluation to obtain one label per item as the estimate to an item’s distribution, but we need to label many items in order to correctly rank classifiers. The last finding is that we

might map distributions into labels first, and then use the labels to evaluate classifiers and still be able to rank classifiers correctly, even though the mapping throws away information. I will discuss more about these findings as the chapter develops.

The rest of the chapter is organized as follows. Section 4.1 discusses related work concerning algorithm evaluation. Section 4.2 proposes two evaluation schemes to evaluate classifiers under the distribution as ground truth model. Section 4.3 formulates the chapter's research question as assessing different "evaluation schemes" in terms of their ability to discriminate good classifiers from bad ones. Sections 4.4 and 4.5 describe the design of the computer simulation methodology and the simulation's results. Finally, section 4.6 summarizes this chapter and discusses future work.

4.1 Related Work

Relevance Judgments in Information Retrieval

Algorithm evaluation with relevance judgments in the information retrieval (IR) literature also studies the problem of unreliable ground truth data. A typical IR evaluation is based on the Cranfield framework, where IR researchers can evaluate their IR systems against a set of reusable "relevance judgments" on queries and documents collections, composed by experts for repeated usage (Voorhees 2002).

The key criticism of the Cranfield framework is that "relevance" is inherently subjective, and thus relevance judgments could vary for different assessors and even for the same assessor at different points (Voorhees 2002). Critics question how valid conclusions can be drawn when the process is based on something as volatile as relevance. One response is pragmatic: experience has shown that system improvements developed on test collections prove beneficial in other environments including operational settings (Voorhees 2000). Another response is a small set of studies that show that the comparative performance of two retrieval strategies is quite stable despite marked differences in the relevance assessments themselves (Lesk and Salton, 1969; Burgin, 1992; Cleverdon 1970). In particular, Voorhees (2002) argued that the question is not

whether relevance judgments can represent “ground truth”, but rather whether they can distinguish different IR algorithms. And Voorhees (2000) showed that after dividing a testing set into different subsets, system rankings of IR algorithms are still closely correlated, which, according to the paper, means that the testing set is reliable, despite the fact that “relevance judgments” are subjective and volatile.

Lesk and Salton (1969) gave three reasons for the stability of IR system rankings despite differences in relevance judgments. First, evaluation results are reported as averages over many topics, and thus random errors are averaged out. Second, disagreements among judges affect borderline documents, which in general are ranked after documents that are unanimously agreed upon, and thus do not affect much the top-k ranking performance of IR systems. Third, recall and precision depend on the relative position of the relevant and non-relevant documents in the relevance ranking, and changes in the composition of the judgment sets may have only a small effect on the ordering as a whole.

Another line in IR evaluation research is to study whether the difference between IR systems A and B (in terms of precision, recall and a few other measurements) is statistically significant or not, given that the relevance judgments are subjective and erroneous (Zobel 1998). Only when the difference is significant can the conclusions be made that one system is indeed better than the other, not because of a fluke. Some studies compared different statistical tests among Wilcoxon’s signed-rank test, t-test, ANOVA, Fisher’s randomization test, and sign test and concluded that Wilcoxon’s signed-rank test is optimal (Zobel 1998, Sanderson and Zobel 2005, Smucker et al 2007). This line of work is inspiring, and future research could apply the method of statistical tests to study classifier evaluation. However, due to time and resource constraints, this chapter only explored using another method to study classifier evaluation, which will be discussed in section 4.3.

Another line in IR evaluation research studies the optimal number of documents per query and the optimal number of queries in order to reduce the cost of relevance

judgments, while still maintaining reasonable discrimination power of system ranking (Carterette and Allan 2005, Sanderson and Zobel 2005, Carterette et al 2006, Carterette and Smucker 2007, Carterette et al 2008). All of these studies concluded that more queries are better than more documents for each query. This leads to another line of research on how to sample documents for queries in order to evaluate IR systems with incomplete relevance judgments (Aslam et al 2006). Carterette and Soboroff (2010) summarized eight assessor models and studied how to use different assessor models to improve the quality of relevance judgments. This chapter makes an analogous finding that one label per item with many items is the optimal setting to obtain labeled items for evaluation.

This chapter also draws many ideas from the IR evaluation literature and reaches similar conclusions, as can be seen later. However, this chapter still differs from the IR evaluation literature. Obviously, the problem settings of IR research and classification research are quite different. A typical IR system responds to multiple queries and returns the most relevant documents, while a classification system classifies many items into a few categories. Also, most IR systems are only evaluated against the top-K query results, while classification systems are evaluated against all items. Despite some similarities between the two types of research, it is not clear how to directly apply findings about relevance judgments from IR research to classification research.

Furthermore, the relevance judgments datasets in IR are designed to be reusable, such as the ones composed for TREC (Manning, Raghavan and Schutze 2008). The validity and reliability of these datasets are scrutinized with scientific rigor, and IR researchers can simply reuse these datasets for training and evaluation purposes without having to worry about their quality. However, many classification problems are different and reusable datasets are not available, so researchers and practitioners of classification algorithms should learn how to obtain labeled items properly for their own classification problems, rather than simply assuming the labeled dataset in use is reliable.

Algorithm Evaluation with No Ground Truth

There are a few examples of algorithm evaluation with no ground truth data. They are inspiring, but are not directly applicable to this chapter because there were no ground truth data involved. I will summarize the work here.

Grimmer and King (2011) proposed a clustering algorithm, and in order to show that it works better than other alternatives, the authors used three evaluation approaches. Their first approach was to have a few human coders manually rank a few randomly selected document pairs in clusters and then compute the cohesiveness of clusters as the measurement of clustering algorithm performance. Their second approach was to have multiple domain experts evaluate the clusters directly. The third approach was to show, practically, that using the clusters generated from the algorithm can really help with their qualitative research.

Another set of examples involves diagnostic test evaluation in the Medical and Biometrics literature. The research goal of this type of work (e.g., Hui and Walter 1980, Hui and Zhou 1998, Phelps and Hutson 1995, Albert and Dodd 2004) is to be able to evaluate diagnostic tests (tests that diagnose whether a disease is present in a patient or not) without knowing the ground truth (whether a patient indeed presents a disease or not). This type of work is quite similar to the line of work by Dawid and Skene (1995), which was discussed earlier. Moreover, even though ground truth is absent when conducting the diagnostic tests, in many cases it would be revealed eventually (through autopsy for example) for *ex post* study.

The last set of examples involves credit scoring for credit card companies. Kelly et al (1998, 1999, 1999a, 1999b) studied the problem of designing computer algorithms to classify credit card holders as good customers or bad customers. The challenge here was the absent ground truth in terms of whether a customer is good or bad, and therefore it is impossible to directly evaluate the performance of classification algorithms. Instead of direct evaluation, the researchers proposed to evaluate the algorithms based on whether an algorithm eventually leads to revenue increase of credit card companies.

4.2 Classifier Evaluation Schemes

Let C be a political leaning classifier. Let $w_i = C(i) \in \mathcal{L}$ be the classification result of C for i . The classification output on a dataset I is denoted as $C(I)$. To evaluate the performance of a classifier C , we need to compare its classification output $C(i)$ and the estimated ground truth \hat{H}_i for all $i \in I_{test}$. Denote an evaluation scheme as Ψ . As introduced in the beginning of the chapter, Ψ is a particular way to evaluate classifiers in order to be able to discriminate good classifiers from bad ones. This section introduces three evaluation schemes, Ψ_H , Ψ_l and Ψ_c , and I will discuss how to evaluate them in section 4.3.

4.2.1 Ψ_H : Direct Evaluation with Distributions

Ψ_H (where the subscript H stands for a distribution H_i), or colloquially “direct distribution evaluation scheme”, applies the concept of “cost” introduced in chapter 2 (see section 2.5, Equation 5 and Equation 6) and directly evaluates classifiers with distributions: it ranks the classifier that minimizes the total expected cost, \mathbb{L} , as the best classifier among a set of alternative classifiers.

I’d like to explain the intuition of Ψ_H with an example. Suppose we have an article i with $H_i = \langle 0.7, 0.2, 0.1 \rangle$. If a classifier C_a classifies i as red, then using the quadratic cost function to compute a penalty, $\mathbb{L}_i = .7 \times 0 + .2 \times 1 + .1 \times 4 = 1.3$. Recall that \mathbb{L}_i is the expected cost of showing w_i (the classification result of i) to the target population (see Equation 5). 70% viewers will think the item red, and see a label red, incurring no cost; 20% will think the item gray and see a red label at cost 1; 10% will think the item blue, in which case the red label will seem particularly strange, at a cost of 4. On the other hand, if another classifier C_b classifies i as blue, then $\mathbb{L}_i = 3.0$. That is, classifying i as red incurs less cost than as blue. If C_a consistently classifies items that incur less cost than C_b , Ψ_H would then evaluate C_a as a better classifier because its classification results incur less total expected cost.

Since we are only able to observe $\widehat{\mathbb{L}}$ instead of \mathbb{L} , Ψ_H evaluates classifiers based on $\widehat{\mathbb{L}}$ instead of \mathbb{L} . Therefore, the classifier that minimizes $\widehat{\mathbb{L}}$ does not necessarily mean that it can also minimize \mathbb{L} . Thus Ψ_H might incorrectly evaluate classifiers because of the unreliably labeled items. I will discuss this in more detail in the next few sections.

Note that even the best classifier, when evaluated with Ψ_H , would still have a non-zero \mathbb{L} , which indicates that no matter how we classify the articles some people would be dissatisfied when seeing the classification results. But the best classifier is the one that classifies the items in the way that minimizes the total cost. We cannot treat a classification as either fully correct or incorrect when computing the accuracy of a classifier. Thus, I'd like to introduce the normalized accuracy score to measure the performance of classifiers under Ψ_H .

Define the minimum and maximum possible cost of \mathbb{L} as \mathbb{L}^+ and \mathbb{L}^- respectively. Given a labeled dataset I , both \mathbb{L}^+ and \mathbb{L}^- are constants, and we can always compute them in post-hoc analysis using the evaluation dataset I . Then for any classifier C and its classification cost \mathbb{L} , define its normalized accuracy score in Equation 11. Here, $\widehat{\mathbb{L}}$, $\widehat{\mathbb{L}}^+$, $\widehat{\mathbb{L}}^-$ and \hat{A} are the corresponding sample cost and normalized accuracy.

$$A = \frac{\mathbb{L}^- - \mathbb{L}}{\mathbb{L}^- - \mathbb{L}^+}, \quad \text{and} \quad \hat{A} = \frac{\widehat{\mathbb{L}}^- - \widehat{\mathbb{L}}}{\widehat{\mathbb{L}}^- - \widehat{\mathbb{L}}^+}$$

Equation 11. Normalized accuracy under Ψ_H

Obviously, $\mathbb{L} \in [\mathbb{L}^+, \mathbb{L}^-]$, and thus $A \in [0,1]$, where $A = 1$ means that C is equivalent to the perfect classifier that minimizes \mathbb{L}_i for all $i \in I$, and $A = 0$ means that C is equivalent to the anti-minimizing classifier that maximizes \mathbb{L}_i for all $i \in I$. Since both \mathbb{L}^+ and \mathbb{L}^- are constants, A is a linear transformation of \mathbb{L} . Comparing classifiers according to A (or \hat{A}) would have the same ordering as comparing them according to \mathbb{L} (or $\widehat{\mathbb{L}}$).

Note that $\mathbb{L}^- - \mathbb{L}^+ = K$ is a constant given a fixed labeled dataset I . Then, $\mathbb{L}^- - \mathbb{L} = K - (\mathbb{L} - \mathbb{L}^+) = K - \lambda$, where the second term $\mathbb{L} - \mathbb{L}^+ = \lambda$ is the "regret" of the classifier, relative to the best possible. Therefore, Equation 11 is equivalent to Equation 12. The

intuitive explanation of Equation 12 is that $A = 1$ is achieved when regret is 0, and $A = 0$ is achieved when regret is the maximum. Therefore, the normalized accuracy scores A and \hat{A} are also defined according to the regret.

$$A = \frac{K - \lambda}{K}, \quad \text{where } K = \mathbb{L}^- - \mathbb{L}^+ \text{ and } \lambda = \mathbb{L} - \mathbb{L}^+$$

$$\hat{A} = \frac{\hat{K} - \hat{\lambda}}{\hat{K}}, \quad \text{where } \hat{K} = \hat{\mathbb{L}}^- - \hat{\mathbb{L}}^+ \text{ and } \hat{\lambda} = \hat{\mathbb{L}} - \hat{\mathbb{L}}^+$$

Equation 12. Normalized accuracy defined as regret

To sum up, the procedure that uses Ψ_H to evaluate a set of classifiers and rank them according to their performance is as follows:

- Step 1: For each of the classifiers C_1, C_2, \dots to be evaluated, compute their classification results $w_l = C(I)$ against the testing dataset I .
- Step 2: Compute their sample accuracy $\hat{A}_1, \hat{A}_2, \dots$ using Equation 11.
- Step 3: The classifier that has the highest sample accuracy \hat{A} would be ranked as the better classifier.

4.2.2 Ψ_l : Evaluation with Labels Mapped From Distributions

Ψ_l (where the subscript l stands for a label l_i), or colloquially “indirect distribution evaluation scheme”, first maps a distribution H_i into its cost-minimizing label l_i using the principled approach discussed in section 2.5, and then uses l_i as ground truth to evaluate classifiers. Indeed, Ψ_l assumes the distribution as ground truth model, even though it uses labels l_i for classifier evaluation.

One reason to map distributions into labels first is because it is more conventional to evaluate classifiers that output labels against labels as well. In addition, it is possible to use “accuracy”, “precision” and “recall” to measure the performance of classifiers and the interpretations of such measurements are more straightforward. On the other hand, however, mapping distributions into labels implies information loss. Whether it is justifiable to map distributions into labels for classifier evaluation will be one of the research questions of this chapter.

The intuition of Ψ_l is like this. If a classifier is able to classify an item i that matches l_i , it means the classifier has made a “correct” classification for i in terms of minimizing the cost \mathbb{L}_i . A classification that does not match l_i means that the classification is not cost-minimizing, and thus is “incorrect”. An incorrect label is penalized under Ψ_l . The penalty depends on the cost of $L(w_i, l_i)$ for that particular kind of miss. For example, under the quadratic cost function, the penalty is higher for a red label w_i when the best (cost-minimizing) label l_i would have been blue than the penalty when the best label would have been red. The difference from Ψ_H is that the amount of the penalty under Ψ_l depends only on w_i and l_i , whereas under Ψ_H the amount of the penalty depends on how different \mathbb{L}_i , the expected cost, is for that item. If, for some item i , red is the cost-minimizing label, but gray is a close second (e.g., $H_i = \langle 0.51, 0.49, 0 \rangle$), the penalty under Ψ_H for classifying i as gray would be very small. When gray is much worse (e.g., $H_i = \langle 0.99, 0.01, 0 \rangle$), the penalty would be higher. Under Ψ_l , the penalty for the red-gray mismatch is the same, regardless of whether the difference in expected cost between red and gray is large or small for that item.

Again, since we are only able to observe \hat{l}_i , and thus Ψ_l can only evaluate classifiers based on \hat{l}_i instead of l_i and might incorrectly rank classifiers. Similar to Ψ_H , Ψ_l uses the normalized accuracy score to measure and rank the performance of classifiers. The process of Ψ_l to evaluate classifiers is then described as follows.

- Step 1: For each $i \in I$, map \hat{H}_i into the cost-minimizing label \hat{l}_i .
- Step 2: For each of the classifiers in C_1, C_2, \dots , compute its classification results $w_I = C(I)$.
- Step 3: Compute the classifiers’ accuracy $\hat{A}_1, \hat{A}_2, \dots$ using the following rules:
 - If $w_i = \hat{l}_i$, add i to the “hit” set, denoted as \oplus ;
 - If $w_i \neq \hat{l}_i$, add i to the “miss” set, denoted as \ominus ;
 - Compute \hat{A} according to Equation 13.
- Step 4: Rank the classifiers according to $\hat{A}_1, \hat{A}_2, \dots$, and the classifier with higher accuracy is the better classifier.

$$\hat{A} = \frac{|\oplus|}{|\oplus| + \sum_{i \in \ominus} L(w_i, \hat{l}_i)}$$

Equation 13. Normalized accuracy under Ψ_l

When all items are classified correctly, $\hat{A} = 1$; when all items are classified incorrectly, $\hat{A} = 0$. Thus, $\hat{A} \in [0,1]$. Compared to the normalized accuracy score for Ψ_H (Equation 11 and Equation 12), here Equation 13 simply counts the number of correct and incorrect classifications (weighted by L) without having to compute classification “regret” in post-hoc analysis.

In this chapter both Ψ_H and Ψ_l use the quadratic cost function for the political leaning classification problem. However, for other problems, both Ψ_H and Ψ_l are not limited to the quadratic cost function, but can use other cost functions as well. The setup of Ψ_H and Ψ_l allows the use of any cost functions. When Ψ_l is used with a 0-1 cost function, rather than quadratic, it becomes equivalent to the evaluation scheme used under label as ground truth as the next section shows.

4.2.3 Ψ_c : Evaluation under Label as Ground Truth

Finally, I’d like to formally introduce the evaluation scheme corresponding to that typically used under the label as ground truth model for objective classification problems, denoted as Ψ_c (where the subscript c stands for “classical” or “consensus”), or colloquially “label evaluation scheme”. It is widely used in the current practice of classifier evaluation.

The process of Ψ_c is the same as the process of Ψ_l using the 0-1 cost function instead of the quadratic in two places: first in mapping the distribution of labels to a single best label \hat{l}_i , and second when assessing penalty for mistakes between classifier output and \hat{l}_i . Note that under the 0-1 cost function, the label selected by the majority of labelers is the cost-minimizing label. Thus, \hat{l}_i corresponds to a majority vote process. Also note that since all misclassifications are penalized equally under the 0-1 cost function, the performance of a classifier is measured by counting of the number of mismatches between the classifier and the majority of labelers, which is the “accuracy” of the classifier as shown in Equation 14. For this reason, accuracy in Ψ_c is usually represented as a percentage instead of a normalized score in the range of 0 and 1 as is in Ψ_H and Ψ_l .

$$\hat{A} = \frac{|\oplus|}{|\oplus| + \sum_{i \in \ominus} L(w_i, \hat{l}_i)} = \frac{|\oplus|}{|\oplus| + |\ominus|} = \frac{|\oplus|}{|I|}$$

Equation 14. Accuracy under Ψ_c

All in all, the differences among Ψ_H , Ψ_l and Ψ_c are summarized in Table 7.

Table 7. Differences among evaluation schemes

	Ψ_H	Ψ_l	Ψ_c
Underlying ground truth model	Distribution as ground truth		Label as ground truth
Cost function (or partitioning)	Quadratic (or any other cost function per different applications)		0-1
Classification output evaluated against	Distributions	Labels (mapped from distributions using the quadratic cost function with Ψ_l , or by majority vote with Ψ_c)	
Classifiers ranked according to	Classification regret	Count of correct and incorrect classifications	
Performance of classifiers measured by	Normalized accuracy (Equation 11)	Normalized accuracy (Equation 13)	Accuracy

4.3 Problem Formulation

This chapter has four main research questions as follows, where the first one studies Ψ_H and Ψ_l versus Ψ_c (or, distribution as ground truth versus label as ground truth) and the other three study properties of Ψ_H and Ψ_l (or, how to do classifier evaluation properly with distribution as ground truth).

- Is classifier evaluation with distributions different from classifier evaluation with labels? That is, what are the consequences of using Ψ_c when we should use Ψ_H or Ψ_l because distributions are actually the ground truth?
- Are Ψ_H and Ψ_l able to correctly evaluate classifiers even when the ground truth dataset is unreliable due to sample errors with a small m (the number of labels per item)?
- What is the optimal allocation of k ratings for Ψ_H and Ψ_l ; that is, what is the optimal number of labels per item (m) and number of items (n) given a budget constraint for the total number of labels $k = m \times n$?
- Should we use Ψ_H or Ψ_l for classifier evaluation under distribution as ground truth?

I will use computer simulations to answer these research questions. In this section, I will formulate the research problems and introduce the framework for the computer simulations. Note that one should not confuse simulated items and simulated classifiers with real items and real classifiers in the real world. A simulated item does not have any text features but only ground truth distribution H_i and samples from it, \hat{H}_i ; a simulated classifier does not use SVM or Naïve Bayesian algorithms but produces labels that are a function of the ground truth H_i , together with stochastic sources or “error”.

4.3.1 Ranking of Classifiers

Let $\mathbb{C} = \{C_1, C_2, \dots\}$ be a set of political leaning classifiers. Recall that $C(i)$ is the classification result for item $i \in I$, and $C(i) \in \mathcal{L}$. $C(I)$ is C 's classification results for all items in I . The observed sample ground truth for $i \in I$ is \hat{H}_i , and $\hat{H}_I = \{\dots, \hat{H}_i, \dots\}$ is the annotated ground truth dataset $\forall i \in I$, and $\hat{l}_I = \{\dots, \hat{l}_i, \dots\}$ is the set of cost-minimizing labels mapped from \hat{H}_I according to the quadratic partitioning.

Let Ψ be an evaluation scheme, which defines how to evaluate the performance of a classifier against a ground truth dataset. Let $\bar{\Psi} = \{\Psi_H, \Psi_l, \Psi_c\}$ be a set of evaluation schemes. For any $\Psi \in \bar{\Psi}$, define Ψ as a function:

$$\hat{A} = \Psi(C, \hat{H}_I)$$

Equation 15. Ψ as a function

The input of Ψ is a classifier C and the sample ground truth \hat{H}_I on dataset I , and the output is the sample accuracy \hat{A} of the classifier C evaluated against the sample ground truth \hat{H}_I under the evaluation scheme Ψ . Here, \hat{A} is called “sample accuracy” or “observed accuracy” because it is computed against the sample ground truth \hat{H}_I . Let $A = \Psi(C, H_I)$ be the “population accuracy” or “true accuracy” of classifier C evaluated against the population ground truth H_I under Ψ . Note that \hat{A} might be different from A due to the sample errors introduced from observing H_I as \hat{H}_I .

