DETECTING ANOMALOUSLY SIMILAR ENTITIES IN UNLABELED DATA

A Dissertation Presented

by

LISA D. FRIEDLAND

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 2016

College of Information and Computer Sciences

DETECTING ANOMALOUSLY SIMILAR ENTITIES IN UNLABELED DATA

A Dissertation Presented

by

LISA D. FRIEDLAND

Approved as to style and content by:

David D. Jensen, Chair

David A. Smith, Member

James Allan, Member

Michael L. Lavine, Member

James Allan, Chair of the Faculty
College of Information and Computer Sciences

DEDICATION

To my tribe—my parents, Brad and Jeanie, and brothers, Jeremy and David.

ACKNOWLEDGMENTS

I would like to thank my advisor, David Jensen, for planting the seeds that eventually grew into this work. He has my gratitude for his longstanding support and patience, and for practices he taught me about science: to seek out the interesting "why" questions, and to appreciate both the importance and difficulty of good scientific communication. I also thank the other members of my committee for taking the time to offer ideas, feedback and encouragement at various steps along the way.

Within the Knowledge Discovery Lab and the greater Computer Science Department (/School/College) at UMass, I have been intellectually fed and challenged, and I have learned from friends and colleagues too numerous to name. A special thanks to some who made a particular impact over the years: Ann Guo, Victoria Manfredi, Pippin Wolfe, Michael Hay, Matt Rattigan, Jennifer Neville, Huseyin Oktay, Pallika Kanani, Laura Dietz, Amanda Gentzel, Mike Bendersky, Emily Horrell and Özgür Şimşek. I am also grateful to the Lazer Lab at Northeastern University for hosting me these past few years and gradually reawakening me to subjects outside the algorithm. While in Amherst, I greatly valued my ties to the orchestra and salsa communities; no matter how research was going, these worlds helped ground me to a life outside the student bubble.

Though my family has been unwaveringly supportive, it was friends that first helped me summon the courage and enthusiasm to apply to graduate school, and friends that helped me build and debug the skill set needed to finish. From the early days, Janet Rosenbaum, Merav Shohet, and Michael Cuthbert served as guides, shaping my perspectives on academia and doing research. When it came to perseverance and the mechanics of the daily grind, Yoonheui Kim and Christoph Reichenbach were at my side and understood the value of concretely addressing the little things. I thank Christoph, above all, for listening, teaching, and being there when it matters.

ABSTRACT

DETECTING ANOMALOUSLY SIMILAR ENTITIES IN UNLABELED DATA

SEPTEMBER 2016

LISA D. FRIEDLAND

A.B., HARVARD UNIVERSITY

M.S., UNIVERSITY OF MASSACHUSETTS AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor David D. Jensen

In this work, the goal is to detect closely-linked entities within a data set. The entities of interest have a tie causing them to be similar, such as a shared origin or a channel of influence. Given a collection of people or other entities with their attributes or behavior, we identify unusually similar pairs, and we pose the question: Are these two people linked, or can their similarity be explained by chance?

Computing similarities is a core operation in many domains, but two constraints differentiate our version of the problem. First, the score assigned to a pair should account for the probability of a coincidental match. Second, no training data is provided; we must learn about the system from the unlabeled data and make reasonable assumptions about the linked pairs. This problem has applications to social network analysis, where it can be valuable to identify implicit relationships among people from indicators of coordinated activity. It also arises in situations where we must decide whether two similar observations correspond to two different entities or to the same entity observed twice.

This dissertation explores how to assess such ties and, in particular, how the similarity scores should depend on not only the two entities in question but also properties of the entire data set. We develop scoring functions that incorporate both the similarity and rarity of a pair. Then, using these functions, we investigate the statistical power of a data set to reveal (or conceal) such pairs.