When the classifiers in \mathbb{C} are ranked according to their population accuracy A in the descending order, the resultant ranked classifier sequence is called “population ranking”, or χ . Both A and χ are not observable in the real world, but can be simulated in computer simulations. The “sample ranking” of classifiers, or $\hat{\chi}$, is the ranking of all $C \in \mathbb{C}$ according to their sample accuracy \hat{A} in the descending order. Note that both χ and $\hat{\chi}$ are determined by the classifiers \mathbb{C} , the labeled dataset H_I or \hat{H}_I , and the evaluation scheme Ψ . Different \mathbb{C} , H_I (or \hat{H}_I), and Ψ might lead to different χ (or $\hat{\chi}$).

Natural Ranking

I’d like to denote χ^* as the “natural ranking” of classifiers in \mathbb{C} . Suppose we have a natural ranking of classifiers $\chi^* = \{\dots, C_j, \dots, C_k, \dots\}_{ranked}$ where $C_j, C_k \in \mathbb{C}$. Then it means for some reasons to be discussed later, C_j is “naturally” better than C_k . Note that χ^* is defined only on \mathbb{C} , without reference to any particular H_I (or \hat{H}_I) or Ψ .

The primary approach to obtain the natural ranking of classifiers is called “degradation to random”. Suppose we have a perfect classifier that is able to observe the full information of items and then make correct classifications for all items, and we have a random classifier that makes random classifications, which would be the worst classifier¹⁰. Then between the two extremes of the perfect classifier and the random classifier, we can introduce a few classifiers with monotonically decreasing quality and increasing errors and thus construct the “degradation to random” natural ranking. I will discuss the “degradation to random” simulated classifiers in section 4.3.3.

Another approach to obtain the natural ranking of classifiers, although not used extensively in this chapter, is by arbitrary choice. In different application context, one might prefer one type of classifier to another, and thus is able to arbitrarily set the natural ranking of classifiers for the particular context.

¹⁰ A random classifier is the worst classifier because it has absolutely no information to classify items. The “anti-perfect” classifier that always incorrectly classifies items has zero accuracy, but it still has information about items, although systematically biased, and can be easily fixed to be a good classifier. See more discussions in Ipeirotis, Provost, and Wang (2010).

4.3.2 Evaluating Evaluation Schemes

The main research challenge of this chapter is to evaluate an evaluation scheme $\Psi \in \bar{\Psi}$ in terms of whether it is able to correctly discriminate good classifiers from bad ones in spite of non-negligible errors in the ground truth dataset. The key idea is to compare the sample ranking of classifiers $\hat{\chi}$ under Ψ against the classifiers' natural ranking χ^* , and $\hat{\chi}$ and χ^* should match in a good evaluation scheme. I'll formulate the problem as follows.

Let χ_1 and χ_2 be any two arbitrary ranking sequences of \mathbb{C} . The difference between χ_1 and χ_2 is measured by Kendall's τ rank correlation coefficient. $\tau(\chi_1, \chi_2) \in [-1, 1]$, where $\tau = 1$ means perfect correlation between χ_1 and χ_2 and $\tau = -1$ means perfect negative correlation between χ_1 and χ_2 . The objective is then formulated as finding and optimizing $\Psi \in \bar{\Psi}$ that maximizes $\tau(\chi^*, \hat{\chi})$ in terms of Equation 16.

$$\operatorname{argmax}_{\Psi} \tau(\chi^*, \hat{\chi}), \quad \Psi \in \bar{\Psi}$$

Equation 16. Evaluating Ψ

Intuitively, τ measures the performance of an evaluation scheme Ψ in terms of its power to correctly discriminate good classifiers from bad ones by comparing the classifiers' sample ranking to their true natural ranking. Therefore, I will also call τ the **"discrimination power"** of Ψ , or simply the power of Ψ . A good evaluation scheme Ψ should have high power, which means classifiers' sample ranking matches their true natural ranking quite well.

Sometimes I'll break down Equation 16 into two separate steps in Equation 17. The first step compares the classifiers' natural ranking to their population ranking, and the second step compares the classifiers' population ranking to their sample ranking. This breakdown makes the effects of sample errors more salient.

$$\operatorname{argmax}_{\Psi} \tau(\chi^*, \chi), \quad \Psi \in \bar{\Psi}$$

$$\operatorname{argmax}_{\Psi} \tau(\chi, \hat{\chi}), \quad \Psi \in \bar{\Psi}$$

Equation 17. Two steps evaluation of Ψ

In addition, note that both χ and $\hat{\chi}$ are dependent on the particular choice of \mathbb{C} , H_I (and \hat{H}_I), and Ψ , as I have discussed earlier. Therefore, in computer simulations, I will use different settings of \mathbb{C} , H_I (and \hat{H}_I), and Ψ in order to check for robustness. Furthermore, items in a simulated dataset I are randomly generated according to some distributions and will add variance to τ especially when $n = |I|$ is small. Therefore, sometimes I'll run multiple rounds of simulations and take the average of τ , denoted as $\bar{\tau}$, as the discrimination power of Ψ .

Then, what is the acceptable value of $\bar{\tau}$ for Ψ ? Figure 20 lists a few examples: the differences between two ordered sequences A and B are highlighted and τ is computed. Overall, in computer simulations, I'd like to consider $\bar{\tau} \geq 0.9$ as acceptable, although in practice we might consider a much lower τ as acceptable too.

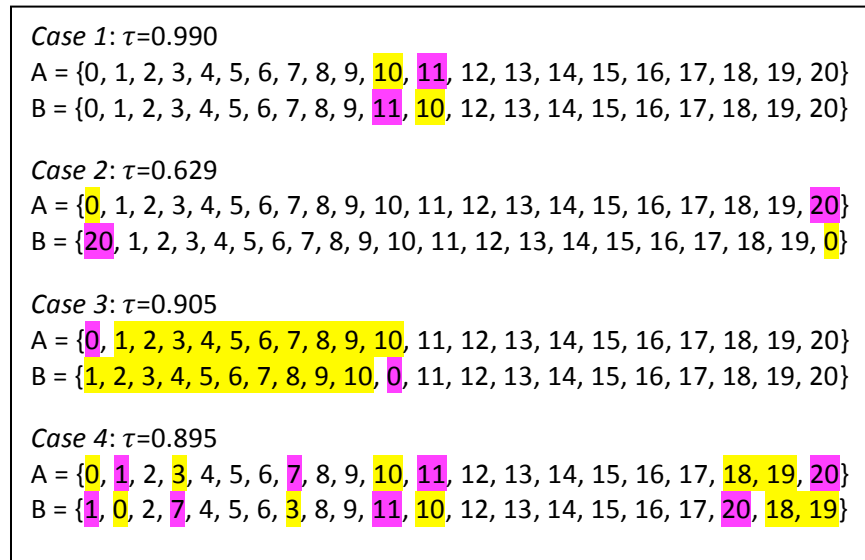


Figure 20. Intuition about the values of τ .

4.3.3 Simulation Design

Simulated Item, i

A simulated item represents a political article to be classified. For each item $i \in I$ in the simulation, I will randomly generate its population ground truth $H_i = \langle R_i, G_i, B_i \rangle$ from a Dirichlet distribution (which will be discussed later). Recall that the distribution H_i is

interpreted as R_i , G_i and B_i of the population consider i as red, gray, and blue respectively.

Also, each item $i \in I$ is simulated to receive m labels from human coders, randomly drawn i.i.d. with replacement from the multinomial distribution $H_i = \langle R_i, G_i, B_i \rangle$. m remains the same for all items i in I , but varies across different simulation settings. Denote m_i^r, m_i^g and m_i^b as the number of red, gray, and blue labels randomly drawn for item i , $m_i^r + m_i^g + m_i^b = m$. The sample ground truth is then $\hat{H}_i = \langle \hat{R}_i, \hat{G}_i, \hat{B}_i \rangle = \langle \frac{m_i^r}{m}, \frac{m_i^g}{m}, \frac{m_i^b}{m} \rangle, \forall i \in I$. Note that each draw of a sample of m coders might result in different m_i^r, m_i^g, m_i^b and \hat{H}_i for the same item i . Therefore, it is necessary to repeat simulations multiple times to get a distribution of values for \hat{H}_i .

I will use $m^* = 10^6$ as the maximum possible value of m . By definition, $\lim_{m \rightarrow \infty} \hat{H}_i = H_i$. I will consider $H_i = \hat{H}_i$ when using $m = m^*$ as the approximation of $m \rightarrow \infty$. I will usually use $m \ll m^*$ to get \hat{H}_i .

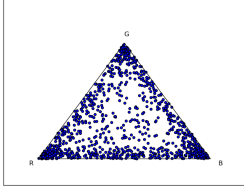
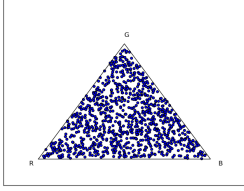
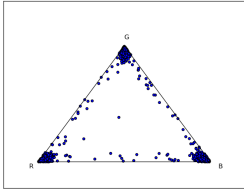
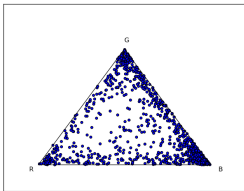
Simulated Datasets, I

A simulated dataset consists of $n = |I|$ simulated items, where each item gets the same number, but not same labels. The total number of labels to simulate a dataset is then $k = m \times n$. n is a variable under different simulation settings. For convenience purposes, I'll denote $n^* = 10^6$ as the maximum value of n instead of ∞ .

Since H_i follows a multinomial distribution, it is natural to use the Dirichlet distribution to generate instances of H_i to form a dataset I . Here the multinomial distribution H_i has three random variables (R_i, G_i , and B_i) with two degrees of freedom, and thus the Dirichlet distribution has three parameters, denoted as $\alpha = \langle \alpha^r, \alpha^g, \alpha^b \rangle$, each of which controls the "concentration" of R_i, G_i , and B_i . Using different settings of α , I will construct 4 types of simulated datasets to be used in the simulation, as shown in Table 8.

Table 8. Simulated datasets

Notation	α	Description	Simplex plot
----------	----------	-------------	--------------

I_{main}	$\langle 0.4, 0.3, 0.4 \rangle$	This dataset closely matches the real dataset I got in chapter 3, with half items as gray, and half as red and blue under quadratic partitioning. It is a typical dataset for a subjective classification problem.	
I_{gray}	$\langle 1, 1, 1 \rangle$	Items in this dataset are uniformly distributed across the simplex, which results in many gray items under quadratic partitioning.	
I_{clean}	$\langle 0.02, 0.02, 0.02 \rangle$	Almost all items in this dataset are concentrated in the three corners of the simplex, meaning that human coders have high agreement on how to label items.	
I_{biased}	$\langle 0.2, 0.3, 0.5 \rangle$	Items in this dataset are biased in favor of blue and gray, while red items are quite rare. This maps relatively closely to the real Digg political dataset, as I will discuss in chapter 5.	

Throughout the simulation, I will mainly use I_{main} to get the main results because it closely matches the real dataset I obtained in chapter 3, and then I will use the other three datasets for robustness check. I would consider the results learned from these four simulated datasets generalizable to many real world datasets, although each real world dataset has its own unique distributions.

I_{clean} is different from the other three datasets and deserves extra explanation. In the real world, a subjective classification problem where people don't agree on many items is not likely to result in a dataset like I_{clean} : I_{clean} is more likely a model of an objective classification problem. For the political leaning classification problem, we might only see a dataset like I_{clean} when there is a comprehensive codebook and coders are well trained to classify items according to the codebook, which means the classification

problem is made objective. Nevertheless, I can still use Ψ_H and Ψ_l to evaluate classifiers on such a dataset, as I have discussed in section 4.2.3.

Simulated Classifiers, \mathbb{C}

I'll introduce two sets of simulated classifiers or "classifier families", \mathbb{C}^{main} and \mathbb{C}^{coder} , as shown in Figure 21.

$$\mathbb{C}^{main} = \{C_1^{main}, C_{0.95}^{main}, C_{0.9}^{main}, \dots, C_\beta^{main}, \dots, C_{0.1}^{main}, C_{0.05}^{main}, C_0^{main}\}$$

$$\mathbb{C}^{coder} = \{C_1^{coder}, C_{0.95}^{coder}, C_{0.9}^{coder}, \dots, C_\beta^{coder}, \dots, C_{0.1}^{coder}, C_{0.05}^{coder}, C_0^{coder}\}$$

Figure 21. Two simulated classifier families

Each classifier $C_\beta^{main} \in \mathbb{C}^{main}$ is able to classify i as the cost-minimizing label l_i according to the quadratic partitioning as if it observed H_i , but with $1 - \beta$ random error producing any label (with equal probability), $\forall i \in I$. For example, C_1^{main} is able to classify any item as l_i without making any error, but $C_{0.95}^{main}$ has 5% chance to make random errors, and C_0^{main} always makes random errors. Therefore, the sequence of the 21 classifiers $C_\beta^{main} \in \mathbb{C}^{main}$ follows the "degradation to random" natural ranking.

I'd like to argue that the \mathbb{C}^{main} family is a reasonably good model of an idealized classifier. The performance of a real classifier is usually measured by its accuracy, where high accuracy indicates that the classifier is able to classify items correctly with small errors, and low accuracy indicates that the classifier makes many errors. The degradation of β in the \mathbb{C}^{main} family basically maps to the degradation of accuracy in a set of real classifiers¹¹.

Define the "step size" of a classifier family such as \mathbb{C}^{main} as the difference between two adjacent classifiers. Since $\beta \in \{1, 0.95, 0.9, \dots, 0.05, 0\}$, the step size of \mathbb{C}^{main} is then 5%.

¹¹ One might argue that the extreme cases such as C_1^{main} and C_0^{main} are hardly seen in the real world, and thus should be excluded. However, the challenge is to define what cases are considered "more common". Further considerations along this line would be left to future work.

If $\beta \in \{1, 0.99, 0.98, \dots, 0.01, 0\}$, then the step size would be 1%. Obviously, a 1% step size means that the classifiers are less distinguishable than that of 5%, and it would be much harder for an evaluation scheme to correctly rank them. That is, the step size of a classifier family also affects the discrimination power of Ψ . In this chapter, however, I choose to use a step size of 5% for the \mathbb{C}^{main} family.

Each classifier $C_\beta^{coder} \in \mathbb{C}^{coder}$ simulates the process of having a certain number of human coders observe \hat{H}_I and then classify items into the corresponding \hat{l}_I . Specifically, C_1^{coder} simulates having the entire population classify items. By definition, the outcome of C_1^{coder} , if represented as distributions, is identical to the population ground truth H_I , except that C_1^{coder} as a “human classifier” needs to output labels l_I mapped from H_I . Similarly, $C_{0.95}^{coder}$ simulates 19 coders per item (which then produces the sample \hat{H}_I instead of H_I), $C_{0.9}^{coder}$ 18 coders, and so on, and C_0^{coder} simulates no coder but makes random classifications. Less coders means more errors. Therefore, the sequence of classifiers in \mathbb{C}^{coder} also follows the “degradation to random” natural ranking, and models an idealized set of classifiers that are perfect except for sampling errors. Note that $\beta \in \{1, 0.95, 0.9, \dots, 0.05, 0\}$ corresponds to $m = \{m^*, 19, 18, \dots, 1, 0\}$. So, for example, $C_{\beta=0.95}^{coder} = C_{m=19}^{coder}$. Both forms of using β and m are equivalent.

For both \mathbb{C}^{main} and \mathbb{C}^{coder} , the best classifier is always based on the full information of items and makes 100% accurate classifications. Indeed, $C_1^{main} = C_1^{coder} = C_1$ is the perfect classifier. The worst classifier is always the random classifier that makes random classifications. Denote $C_0^{main} = C_0^{coder} = C_0$ as the random classifier. Both \mathbb{C}^{main} and \mathbb{C}^{coder} simulate a set of 21 classifiers whose performance degrade from C_1 down to C_0 . However, each pair of C_β^{main} and C_β^{coder} where $\beta \in \{0.95, 0.9, \dots, 0.05\}$ is different by construction: performance degrades differently.

Figure 22 illustrates the 21 classifiers in \mathbb{C}^{main} and \mathbb{C}^{coder} degrading from C_1 down to C_0 with decreasing accuracy. Classifiers’ normalized accuracy was computed against I_{main} ($n = n^*$ and $m = 4$) under Ψ_H and Ψ_l respectively. Note that \mathbb{C}^{main} degrades in a linear

fashion as β decreases, whereas \mathbb{C}^{coder} degrades slowly when β is large and plummets when β drops to around zero. That is, the step size of \mathbb{C}^{coder} varies. We would expect to see lower discrimination power τ with \mathbb{C}^{coder} , because the classifiers are quite close when β is large. Also note that \mathbb{C}_1 does not have perfect sample accuracy score when evaluated against the sample ground truth with $m = 4$, because the sample ground truth has sample errors.

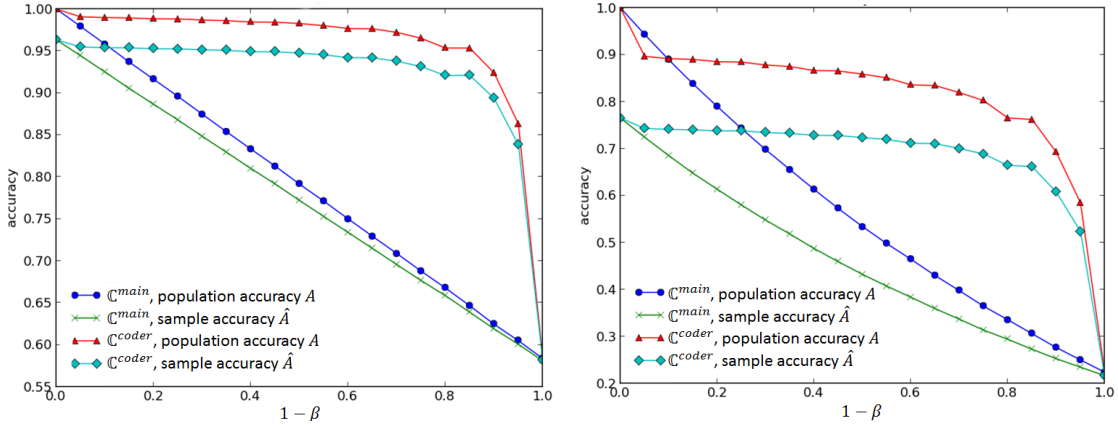


Figure 22. \mathbb{C}_1 degrades to \mathbb{C}_0 for \mathbb{C}^{main} and \mathbb{C}^{coder} : (a) Ψ_H , (b) Ψ_l

4.4 Simulation Results

This section merely presents simulation results. Interpretation of the results will be discussed in section 4.5. Step 1 is the only step that studies Ψ_c (which assumes the label as ground truth model) against Ψ_H and Ψ_l (which assumes the distribution as ground truth model). Steps 2, 3, and 4 study properties of Ψ_H and Ψ_l and no longer consider Ψ_c .

Step 1: Comparing Ψ_H and Ψ_l versus Ψ_c

I'd like to introduce $\mathbb{C}_{majority}^{main}$ and $\mathbb{C}_{majority}^{coder}$ as variations of \mathbb{C}^{main} and \mathbb{C}^{coder} that use the 0-1 cost function (or majority vote) to classify items instead of the quadratic cost function. To be consistent, denote $\mathbb{C}_{quadratic}^{main}$ and $\mathbb{C}_{quadratic}^{coder}$ to be the original \mathbb{C}^{main} and \mathbb{C}^{coder} . These new notations are only used in this step.

The $\mathbb{C}_{majority}$ classifiers and $\mathbb{C}_{quadratic}$ classifiers differ in how they classify items in the partitioning disputed region between the 0-1 and quadratic partitioning. For example, $C_{1,majority}$ would classify $H_i = \langle 0.6, 0, 0.4 \rangle$ into red, but $C_{1,quadratic}$ would classify the item into gray, even though both classifiers are able to observe the item's full information with $\beta = 1$.

Table 9 shows the accuracy of $C_{1,quadratic}$ and $C_{1,majority}$ evaluated against I_{main} ($n = n^*$) under Ψ_H , Ψ_l and Ψ_c respectively. Note that under Ψ_H and Ψ_l , $C_{1,quadratic}$ has higher accuracy than $C_{1,majority}$ and thus is ranked as the better classifier. In contrast, under Ψ_c , $C_{1,majority}$ has higher accuracy than $C_{1,quadratic}$ and is ranked as the better classifier instead. The same ranking between $C_{1,quadratic}$ and $C_{1,majority}$ holds for the other three datasets (I_{gray} , I_{clean} , and I_{biased}) with different m values (e.g., $m = 4$). The same ranking between $C_{\beta,quadratic}^{main}$ and $C_{\beta,majority}^{main}$ and between $C_{\beta,quadratic}^{coder}$ and $C_{\beta,majority}^{coder}$ also holds for any $\beta \in \{0.95, 0.9, \dots, 0.05\}$.

Table 9. Comparisons of classifiers' ranking under different evaluation scheme

Ψ_H	$A (m = m^*)$	$\hat{A} (m = 1)$
$C_{1,quadratic}$	1	0.858
$C_{1,majority}$	0.951	0.823

Ψ_l	$A (m = m^*)$	$\hat{A} (m = 1)$
$C_{1,quadratic}$	1	0.585
$C_{1,majority}$	0.755	0.569

Ψ_c	$A (m = m^*)$	$\hat{A} (m = 1)$
$C_{1,quadratic}$	75.5%	63.7%
$C_{1,majority}$	100%	74.5%

Step 2: Changing n at Fixed m

This step studies how changing n (the number of items in the dataset) affects the power of Ψ_H and Ψ_l . The approach is to compare the sample ranking of classifiers in \mathbb{C}^{main} and \mathbb{C}^{coder} against their "degradation to random" natural ranking respectively and compute

τ under both Ψ_H and Ψ_l , using a dataset with an increasing n (from 100 to 10000, with a step of 100) at fixed m . Simulation is repeated 100 times to compute $\bar{\tau}$.

Figure 23 shows the results of $\bar{\tau}$ for \mathbb{C}^{main} and \mathbb{C}^{coder} using the I_{main} dataset. First, note that $\bar{\tau}$ increases as n increases. This result holds for the other three datasets too. In addition, note that \mathbb{C}^{coder} has a lower $\bar{\tau}$ than \mathbb{C}^{main} , and does not reach $\bar{\tau} \geq 0.9$ even at $n = 10000$. However, as n continues to increase, $\bar{\tau}$ will eventually reach $\bar{\tau} \geq 0.9$. This result also holds for the other three datasets. Finally, note that Ψ_l has a higher $\bar{\tau}$ than Ψ_H for \mathbb{C}^{main} , but then Ψ_H has a higher $\bar{\tau}$ than Ψ_l for \mathbb{C}^{coder} . This result does not always hold for the other datasets. For example, using the I_{gray} dataset, Ψ_l has a higher $\bar{\tau}$ than Ψ_H for \mathbb{C}^{main} when $m = m^*$, but when $m = 1$, Ψ_l then has a lower $\bar{\tau}$ than Ψ_H .

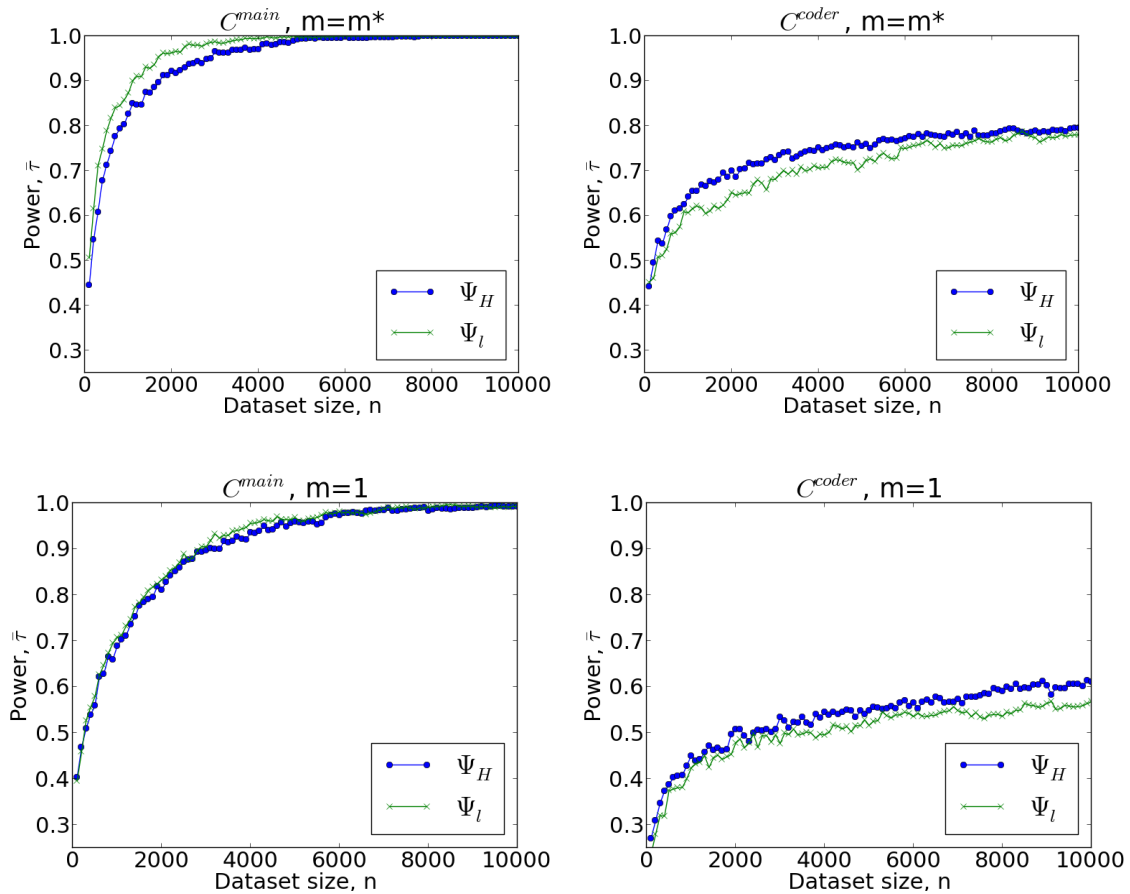


Figure 23. $\bar{\tau}$ changes according to n , with fixed m

Step 3: Changing m at Fixed n

This step studies how changing m (the number of labels per item) affects the power of Ψ_H and Ψ_l . The approach is to compare the sample ranking of classifiers in \mathbb{C}^{main} and \mathbb{C}^{coder} against their “degradation to random” natural ranking respectively and compute τ under both Ψ_H and Ψ_l , using a dataset with an increasing m (from 1 to 20, with a step of 1) at fixed n . Simulation is repeated 100 times to compute $\bar{\tau}$.

Figure 24 shows the results for \mathbb{C}^{main} and \mathbb{C}^{coder} using I_{main} , fixing n at 1000. Note that as m increases, $\bar{\tau}$ also increases slowly. This result does not hold for I_{clean} though, where increasing m does not obviously increase $\bar{\tau}$, which is expected because labels for the same items in I_{clean} almost always agree with each other and adding the same labels does not increase power. Also, simulation shows that even when m increases to $m = m^*$, we still do not have $\bar{\tau} \geq 0.9$ at $n = 1000$. However, fixing n at $n = n^*$, $\bar{\tau}$ is always 1 or close to 1 with $m \in [1, m^*]$ for all four datasets. Finally, note that Ψ_H and Ψ_l have different power in Figure 24: Ψ_l has higher power for \mathbb{C}^{main} , while Ψ_H has higher power for \mathbb{C}^{coder} .

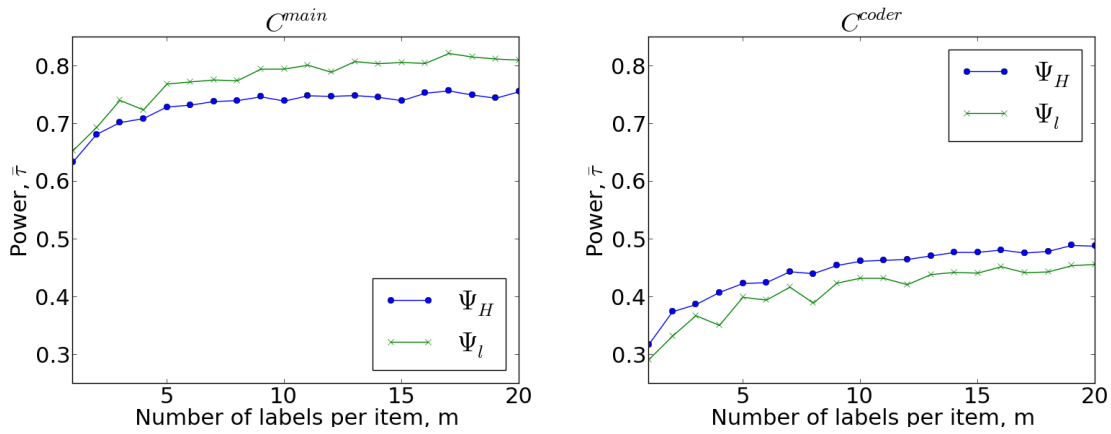


Figure 24. $\bar{\tau}$ changes according to m , with fixed $n = 1000$

Step 4: Optimizing $k = m \times n$

The previous two steps have shown that increasing either m or n will increase $\bar{\tau}$ for both Ψ_H and Ψ_l , although with diminished return. This step is to study the optimal allocation of m and n given a budget constraint for the total number of labels $k = m \times n$. The approach is to study the minimum n required (at a step of 100) in order to have $\bar{\tau} \geq 0.9$ when ranging m from 1 to 20. Again, τ is computed by comparing the sample ranking of classifiers in \mathbb{C}^{main} and \mathbb{C}^{coder} against their “degradation to random” natural ranking respectively, and $\bar{\tau}$ is computed by averaging over 1000 repeated rounds of simulations.

Table 10 lists the minimum n required in order to have $\bar{\tau} \geq 0.9$ for \mathbb{C}^{main} , using the four datasets with selected m values. Figure 25 illustrates the minimum k required in order to have $\bar{\tau} \geq 0.9$ for \mathbb{C}^{main} , using I_{main} with $m \in [1, 20]$. Note that although we only need a smaller n as we increase m , the minimum required k still increases. Results for \mathbb{C}^{coder} is similar, but with a much larger n and k . Also, note that Ψ_H requires a larger n and k than Ψ_l to reach the same level of power to discriminate classifiers in \mathbb{C}^{main} .

Table 10. Minimum n required to get $\bar{\tau} \geq 0.9$ for \mathbb{C}^{main}

	I_{main}		I_{gray}		I_{clean}		I_{biased}	
	Ψ_H	Ψ_l	Ψ_H	Ψ_l	Ψ_H	Ψ_l	Ψ_H	Ψ_l
$m = 1$	3000	2800	3000	4200	1400	1000	2400	2000
$m = 4$	2400	2100	2300	2400	1300 ¹²	1000	1900	1400
$m = 20$	1800	1400	1900	1500	1400	1000	1900	1400
$m = m^*$	1800	1100	1800	1300	1400	1000	1900	1200

¹² Even with 1000 repeated simulation trials, there is still considerable variance introduced by the sample ground truth, the randomness of classifiers, and ranking of the classifiers. More simulation trials should be able to decrease variance and make this number match the other numbers.

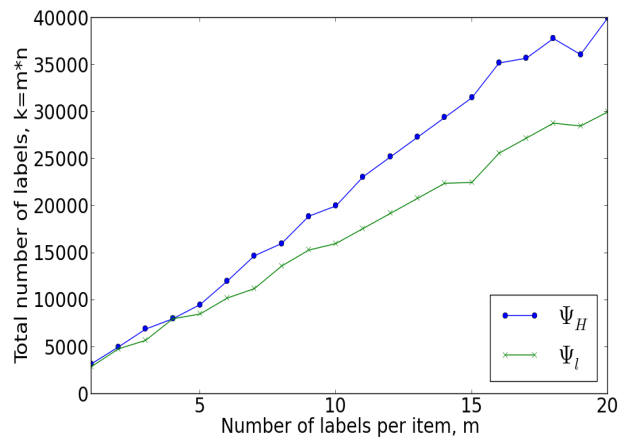


Figure 25. Minimum $k = m \times n$ required for \mathbb{C}^{main} on I_{main}

4.5 Discussion

Using the Wrong Ground Truth Model

One major research question asks how important it is to use the distribution as ground truth model instead of the traditional label as ground truth model for subjective classification problems, or to put it differently, what if anything goes wrong when evaluating classifiers assuming labels were ground truth when really distributions are ground truth. The simple answer is that we might rank classifiers incorrectly using the wrong ground truth model and evaluation scheme.

I have shown in chapter 3 that many low agreement items are in the partitioning disputed region: incorrectly using the label as ground truth model and majority vote, those items would be incorrectly labeled as either red or blue instead of gray. Using those incorrectly labeled items as ground truth to train and evaluate classifiers will drive classifiers to label similar items incorrectly as red and blue instead of gray.

Classifications like that would not be optimal, because those items should have been classified as gray but were incorrectly classified as red and blue, and showing them to users will incur more cost, assuming the true cost function is quadratic.

The intuition above is supported by results from the first step of computer simulation (see Table 9). The natural ranking between $C_{1,quadratic}$ and $C_{1,majority}$ is $\chi^* = \{C_{1,quadratic}, C_{1,majority}\}$, because $C_{1,quadratic}$ can correctly classify items in the partitioning disputed region as gray in order to minimize cost under the quadratic cost function, whereas $C_{1,majority}$ cannot. According to the ranking of both classifiers in Table 9, $\tau(\chi^*, \hat{\chi}) = 1$ under Ψ_H and Ψ_l , and $\tau(\chi^*, \hat{\chi}) = -1$ under Ψ_c . This result holds even when $m = 1$. That means Ψ_H and Ψ_l are better evaluation schemes than Ψ_c . In the real world, using Ψ_c for evaluation will drive classifiers to be optimized closer to $C_{1,majority}$ instead of $C_{1,quadratic}$, which is clearly not optimal.

Reliability of Classifier Evaluation

Chapter 3 has argued that for a typical subjective classification problem, labeled items are not reliable when $m \ll m^*$ due to large sample errors. One main finding of this chapter is this: even though individual items in the labeled dataset are not reliable, classifier evaluation and ranking using either Ψ_H or Ψ_l under the distribution as ground truth model is still reliable with power $\bar{\tau} \geq 0.9$, if n is large enough (e.g., $n \geq 2800$).

As m increases, we could use a smaller n in order to get $\bar{\tau} \geq 0.9$ due to decreased sample errors. However, even when $m = m^*$, we still need a minimum $n = 1000$ in order to have good discrimination power. Suppose $n = 1$, then even the random classifier C_0 is able to guess the one item's correct cost-minimizing label and thus is not discriminated from the perfect classifier C_1 one out of three times. Table 11 shows the probability that the better classifier ($C_{0.8}^{main}$) is discriminated correctly from the worse one ($C_{0.75}^{main}$) with a varying n (evaluated using Ψ_l on I_{main} with $m = m^*$, averaging over 1000 times). When $n \leq 10$, there is a less than 50% chance that the better classifier is discriminated¹³, even though each item in the labeled dataset for evaluation is perfectly reliable with $m = m^*$. When $n = 200$, there is still only 78.1% chance to correctly

¹³ This does not mean the better classifier is ranked lower than the worse classifier more than 50% chance, because I consider the better classifier having the same accuracy as the worse classifier not being discriminated correctly.

discriminate them. In short, it is more important to have a sufficiently large n than decreasing sample errors with a large m in terms of correctly ranking classifiers.

Table 11. Probability of correctly discriminating the first classifier as better

	$n = 1$	$n = 2$	$n = 10$	$n = 100$	$n = 200$	$n = 500$	$n = 1000$
$C_{0.8}^{main}$ vs. $C_{0.75}^{main}$	14.3%	23.9%	48.3%	70.0%	78.1%	90.9%	95.3%

I'd like to make a final point here in terms of sample errors and reliability of classifier evaluation. As argued earlier, by definition the output of $C_{m=1}^{coder}$ is equivalent to the ground truth with $m = 1$. Indeed, Figure 23(d) is to use the output of $C_{m=1}^{coder}$ as ground truth, which is quite "buggy", to evaluate a set of C_m^{coder} ($m > 1$) classifiers which have more coders per item as well as better output. Clearly, here the ground truth data present more errors than the classifiers to be evaluated. How is a low quality ground truth dataset able to evaluate classifiers that produce higher quality classifications and still get good power? It is because a good classifier such as $C_{m=19}^{coder}$ agrees more with $C_{m=1}^{coder}$ over a large number of items than $C_{m=1}^{coder}$ agrees with itself, and therefore a good classifier like $C_{m=19}^{coder}$ is still discriminated as better than $C_{m=1}^{coder}$ even when using a more erroneous dataset as ground truth.

Optimal Allocation of $k = m \times n$ Ratings

Simulation results suggest that the optimal allocation of k ratings for both Ψ_H and Ψ_I is to have $m = 1$ and $k = n$ (Figure 25 and Table 10). One possible explanation is that a second label on the same item partly overlaps with the first label, and thus is less informative. Therefore getting an extra label for the same item is less informative than getting a new label for another new item in terms of the total amount of information obtained.

This result is consistent with findings from the IR evaluation literature (e.g., Carterette and Smucker 2007), which says that even though relevance judgments are subjective and erroneous, evaluations of IR systems are still valid; and a large number of topics with a small number of labels per topic (comparable to a large n and a small m) is better

than a small number of topics with a large number of labels per topic (comparable to a small n and a large m). The result is also consistent with findings from Jordan and Wellman (2009), who used simulations and the framework of *generalization risk minimization* to find that, in the context of multi-agent systems, it could be more beneficial to have a small amount of data for each strategy profile and a large number of strategies and profiles (comparable to a small m and a large n).

In addition, the result does not contradict to Sheng et al (2008), who found that occasionally it is more optimal for classifier training purposes to get another label for the same item than getting a new label for a new item, because it could prevent error propagation in the training process. Obviously, classifier training and evaluation are two different processes, and opposing results (in terms of whether another label on the same item is preferable over a new label on a new item) are not inherently contradictory. If we have to use multiple labels per item in order to improve the training process, we can still just use one label per item to prepare held out data for testing.

When $m = 1$, it seems that Ψ_H (which directly evaluates classifiers with distributions) and Ψ_l (which maps distributions into labels first and then evaluate classifiers with labels) would be the same since we only get one label for an item instead of a full distribution. I'd like to point out that simulation result shows that $\bar{\tau}$ for Ψ_H and Ψ_l are still different when $m = 1$. The reason is that Ψ_H and Ψ_l still use different equations to compute classifier accuracy (Equation 11 versus Equation 13) regardless of the value of m , where Ψ_H takes into account "classification regret" and Ψ_l simply counts the number of correct and incorrect classifications.

Even though simulation suggests $m = 1$ is optimal for classifier evaluation, it does not mean that researchers should always obtain labeled datasets with $m = 1$, because a labeled dataset can be used for purposes other than classifier evaluation. For example, if we just want to use human coders to classify the political leaning of a few articles (e.g., Munson and Resnick 2010), then we need $m > 1$ in order to get reasonably accurate

results. Or, we might want to use multiple coders to check each other's quality: a coder might be a spammer if he rarely agrees with other coders more than at random.

Finally, simulation suggests that a classifier evaluated against a sample ground truth with $m \ll m^*$ would result in a lower accuracy score than with $m = m^*$. This finding is consistent with existing studies (e.g., Phelps et al 1995) and is understandable: even the perfect classifier would have imperfect accuracy when evaluated against an erroneous ground truth dataset. Therefore, in order to learn a more precise accuracy score for a classifier in addition to correctly ranking the classifiers, we might want to have $m > 1$ to decrease errors in the ground truth dataset.

Comparing Ψ_H and Ψ_l

One research question is to ask whether Ψ_H or Ψ_l is a better evaluation scheme with the distribution as ground truth model, where the exact location of a point on the simplex (i.e., an item's distribution) matters under Ψ_H , but only the partition of a point on the simplex matters under Ψ_l . Simulation at its current form does not provide a clear answer: Ψ_H has higher power than Ψ_l to discriminate classifiers in the \mathbb{C}^{coder} family; Ψ_l has higher power than Ψ_H to discriminate classifiers in the \mathbb{C}^{main} family, but not with the *Igray* dataset at $m = 1$.

What we did learn from simulation, however, is that both Ψ_H and Ψ_l are valid evaluation schemes that could achieve high classifier discrimination power with a large n , although sometimes one has a higher power than the other and thus requires a smaller n . It is quite surprising because Ψ_l essentially throws away information, yet it can still achieve high power, sometimes even higher than Ψ_H , to evaluate classifiers. I'll leave it to the future work to study when Ψ_H or Ψ_l is more suitable in what situations.

There are other practical reasons to choose between Ψ_H and Ψ_l regardless of their power. The benefit of Ψ_l is that we only need to map distributions into labels once, and then it is convenient to use the cost-minimizing labels to evaluate multiple classifiers that also classify items into labels. In addition, with Ψ_l , it is more straightforward to

report other evaluation metrics such as “precision” and “recall” simply by comparing the classification results against testing items’ cost-minimizing labels. For these reasons, I’d suggest using Ψ_l unless there are specific reasons to prefer Ψ_H for a particular situation. In chapter 5, I will use Ψ_l to evaluate political leaning classifiers.

Finally, I’d like to provide some intuitions to explain the simulation result about comparing Ψ_H and Ψ_l . Recall that Ψ_H ranks classifiers based on the exact classification cost of each item, while Ψ_l ranks classifiers based on the cost-minimizing labels instead. For example, under Ψ_H , misclassifying $\hat{H}_i = \langle 0.51, 0.49, 0 \rangle$ into gray is not as bad as misclassifying $\hat{H}_j = \langle 0.99, 0.01, 0 \rangle$ into gray, because i is “less red” than j . Under Ψ_l , however, misclassifications of both items incur the same cost because the cost-minimizing labels for both items are the same. The setup of the \mathbb{C}^{main} classifier family does not treat items like i and j differently: i and j have the same random error rate to flip into an incorrect label. Therefore, Ψ_l is more advantages for \mathbb{C}^{main} because Ψ_l does not treat items like i and j differently either, resulting in higher power. On the other hand, if the classifiers are designed to classify items into *distributions* instead of *labels*, one can imagine that Ψ_H might turn out to be the better than Ψ_l . I will discuss more about it in the future work.

4.6 Conclusions and Future Work

In the previous two chapters, I have proposed the distribution as ground truth model for subjective classification problems, and I have shown with empirical dataset that SCPs do exist with many low agreement items and different partitioning does result in different cost minimizing labels. This chapter uses computer simulation to study using the distribution as ground truth model for classifier evaluation. Simulation shows that using the wrong ground truth model and evaluation scheme could lead to ranking classifiers incorrectly.

For a subjective classification problem that uses distribution as ground truth, one can no longer assume the labeled dataset obtained from a small number of raters is reliable.

Another main result of chapter is this: even though individual items in the labeled dataset are unreliable, we may still get reliable results about classifier ranking as long as we have a large number of items in the evaluation dataset.