In the dissertation, we develop generative models of linked pairs and independent entities and use them to derive scoring functions for pairs in three different domains: people with job histories, Gaussian-distributed points in Euclidean space, and people (or entities) in a bipartite affiliation graph. For the first, we present a case study in fraud detection that highlights the potential, as well as the complexities, of using these methods to address real-world problems. In the latter two domains, we develop an inference framework to estimate whether two entities were more likely generated independently or as a pair. In these settings, we analyze how the scoring function works in terms of similarity and rarity; how well it can detect pairs as a function of the data set; and how it differs from existing similarity functions when applied to real data.

TABLE OF CONTENTS

Page

CHAPTER

APPENDICES

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1
INTRODUCTION

1.1 Overview

In this thesis, we consider a problem that has appeared in many variations but surprisingly, has not previously been described abstractly. Suppose we have a collection of objects or people in which most entities are generated independently, while a small number are generated in dependent pairs. The paired entities closely resemble each other, but they are otherwise unremarkable. We wish to identify these pairs.

This problem is interesting to study because it can represent diverse scenarios; for example, the paired entities could be people who shop together, bots online that spam together, or database records that describe the same object. To identify such pairs within the collection, we might start by noting some candidate pairs that are unusually similar in attributes or behavior. For each candidate, either the entities were generated independently—so their similarity is due to coincidence—or they were generated dependently—that is, their similarity is due to an unobserved tie. A central question in this thesis is how to estimate the probability that such a tie (or "link") exists, given the two entities and the rest of the data set. We assume that we have unlabeled data, that we wish to identify the pairs (or small groups) of linked entities by assigning them probability scores, and that we want the scores to account for whether the similarity could have arisen by chance. We call this problem the detection of Anomalously Similar Objects in UNlabeled Data, or (detecting) ASOUND.

In the thesis, we look closely at ASOUND in three domains, with a focus on the scoring functions for candidate pairs. The first piece of research is a real-world case study in fraud detection that serves as an introduction to the task. For the latter two, starting from the abstract problem statement, we construct generative models, derive scoring methods, and validate them on real and synthetic data. The work culminates in a set of interpretable methods for inferring latent ties among nodes in a bipartite graph or among binary vectors.

The methods are based on likelihood ratios, they adjust for the rarity of shared elements, and they are applicable to a wide range of problems.

In parallel, we investigate a higher-level story regarding pair detection. Many well-established areas of research, such as entity resolution and information retrieval, share this same core problem of detecting and scoring pairs of anomalously similar objects (ASOs). In contrast, newer applications, in particular those involving social network inference, could stand to benefit from firmer theoretical foundations. However, despite the existence of sophisticated methods for specific ASO problems, little work addresses commonalities across domains, and we found little guidance for generalizing their approaches to new domains.

In response to this gap, we formalize ASOUND as a domain-independent problem. We view this problem statement as a previously missing abstraction layer, a means of unifying existing work and facilitating the sharing of methods and results across communities. Working within this abstraction layer, we focus on two natural and important questions. The first, described above, is how to derive a scoring function for pairs.

The second question concerns mapping the problem space of ASOUND. Even using an optimal method, pairs will be statistically more difficult to distinguish within some data sets than others. Furthermore, as we vary characteristics of the data set, not only does the amount of information present vary, but so do the primary sources of that information. Because related literature has mostly ignored these issues, we make an effort to systematically explore the problem space. We show that a data set's location in the problem space influences both the detectability of its pairs and the relative effectiveness of various scoring methods.

The contributions of this thesis, which result from investigating the questions above, are in the following four areas. These points are more fully elaborated in Section 1.5.

- **Domain-independent task formalization and modeling.** We present the first abstract characterization of ASOUND and provide an approach to developing scoring functions for any domain.
- **Domain-specific scoring functions.** We derive methods for scoring pairs of Gaussian-distributed points and pairs of nodes in a bipartite graph, and we examine tradeoffs among these and other new and existing alternatives.