The optimal way to obtain labels for classifier evaluation is to obtain one label per item with many items, and it is usually advisable to map the distributions into cost-minimizing labels first and then evaluate classifiers with the labels. According to simulation, we need to label about 1000 to 3000 items with one label per item in order to rank classifiers correctly with good classifier discrimination power. The exact number of items needed is subject to many factors, such as the step size of classifier difference, the level of disagreement among coders, the number of classification categories, and so on. However, the caveat here is that too few labeled items for classifier evaluation is likely to lead to incorrect evaluation result.

When we only have one label from a human rater instead of a full distribution of an item, one might ask whether the distribution as ground truth model still matters. Conceptually (which has been discussed in chapter 2), the distribution as ground truth model is defined regardless of the number of labels to obtain for each item: even if we only get one label per item, the one label is an estimate to a distribution instead of an estimate to a label. Practically, I have shown in this chapter that different ground truth models and evaluation schemes use different equations to compute classifier accuracy, have different classifier evaluation power, and produce different classifier rankings, even when we only have one label per item.

The findings of this chapter echo the research of Hand (2006), who argues that the progress of classification application research is an illusion. One may easily find many examples in existing literature where researchers use only a small number of labeled items (e.g., 100~300) to evaluate their classifiers and conclude that their inventions are superior to alternatives, without realizing that their results could be unreliable due to the unreliable labeled dataset in the evaluation process. Practitioners and researchers of machine learning applications should be aware of this danger and try to avoid it.

To conclude this chapter, I'd like to propose a set of guidelines to collect labeled items for classifier evaluation in the machine learning area. Note that this process is intended to prepare labels for evaluation, not for training. And it is *not* supposed to be used with qualitative study which requires the qualitative coding process as I have discussed earlier.

Step 1: Prepare a codebook with a set of objective instructions on how to label items. In order to increase generalizability of the classifications, the codebook should not represent researchers' own specific subjective opinions. This process also helps to determine whether the classification problem at hand is objective or subjective: for a subjective classification problem, it will be hard to write down classification rules without giving subjective specifics.

Step 2: Use multiple coders to label a subset of items, and compute inter-rater reliability score. If IRR is high, then the classification problem should be objective. Researchers could assume the label as ground truth model and just use one coder to label the rest of the items. On the other hand, if IRR is low and it is hard to define a codebook, then the classification problem is likely to be subjective, and researchers should proceed to the next step.

Step 3: Use one coder per item to label 1000 to 3000 items as the classifier evaluation dataset. Note that the exact number of items required is dependent on the application specifics, but it cannot be too small to have good classifier discrimination power. In order to avoid bias, researchers should randomly draw a rater from a pool of raters for each item (perhaps through Amazon Mechanical Turk, for example), instead of having one of the researchers label all items himself.

Step 4: Follow the steps of Ψ_l to evaluate the classifiers: use Equation 13 to compute accuracy for each classifier, and rank the classifiers according to their normalized accuracy. The classifier with higher accuracy would be the better classifier, and we know it is reliable. Note that the resultant accuracy score would be lower than its actual value because the ground truth data has sample errors.

I will discuss future work next. One challenge of future work is to increase the power of computer simulation. Currently, computer simulation relies heavily on the \mathbb{C}^{main} classifier family, which is a simplified model of real world classifiers: it only models different classifiers according to a set of fixed error rates, without considering items' rich features commonly seen in the real world. For example, a real classifier might be able to do very well on high agreement items, but make more mistakes on low-agreement items. Such a classifier is not modeled in the current simulation. If we add such a classifier to simulation and want to discriminate it as a better classifier, we might hypothesize that one label per item is not optimal for evaluation because multiple labels per item is required to distinguish high agreement items from low agreement items.

One of the research goals of this chapter is to compare two classifier evaluation schemes – whether to evaluate classifiers directly with distributions (Ψ_H), or to map distributions into labels first, and then evaluate classifiers with the mapped labels (Ψ_I) – in terms of their power of correctly ranking classifiers. Statisticians might find that the set of simulated classifiers (\mathbb{C}^{main}), which takes one parameter to control how much randomness is in the classification results, is comparable to a set of models with varying randomness, and therefore, ranking a pair of classifiers is comparable to two-sided hypothesis testing about whether their samples come from the same distribution. In that sense, studying the power of a classifier evaluation scheme in terms of correctly ranking classifiers is comparable to studying the power of the evaluation scheme as a statistical test. This is not new to statistics: a plethora of work is done on efficient estimators, and many conclusions have been made about which statistical test is more powerful under what conditions (e.g., Casella and Berger, 2002).

This chapter does not attempt to make general conclusions about the statistical power of the classifier evaluation schemes. It makes assumptions about the structure of errors made by classifiers (the families \mathbb{C}^{main} and \mathbb{C}^{coder}) only for illustrative purposes, to show that, at least for some families of classifiers, some particular evaluation schemes are better than others. If we assumed that real classifiers did make errors following

particular structures such as \mathbb{C}^{main} , then it would be valuable to map results from statistical theory to identify other evaluation schemes that might be more efficient, for that family of classifiers, than those considered in this chapter.

Also, I'd like to define "boundary items" as those items whose \hat{H}_i are plotted on the simplex partitioning boundaries. For example, when we get two labels per item, there are only 6 possible combinations of the two labels, and we would get many cases such as $\hat{H}_i = \langle 0.5, 0.5, 0 \rangle$ (or $\langle 0.5, 0, 0.5 \rangle, \langle 0, 0.5, 0.5 \rangle$) on the partitioning boundary between red and gray. Boundary items are not a problem for Ψ_H , which directly uses the distributions for evaluation. But they do create a problem for Ψ_l because Ψ_l needs to map \hat{H}_i into either one of the two bordered cost-minimizing labels. In this chapter, I just randomly pick one label to map. But future work should consider other possibilities such as arbitrarily picking the gray label, obtaining another label to move the item out of the boundary, or discarding the item for evaluation.

Finally, generalization of the current study is limited by the particular choice of the political leaning classification problem, where I have defined three classification categories (red, gray and blue) and used the quadratic cost function. Future work should address whether the result from this chapter is generalizable to a broader range of subjective classification problems where the classification categories are not three and the cost function is not quadratic.

Chapter 5. LabelPropagator: A Semi-Supervised Classifier

This chapter proposes a new political leaning classification algorithm called LabelPropagator, which automatically classifies people and items as liberal or conservative. The intuition of LabelPropagator is that a few manually coded labels may be further propagated to identify other people and articles, since liberals are likely to endorse liberal articles, and likewise for conservative people and articles. Different from other traditional classification algorithms, LabelPropagator does not rely primarily on textual information, although I do consider it as an extra information source.

There is a naturally occurring source for the data needed to propagate: a large pool of subjective “votes” for individual articles. Digg.com, a popular social news aggregator, has links to political stories from both blogs and news sites. Users can “digg” stories they like. Individual diggs are visible on the website and accessible through a public API, or rather were at the time of data collection. Other sources could include tweet mentions, Facebook “I Like”, or Google “+1”.

Figure 26 illustrates the potential propagation of a few initial labels through the diggs network. The links in the graph represent diggs, the votes by users for particular articles. The articles dugg by the red user can be colored red. Similarly, the people who dugg the blue articles can be colored blue. In subsequent rounds, those colorings can be propagated still further. Eventually there will be articles dugg by both red and blue users: the LabelPropagator algorithm will handle those cases to classify articles properly.

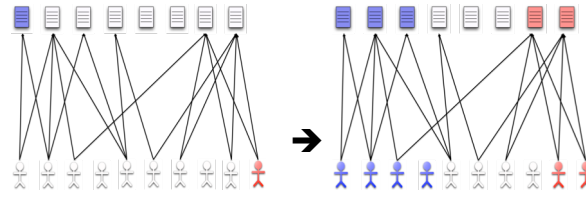


Figure 26. Intuition of LabelPropagator

This chapter has two parts. The first part was completed before beginning this thesis. It proposed and evaluated the LabelPropagator algorithm. However, the main limitation of the original study was that the labeled dataset to train and evaluate classifiers only consisted of clearly red and blue items without gray items. Items where human labelers disagree were excluded. The second part of this chapter aims to fix the unreliable ground truth dataset problem by using the dataset obtained from chapter 3 and the classifier evaluation scheme proposed in chapter 4. I will show that using a different ground truth model and an evaluation scheme indeed leads to retuning parameters of the classifier and to different conclusions about its overall accuracy.

This chapter is organized as follows. Section 5.1 discusses related work. Section 5.2 proposes three variations of the graph propagation algorithm. Section 5.3 discusses the original study of the algorithm. Section 5.4 discusses the evaluation of the algorithm with new ground truth data and evaluation scheme from chapters 3 and 4. Section 5.5 summarizes the chapter and discusses future work. Sections 5.1 to 5.3, describing the original study, first appeared in Zhou et al (2011). Sections 5.4 and 5.5 belong to the second part of the chapter, which is the follow-up study. Note that some notations to be introduced in this chapter may differ from the other chapters in order to be compatible with earlier publication.

5.1 Related Work

Political Leaning Classification

Literature from both political science and computer science has studied the problem of classifying political positions from texts. One line of work used word frequencies,

Bayesian statistical models, and topic models (Laver et al 2003, Martin and Vanberg 2008, Lin 2006, Lin et al 2008, Monroe et al 2008, Slaping and Proksch 2008). Another line of work focused on using SVM with optimization of text feature selection (Jiang and Argamon 2008, Oh et al 2008, Yu et al 2008, Hirst et al 2010), as well as complementing that with sentiment analysis (Durant and Smith 2006, Mullen and Malouf 2006, Malouf and Mullen 2007, Holtzman et al 2007). Rao et al (2010) studied how to use text features in tweets to classify twitter users.

Rather than using text features, Efron (2004) used co-citations from Google search to classify political blogs. Park et al (2011) used a panel of “predictive users” and their comments’ sentiments to predict the political leaning of articles. Pennacchiotti and Popescu (2011) used features from user profiles, tweet behavior, tweet content, and tweet networks to classify twitter users. Conover et al (2011) hired two coders to manually classify the political leaning of twitter users from their tweets.

Semi-supervised Learning

Semi-supervised learning approaches use a large amount of unlabeled data in the classification process, and thus achieve good classification performance even if only a small set of labeled examples are available. A particular family of semi-supervised classification algorithms, which we draw on, cast the classification task as a process of label propagation in the graph structure of labeled and unlabeled data (Zhu et al. 2003, Zhou et al. 2004).

The closest study to ours is Lin and Cohen (2008), who used a semi-supervised learning algorithm called “multirank” to classify political blogs using the HTML links between blog stories. We use different algorithms and find that, once we propagate via diggs, adding propagation via blog-to-blog HTML links actually decreases accuracy.

5.2 Semi-Supervised Learning Algorithms

5.2.1 Problem Formulation

From Digg, we have a set of stories V_{story} , users V_{user} , and the diggs from users to stories as approval votes, denoted as $E_{\text{digg}}=\{(i,j)\}$, where $i \in V_{\text{user}}$ and $j \in V_{\text{story}}$. We also have 1) other types of nodes, V_{extra} , such as the domain names of blogs, and 2) other types of links, E_{extra} , which will be discussed in the Datasets section. Let $V=V_{\text{user}} \cup V_{\text{story}} \cup V_{\text{extra}}$, $E=E_{\text{digg}} \cup E_{\text{extra}}$, and then we can construct a graph $G = \langle V, E \rangle$. Edges $e \in E$ are undirected because the color label of either node of e should propagate to the other.

From G , we construct the symmetric affinity matrix W , where $W_{ij}=1$ if $(i,j) \in E$, or 0 otherwise. Define D as the diagonal matrix of degrees of the nodes. That is, $D_{ii}=\sum_j W_{ij}$.

Denote the initially labeled nodes as L (LCV). Let $C=\{\text{red}, \text{blue}, \text{gray}\}$ be the set of category labels. For each $i \in L$, we have an initial label $c_i \in C$. Let $T=V-L$ be the set of unlabeled nodes (i.e., nodes that need to be classified). The labels will be divided into a training set L_{training} and a testing set L_{testing} , where $L = L_{\text{training}} \cup L_{\text{testing}}$. The goal of our algorithms is to use the initial labels from L_{training} and assign a label $c'_j \in C$ as the classification output to each node $j \in T \cup L_{\text{testing}}$.

We also introduce a $|V| \times 2$ matrix Y , where the 2 column vectors are indexed as R for *red* and B for *blue*. $Y_{iR}=1$ if $i \in L_{\text{training}}$ and $c_i=\text{red}$; $Y_{iB}=1$ if $i \in L_{\text{training}}$ and $c_i=\text{blue}$; 0 for the other elements.

Note that any *gray* labels in L are simply ignored for the purposes of training. Intuitively, we do not treat *gray* as an independent classification category but rather as nodes for which the classification is mixed or uncertain. Thus, we do not propagate *gray* classifications. However, our algorithms could still output *gray* for borderline items.

5.2.2 Random Walk with Restart (RWR)

Our first semi-supervised algorithm, shown in Figure 27, is based on the popular “random walk with restart” model (Grinstead and Snell 1997). After convergence, F^* is the stationary distribution specifying the probability that a random walk with restart at Y' will be at each of the nodes.

In our case, the restart matrix Y' has two columns, corresponding to two separate random walks, leading to two stationary distributions. The first walk, with the R column of Y' , restarts from only the *red* labeled nodes, and the F_{iR}^* scores indicate the stationary probabilities restarting only from the *red* nodes. Similarly, the second walk, with the B columns of Y' , yields the F_{iB}^* scores. Intuitively, $F_{iR}^* > F_{iB}^*$ means a walk starting from the *red* nodes is more likely to reach node i than a walk starting from the *blue* nodes, and thus we should label i as *red*.

When F_{iR}^* is close to F_{iB}^* , there is not a clear classification and we label them *gray*. Two threshold parameters θ_R and θ_B control how big the ratio of F_{iR}^* and F_{iB}^* has to be in order to make a *red/blue* classification. When $\theta_R = \theta_B = 1$, the algorithm simply compares F_{iR}^* and F_{iB}^* and does not generate *gray* classifications (except for the rare cases where $F_{iR}^* = F_{iB}^*$). Increasing the thresholds leads to output of more *gray* labels for borderline nodes.

5.2.3 Local Consistency Global Consistency (LCGC)

Our second semi-supervised learning algorithm, also shown in the algorithm box, follows the “local consistency global consistency” classification algorithm proposed in Zhou et al (2004). Note that the iteration process differs from RWR only in normalization factors for S and Y .

The intuition behind the algorithm is to optimize for two conditions: a) the labels assignment should not change too much between nearby nodes (“local consistency”), and b) the initial labels assignment should not change too much after propagation

(“global consistency”). The parameter α controls the trade-off between the two objectives.

5.2.4 Absorbing Random Walk (ARW)

The “absorbing random walk” model is different from the original random walk model in that it has “absorbing states” without outgoing links (Grinstead and Snell 1997). Let ACV be a set of absorbing states. After convergence, each node $i \in V$ has a probability score $P(a|i)$ for each absorbing state $a \in A$, indicating the probability that i would eventually be absorbed into a . We have $\sum_{a \in A} P(a|i) = 1$, and $P(a|a)=1$. The use of absorbing random walks in classification is closely related to Zhu et al (2003).

In our case, we did not use the labeled dataset L as the absorbing states, because the initial labels are not 100% accurate and should be allowed to change color during propagation. Therefore, we added two new absorbing states a_{red} and a_{blue} to V , and added directed edges $\{(i, a_{red})\}$ and $\{(j, a_{blue})\}$ if $i, j \in L$ and $c_i = red, c_j = blue$.

In step 1, $k \in [0, \infty)$ is the weight of the newly added edges $\{(i, a_{red})\}$ and $\{(j, a_{blue})\}$. In step 2, weights on edges (including the new edges) are normalized to create probability distributions over transitions from each node. The normalized matrix decomposes into Q , which gives probabilities of transitions to edges in the original graph, and Y' , which gives probabilities of transitions to the absorbing states a_{red} and a_{blue} . After the random walk converges, F_{iR}^* and F_{iB}^* are the probabilities for each $i \in V$ eventually getting absorbed in a_{red} and a_{blue} respectively. In steps 4 and 5, we classify the nodes using the “class mass normalization” method suggested by Zhu et al (2003).

Intuitively, this algorithm classifies $i \in V$ as *red* if it has a much higher probability to be absorbed in a_{red} , and the same for *blue*.

Algorithm 1: RWR

Input: W, D, Y

Algorithm:

1. Construct the transition matrix $S=D^{-1}W$
2. Construct $Y'=(D_Y^{-1}Y^T)^T$, where D_Y is a diagonal matrix with $D_{Y-ii}=\sum_j Y_{ji}$.
3. Iterate $F(t+1)=(1-\alpha)SF(t)+\alpha Y'$, where $F(t)$ is a $|V|\times 2$ matrix, $F(0)=Y'$, and α is a tunable teleport factor. Let F^* denote the limit of the sequence $\{F(t)\}$

Classification: Label $i\in V$ as:

- a. *red* if $F_{iR}^*/F_{iB}^* > \theta_R$
 - b. *blue* if $F_{iB}^*/F_{iR}^* > \theta_B$
 - c. *gray* otherwise
- (θ_R and θ_B are tunable threshold parameters.)
-

Algorithm 2: LCGC

Input: W, D, Y

Algorithm:

1. Construct the matrix $S=D^{-1/2}WD^{-1/2}$
2. Iterate $F(t+1) = (1-\alpha)SF(t)+\alpha Y$, where α is a tunable parameter in $(0, 1)$, and $F(0)=Y$. Let F^* denote the limit of the sequence $\{F(t)\}$

Classification: *the same as in RWR*

Algorithm 3: ARW

Input: W, D, Y

Algorithm:

1. Construct $W'=\begin{bmatrix} W & kY \\ 0 & I \end{bmatrix}$, where $k=(1-\alpha)/\alpha$, $\alpha\in(0,1]$ is a tunable parameter; I is a 2×2 identity matrix.
2. Construct $S'=D'^{-1}W'$, where D' is a diagonal matrix with $D'_{ii}=\sum_j W'_{ij}$. S' has form $\begin{bmatrix} Q & Y' \\ 0 & I \end{bmatrix}$.
3. Iterate $F(t+1)=Y'+QF(t)$, where $F(0)=Y'$. Let F^* denote the limit of the sequence $\{F(t)\}$.
4. Let $D_p=\begin{bmatrix} P(a_{red}) & 0 \\ 0 & P(a_{blue}) \end{bmatrix}$, where $P(a_{red})=\sum_{i\in V}(F_{iR}^*/|V|)$, $P(a_{blue})=\sum_{i\in V}(F_{iB}^*/|V|)$. $P(a_{red})+P(a_{blue})=1$.
5. Calculate $F^{*'}=(D_p^{-1}F^{*T})^T$

Classification: *the same as in RWR using $F^{*'}$*

Figure 27. Three versions of the semi-supervised learning algorithms

5.3 Original Study

5.3.1 Datasets

Graph Structure

Diggs as votes from user nodes to story nodes ($V_{story}, V_{user}, E_{digg}$). This is the primary dataset for our study. With the Digg API, we harvested 480,932 stories and their 6,298,104 diggs from 2 categories, ‘political news’ and ‘political opinions’, from 2009-5-25 to 2010-8-11. That is an average of 1,083 and 14,185 new stories and diggs each day, with an average of 13 diggs per story. Our study only used the $|V_{story}|=84,433$ “popular” stories that received more than 10 diggs, and the $|V_{user}|=74,844$ “frequent” users who submitted more than 5 diggs. Those “frequent” users made a total of $|E_{digg}|=5,216,273$ diggs to the “popular” stories. The median degree for items (number of frequent users in the dataset who dugg the popular item) was 22. Note that each user could submit an unlimited number of diggs, but only one per story. The largest connected component covers 99.98% of the nodes.

Domain source links (V_{source}, E_{source}). For each story in V_{story} , we have its source domain name. For example, stories posted on HuffingtonPost.com would all share the same domain name. We hypothesize that stories with the same domain name would share the same political leaning. This is clearly true for political blogs like HuffingtonPost.com, but not necessarily true for NYTimes.com or Blogspot.com. We created domain source nodes V_{source} and edges $E_{source}=\{(i,s): i \in V_{story} \text{ was posted on source domain } s \in V_{source}\}$.

Links from blogs to stories ($V_{link-to}, E_{link-to}$). Munson et al (2008) created a dataset consisting of political blogs and their HTML links to other stories. We hypothesize that stories linked to by the same blog would have the same political leaning as the blog. We created $V_{link-to}$ for the 396 political blogs that linked to any stories in V_{story} , and undirected edges $E_{link-to}=\{(i,j): i \in V_{link-to} \text{ HTML links to } j \in V_{story}\}$. $|E_{link-to}|=17,372$.

User friendship links (E_{user}). Digg.com allows users to mark other users as friends, by mutual consent. We hypothesize that users who are friends on Digg.com will tend to

share the same political leaning. Using the Digg API, we harvested a total of 2,247,591 user-user friendship links, among which 755,303 are between $u \in V_{\text{user}}$. We created $E_{\text{user}} = \{(i, j) : i, j \in V_{\text{user}} \text{ are friends on Digg.com}\}$.

Story similarity links (E_{story}). Using the Digg API, we obtained the title and a short text snippet for each story, which was used to calculate text similarity between each story pair. We hypothesize that stories that have similar text would also share similar political leaning. We used Apache Lucene to calculate text similarity. We considered only terms that appeared in at least 5 stories but not more than 40% of the total stories. For each story, we selected its 20 terms with the highest tf*idf scores and used them to calculate a cosine similarity with each other story. If i was one of the 10 stories with highest similarity to j , and also j was one of the 10 most similar to i , an undirected edge (i, j) was added to E_{story} . $|E_{\text{story}}| = 107,961$, so each story had an average of 1.3 text similarity links.

Labels

Users identified in a news article ($L_{\text{user-reported}}$). A news article¹⁴ reported a group of conservative Digg users who created a Yahoo group called the “Digg Patriots” and self-organized themselves to deliberately bury liberal stories on Digg. The article identified 106 conservative users of the group. It also identified 44 liberal users on Digg who were their primary targets (i.e., stories suggested to Digg by those 44 liberal users were voted down). Digg.com has purged some members of the “Digg Patriots” from the system to prevent manipulation. But we still have 104 labeled Digg users, 68 *red* and 36 *blue*. We denote these labeled users as $L_{\text{user-reported}}$. These users dugged 69,785 (83%) of the stories in V_{story} .

Labeled blogs (L_{blogs}). From five sites that classify blogs¹⁵, we compiled 1,635 blogs tagged as conservative or liberal. Of these, only 240 (15%) had any stories in V_{story} . Those

¹⁴ <http://blogs.alternet.org/oleoleolson/2010/08/05/massive-censorship-of-digg-uncovered>

¹⁵ They are: blogcatalog.com, blogarama.com, httpetalkinghead.com, blogs.botw.org, and wonkosphere.com

blogs form a subset of V_{source} that we call L_{blogs} . 25,643 (30%) of the stories in V_{story} were from one of these labeled blogs.

More labeled blogs ($L_{\text{link-to}}$). For the 396 political blogs in $V_{\text{link-to}}$, Munson et al (2008) also labeled them as liberal or conservative. We used them as $L_{\text{link-to}}$, which overlapped but was distinct from the L_{blogs} above.

Manually coded stories from Amazon Mechanical Turk (L_{mturk}). We randomly selected 1000 stories from V_{story} and posted them on AMT. For each story, we accepted 6 ratings (3 from self-identified liberals and 3 from self-identified conservatives), at the cost of 3 cents per rating and a \$1-\$2 weekly bonus to the most productive turkers. We collected the ratings from 2010-7-8 to 2010-8-22, with 50-500 incoming ratings per week. 41 turkers coded at least one story, and 13 (8 liberals and 5 conservatives) coded more than 50 stories.

For quality control purposes, we required the turkers to pass a qualification test with 9 correct answers out of 10 questions: 5 questions on basic political knowledge (e.g., who was the Republican candidate in the 2008 presidential election?) and 5 questions on the real coding tasks to test their understanding of the coding guideline. They also had to meet the following criteria: a) located in the US, b) > 90% acceptance rate on other AMT tasks, and c) complete our survey on their political leaning

We also randomly inserted verification questions (e.g., “1+4=?”) into 100 stories, and got correct answers from all turkers who encountered them. The turkers spent an average of 63 seconds on each story. The Fleiss inter-rater reliability score was 0.53, a “moderate agreement” (Landis and Koch 1977).

The limited inter-rater reliability suggests that there is not universal agreement about the liberal-conservative categories and that they apply more clearly to some stories than others. We considered those stories where all 6 turkers agreed to be clear examples. There were 73 red and 234 blue stories in our labeled dataset L_{mturk} . In section 5.4, we return to consideration of stories where raters were not unanimous.

Manually coded users ($L_{user-coded}$). We selected 220 Digg users from V_{user} who had made more than 15 comments, with more than half of their most recent comments on political stories. Then, we took a snapshot of their 15 most recent comments, and hired 2 undergraduate students to code them into *red*, *blue* and *gray* based on the political leaning inferred from the 15 comments.

Before the coding process, we trained the coders to follow coding guidelines. For quality control purposes, we inserted six known Digg users from $L_{user-reported}$ into the 220 user pool, and both coders correctly classified them. When both coders felt confident enough to assign a *red* or *blue* label, their agreement was 94.5% and their Cohen’s kappa score was 0.89. We took the 69 red users and 62 blue users that both coders agreed upon as clear examples, and formed dataset $L_{user-coded}$.

5.3.2 Evaluation

We organized our evaluation process into 4 steps. Due to limited space, we only document the detailed optimization process for the RWR algorithm, and simply report the results for the other 2 algorithms in the first 3 steps. In the last step, we compared the three algorithms. For all 4 steps, we used 10-fold cross validation, repeatedly holding out one tenth of the nodes with known labels for testing.

The primary measurement was “accuracy”, averaged across the 10 folds of cross validation. Let $O_{testing} \subseteq O$ be the output for $i \in L_{testing}$. Let $O_{testing}^*$ be the subset of $O_{testing}$ that are correctly classified.

$$\text{Accuracy} = \frac{|O_{testing}^*|}{|O_{testing}|}$$

Step 1: Optimizing Parameters

We were not confident on the usefulness of the extra structural datasets beyond the diggs, nor of two of the label datasets, L_{blogs} and $L_{link-to}$. Thus, to tune the algorithm parameters, we used the limited network $G' = \langle V', E' \rangle$, where $V' = V_{story} \cup V_{user}$, $E' = E_{digg}$, and labeled data $L' = L_{user-reported} \cup L_{user-coded} \cup L_{mturk}$. On average, for nodes in a cross-

validation test set, the shortest path from a node in the training set is 2.55, suggesting that a small but non-trivial amount of propagation will be necessary to propagate labels to nodes in the test set.

First, we optimized the teleport factor α for RWR, fixing θ_R and θ_B at 1.0. Figure 28(a) shows that accuracy was not sensitive to α in the range 0.1 to 0.7. The optimal was $\alpha=0.3$, yielding accuracy 94.8%. For the LGC and ARW algorithms, the optimized α was 0.3 and 0.1 respectively. We used these values for the rest of the evaluation.

Since our labeled datasets include only definitively labeled items (*reds* and *blues*, but no *grays*), overall accuracy will be optimized only when all items are assigned a *red* or *blue* label definitively. Not surprisingly, then, holding $\theta_R=1.0$ the optimal value for θ_B was also 1.0, and vice versa. Thus, we assign red labels whenever $F_R^* > F_B^*$ and blue labels when $F_B^* > F_R^*$.

However, θ_R and θ_B could be used to trade off precision and recall rather than simply optimizing for overall accuracy. Precision and recall for *red* are defined as follows (for *blue* they are defined analogously):

$$\text{Precision}_R = \frac{|O_R^*|}{|O_R|}, \text{Recall}_R = \frac{|O_R^*|}{|L_{\text{testing},R}|}$$

In the formula, O_R is a subset of O_{testing} that are *red*. O_R^* is the subset of O_{testing} that are correctly classified as *red*. $L_{\text{testing},R}$ is the set of *red* nodes in the initial testing set.

The results are shown in Table 12. At $\theta_R=\theta_B=1.0$, *red* had higher recall and *blue* had higher precision. That means our algorithm tended to over-classify nodes as *red*.

Table 12. High recall for red; high precision for blue

	Precision	Recall
<i>Red</i>	88.0%	99.7%
<i>Blue</i>	99.6%	92.1%

To trade off precision and recall, we could adjust the θ_R and θ_B threshold parameters. In general, as we increase θ_R , fewer things are classified as *red* and more are left as *gray*, increasing precision but decreasing recall for red items, and similarly for θ_B . We have similar results for LCGC and ARW algorithms, too. We will return to the θ parameters in section 5.4, where items that turkers did not agree on are labeled *gray* for the purpose of evaluation.

Step 2: Evaluating Source and Link-to Relations from Labeled Blogs

In this step, we evaluated the usefulness of the two labeled sets of blogs, L_{blogs} and $L_{\text{link-to}}$. We added nodes for the blogs in L_{blogs} and edges from them to stories that appeared in those blogs, and added nodes for the blogs in $L_{\text{link-to}}$ and edges for their HTML links to stories. Table 13 shows the effects on accuracy. We get similar results using LCGC and ARW. Since the labels (and nodes and edges) associated with L_{blogs} were useful, we included them in the baseline for assessments in step 3. But we excluded $L_{\text{link-to}}$ and the associated $V_{\text{link-to}}$ and $E_{\text{link-to}}$ because they did not increase accuracy. The optimized labeled dataset was then $L^* = L_{\text{user-reported}} \sqcup L_{\text{user-coded}} \sqcup L_{\text{mturk}} \sqcup L_{\text{blogs}}$.

Table 13. Blog sources are useful; not blog links

		Add $L_{\text{link-to}}$?	
		No	Yes
Add L_{blogs} ?	No	94.8%	92.1%
	Yes	95.4%	92.9%

Step 3: Evaluating Structural Datasets

In this step, we evaluated the usefulness of the three extra structural datasets V_{source} , E_{source} , E_{user} , and E_{story} that added additional nodes and links without adding any additional labels. Prior to this step, we used $E = E_{\text{digg}}$, and the weight w_{ij} for each $(i,j) \in E$ was set to 1. Since we have more than 5 million links in E_{digg} and the number of extra links, $|E_{\text{domain}} \sqcup E_{\text{user}} \sqcup E_{\text{story}}|$, is only 10% of $|E_{\text{digg}}|$, adding the extra links to E with the same weight as E_{digg} would not have much effect. Therefore, we also optimized the weight for the extra links by varying their values from 1 up to 100.

First, we added V_{source} and E_{source} to G . Note that L_{blogs} was already included in G , so the only additional source nodes added were those not associated with known labeled blogs. Figure 28(b) shows that their addition, with weight 1, decreased accuracy from 95.4% to below 95%. Increasing the weight of edges $e \in E_{\text{source}}$, including the edges from the labeled blogs, increased accuracy, up to an optimal weight of 50, which yielded accuracy of 96.6%.

Second, we added E_{user} to the original G (without V_{source} and E_{source}). As shown in Figure 28(c), adding the friendship links reduced accuracy. Even though accuracy peaked at weight=10, it was still lower than the previous optimum.

Finally, we added E_{story} to G . As shown in Figure 28(d), accuracy dropped steadily as the weight of $e \in E_{\text{story}}$ increased, and it was always lower than the previous optimum of 95.4% without E_{story} .

Thus, the optimal graph structure $G^* = \langle V^*, E^* \rangle$ has $V^* = V_{\text{story}} \cup V_{\text{user}} \cup V_{\text{source}}$, and $E^* = E_{\text{digg}} \cup E_{\text{source}}$ with the weight of $e \in E_{\text{source}}$ equal to 50. We found similar results for LCGC and ARW algorithms too, where E_{source} improved accuracy, but E_{user} and E_{story} did not.

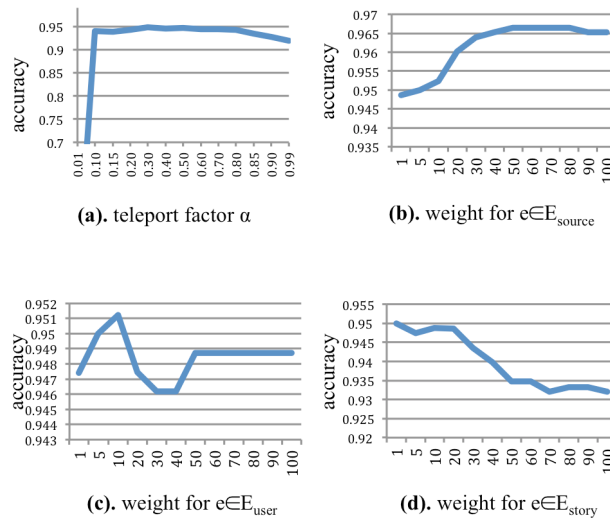


Figure 28. (a) optimal $\alpha = 0.3$ (b) $V_{\text{source}}/E_{\text{source}}$ helpful, with optimal weight = 50 (c) E_{user} not helpful (d) E_{story} not helpful.

Step 4: Semi-Supervised Algorithms Comparison

Next, we compared the performance of the three semi-supervised learning algorithms, using the optimized parameters from step 1 and the optimized L^* and G^* from steps 2 and 3. In addition to the overall accuracy, we also show accuracy for stories and users separately in Table 14.

Table 14. Algorithms comparison

	Accuracy (overall)	Accuracy (stories)	Accuracy (users)
RWR	96.6%	95.4%	99.5%
LCGC	96.9%	95.6%	100%
ARW	97.3%	96.3%	99.6%

5.3.3 Discussion

Overall, the results are quite promising, suggesting that a relatively small number of seed people and stories that are clearly liberal and conservative, together with a large number of people to item votes, can be used to classify, with high precision and recall, the other people and items that are clearly liberal or conservative. The three algorithms all had very high accuracy, with the absorbing random walk performing the best of the three.

Precision was higher for *blue* classifications, but true *reds* had higher recall. We suspect that this is because Digg is quite skewed in favor of liberal stories, but the color distribution in our training data is more balanced.

The liberal/conservative labels of source blogs, together with links from source blogs to the items that appeared in those blogs, proved to be useful inputs to the propagation algorithms. A closer look at the results of the optimized RWR algorithm reveals that 98.5% of stories in L_{blogs} had the same political leaning as their source blogs. We suspect that congruence, together with reasonably high quality classifications of the blogs, made those labels and links useful for classification.

Propagating liberal/conservative labels from blogs through their HTML links to stories, however, decreased overall classification accuracy. Although Adamic and Glance (2005) found that HTML links between blogs that have different ideologies are rare, such links are frequent enough to make propagating labels over those links misleading. According to the RWR algorithm's classification, only 76.5% of HTML links led from blogs to stories that had the same political leaning.

Adding nodes for website domains (including the non-labeled ones) with links to stories that appeared on those domains was helpful for the classification. This implies that most Digg stories posted on the same website indeed share the same political leaning. Classification improved most when the links from sites to stories were given weight equal to fifty individual diggs. We suspect that if we had a dataset with many more diggs per story, the optimal weight for these site-story links might be even higher.

Adding links between declared "friends" decreased classification accuracy. The classification results from our algorithm suggest that only 63.3% of the 755,303 friendship pairs share the same political leaning. One possible explanation is that since Digg is not for political news only, and friendship on Digg is not necessarily based on political preference but perhaps on other non-political factors such as the same hobbies or locations.

Adding links between stories with textual similarities also decreased classification accuracy. One plausible explanation is that stories that have similar topics and content features do not necessarily share the same opinions. For example, "thumbs up to healthcare reform" has similar text to "thumbs down to healthcare reform", but clearly they have opposite political leanings. Another reason was that we didn't have the full text of the stories, which prohibited further optimization on the text similarity links. It could be, however, that more sophisticated text comparisons, perhaps based on topic modeling and sentiment analysis, would provide better inputs to our graph propagation algorithms.

Comparison to SVM

We compared the semi-supervised learning algorithms to the supervised learning algorithm, SVM, which was the most frequently used approach in prior studies on political leaning classification. Our semi-supervised learning algorithms outperformed the SVM algorithm.

We tried three versions of the SVM classifier. All text features were generated using Apache Lucene, and we used the SVM-light¹⁶ application to run the algorithm.

In the first version, we simply generated the uni-gram text features from each story's title and text snippet, and ran SVM. In the second version, we followed the optimization process in (Oh et al, 2009): for the text features, we used the combination of uni-gram, bi-gram and tri-gram, and then used χ^2 feature selection that selected 5,440 out of 3,079,759 features with $p < 0.1$.

The first two versions only worked for stories that have text features. In the third version, we first ran SVD on W to generate 6 major components for both stories and users, and then used them as 6 features the classifier.

The result is in Table 15. SVM with feature selection worked quite well, achieving 92% accuracy for stories. However, it has two limitations. First, after feature selection, SVM was not able to classify 9.8% of the stories that didn't have any selected features. To be able to classify those stories, we would have had to add more features, possibly noisy ones, which would have driven down accuracy. Second, a text classifier was not able to classify the users (except for using features from SVD) that didn't have any text features.

Table 15. Result of SVM

	Accuracy (overall)	Accuracy (stories)	Accuracy (users)
All features	N/A	76.2%	N/A
χ^2 feature	N/A	92.0%	N/A

¹⁶ <http://svmlight.joachims.org/>

selection			
SVD features	81.4%	87.6%	76.0%

Extensions

Online updating. Running the algorithm with full iteration would be too time-consuming in an online setting each time a new digg arrived. For the RWR algorithm, we developed a simple online algorithm to classify stories and users using 1-level propagation of previously-computed RWR scores $F^*_i=(F^*_{iR}, F^*_{iB})$ through the complete set of diggs, including those that had newly arrived:

$$F_j = \sum_i \frac{F^*_i}{\text{degree}(i)}, \text{ where } (i, j) \in E$$

Using the two automatically collected datasets $L_{\text{training}}=L_{\text{blogs}} \boxplus L_{\text{user-reported}}$ to generate the pre-computed F^*_i scores with RWR, we evaluated the 1-level propagation algorithm using the two manually coded datasets for testing, $L_{\text{testing}}=L_{\text{mturk}} \boxplus L_{\text{user-coded}}$. Compared to the first row of Table 16, accuracy of 1-level propagation was only a little lower. We conclude that it would be reasonable to use 1-level propagation online and periodically re-compute the stationary F^*_i scores. We leave it to future work to develop similar approximations of the LCGC and ARW algorithms.

Table 16. Result of 1-level propagation on , with RWR

	Accuracy (overall)	Accuracy (stories)	Accuracy (users)
$L_{\text{training}}=L_{\text{domain}} \boxplus L_{\text{user-reported}}$ $L_{\text{testing}}=L_{\text{mturk}} \boxplus L_{\text{user-coded}}$	95.2%	94.1%	97.7%
1-level propagation	93.8%	92.5%	96.9%

Preliminary study of using gray labels in testing set. So far, we have showed the semi-supervised learning algorithms achieved high accuracy on clearly labeled *red* and *blue* items. Some stories and users, however, do not fit cleanly into either category. In some contexts, either *red* or *blue* labels for ambiguous items would be acceptable. In others, however, it would be better to mark such ambiguous items as *gray*, and classifying them as either *red* or *blue* would be considered erroneous. In that case, excluding *gray* items

from the calculation of error rates, as we have done, would lead to overestimates of the precision of the classifications.

We have conducted some preliminary analysis of how the algorithms would perform if classifications of *gray* items as *red* or *blue* counted as errors. From the 1000 Mechanical Turk stories, we defined the rest of the 653 stories (excluding 40 broken link stories) that were not in L_{mturk} , those without unanimous ratings from turkers, as *gray*. Adding the new *gray* labels to the testing set, we got the optimal threshold parameters as $\theta_R=1.6$ and $\theta_B=1.45$ for RWR, which switched some of the *red* and *blue* classifications to *gray*. Accuracy overall dropped to 72.4%. Accuracy for the clearly labeled *red* and *blue* items dropped to 89.9% with the new threshold parameters.

For comparison, we used two binary SVMs (one classifies *red* vs. *not-red*, the other classifies *blue* vs. *not-blue*) to classify *red* (as *red* and *not-blue*), *blue* (as *blue* and *not-red*), and *gray* (otherwise) using the new testing data. Accuracy was 85.7%, higher than RWR. Note that SVM could not classify 13% of the stories that did not have any selected features, and we simply labeled them as *gray*. This helped SVM because there are many *gray* labels in the testing set.

With a slightly different definition of true *red*, *blue*, and *gray*, the results turned out differently. For Mechanical Turk stories, we defined *red* as any story having $>2/3$ *red* ratings from the turkers, *blue* as having $>2/3$ *blue* ratings, and *gray* for the rest, which resulted in 490 *blue*, 203 *red*, and 267 *gray*. Using these new data in the testing set for RWR, we got the optimal $\theta_R=1.15$, $\theta_B=1.1$, and overall accuracy 74.8%. For clearly labeled *red* and *blue* items, we still have 95.6% accuracy using the new threshold parameters. The SVM algorithm got accuracy 73.9%, now slightly lower than RWR.

The lack of a principled approach to mapping multiple ratings into *red*, *gray* and *blue* prevents us from drawing reliable conclusion on whether RWR or SVM is a better algorithm when classifying items as *gray*, which is one major limitation of the original study. I will discuss this in more detail in the follow-up study.

5.4 Follow-up Study

The main limitation of the original study was that we only used clearly red and blue items in the labeled ground truth dataset. Low agreement items were entirely excluded. The limitation was due to the lack of a principled approach to defining “gray” items when human raters don’t agree on the correct label of items. As a result, classifiers would only classify items as either red or blue but not gray in order to be evaluated favorably against such a ground truth with no gray items. Although we learned that LabelPropagator worked quite well for clearly red and blue items, we still didn’t know how well it could work for gray items.

Furthermore, even though we had 6 ratings per item in the original study and we only used those items with consensus agreement from the raters, the labeled dataset was still not reliable: 13.7% of those items with 6 out of 6 consensus ratings in the original study would get different cost-minimizing labels when labeled again by 20 different coders. One might challenge the reliability of the results from the original study based on the unreliable labeled dataset.

Therefore, the follow-up study tries to answer two questions. First, how does LabelPropagator perform with gray items? Second, does LabelPropagator work better than alternative classifiers? The primary task of the follow-up study is to re-evaluate LabelPropagator using the labeled dataset obtained from chapter 3 in order to study the performance of LabelPropagator when tuned to produce gray labels as well as red and blue. We also want to be sure that comparing the performance of LabelPropagator to that of alternative classifiers lead to reliable result by following the suggestions from chapter 4, despite the presence of sample errors in the labeled dataset.

The rest of the section is organized as follows. Section 5.4.1 summarizes the datasets involved in the follow-up study. Section 5.4.2 discusses the evaluation of LabelPropagator using the labeled dataset obtained from chapter 3, and section 5.4.3 summarizes the main findings and insights of the follow-up study.

5.4.1 Datasets

Throughout the course of the thesis, I have done three separate rounds of political leaning annotations on Amazon Mechanical Turk. I will summarize them in chronological order below.

The first round was done in 2010 in order to collect labeled articles as ground truth to study LabelPropagator. This labeled dataset was discussed in detail in section 5.3.1. To better distinguish this dataset from the other two rounds, I'd like to rename the notations used in section 5.3.1 – V_{story} , V_{user} , E_{digg} , and L_{mturk} – as $V_{story-2010}$, $V_{user-2010}$, $E_{digg-2010}$ and $L_{mturk-2010}$. In the original study, L_{mturk} had only 307 clearly red and blue stories with 6 out of 6 consensus ratings from raters. Here, let $L_{mturk-2010}$ denote all 960 labeled stories that received 6 valid ratings, regardless of whether the ratings were consistent or not. Let $L_{original}$ then be the labeled dataset used in the original study that only consists of the 307 clearly red and blue stories.