- **Demonstration of practical utility.** In a large-scale case study, we show that these inferred social ties have value as indicators of fraud risk. Using other real-world data sets in which ground truth pairs are known (yet non-trivial to identify), we demonstrate that our recommended scoring functions can yield substantial improvements over naive methods.
- **Problem space analysis.** Using theory and simulations, we manipulate the parameters of ASOUND data sets to identify which characteristics make pair detection easier. Similar analysis also reveals how some methods can be near-optimal in certain parts of the problem space only.

1.2 Problem Details

A broad range of tasks require judgments about anomalously similar pairs. Here are examples from several domains and application areas.

Employment: Two people worked at the same office as each other, at three different times (and employers) over the course of their careers. Could this have happened by accident, or is it evidence that they stayed together on purpose?

Consumer purchasing: Two customers at a bookstore purchased five of the same titles last week. Are they in a book club or class together?

Social media: Two Twitter accounts are posting links to the same spam website (a blog advertising pharmaceuticals). Are they Sybil accounts controlled by a single person?

Entity resolution: A publication database lists two articles by the same author, in which the titles and co-authors differ by only a few characters. Is one record just a noisy duplicate of the other?

Fraud prevention: A visitor to a lost and found claims she lost a Gucci watch with a brown leather band. Is this description enough to uniquely identify the object and prove ownership?

Individual re-identification: At a professional event, you meet someone of the same nationality, institution, and research area as someone (whose name and face you cannot recall) you met yesterday. What are the odds this is the same person twice?

The upper half of this list is representative of our primary interests, but our approach is via an abstract characterization of the task, next.

1.2.1 Task Specification

Problems involving ASOs span not only a wide range of domains, but also a variety of assumptions and problem setups. We differentiate ASOUND from the larger class of ASO problems with the following constraints.

First, we assume in ASOUND that we have no labeled examples, but rather, that linked pairs will be identifiable through being "anomalously similar." This restriction is important because in many practical settings, there are in fact few or no known positives.

Second, we require that the pairwise scores take into account whether the similarity could be coincidental—that is, produced by independent entities. Notice, however, that the phrase "anomalously similar" reflects only part of this requirement. To estimate the probability of a link, a scoring method needs to take into account not only a pair's similarity, but also its rarity—how unusual the two entities are in the population. All else being equal, a pair that shares rare attributes is more likely to be linked than one that shares common attributes.

Note that we assume that the linked entities are similar, but we do not assume that they are all detectable. Some linked pairs, even perfect duplicates, might be statistically indistinguishable from independent. Our emphasis is not on identifying the linked pairs using any means possible, such as external data, but rather, on optimally combining the available information.

Finally, we assume that linked pairs are rare, and the remaining entities—the vast majority—are independent. Treating most entities as independent is a reasonable simplifying assumption; it means they can be modeled as separate draws from a single distribution, but it does not prevent that distribution from having a complex substructure. Assuming a strong class imbalance provides a source of leverage: it means we can use the data set as a whole to learn the properties of typical entities. If the models need to take rarity into account, they can estimate from the original data how rare a given attribute is. If they need to determine whether a pair is unusually similar, they can estimate from original data the distribution of similarities for independent pairs.

Some ASO tasks only ask for certain pairs to be scored or involve additional constraints on the outputs. For example, some problems need to determine the best matches (if any) to a specific entity, such as when searching a database for the best match to a DNA sample, a fingerprint, or a plagiarized document. Others consider all links between one database and another, such as when merging two feature sets describing the same entities or using one data set to de-anonymize another. Still others must aggregate the inferred pairs into clusters of entities corresponding to distinct real-world objects or groups of collaborators. To avoid getting caught up in phenomena specific to those scenarios, we define the output to ASOUND to be simply an independent score for every pair of entities. While this limits us from exploring constrained matching scenarios, such as picking a single best match per entity, it helps home in on our main questions, about pairwise scoring functions and pair detectability.