The second round of data collection was done in 2011 between May 30th and November 30th. Each day, an average of 200-300 popular political stories were crawled from Digg.com and the most popular ones on the Digg.com homepage between 2011-07-28 and 2011-11-28 were immediately posted to AMT to get political leaning annotations with 4 labels per story. I'd like to denote the dataset collected in this round as $V_{story-2011}$, $V_{user-2011}$, $E_{digg-2011}$, and $L_{mturk-2011}$, where $|V_{story-2011}|=29,313$, $|V_{user-2011}|=19,614$, $|E_{digg-2011}|=1,711,449$, and $|L_{mturk-2011}|=14,568$. To be consistent with the original study, I have removed stories that received less than 10 diggs, and removed users that did not make at least 5 diggs.

The last round of data collection was done in 2012, and was discussed in detail in chapter 3. This round did not crawl any new stories from Digg.com. It only had AMT workers annotate again the same stories sampled from $L_{mturk-2010}$ and $L_{mturk-2011}$, but this time with 20 labels per story. The process was discussed in chapter 3 and will not get repeated here. To use consistent notations, I will denote the labeled dataset

obtained in this round as $L_{mturk-2012}$ (which was denoted as \hat{H}_i^* , $\forall i \in I$ in chapter 3 as the semi-true ground truth dataset). Recall that $|L_{mturk-2012}|=1911$. Table 17 lists the differences of the three labeled datasets: $L_{mturk-2010}$, $L_{mturk-2011}$ and $L_{mturk-2012}$.

Table 17. AMT annotations in 2010, 2011, and 2012

	$L_{mturk-2010}$	$L_{mturk-2011}$	$L_{mturk-2012}$
Number of labels per article (m)	6	4	20
Number of valid articles (n)	960	14,568	1,911
Articles selected from:	$V_{story-2010}$	$V_{story-2011}$	$V_{story-2010}$ and $V_{story-2011}$ (only those in $L_{mturk-2010}$ and $L_{mturk-2011}$)
Articles pre-processed (removing HTML and source)	No	No	Yes
Political leaning labels	Liberal, Conservative, Neither, Both, Broken	Liberal, Conservative, Neither, Both, Broken	Red, Blue Gray
Price per label	\$0.03	\$0.01~0.05	\$0.05
Bonus	Yes	Yes	Yes
Quality control	Qualification test, threats, captchas	Qualification test, threats	Qualification test
Annotated in	Jul. ~ Aug., 2010	Jul. ~ Nov., 2011	Oct., 2012
Sample stratification	No	No	Yes

I'd like to make two clarification points. First, $V_{user-2010}$ and $V_{user-2011}$ overlap with each other: 20.9% of users in $V_{user-2011}$ are also in $V_{user-2010}$. Therefore, labels in the original graph (constructed from $V_{story-2010}$, $V_{user-2010}$, and $E_{digg-2010}$) were able to propagate to new stories and users in 2011 through those overlapping users. Similarly, labels on the new stories are able to propagate back to nodes in the original graph. In other words, I can append $V_{story-2011}$, $V_{user-2011}$ and $E_{digg-2011}$ to the original graph and construct a new connected graph for LabelPropagator.

Second, stories labeled in $L_{mturk-2012}$ overlap with stories labeled in $L_{mturk-2010}$ and $L_{mturk-2011}$, but the labels are different. That is, each story in $L_{mturk-2012}$ is (or is intended to be) labeled for the second time in 2012 although it has already been labeled in either 2010 or 2011. I intended to get 1000 stories from $V_{story-2010}$ labeled in both 2010 and 2012, but due to various technical problems (e.g., broken links, network timeout, etc), only 960 and 744 of them got labeled in $L_{mturk-2010}$ and $L_{mturk-2012}$ respectively. The overlap consists of 733 stories, and I'd like to denote L_{new} as the overlap in $L_{mturk-2012}$ with 20 labels per story from 2012. All the 1,165 stories from $V_{story-2011}$ that are labeled in $L_{mturk-2012}$ are also labeled in $L_{mturk-2011}$. I'd like to denote $L_{heldout}$ as the rest of the 13,421 labeled stories in $L_{mturk-2011}$ but not in $L_{mturk-2012}$. The relationships among these datasets are illustrated in Figure 29.

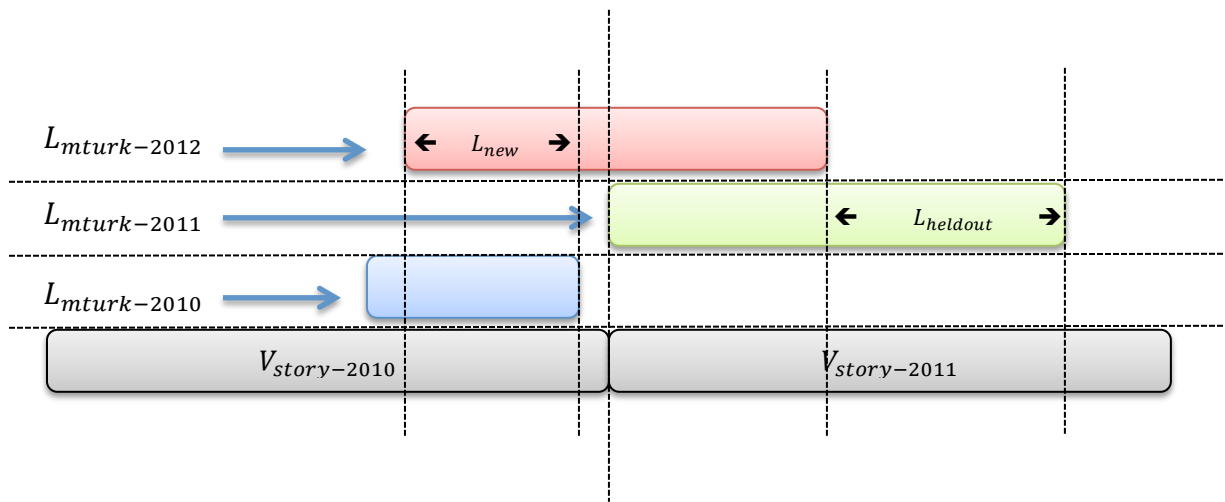


Figure 29. Labeled datasets overlap

5.4.2 Evaluation

I'd like to make a few clarification points. Firstly, the original study found that ARW has higher accuracy than the other two variations of LabelPropagator (RWR and LCGC), and that a story's sources are useful for propagation while text similarities, users' friendships and blog reference links are not. In this follow-up study, I will use ARW and stories' sources for LabelPropagator without studying again which variation of LabelPropagator (e.g., RWR, LCGC or ARW) works better or which additional dataset (e.g., stories sources,

blog reference links, text similarities or users' friendship) is useful for propagation, although future work should address whether these results still hold when adding gray items to the ground truth dataset for training and evaluation.

Secondly, in the follow-up study, I will treat the underlying ground truth of stories' political leaning as distributions instead of labels. However, I will use the evaluation scheme Ψ_l instead of Ψ_H to evaluate the classifiers, as suggested by chapter 4. That is, I will first map the distributions (\hat{H}_i) into their cost-minimizing labels (\hat{l}_i) by using the quadratic partitioning and then use the cost-minimizing labels to evaluate classifiers, instead of directly evaluate classifiers with the distributions. Using the distribution as ground truth model and Ψ_l , I am finally able to define gray items in a principled way (in terms of cost-minimizing) and solve this major problem in the original study. Suppose we have 6 labels per story, then $\langle 3 \text{ red}, 3 \text{ blue} \rangle$ is gray, $\langle 3 \text{ red}, 3 \text{ gray} \rangle$ is gray, $\langle 4 \text{ red}, 2 \text{ gray} \rangle$ is red, $\langle 4 \text{ red}, 1 \text{ gray}, 1 \text{ blue} \rangle$ is gray, $\langle 5 \text{ red}, 1 \text{ blue} \rangle$ is red, and so on. The labeled dataset might still be unreliable due to sample errors, but we know from chapter 4 that classifier evaluation is still reliable if we have many items in the evaluation dataset, which we do. Also, Ψ_l uses Equation 13 to compute classifier accuracy, which is discounted according to the quadratic cost function. As I have explained in chapter 4, I will use a decimal number instead of a percentage to denote the accuracy score, and it is not comparable to the original study.

Thirdly, I will not re-invent LabelPropagator to take advantage of the distribution as ground truth model for *training* purpose. Instead, I will first map the distributions into their cost-minimizing labels, and then use the labels to *train* LabelPropagator and other classifiers, just like I will use the labels to *evaluate* the classifiers too. Although I will use the cost-minimizing labels instead of distributions for both training and evaluation purposes, the process still assumes the distribution as ground truth model, which is quite different from the traditional label as ground truth model. I have discussed this in both chapter 2 and chapter 3 and will not repeat them here. I will show more evidence later to support this argument.

Step 1: Ground Truth – Old versus New

This step is to evaluate LabelPropagator using the new ground truth dataset L_{new} , and to contrast the result to evaluation using the old ground truth dataset $L_{original}$ in the original study. In this step, I will use the same graph structure as in the original study, constructed from $V_{story-2010}$, $V_{user-2010}$, $E_{digg-2010}$. But I will use different labels (L_{new} instead of $L_{original}$) to propagate in the graph as well as to evaluate the classifier.

Similar to the original study, I will still use 10-fold cross validation for classifier training and testing: nine tenth of L_{new} will be used for training, and the rest one tenth for testing. In the training process, even though the nine tenth labeled items include gray items, I will *not* use them for propagation. I will explain it in the “future work” section. I will use gray items in the one tenth for testing, which incentivizes classifiers to classify items into gray in order to achieve higher accuracy. As I have discussed earlier, the original study did not use gray items at all, and therefore the classifiers had no reasons to classify items into gray even though they were capable to do so.

Recall that $L_{original}$ only includes 307 clearly red and blue stories with 6 out of 6 consensus ratings from raters. 25% of them were not labeled again in L_{new} due to technical problems explained earlier. But 18% of the items in $L_{original}$ that got labeled again in L_{new} received different cost-minimizing labels in L_{new} . Also, L_{new} includes other items that were not originally included in $L_{original}$ because they did not receive consensus ratings in 2010. But now we are able to map them into red, gray and blue using the principled cost-minimizing approach.

Due to these changes in the ground truth dataset, the optimized parameters of LabelPropagator change from $\theta_R = \theta_B = 1.0$ to $\theta_R = 1.004$ and $\theta_B = 1.008$ and accuracy changes from 96.2% to 0.773¹⁷. A total of 13.9% of the stories in the classification results get classified differently. Table 18 shows the confusion matrix of

¹⁷ The two accuracy scores, 96.2% and 0.773, were computed under Ψ_c and Ψ_l respectively, which followed different equations. Thus the two scores were not directly comparable.

the classification result changes. Note that many items that were originally classified as either red or blue are now classified as gray.

Table 18. Confusion matrix: classification outcome changes

		L_{new}		
		r	g	b
$L_{original}$	r	-	3.2%	0.1%
	g	0%	-	0%
	b	0.4%	10.2%	-

Total changes: 13.9%

Step 2: Evaluation against Held-out Data

This step is to compare the performance of LabelPropagator to alternative classifiers. I will use all items in $L_{mturk-2012}$ for training purpose and all items in $L_{heldout}$ as held-out dataset for testing. Recall that $L_{heldout}$ has about 14 thousand labeled items with 4 labels per item. According to chapter 4, classifiers' ranking would be quite reliable even though the cost-minimizing label of each individual item in $L_{heldout}$ is not reliable.

Now I expand the original graph to include new stories, users and Diggs from $V_{story-2011}$, $V_{user-2011}$ and $E_{digg-2011}$. As I have explained earlier, the new graph is still connected because of the overlapping users in $V_{user-2010}$ and $V_{user-2011}$. Using the new graph and ground truth data $L_{mturk-2012}$ to propagate labels, LabelPropagator has the new optimized parameters $\theta_R = 1.038$ and $\theta_B = 1.018$. Accuracy is shown in Table 19.

There are three alternative classifiers as comparison baselines. The first alternative classifier, SVM, has been described in the original study and I will not repeat it here. Another alternative, SourceClassifier, simply takes the source of any story, and classifies the story as having the same political leaning as its source. It works on any item as long as the bias of the source is known, or else the item is labeled as gray. The political leaning of the sources used in SourceClassifier is from a separate ongoing project¹⁸,

¹⁸ This project is in its early phase led by Munson(smunson@um.edu). The political leaning of sources are aggregated over multiple sources such as those described in the original study. Source code is available upon request.

which uses a set of heuristics to infer the political leaning of sources. The last alternative classifier, RandomClassifier, simply classifies items uniformly at random into red, gray, and blue. This is the worst classifier, and establishes the lower bound. Accuracy scores of alternative classifiers are listed in Table 19.

Table 19. Accuracy evaluated against $L_{heldout}$ using Ψ_l

	Accuracy
LabelPropagator	0.679
SVM	0.384
SourceClassifier	0.543
RandomClassifier	0.210*

(Note: accuracy marked with * are averaged over 1000 repeated trials)

Table 20. Confusion matrix: classification results from different classifiers

		LabelPropagator			SVM			SourceClassifier		
		<i>r</i>	<i>g</i>	<i>b</i>	<i>r</i>	<i>g</i>	<i>b</i>	<i>r</i>	<i>g</i>	<i>b</i>
$L_{heldout}$	<i>r</i>	11.7%	5.3%	0.04%	4.1%	12.4%	0.59%	6.7%	9.6%	0.68%
	<i>g</i>	7.4%	31.2%	6.7%	8.2%	34.6%	2.4%	4.2%	31.1%	10.0%
	<i>b</i>	0.25%	13.5%	23.9%	3.9%	28.8%	4.9%	0.10%	19.8%	17.7%

Step 3: Comparison to Human Coders

This step is to compare LabelPropagator against human coders as the “human classifier”, such as the \mathbb{C}^{coder} family described in chapter 4. In this step, I will use all items in $L_{mturk-2012}$ as held-out data for testing purpose instead of for training purpose. For LabelPropagator, the initial labels to propagate in the expanded graph are only from $L_{user-reported}$, $L_{user-coded}$, and L_{blogs} that were introduced in the original study. With this setting, accuracy of LabelPropagator is listed in Table 21.

I will denote the “human classifier” as “*k*-coder”, which randomly sub-samples *k* labels from 20 labels per item for each item in $L_{mturk-2012}$ and then uses quadratic partitioning to map the *k* labels into an item’s cost-minimizing label as the classification result. Accuracy of *k*-coder is listed in Table 21.

Table 21. Accuracy evaluated against $L_{mturk-2012}$ using Ψ_l

	Accuracy
LabelPropagator	0.644
1-coder	0.602*
2-coders	0.736*

(Note: accuracy marked with * are averaged over 1000 repeated trials)

5.4.3 Discussion

The political leaning classification problem is subjective. I have argued in chapters 2 and 3 that the labeled ground truth dataset for such a problem is unreliable. Here, “unreliable” means that many factors (such as who the raters are) will affect the labels of items, and therefore repeatedly labeling items will lead to different results. The first step of the follow-up study shows that using different ways to define ground truth on the same set of items will affect classifier optimization and evaluation. Decisions regarding whether to add gray items, whether to increase the number of labels per item to decrease sample errors, and which partitioning to choose in order to map distributions into labels all have a considerable impact on classifier evaluation and optimization. Different decisions will change LabelPropagator’s optimized parameters (θ_R and θ_B), cause classification results to vary from 13.9%, and changes accuracy from 96.3% to 0.773 (step 1). In short, different ways to define the ground truth dataset do matter. It is imprudent to define ground truth using any arbitrary approach and expect to draw meaningful conclusions about the classifiers without considering all factors that might affect the ground truth dataset.

Using the new ground truth dataset causes accuracy to drop from 96.3% to 0.773. I’d like to point out that the original study and the follow-up study use different evaluation scheme to compute classifier accuracy, and thus the two accuracy scores are not directly comparable. However, in the follow-up study with gray items, LabelPropagator and other alternatives indeed made more incorrect classifications (see Table 20). Studies (e.g., Bishop 2006) have consistently shown that multi-class classification problems are harder than binary classification problems. Although accuracy was usually

higher than 95% for LabelPropagator and above 90% for alternative classifiers such as SVM in the original study without gray items, adding gray items makes the classification problem much harder, and not surprisingly, accuracy drops considerably for all classifiers. Almost all previous studies on political leaning classification only used items that fit cleanly into the red or blue category, and would have suffered the same problem too, had they also included the more difficult items in their evaluations.

Even though the performance of LabelPropagator degrades after using the gray items, it still outperforms SVM and SourceClassifier (step 2). Classification results from LabelPropagator are even better than the output of 1 human coder as the “human classifier” (step 3). According to chapter 4, the conclusion that LabelPropagator is better than the alternative classifiers is reliable because the classifiers are evaluated against powerful ground truth datasets containing many items and many labels per item. We are confident that the better accuracy of LabelPropagator is not simply a misleading case of comparing the results of LabelPropagator with an unreliable labeled dataset.

Furthermore, LabelPropagator has other advantages besides higher accuracy. It doesn't rely on any textual information of the items, and therefore it can classify images, videos, short messages such as tweets, and people, as long as they are on the propagation graph. Another advantage is that it does not require a lot of expensive training data, which is usually a requirement for supervised learning algorithms such as SVM. These benefits, together with the higher accuracy, make LabelPropagator one of the better choices for political leaning classification.

SVM does not work very well with gray items: accuracy drops from 92.0% in the original study down to 0.384 (step 2). Most previous studies about political leaning classification using SVM have shown higher accuracy (e.g., Oh et al 2009). But I'd like to point out that those studies did not take into account gray items, which would make the classification problem much harder. Also, Table 20 shows that SVM has generated many false gray classifications, indicating the lack of training examples necessary to classify the large number of articles with many new text features in the held-out dataset. I suspect that

many previous studies might have suffered the problem of overfitting due to the lack of labeled items for training and evaluation, thus resulting in higher accuracy. On the other hand, admittedly, SVM in this section was not fully optimized. For example, if SVM could simply take an article's source as a feature during the training process, it should perform at least as well as SourceClassifier. Also, I have used two binary SVM classifiers to classify items into red, blue and gray; a more sophisticated multi-class SVM classifier could have performed better. Further optimization of SVM will be left to future work.

SourceClassifier does not work very well either. Here I will give a few examples to illustrate why. There were 107 NYTimes articles in the ground truth dataset ($L_{mturk-2012}$), but only 36% of them were labeled as blue (using the quadratic partitioning) and the other 64% of stories were labeled as gray. Similarly, there were 23 FoxNews stories, but only 48% of them were labeled as red and the other 52% were labeled as gray. Recall that each article in the ground truth dataset was labeled by 20 coders without knowing where the article came from. Simply by reading the content of the articles, many people did not think the articles shared the exact same political leaning as their obviously biased sources. In short, although it is intuitive to label items according to their sources, it is not optimal to do so.

It can be seen that these alternative classifiers are complementary to each other. In fact, the political leaning of articles' sources used by SourceClassifier can be and has already been used by LabelPropagator to propagate to other nodes on the graph. Also, we can treat an article's source as a feature to train SVM. I will talk more about these possibilities in the "future work" section as well as in the last chapter.

5.5 Conclusions and Future Work

This chapter proposes the LabelPropagator algorithm that classifies articles and people's political leaning by propagating political leaning of known articles and users to the target nodes. Overall, the results are quite promising, suggesting that a relatively small number of seed people and stories that are clearly liberal and conservative, together

with a large number of people to item votes, can be used to classify, with high precision and recall, the other people and items that are clearly liberal or conservative. The three variations of LabelPropagator all had quite high accuracy when evaluated without gray items, with the absorbing random walk performing the best of the three.

This chapter also found that the liberal/conservative labels of source blogs, together with links from source blogs to the items that appeared in those blogs, proved to be useful input to the propagation algorithms. Propagating liberal/conservative labels from blogs through their HTML links to stories, however, decreased overall classification accuracy. Adding nodes for website domains (including the non-labeled ones) with links to stories that appeared on those domains was helpful for the classification. Adding links between declared “friends” decreased classification accuracy. Adding links between stories with textual similarities also decreased classification accuracy.

In the follow-up study, the underlying ground truth of articles’ political leaning was treated as distributions instead of labels. It allowed us to use cost-minimizing labels to define gray items in a principled way, and then to evaluate LabelPropagator to see how it classifies items into all three categories of red, gray and blue. Adding the gray items to the ground truth dataset and using the more reliable ground truth dataset with more labels per item for the same items did change LabelPropagator’s optimized parameters and classification results, and accuracy dropped from 96.3% to 0.773. Even so, LabelPropagator still has higher accuracy than the other alternative classifiers, which is a reliable conclusion as suggested by chapter 4.

The primary challenge of future work is to improve LabelPropagator. Obviously, LabelPropagator depends on the graph to propagate known labels. As the thesis is written, Digg.com has been officially closed and re-built, and the public API to crawl diggs data is not available anymore. In order for LabelPropagator to work, it has to utilize other “voting” data from people to items, such as Twitter mentions, Facebook “I like”, or Google “+1”. A preliminary study on using the LabelPropagator algorithm to

classify the political leaning of Korean twitter messages has shown positive results. Future work should continue this line of study.

Another improvement to LabelPropagator is to use gray items for training. Currently the algorithm only propagates red and blue labels, and uses threshold parameters (θ_R and θ_B) to determine whether an item should be classified as red, blue or gray. Propagating gray labels could increase classification accuracy, but at the same time it could be detrimental because it is not necessarily clear that gray people are likely to vote for gray items and vice versa, which differs from the algorithm's intuition that blue/red people are likely to vote for blue/red items and vice versa. Nevertheless, future research should study this possible improvement.

Another interesting direction for future work is to try to understand and characterize the properties of datasets for which the different versions of the propagation algorithms – RWR, LCGC, and AWR – will perform better or worse. One promising possibility is to take an axiomatic approach based on identifying how the classification should change in response to changes in the graph structure.

We note that the propagation algorithms gained accuracy with the addition of datasets such as domain source links, where the linked items tend to have high correlation in their labels, but lost accuracy with the addition of datasets such as friendship links where the correlation was lower. We therefore discarded those datasets. Clearly, this is not optimal, since even a positive correlation much less than 1 in principle provides some information. Future research should find ways to make use of these noisy datasets rather than discarding them entirely.

Finally, LabelPropagator is limited in that it requires interactions between stories and users (i.e., diggs). For unpopular articles not covered in social news sites such as Digg, our algorithm will not be able to classify them. However, its advantage is that it does not require much training data. This is complementary to SVM (see Table 22), which requires lots of training data, but does not require user-story votes. Therefore, one idea is to use the propagation algorithms to generate many labeled data with high accuracy,

feed this data to train SVM, and then use the well-trained SVM model to classify any textual items. The process is illustrated in Figure 30.

Table 22. Comparison between LabelPropagator and text analysis classifiers

Classifier	Pros	Cons
Text analysis: SVM	Doesn't require a graph; high coverage	Expensive (requires large amount of training data); mediocre accuracy
LabelPropagator	Inexpensive; high accuracy	Requires a graph; limited coverage

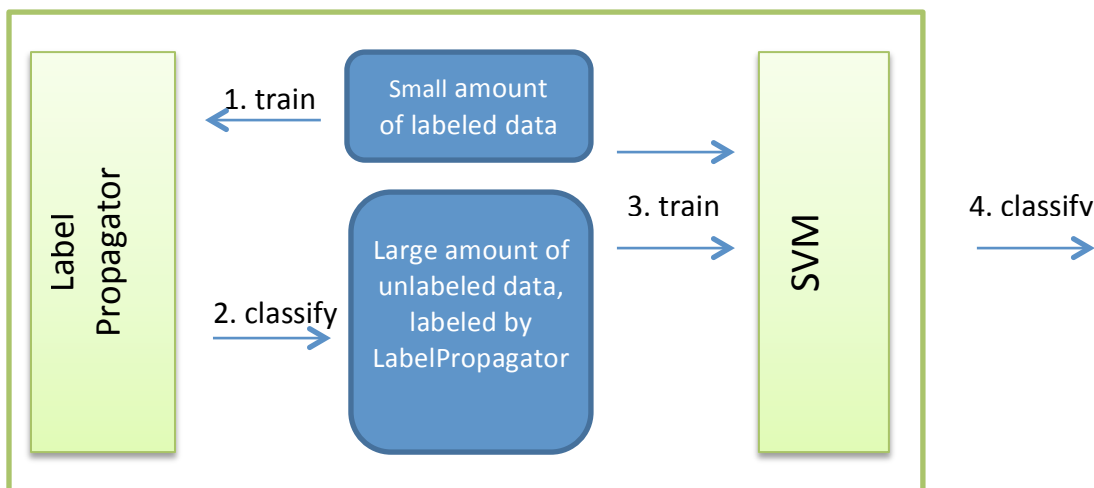


Figure 30. Co-SVM illustration

This type of approach was invented by Blum and Mitchell (1998) as the “co-training” approach. The version of LabelPropagator/SVM co-training as I have proposed and illustrated in Figure 30 is very much simplified, which only iterates once by feeding the output of LabelPropagator into SVM. Preliminary study shows that the simplified co-training classifier has achieved 0.534 accuracy when evaluated against the held-out dataset ($L_{heldout}$). It is not as good as LabelPropagator, but still is an improvement over SVM. Future work should explore this promising direction to further improve both LabelPropagator and SVM through co-training, perhaps with more iteration between them.

Chapter 6. Closing Remarks

6.1 Summary

This thesis started with the motivation to develop an algorithm able to classify a large number of political articles into conservative, liberal and other. The challenge is not only about classifier invention, but also about how to evaluate the invented classifier with unreliable ground truth. Therefore, this thesis has two main themes: the “ground truth” theme and the “classifier” theme. The “ground truth” theme studies how to define the “ground truth” of articles’ political leaning, how to elicit annotations from human coders, and how to evaluate classifiers with inaccurate labeled data. The “classifier” theme aims to develop a better political leaning classifier that does not rely primarily on text analysis. Next, I will summarize the main findings of both themes respectively.

The “ground truth” theme is mainly studied in chapters 2, 3 and 4. I have argued that there exists a set of subjective classification problems such as the political leaning classification problem where people don’t agree upon the correct label of items due to the lack of objective classification rules. This is in contrast with the traditional objective classification problems commonly found in the machine learning literature, where people agree on the correct label of most items according to objective rules for categorizing items, resulting in a reliable labeled dataset with few or no errors. For a subjective classification problem, I have argued for the use of distributions as the underlying ground truth model instead of labels. Under this perspective, disagreements among coders are not treated as human errors to be eliminated, but rather as useful information reflecting the distribution of people’s subjective opinions.

Empirical data shows that more than half of the labeled items are in the middle of the simplex, where more than 5 out of the 20 labels received for an item do not agree with the majority label of that item. Many of the low agreement items, that is, 19.4% of the total items, are in the partitioning disputed region, where different cost functions (or partitioning of the simplex) would map them into different cost-minimizing labels. Choosing a non-optimal cost function (i.e., the default 0-1 instead of the quadratic for political leaning classification) that does not correctly model users' true cost would result in labeling those items incorrectly: showing the incorrectly labeled items to the users would incur more cost than the optimal.

For a subjective classification problem, one can no longer assume the labels from a small number of coders are reliable due to large sample errors: 24.8% of the items, when labeled again, changed their cost-minimizing labels. I have found, however, that even though individual items in the labeled dataset are unreliable, we may still reliably evaluate and rank classifiers with distribution as ground truth, as long as we have a large number of items in the evaluation dataset. Treating a subjective classification problem as if it were objective runs the risk of incorrectly ranking the classifiers.

I have proposed two classifier evaluation schemes with the distribution as ground truth model using the concept of cost-minimization: one scheme directly uses distributions for evaluation and the other maps distributions into labels first and then evaluates. Using computer simulation, I have demonstrated that both evaluation schemes can obtain high classifier discrimination power with a large number of labeled items for evaluation. The optimal setting is to obtain one label per item with many items, and it is usually advisable to map items' distributions into their cost-minimizing labels first and then evaluate with the labels. According to the simulation, we need roughly 3000 items for evaluation in order to get good discrimination power for a subjective classification problem. For an objective classification problem, where labels for each item are drawn from tighter distributions, we only need roughly 1000 items.

The “classifier” theme is mainly studied in chapter 5. I have proposed the LabelPropagator algorithm that classifies the political leaning of articles and people by propagating the political leaning of known articles and users to the target nodes. There is a temptation to leave unclear items out of evaluation sets, but that greatly simplifies the task for classifiers, allowing them to get away with classifying all the gray items as red or blue without penalty. When classifiers are assessed based on their correct labeling of gray as well as red and blue, it is much harder to achieve high accuracy. Nevertheless, I have shown that LabelPropagator achieves higher accuracy than the other alternative classifiers, with and without gray items, suggesting that a relatively small number of labeled people and stories, together with a large number of people to item votes, can be used to classify the other people and items.

I have also found that the liberal/conservative labels of source blogs, together with links from source blogs to the items that appeared in those blogs, proved to be useful input to the propagation algorithms. Propagating liberal/conservative labels from blogs through their HTML links to stories, however, decreased overall classification accuracy. Adding nodes for website domains (including the non-labeled ones) with links to stories that appeared on those domains was helpful for the classification. Adding links between declared “friends”, however, decreased classification accuracy. Similarly, adding links between stories with textual similarities also decreased classification accuracy.

6.2 Future Work

I have discussed future work specific to each chapter in that chapter. In this section, I will discuss future work not covered in the previous chapters.

Distributions as Ground Truth for Training

In terms of using the distribution as ground truth model to design and evaluate classifiers, this thesis essentially proposes to map distributions into labels first, and then use the mapped labels to train and evaluate classifiers that also classify items into labels. Another direction, which is not discussed in the thesis, is to directly train classifiers with

distributions and classify items into distributions over labels as well. Then, if needed, we could do a separate step of mapping the classification results as distributions into labels in order to show the labels to end-users. Future work should fully compare these two different approaches of using distributions as ground truth.

A few previous studies have taken the approach of training and evaluating classifiers directly with distributions (e.g., Lugosi 1992, Cook and Stefanski 1994, Friedman 1997, Kuchenhoff et al 2006, Rogers et al 2009, Dekel and Shamir 2009b, Hopkins and King 2010). Another line of work, which is sometimes dubbed as the “multiple raters classification problems”, is to train classifiers with multiple labels from raters, and then classify them into discrete categories (e.g., Yan et al 2010a, 2010b; Raykar et al 2009; Zhou, Platt, Basu, and Mao 2012).

These prior works share a common limitation in that all of them assume that the labeled ground truth dataset for evaluation is reliable, which, as I have argued earlier, is not true due to non-negligible sample errors with a small number of raters. Therefore, their conclusions about the classifiers could be misleading due to evaluation with unreliable ground truth datasets. Future work could re-evaluate these existing works using the evaluation scheme discussed in this thesis to determine if their conclusions still hold.

I have shown that it is optimal to elicit annotations using one label per item for classifier evaluation purpose. But Sheng et al (2008) argued that it might be optimal for classifier training purposes to obtain multiple labels for the same item. In other words, the optimal number of labels per item for training and evaluation could be different. It is fine as long as training data and evaluation data are kept separate. But it does create a problem for cross-validation where each labeled item is used for both training and evaluation. Future work should address how to optimally obtain labeled data for both training and evaluation instead of obtaining training and evaluation datasets separately.

Finally, one limitation of LabelPropagator is that it doesn't use labeled items' distributions in the training process but simply maps distributions into labels first and then propagate the labels, resulting in information loss. Future work should study how

to directly propagate distributions (or annotations from multiple coders) to better classify the items.

Ensemble Method for Political Leaning Classification

There are a few algorithms that have or could have been devised as political leaning classifiers, listed as follows:

- Text mining based supervised learning algorithms
 - SVM and its variations (N-gram, feature selection)
 - Naïve Bayesian and its variations (N-gram and feature selection)
 - Neural networks
 - Topic models and generative models
 - Perceptron, logistic regression, and linear regression
 - Sentiment analysis (Jiang and Argamon, 2008)
 - Text analysis with memes and shingles
- Unsupervised learning algorithms
 - SVD (Baio, 2008)
 - K-means clustering
- Semi-supervised algorithms, graph-based algorithms
 - LabelPropagator: RWR, LCGC, ARW
 - Community finding algorithms
- Miscellaneous
 - Links from source
 - Predictions from comments (Park et al, 2012)
 - Google search co-occurrence (Efron, 2004)
 - Human “classifiers”

Schapire (1990) showed, analytically, that combining multiple weak learners would produce a strong learner that outperforms each of its components. Here, I will use “ensemble method” as an umbrella term for such approaches that combine different classifiers into a stronger classifier. There are many such ensemble methods developed and used successfully in practice. For example, Freund and Schapire (1995) proposed the AdaBoost algorithm. Sill, Takacs, Mackey, and Lin (2009) proposed the “feature-weighted linear stacking” approach to ensemble recommender algorithms using linear regression. Bell & Koren (2007) discussed their linear regression approach to combine 107 recommendation algorithms to form the Netflix winner algorithm. A few review

papers (Polikar 2006; Dietterich 2000; Jahrer, Toscher, and Legenstein 2011) have studied various ensemble methods, listed as follows:

- Boosting, AdaBoost
- Bagging
- Stack generalization
- Pasting from small votes
- Logistic regression
- Gradient boosting decision trees (GBDT)
- Bootstrap
- Mixture of experts (e.g., Dawid 1978)

Future work could study how to use the ensemble method to combine different political leaning classifiers in order to achieve better classification accuracy.

Application

As I have introduced in the first chapter, the motivation of this thesis is to be able to classify the political leaning of a large number of articles in order to feed into other research or applications that need the classification results. It will have tremendous practical value to put everything together from this thesis and build the state-of-the-art political leaning classification system. Ideally, the system will run in a pipeline that crawls political articles, classifies them into red/blue/gray categories, uses human assessments to increase classification accuracy, provides classification results to the wide world through web services, and automatically optimizes the classification parameters as the system evolves over time.

At a high level, the classification system will have the following features.

- Classification accuracy will increase over time after accumulating more labeled data as well as more unlabeled data (for propagation).
- Parameters can automatically adjust to optimize classification results.
- It can classify the political leaning of any English news articles or snippets.
- The classification results are available to the world via web services.

Below is a list of components to be included in the system.

- The *crawler* component will crawl political news articles from all over the web, clean them, and save them to the database. The component will have a plugin design, so that new crawlers can be easily plugged into the system.
- The *ground truth* component serves as both training and testing dataset for the classifiers. Using this dataset, the system could automatically re-adjust parameter settings to optimize for classification outputs. The dataset will grow in size over time as the system acquires more labels from human coders.
- The *human* component is based on the SignalElicitor/PredictionElicitor discussed in Chapter 3. It serves as both a real-time classifier for difficult articles that the automated classifiers are not able to handle, and also as a way to obtain more labeled data.
- The *ensemble of classifiers* component uses an ensemble algorithm to combine results from multiple political leaning classifiers, including LabelPropagator, PredictionElicitor, and other text analysis tools such as Naïve Bayes. The final output should be more accuracy than any of the individual classifiers.
- The *parameter optimization* component works with both the ground truth component and the ensemble of classifiers component to periodically update the configurations of the system for better outputs.
- The *web service* component exposes the political leaning data to the outside world through REST, XML-RPC, SOAP or other web services.

6.3 Broader Impact

The academic contribution of this thesis is two-fold, which corresponds to the two themes. In terms of the “ground truth” theme, the thesis proposes the “distribution as ground truth” model and gives clear instructions on how to elicit labels from human coders and how to evaluate classifiers even though not all coders agree on all items. The findings can apply to a broad range of subjective classification problems beyond just political leaning classification. It is also the author’s hope that this thesis can serve as a first step towards a systematic guidance to machine learning application designers and practitioners on how to obtain labeled dataset properly.

In terms of the “classifier” theme, the thesis proposes the LabelPropagator algorithm that has higher accuracy compared to traditional text analysis approaches. Using this classifier, researchers can obtain good quality political leaning classification results to do other studies that require such data, such as those discussed in section 1.1.

In addition to academic contribution, this thesis could give rise to many real world applications. For example:

- Search engines like Google can use the classification as important metadata to allow users to search political articles within particular political categories.
- Major media like the New York Times can monitor the political leaning of their content and add or remove articles when certain political views are under- or over-represented.
- National online forums can use the technology to monitor online discussion threads and make sure they represent balanced opinions to attract readers from both sides.
- Political campaigns can promote popular articles from both sides (selected from our labeled dataset) to elicit bi-partisan support.
- Website designers can use the classification to highlight the political leaning of articles for better user experience.
- News consumers can receive feedback about the aggregated political leaning of their previous reading history, and adjust the articles they want to read to avoid reading only one side of the story.
- The government can monitor the level of polarization and fragmentation on mass media and make policies to reduce it and enhance democratic functions.

To sum up, this thesis will make a positive impact upon academic research, news consumers, media publishers, the government, and the broader culture.

References

- Adamic, L. A., & Glance, N. (2005). *The political blogosphere and the 2004 US election: divided they blog*. Paper presented at the Conference on Knowledge Discovery in Data, Proceedings of the 3rd international workshop on Link discovery.
- Adar, E. (2011). *Why I Hate Mechanical Turk Research (and Workshops)*. Paper presented at the CHI'11 Workshop on Crowdsourcing and Human Computation.
- Albert, P. S., & Dodd, L. E. (2004). A cautionary note on the robustness of latent class models for estimating diagnostic error without a gold standard. *Biometrics*, 60(2), 427-435.
- Alpaydin, E. (2010). *Introduction to Machine Learning (2nd Edition)*. *Machine Learning* (2nd Editio.). MIT Press.
- Aslam, J. A., Pavlu, V., & Yilmaz, E. (2006). *A statistical method for system evaluation using incomplete judgments*. Paper presented at the Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval.
- Baio, A. (2008). Memeorandum Colors: Visualizing Political Bias with Greasemonkey [Electronic Version], from http://waxy.org/2008/10/memeorandum_colors/
- Baum, M. A., & Gussin, P. (2007). In the eye of the beholder: How information shortcuts shape individual perceptions of bias in the media. *Quarterly Journal of Political Science*, 3, 1-31.
- Bell, R., & Koren, Y. (2007). *The BellKor solution to the Netflix Prize A factorization model. KorBell Teams Report to Netflix*. Retrieved from <http://www.mendeley.com/research/bellkor-solution-netflix-prize/>
- Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis* (Second ed.). Springer.
- Bernstein, M. S., Brandt, J., Miller, R. C., & Karger, D. R. (2011). *Crowds in Two Seconds: Enabling Realtime Crowd-Powered Interfaces*. Paper presented at the UIST 2011.
- Bernstein, M. S., Little, G., Miller, R. C., Hartmann, B., Ackerman, M. S., Karger, D. R., et al. (2010). *Soylent: a word processor with a crowd inside*. Paper presented at the Proceedings of the 23nd annual ACM symposium on User interface software and technology.

- Bickel, J. E. (2007). Some comparisons among quadratic, spherical, and logarithmic scoring rules. *Decision Analysis*, 4(2), 49-65.
- Bigham, J. P., Jayant, C., Ji, H., Little, G., Miller, A., Miller, R. C., et al. (2010). *VizWiz: nearly real-time answers to visual questions*. Paper presented at the Proceedings of the 23rd annual ACM symposium on User interface software and technology.
- Bishop, C. M. (2006). *Pattern recognition and machine learning*: Springer New York.
- Bishop, B. (2008). *The Big Sort: Why the Clustering of Like-minded America is Tearing Us Apart*: Houghton Mifflin Company.
- Blum, A., & Mitchell, T. (1998). *Combining labeled and unlabeled data with co-training*. Paper presented at the Proceedings of the eleventh annual conference on Computational learning theory.
- Bohman, J., & Rehg, W. (2011). Jürgen Habermas. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy (Winter 2011 Edition)*.
- Bowker, G. C., & Star, S. L. (1999). *Sorting things out: classification and its consequences*. Cambridge, MA: MIT Press.
- Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1), 1-3.
- Camerer, C., & Fehr, E. (2004). Measuring Social Norms and Preferences Using Experimental Games: A Guide for Social Scientists. In J. Henrich, R. Boyd, S. Bowles, C. Camerer, E. Fehr & H. Gintis. (Eds.), *Foundations of Human Sociality -- Economic Experiments and Ethnographic Evidence from Fifteen Small-Scale Societies*.
- Casella, G., & Berger, R. L. (2002). *Statistical inference (2nd Edition)*.
- Carpenter, B. (2008). Multilevel bayesian models of categorical data annotation.
- Carterette, B., & Allan, J. (2005). Incremental test collections. Paper presented at the Proceedings of the 14th ACM international conference on Information and knowledge management.
- Carterette, B., Allan, J., & Sitaraman, R. (2006). *Minimal test collections for retrieval evaluation*. Paper presented at the Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval.
- Carterette, B., Pavlu, V., Kanoulas, E., Aslam, J. A., & Allan, J. (2008). *Evaluation over thousands of queries*. Paper presented at the Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval.

- Carterette, B., & Smucker, M. D. (2007). *Hypothesis testing with incomplete relevance judgments*. Paper presented at the Proceedings of the sixteenth ACM conference on Conference on information and knowledge management.
- Carterette, B., & Soboroff, I. (2010). *The effect of assessor error on IR system evaluation*. Paper presented at the Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval.
- Chandler, D., & Horton, J. (2011). Labor Allocation in Paid Crowdsourcing: Experimental Evidence on Positioning, Nudges and Prices.
- Chandler, D., & Kapelner, A. (2010). Breaking monotony with meaning: Motivation in crowdsourcing markets. *University of Chicago mimeo*.
- Chen, D. L., & Dolan, W. B. (2011). Building a Persistent Workforce on Mechanical Turk for Multilingual Data Collection.
- Chilton, L. B., Horton, J. J., Miller, R. C., & Azenkot, S. (2010). *Task search in a human computation market*. Paper presented at the HCOMP '10.
- Collier, D., & Mahon Jr, J. E. (1993). Conceptual" stretching" revisited: Adapting categories in comparative analysis. *American Political Science Review*, 845-855.
- Conover, M. D., Ratkiewicz, J., Francisco, M., Goncalves, B., Menczer, F., & Flammini, A. (2011). *Political Polarization on Twitter*. Paper presented at the ICWSM'11.
- Conover, M., Gonçaves, B., Flammini, A., & Menczer, F. (2012). Partisan asymmetries in online political activity. *EPJ Data Science*.
- Cook, J., and L. Stefanski. 1994. "Simulation-Extrapolation Estimation in Parametric Measurement Error Models." *Journal of the American Statistical Association* 89: 1314–28.
- Dawid, A. P., & Skene, A. M. (1979). Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 20-28.
- Dawid, A. P. (1984). Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society. Series A (General)*, 278-292.
- Dekel, O., & Shamir, O. (2009a). *Vox populi: Collecting high-quality labels from a crowd*. Paper presented at the COLT 2009: Proceedings of the 22nd Annual Conference on Learning Theory.
- Dekel, O., & Shamir, O. (2009b). *Good learners for evil teachers*. Paper presented at the Proceedings of the 26th Annual International Conference on Machine Learning.