1.2.2 Solution Characteristics

The solution we develop for the scoring function, further outlined in Section 1.4.1, distinguishes between two classes of candidate pairs. First, from the entire data set, we learn a model of independent entities, which we call ϕ. Then, to score a candidate pair, we compute whether it has a higher likelihood of being a negative pair—that is, two independent draws from ϕ—or a positive pair—generated from a separate model, one that also uses ϕ. This approach satisfies the above requirements (up to limitations such as the fit of the models), providing a statistical way to combine evidence about pairs.

Once we have such a method, it is natural to ask what properties of the data make the score go up or down. This is important for understanding any method's strengths and weaknesses. Moreover, it is interesting because it is not always intuitively obvious what behavior to expect.

Take, for instance, the bookstore customers from above, who come from a bipartite graph of all customers and their purchases. The likelihood of a link between a pair of customers should depend on the number of titles they co-purchase, and also at least some of the following factors:

Sample size: How many customers visited the store that week? If it is a large online retailer, there is a greater chance that some pair of customers would buy the same items, even without knowing each other, than in a brick and mortar store.

Number of pairs: How many of the store's customers know each other? If the proportion is high—e.g., if the store is in a small town—then it is more likely for the pair to have a social tie affecting their book-buying habits.

Cardinality of attribute set: How many books does the store offer? In a souvenir shop with a limited selection, it is less surprising for customers to buy the same items.

Number of attributes per person: How many books do customers tend to buy? How many other books did these customers buy? If they each only bought the five that overlapped, the similarity seems more likely to be intentional. If one or both of them made a large number of other purchases, then it seems more likely to be coincidental.

Strength and characteristics of true links: When people do share a social link, how does it affect their buying habits? People in the same class might purchase many books at the start of a semester, whereas those in a book club might instead overlap by one title per month.

Distribution of shared attributes within the population: Which particular books did they share? Best sellers seem more likely to occur by chance. So do series, such as Harry Potter. Books that are rare and/or unrelated are stronger signs of a link.

Notice that some of these properties—those describing these particular customers—can only affect the score of one pair. Others—those describing the data set as a whole—affect the scores of all pairs in the data set. Properties that affect all scores, for instance by changing ϕ, are traditionally neglected by the literature. But it is an interesting theoretical question to ask what happens in ASOUND when ϕ changes. One of our goals, beyond developing scoring functions, is to understand how the data set's properties affect the feasibility of detecting pairs—or more broadly, to characterize the problem space of ASOUND.

1.2.3 Existing Literature

Problems involving pairs of ASOs are found in numerous areas of the literature. Calculating similarity or detecting near-duplicates is a core computational need, and different communities approach it in diverse ways, depending on the application at hand. Inferring latent ties between people can be framed as analysis of social networks and human behavior [40, 45, 86], or as a tool to detect spam or cheating in online activities [87, 128, 143]. Detecting the same person twice can indicate a successful forensic identification from a crime scene [125], or else an unwanted privacy breach [93]. Determining whether two database records match is a data cleaning problem to be solved using string similarity measures wrapped into a supervised learning or clustering routine [46], while determining whether two documents match is a question to address using information retrieval models [83] or plagiarism detection software [103].

Yet despite the many related ASO problems—or perhaps because of them—existing literature does not provide a clear starting point for ASOUND. One reason is our particular problem constraints. The prerequisites that the scoring method handle unlabeled data and that it take rarity into account in order to distinguish true links from coincidences differentiate ASOUND from much existing work. Two common approaches are ruled out. In many applications, researchers have access to labeled examples, so they construct useful features and train classifiers to identify the linked pairs. In another segment of applications, a similarity calculation is considered sufficient; the goal is to identify highly similar pairs and do something further with them. There, if there is a concern about pairs being similar "by chance," it might be handled manually, for example by excluding pairs with too little shared material.

A second reason is that most work is too domain-specific to be to be easily generalized. Information retrieval models for computing a document's relevance to a query, refined over the past forty years, address issues we care about, but they assume a setup in which entities are text documents, a short query entity is held constant, and the output is a list of its best matches. Likewise, the entity resolution problem in databases is very similar to ASOUND, but its scoring method expects entities containing text fields, and an accompanying similarity function. In forensic science, significant effort is put into into modeling the properties of a given material, such as fingerprints or glass shards, in order to make high-accuracy probability estimates for relatively few candidate pairs. Chapter 5 discusses these fields in more detail. While their scoring methods might satisfy our requirements, they have not been presented in general enough terms to be extended into new domains. New problems, such as inferring social ties from activity data, have tended to be addressed using the two common approaches mentioned previously, which do not involve modeling the independent entities.

Because existing work is relatively task-focused, there has been little attention to how outcomes are influenced by the properties of a data set. Within a given data set, it is typical to compare multiple algorithms and feature sets. However, it is usually considered outside the scope of interest to ask whether the task would be easier, or if a particular scoring function would perform better, under a different data distribution. We highlight a few exceptions in Chapter 5.

1.3 Research Goals and Outline

Originally, we expected to develop scoring methods for this thesis by borrowing from prior literature on detecting ASOs. However, since existing research is specific to particular domains and tasks, we found no clear starting point for new domains. As a response to this gap, we formulate ASOUND as a domain-independent problem. Framing it as single, abstract problem will help the community discover relevant work and facilitate progress on its more general questions. Although other ASO instances are diverse in their domains, constraints, and positions within the problem space, examining the theoretical aspects of ASOUND should yield answers relevant to a broad audience. We envision that referring to and building upon the models developed here, in addition to thinking about the larger problem space, will help people contextualize results observed on individual data sets.

Within ASOUND, the most immediate question is how to score pairs: Given two entities from a data set, how can we estimate the probability that they are linked, as opposed to independent? Such a pairwise function is at the heart of ASOs in any domain. Even when research focuses on efficiency, as it does in near-duplicate detection, or on clusters or joint configurations of links, as it does in record linkage, some pairwise similarity function is implied.

Our second question concerns the problem space of ASOUND. Given a particular data set containing linked pairs, we would like to be able to forecast how statistically identifiable the pairs will be and how useful the respective information sources will be. These properties will vary as a function of the data. If we can understand how they differ across the problem space, a number of possibilities open up.

The issue of whether pairs are statistically identifiable relates to matters of statistical power—whether there is enough data to make confident inferences—and of class separation—whether the linked pairs are any more similar than independent pairs. If we can determine a priori that a data set's pairs are easy to identify, there may be opportunities for efficiency gains, such as by reducing the number of pairs or features examined or by switching to a cheaper (but less accurate) scoring function. If we determine they are hard to identify, there may be opportunities to remedy that, such as by increasing the number of entities or features, or to respond to it, say, by scoring only the most likely-looking pairs. Such knowledge either way has implications for the complementary problem of privacy protection: if it is easy to match entities to their near-duplicate pairs, their re-identification risk is high (subject to the constraints of each task) [136]. Finally, an understanding of where the information comes from, as a function of the problem space, lets us choose knowledgeably among scoring functions and the features they use. Different scoring functions will provide different tradeoffs among efficiency, performance, and robustness. Overall, understanding the role of the problem space in ASOUND will help us make informed choices about data and methods, and it is a fundamental step beyond empirically testing one algorithm versus another.

In the thesis, we address ASOUND for entities from three domains: first, for entities described by time-stamped sequences of affiliations (Chapter 2); second, for entities with real-valued vectors of features (Chapter 3); and finally, for entities with arbitrary sets of affiliations (Chapter 4). The first piece of research is a fraud detection application that involves complex data and evaluation methods, and it serves as an introduction to the task. The latter two form more of a unit, in which we generalize and formalize the task. The third domain, in which the entities are part of a bipartite graph, is probably the most widely applicable to new problems. In all three, we show how real-world problems can be framed as instances of ASOUND, and we develop successful new scoring methods for data of the given form.

Within ASOUND, the performance of any algorithm will vary as a function of its pairwise scoring method, features used, the problem setup (the form of the inputs and outputs, and the evaluation measures), and the data (properties of the linked pairs and the singletons). Table 1.1 sketches how, across chapters, the thesis's focus shifts as we develop the higher-level story. Roughly speaking, other literature on ASOs tends to experiment with different methods and feature sets, selects from multiple performance measures, and explores the use of different constraints on algorithms' inputs and outputs. It rarely systematically varies the data properties themselves. In Chapter 2, we introduce a new real-world problem that can be described as an instance of ASOUND, then we proceed in the vein of previous literature, examining how findings vary across multiple scoring methods, feature sets, output requirements and evaluation measures.

	Vary problem setup		Vary methods		Vary problem space	
	Inputs, outputs	Evaluation	Features	Models	Properties of true pairs	Distribution of singletons
Previous literature	✘	✘	✘	✘		
Tribes (Chapter 2)	✘	✘	✘	✘		
Points (Chapter 3)				✘	✘	∼
Bipartite (Chapter 4)				✘	✘	✘

Table 1.1: High-level aspects of the pair detection problem addressed in each chapter and in other literature.

In Chapters 3 and 4, taking a more theoretical approach, we hold constant the problem setup and the feature set (using the full description of the entity, subject to what each model can represent), and instead we vary both methods and properties of the data set.

We develop synthetic model systems in which we can systematically explore how the data properties affect overall performance and the strength of different information sources. The real-valued points domain of Chapter 3 has practical applications, but it also serves as a demonstration problem for sketching out the abstract characteristics of ASOUND—namely, the domain-independent scoring approach and the axes along which the problem space varies. In Chapter 4, we return to bipartite data (also present in Chapter 2), here again analyzing the problem space, while also developing and comparing methods that can be applied to many problems.

1.4 Approach to Detecting ASOUND

In ASOUND, we are given a set of entities described by their properties in some domain. We assume the majority of entities can be treated as if they were independent samples from some distribution, which we call ϕ. A small number of pairs of entities have a latent tie. At times, we describe the scoring as if we were inferring the adjacency matrix of a graph, by distinguishing the "linked" pairs from the "independent" entities (or "singletons") among all "candidate pairs." Other times we use the terminology of a classification task, in which the input is a set of "pairs," "links," or "edges," and the purpose of the score is to differentiate the "positive" or "true" pairs from "negative pairs." The meanings should be clear in context.

1.4.1 Modeling Approach

The approach we develop in Chapters 3 and 4 uses generative models and a likelihood ratio. The first modeling decision to make in a given domain is choosing a suitable form for ϕ, the distribution of most entities. Once we specify the form of ϕ, we can estimate its parameters from the data set. Second, we must specify a generative model for the positive pairs. This model needs to be developed specifically for the domain. It should involve ϕ, so that the positives resemble the rest of the data, and it should produce two similar entities.

Then, the score for any pair (X_1, X_2)—the probability the pair is positive—comes out to be a (monotonic) function of the following likelihood ratio, which we abbreviate LR:

$$\mathrm{LR} = \frac{P(X_1, X_2 \mid \text{positive})}{P(X_1, X_2 \mid \text{negative})} = \frac{P(X_1, X_2 \mid \text{positive})}{P(X_1 \mid \phi) P(X_2 \mid \phi)}. \tag{1.1}$$

The numerator in this expression is the joint likelihood of X_1 and X_2 under the generative model for pairs, and the denominator is their likelihood under the model for independent singletons.

In Chapter 3, we present a domain-independent generative model of both singletons and pairs in a data set. We discuss how performing inference in that model leads to (1.1), once we require that each pair's score be computed independently. We then develop the component models $P(X \mid \phi)$ and $P(X_1, X_2 \mid \text{positive})$ for a Gaussian domain. In Chapter 4 we use the same LR with new component models for bipartite graphs.

Chapter 2 predates this formal model; it uses ϕ, but has no model of positive pairs. Its main probabilistic method, which is justified on intuitive grounds, is similar to Chapter 4's SharedWeight method (Section 4.4.3), but extended to handle temporal data.

Generative models are useful for ASOUND because they let us encode assumptions about the problem without needing training data. (In actuality, our models for positive pairs do require one parameter, but we offer guidance for when the parameter is unknown.) Another benefit of generative modeling is the ability to create synthetic data. In Chapters 3 and 4, we run experiments on synthetic data, and in addition, we analyze its theoretical distributions of positive and negative pairs.

1.4.2 Outputs and Evaluations

Of course we do need labeled data to evaluate algorithm performance. In Chapter 2, we exploit additional features in the data to evaluate the quality of the methods. In Chapters 3 and 4, we similarly use additional features of real data, this time to define the "ground truth" linked pairs. With synthetic data, we generate the ground truth labels along with the data.

In a naive implementation, to detect the true pairs in a data set, we would score every pair of entities. In real systems, since the number of pairs scales as $O(n^2)$ in the number of entities, finding ways to make the computation more efficient—mostly by reducing the number of pairs examined—is of enormous practical importance. Because this aspect of the problem is widely studied elsewhere, we do not focus on computational issues, except for requiring that individual scores be quick to compute. Section 5.7 briefly reviews existing techniques, such as blocking, that can be applied in practice. In Chapter 2, where the data set is large, we use a multi-pass approach, in which a cheap similarity criteria is used to enumerate candidate pairs, and only those pairs meeting some minimum threshold are scored fully; the remainder, for the sake of completeness, could implicitly be considered to have been assigned a lower score. In Chapters 3 and 4, we skirt these issues by keeping the number of entities per data set low enough that we can score all pairs.

Our scoring methods use probabilistic models, and the LRs of Chapters 3 and 4 can be converted into probability estimates for the links. However, none of our evaluation measures require the scores to be individually interpretable; all are based on the ranking of pairs induced by the scores. Evaluating based on rankings let us compare scoring methods of any type, probabilistic or not. In Chapter 2, we take the top k pairs, for different methods and values of k, optionally form clusters, and evaluate the pairs or clusters according to multiple criteria. In Chapters 3 and 4, we evaluate the quality of a ranking with respect to the ground truth pairs.

When comparing a ranking to true labels, we use AUC (area under the Receiver Operating Characteristics, or ROC, curve) as the measure of ranking quality. Its main selling point over alternatives such as precision or recall is its invariance to changes in the proportion of true positives [49]. This means we can allow the numbers of positive and negative pairs—and entities—to vary across experiments without inducing changes in the performance measure. In addition, AUC can be interpreted as a measure of the distance between two distributions, the positive and negative scores. In Chapter 4 we take advantage of this property to estimate AUCs for purely theoretical distributions.

1.5 Contributions

The main contributions of the thesis are the following:

Domain-independent task formalization and modeling. We introduce the ASOUND problem, a computational task that has appeared in many variations but has not previously been defined abstractly. We present a generative model in which ASOUND data comprises a mixture of pairs and singletons, and we use it to derive likelihood ratio (1.1) as the scoring function for arbitrary domains. To instantiate it requires creating a model of the true pairs, $P(X_1, X_2 \mid \text{positive})$, and estimating a model of the singletons, $P(X \mid \phi)$.

Demonstration of practical utility. In an initial, applied example of ASOUND, we detect small groups of connected people based on their career histories in the securities industry. These people have unusual sequences of shared jobs, they share jobs in multiple cities, and they have high and correlated risk scores for fraud, in comparison with the general population or with other groups.

Development of domain-specific scoring functions. We construct generative models of paired and singleton entities for Gaussian and bipartite (affiliation) data. The models enable us to generate synthetic data, and they are simple enough to keep the resulting scoring functions efficient.

Gaussian scoring. In Gaussian-distributed data, the LR score we derive is an interpretable function of the pair's distance apart, its distance from the origin, and the model's parameter for distance between true pairs.

Bipartite scoring. In bipartite data, the LR score we derive, called