- Dietterich, T. (2000). Ensemble methods in machine learning. *Multiple classifier systems*, 1-15.
- Donmez, P., Carbonell, J. G., & Schneider, J. (2009). *Efficiently learning the accuracy of labeling sources for selective sampling*. Paper presented at the Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining.
- Donmez, P., Carbonell, J., & Schneider, J. (2010). *A probabilistic framework to learn from multiple annotators with time-varying accuracy*. Paper presented at the Proceedings of the SIAM International Conference on Data Mining (SDM 2010).
- Dow, S., Kulkarni, A., Bunge, B., Nguyen, T., Klemmer, S., & Hartmann, B. (2011). *Shepherding the crowd: managing and providing feedback to crowd workers*. Paper presented at the Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems.
- Downs, J. S., Holbrook, M. B., Sheng, S., & Cranor, L. F. (2010). *Are your participants gaming the system?: screening mechanical turk workers*. Paper presented at the Proceedings of the 28th international conference on Human factors in computing systems.
- Durant, K. T., & Smith, M. D. (2006). *Mining sentiment classification from political web logs*. Paper presented at the Proceedings of Workshop on Web Mining and Web Usage Analysis of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (WebKDD-2006).
- Efron, M. (2004). *The liberal media and right-wing conspiracies: using cocitation information to estimate political orientation in web documents*. Paper presented at the Proceedings of the thirteenth ACM international conference on Information and Knowledge Management.
- Elte, E. L. (1912), *The Semiregular Polytopes of the Hyperspaces*. Groningen: University of Groningen
- Freund, Y., & Schapire, R. (1995). *A desicion-theoretic generalization of on-line learning and an application to boosting*. Paper presented at the Computational learning theory.
- Friedman, J. H. (1997). On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, 1(1), 55-77.
- Fuller, W. A. (2008). *Measurement Error Models*: John Wiley & Sons, Inc.
- Gamon, M., Basu, S., Belenko, D., Fisher, D., Hurst, M., & Konig, A. C. (2008). *BLEWS: Using Blogs to Provide Context for News Articles*. Paper presented at the Proceedings of the 2008 International Conference on Weblogs and Social Media (ICWSM).

- Garrett, R. K. (2005). *Exposure to Controversy in an Information Society*.
- Glanzberg, M. (2009). Truth. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy (Spring 2009 Edition)*.
- Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477), 359-378.
- Grimmer, J., & King, G. (2011). General purpose computer-assisted clustering and conceptualization. *PNAS*.
- Grinstead, C. M., & Snell, J. L. (1997). *Introduction to Probability*.
- Gwet, K. L. (2010) *Handbook of Inter-Rater Reliability* (2nd Edition). Gaithersburg : Advanced Analytics, LLC.
- Hand, D. J. (2006). Classifier technology and the illusion of progress. *Statistical Science*, 21(1), 1-14.
- Harris, C. G. (2011). *You're Hired! An Examination of Crowdsourcing Incentive Models in Human Resource Tasks*. Paper presented at the Crowdsourcing for Search and Data Mining (CSDM 2011).
- Hastie, T., Tibshirani, R., Friedman, J., & Franklin, J. (2005). *The elements of statistical learning: data mining, inference and prediction* (Vol. 27).
- Hausser, D. (1992). Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and computation*, 100(1), 78-150.
- Hirst, G., Riabinin, Y., & Graham, J. (2010). Party status as a confound in the automatic classification of political speech by ideology. Paper presented at the JADT 2010.
- Holtzman, N. S., Schott, J. P., Jones, M. N., Balota, D. A., & Yarkoni, T. (2007). Exploring media bias with semantic analysis tools: validation of the Contrast Analysis of Semantic Similarity (CASS). *Behavior Research Methods*, 1-8.
- Hookway, C. (2010). Pragmatism. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy (Spring 2010 Edition)*.
- Hopkins, D. J., & King, G. (2010). A method of automated nonparametric content analysis for social science. *American Journal of Political Science*, 54(1), 229-247.
- Horton, J. J. (2010). Employer Expectations, Peer Effects and Productivity: Evidence from a Series of Field Experiments. *SSRN eLibrary*.
- Horton, J. J., & Chilton, L. B. (2010). *The labor economics of paid crowdsourcing*. Paper presented at the Proceedings of the 11th ACM conference on Electronic commerce.
- Howe, J. (2006). The rise of crowdsourcing. *Wired magazine*, 14(6), 1-4.

- Howe, J. (2008). *Crowdsourcing: Why the power of the crowd is driving the future of business* (Vol. 14).
- Hui, S. L., & Walter, S. D. (1980). Estimating the error rates of diagnostic tests. *Biometrics*, 167-171.
- Hui, S. L., & Zhou, X. H. (1998). Evaluation of diagnostic tests without gold standards. *Statistical Methods in Medical Research*, 7(4), 354.
- Hyslop, D. R., & Imbens, G. W. (2001). Bias from classical and other forms of measurement error. *Journal of Business and Economic Statistics*, 19(4), 475-481.
- Ipeirotis, P. G. (2010). Worker Evaluation in Crowdsourcing: Gold Data or Multiple Workers?, from <http://www.behind-the-enemy-lines.com/2010/09/worker-evaluation-in-crowdsourcing-gold.html>
- Ipeirotis, P. G., Provost, F., & Wang, J. (2010). *Quality management on amazon mechanical turk*. Paper presented at the Proceedings of the ACM SIGKDD workshop on human computation.
- Jahrer, M., Toscher, A., & Legenstein, R. (2011). *Combining predictions for accurate recommender systems*. Paper presented at the Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining.
- Jin, R., & Ghahramani, Z. (2003). Learning with multiple labels. *Advances in Neural Information Processing Systems*, 921-928.
- Jiang, M., & Argamon, S. (2008). *Political leaning categorization by exploring subjectivities in political blogs*. Paper presented at the Proceedings, 4th International Conference on Data Mining (DMIN 2008).
- Jordan, P. R., & Wellman, M. P. (2009). *Generalization risk minimization in empirical game models*. Paper presented at the Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1.
- Kazai, G. (2010). An Exploration of the Influence that Task Parameters have on the Performance of Crowds. Paper presented at the CrowdConf 2010.
- Kapelner, A., & Chandler, D. (2010). *Preventing Satisficing in Online Surveys*. Paper presented at the CrowdConf 2010.
- Kelly, M. G., Hand, D. J., & Adams, N. M. (1998). *Defining the goals to optimise data mining performance*. Paper presented at the Proc. Fourth International Conference on Knowledge Discovery and Data Mining (R. Agrawal, P. Stolorz and G. Piatetsky-Shapiro, eds.).
- Kelly, M. G., & Hand, D. J. (1999). Credit scoring with uncertain class definitions. *IMA Journal of Management Mathematics*, 10(4), 331.

- Kelly, M., Hand, D., & Adams, N. (1999a). Supervised classification problems: how to be both judge and jury. *Advances in Intelligent Data Analysis*, 235-244.
- Kelly, M. G., Hand, D. J., & Adams, N. M. (1999b). *The impact of changing populations on classifier performance*. Paper presented at the Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining.
- Kilgarriff, A. (1998). Gold standard datasets for evaluating word sense disambiguation programs. *Computer Speech and Language*, 12(4), 453-472.
- Kittur, A., Chi, E. H., & Suh, B. (2008). *Crowdsourcing user studies with Mechanical Turk*. Paper presented at the Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems.
- Klein, D. B., & Stern, C. (2008). Liberal Versus Conservative Stinks. *Society*, 45(6), 488-495.
- Krebs, V. (2008). Political Polarization in Amazon.com Book Purchases. from <http://orgnet.com>
- Krippendorff, K. (2004). *Content analysis: An introduction to its methodology*: Sage Publications, Inc.
- Krupka, E. L., & Weber, R. A. (2012). Identifying social norms using coordination games: Why does dictator game sharing vary?
- Kuchenhoff, H., Mwalili, S. M., & Lesaffre, E. (2006). A general method for dealing with misclassification in regression: The misclassification SIMEX. *Biometrics*, 62(1), 85-96.
- Lambert, N. S. (2011). *Elicitation and Evaluation of Statistical Forecasts*.
- Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1), 159.
- Laver, M., Benoit, K., & Garry, J. (2003). Extracting policy positions from political texts using words as data. *American Political Science Review*, 97(02), 311-331.
- Le, J., Edmonds, A., Hester, V., & Biewald, L. (2010). *Ensuring quality in crowdsourced search relevance evaluation: The effects of training question distribution*. Paper presented at the SIGIR 2010 Workshop on Crowdsourcing for Search Evaluation.
- Levine, A., Simmons, M. P., Adar, E., & Adamic, L. A. (2011). The party is over here: Structure and content in the 2010 election.
- Li, T., Zhang, C., & Zhu, S. (2006). *Empirical studies on multi-label classification*. Paper presented at the Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06).

- Little, G., Chilton, L. B., Goldman, M., & Miller, R. C. (2009). *TurKit: tools for iterative tasks on mechanical Turk*. Paper presented at the Proceedings of the ACM SIGKDD Workshop on Human Computation.
- Lin, W. H. (2006). *Identifying perspectives at the document and sentence levels using statistical models*. Paper presented at the Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume: doctoral consortium.
- Lin, F., & Cohen, W. W. (2008). The multirank bootstrap algorithm: Semi-supervised political blog classification and ranking using semi-supervised link classification. Paper presented at the ICWSM 2008.
- Lin, W. H., Xing, E., & Hauptmann, A. (2008). A joint topic and perspective model for ideological discourse. *Machine Learning and Knowledge Discovery in Databases*, 17-32.
- Little, G., Chilton, L. B., Goldman, M., & Miller, R. C. (2010). *Exploring iterative and parallel human computation processes*. Paper presented at the Proceedings of the ACM SIGKDD workshop on human computation.
- Lugosi, G. (1992). Learning with an unreliable teacher. *Pattern Recognition*, 25(1), 79-87.
- MacQueen, K. M., McLellan, E., Kay, K., & Milstein, B. (1998). Codebook development for team-based qualitative analysis. *Cultural Anthropology Methods*, 10(2), 31-36.
- Malouf, R., & Mullen, T. (2007). *Graph-based user classification for informal online political discourse*. Paper presented at the Proceedings of the 1st Workshop on Information Credibility on the Web.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*: Cambridge University Press New York, NY, USA.
- Martin, L. W., & Vanberg, G. (2008). A robust transformation procedure for interpreting political text. *Political Analysis*, 16(1), 93.
- Mason, W., & Suri, S. (2011). Conducting behavioral research on Amazon's Mechanical Turk. *Behavior Research Methods*, 1-23.
- Mason, W., & Watts, D. J. (2009). *Financial incentives and the "performance of crowds"*. Paper presented at the Proceedings of the ACM SIGKDD Workshop on Human Computation.
- Matlin, C., Singer-Vine, J., & Wilson, C. (2010). Escape From the Echo Chamber -- An interactive test of how open-minded your news diet is. *Slate Magazine*.

- Mehta, J., Starmer, C., & Sugden, R. (1994). The Nature of Salience: An Experimental Investigation of Pure Coordination Games. *American Economic Review*, 84(3), 658-673.
- Miller, N., Resnick, P., & Zeckhauser, R. (2005). Eliciting informative feedback: The peer-prediction method. *Management Science*, 1359-1373.
- Mitchell, T. (1997). *Machine Learning*. McGraw-Hill.
- Monroe, B. L., Colaresi, M. P., & Quinn, K. M. (2008). Fightin'Words: Lexical Feature Selection and Evaluation for Identifying the Content of Political Conflict. *Political Analysis*, 16(4), 372.
- Mullen, T., & Malouf, R. (2006). *A preliminary investigation into sentiment analysis of informal political discourse*. Paper presented at the Proceedings of the AAAI symposium on computational approaches to analyzing weblogs.
- Munson, S., & Resnick, P. (2010). *Presenting Diverse Political Opinions: How and How Much*. Paper presented at the CHI 2010.
- Munson, S., Rosengren, E., & Resnick P. (2011) *Thanks and Tweets: Comparing Two Public Displays*. Proceedings of CSCW 2011. ACM. New York, NY.
- Munson, S., Zhou, D. X., & Resnick, P. (2009). *Sidelines: An Algorithm for Increasing Diversity in News and Opinion Aggregators*. Paper presented at the ICWSM'09.
- Oh, A., Lee, H., & Kim, Y. (2009). *User Evaluation of a System for Classifying and Displaying Political Viewpoints of Weblogs*. Paper presented at the Proceedings of the Third International ICWSM Conference.
- Opitz, D., & Maclin, R. (1999). Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*
- Paolacci, G., Chandler, J., & Ipeirotis, P. G. (2010). Running experiments on Amazon Mechanical Turk. *Judgment and Decision Making*, 5(5), 411-419.
- Park, S., Kang, S., Lee, S., Chung, S., & Song, J. (2008). *Mitigating Media Bias: A Computational Approach*. Paper presented at the Proc. of Web Science: Collaboration and Collective Intelligence.
- Park, S., Kang, S., Chung, S., & Song, J. (2009). *NewsCube: Delivering Multiple Aspects of News to Mitigate Media Bias*. Paper presented at the Proc. of CHI 2009.
- Park, S., Ko, M., Kim, J., Liu, Y., & Song, J. (2011). *The politics of comments: predicting political orientation of news stories with commenters' sentiment patterns*. Paper presented at the Proceedings of the ACM 2011 conference on Computer supported cooperative work (CSCW).

- Pennacchiotti, M., & Popescu, A.-M. (2011). A Machine Learning Approach to Twitter User Classification. Paper presented at the ICWSM'11
- Phelps, C. E., & Hutson, A. (1995). Estimating Diagnostic Test Accuracy Using a " Fuzzy Gold Standard". *Medical Decision Making*, 15(1), 44.
- Polikar, R. (2006). Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, 6(3), 21-45.
- Prelec, D. (2004). A Bayesian truth serum for subjective data. *Science*, 306(5695), 462.
- Quinn, A. J., & Bederson, B. B. (2011a). *Human computation: a survey and taxonomy of a growing field*. Paper presented at the Proceedings of the 2011 annual conference on Human factors in computing systems.
- Quinn, A. J., & Bederson, B. B. (2011b). *Human-Machine Hybrid Computation*. Paper presented at the Position paper for CHI 2011 Workshop On Crowdsourcing And Human Computation.
- Quost, B., & Denoeux, T. (2009). *Learning from data with uncertain labels by boosting credal classifiers*. Paper presented at the Proceedings of the 1st ACM SIGKDD Workshop on Knowledge Discovery From Uncertain Data.
- Rao, D., Yarowsky, D., Shreevats, A., & Gupta, M. (2010). *Classifying latent user attributes in twitter*. Paper presented at the Proceedings of the 2nd international workshop on Search and mining user-generated contents.
- Raykar, V. C., Yu, S., Zhao, L. H., Jerebko, A., Florin, C., Valadez, G. H., et al. (2009). *Supervised Learning from Multiple Experts: Whom to trust when everyone lies a bit*. Paper presented at the Proceedings of the 26th Annual International Conference on Machine Learning.
- Reips, U. D. (2000). The Web experiment method: Advantages, disadvantages, and solutions. *Psychological experiments on the Internet*, 89-114.
- Rogers, S., Girolami, M., & Polajnar, T. (2009). Semi-parametric analysis of multi-rater data. *Statistics and Computing*, 20(3), 317-334.
- Rogstadius, J., Kostakos, V., Kittur, A., Smus, B., Laredo, J., & Vukovic, M. (2011). *An Assessment of Intrinsic and Extrinsic Motivation on Task Performance in Crowdsourcing Markets*. Paper presented at the ICWSM 2011.
- Rosch, E. (1999). Principles of categorization. *Concepts: core readings*, 189-206.
- Rothschild, D., & Wolfers, J. (2011). Forecasting elections: Voter intentions versus expectations. Available at SSRN 1884644.

- Sanderson, M., & Zobel, J. (2005). *Information retrieval system evaluation: effort, sensitivity, and reliability*. Paper presented at the Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval.
- Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2), 197–227.
- Schelling, T. C. (1960). *The Strategy of Conflict*. Cambridge, MA: Harvard University Press.
- Settles, B. (2009). *Active Learning Literature Survey*: University of Wisconsin–Madison.
- Shakhnarovich, G., Viola, P. A., & Moghaddam, B. (2002). *A unified learning framework for real time face detection and classification*. Paper presented at the Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on.
- Sharma, R., Yeasin, M., & Walavalkar, L. A. (2002). Multi-modal gender classification using support vector machines (SVMs): Google Patents.
- Shaw, A. D., Horton, J. J., & Chen, D. L. (2011). *Designing incentives for inexpert human raters*. Paper presented at the Proceedings of the ACM 2011 conference on Computer supported cooperative work.
- Sheng, V. S., Provost, F., & Ipeirotis, P. G. (2008). *Get another label? improving data quality and data mining using multiple, noisy labelers*. Paper presented at the Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining.
- Sill, J., Takacs, G., Mackey, L., & Lin, D. (2009). Feature-Weighted Linear Stacking. *Arxiv preprint arXiv:0911.0460*
- Slapin, J. B., & Proksch, S. O. (2008). A scaling model for estimating time-series party positions from texts. *American Journal of Political Science*, 52(3), 705-722.
- Smucker, M. D., Allan, J., & Carterette, B. (2007). *A comparison of statistical significance tests for information retrieval evaluation*. Paper presented at the Proceedings of the sixteenth ACM conference on Conference on information and knowledge management.
- Smyth, P., Fayyad, U., Burl, M., Perona, P., & Baldi, P. (1995). Inferring ground truth from subjective labeling of Venus images. *Advances in Neural Information Processing Systems*, 1085-1092.
- Smyth, P. (1995). Learning with Probabilistic Supervision. In *Computational Learning Theory and Natural Learning Systems*.

- Snow, R., O'Connor, B., Jurafsky, D., & Ng, A. Y. (2008). *Cheap and fast---but is it good?: evaluating non-expert annotations for natural language tasks*. Paper presented at the Proceedings of the Conference on Empirical Methods in Natural Language Processing.
- Sorokin, A., & Forsyth, D. (2008). *Utility data annotation with amazon mechanical turk*. Paper presented at the Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08.
- Sugden, R. (1995). A theory of focal points. *The Economic Journal*, 533-550.
- Sunstein, C. R. (2001). *Republic. com*: Princeton University Press.
- Tsoumakas, G., & Katakis, I. (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3), 1-13.
- Vapnik, V. (1999). *The nature of statistical learning theory*: springer.
- Vander Wal, T. (2007). *Folksonomy. online posting, Feb, 7*
- Vannoorenberghe, P., & Denoeux, T. (2002). *Handling uncertain labels in multiclass problems using belief decision trees*. Paper presented at the International Conference on Processing and Management of Uncertainty.
- Veronis, J. (1998). *A study of polysemy judgements and inter-annotator agreement*. Paper presented at the Programme and advanced papers of the Senseval workshop.
- von Ahn, L. (2005). *Human Computation*. Doctoral Thesis. UMI Order Number: AAI3205378, CMU.
- Volkmer, T., Thom, J. A., & Tahaghoghi, S. M. M. (2007). Modeling human judgment of digital imagery for multimedia retrieval. *Multimedia, IEEE Transactions on*, 9(5), 967-974.
- Voorhees, E. M. (2000). Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing & Management*, 36(5), 697-716.
- Voorhees, E. (2002). *The philosophy of information retrieval evaluation*. Paper presented at the Evaluation of cross-language information retrieval systems.
- Welinder, P., Branson, S., Belongie, S., & Perona, P. (2010). *The Multidimensional Wisdom of Crowds*. Paper presented at the NIPS 2010.
- Welinder, P., & Perona, P. (2010). *Online crowdsourcing: Rating annotators and obtaining cost-effective labels*. Paper presented at the Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on.

- Whitehill, J., Ruvolo, P., Wu, T., Bergsma, J., & Movellan, J. (2009). Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in Neural Information Processing Systems*, 22, 2035ñ2043.
- Wiebe, J. M., Bruce, R. F., & O'Hara, T. P. (1999). *Development and use of a gold-standard data set for subjectivity classifications*. Paper presented at the Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics.
- Winkler, R. L., & Jose, V. R. R. (2010). Scoring Rules. In *Wiley Encyclopedia of Operations Research and Management Science*.
- Witkowski, J., & Parkes, D. (2012). *A Robust Bayesian Truth Serum for Small Populations*. Paper presented at the Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI'12).
- Witten, I. H., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*.
- Wolfers, J., & Zitzewitz, E. (2004). Prediction Markets. *Journal of Economic Perspectives*, Vol.18(No.2), 107-126.
- Yan, Y., Rosales, R., Fung, G., Schmidt, M., Hermosillo, G., Bogoni, L., et al. (2010a). *Modeling annotator expertise: Learning when everybody knows a bit of something*. Paper presented at the International Conference on Artificial Intelligence and Statistics.
- Yan, Y., Rosales, R., Fung, G., & Dy, J. (2010b). Modeling Multiple Annotator Expertise in the Semi-Supervised Learning Scenario.
- Yu, B., Kaufmann, S., & Diermeier, D. (2008). Classifying party affiliation from political speech. *Journal of Information Technology & Politics*, 5(1), 33-48.
- Zhou, D., Bousquet, O., Lal, T. N., Weston, J., & Scholkopf, B. (2004). *Learning with local and global consistency*. Paper presented at the Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference.
- Zhou, D., Platt, J., Basu, S., & Mao, Y. (2012). *Learning from the Wisdom of Crowds by Minimax Entropy*. Paper presented at the Advances in Neural Information Processing Systems 25.
- Zhou, D. X., & Resnick, P. (2009). Assessment of Conversation Co-mentions as a Resource for Software Module Recommendation. Paper presented at the ACM RecSys'09.
- Zhou, D. X., Resnick, P., & Mei, Q. (2011). *Classifying the Political Leaning of News Articles and Users from User Votes*. Paper presented at the ICWSM 2011.

Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). *Semi-supervised learning using gaussian fields and harmonic functions*. Paper presented at the ICML-03.

Zinovev, D., Feigenbaum, J., Furst, J., & Raicu, D. (2011). *Probabilistic lung nodule classification with belief decision trees*. Paper presented at the Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE.

Zobel, J. (1998). *How reliable are the results of large-scale information retrieval experiments?* Paper presented at the Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval.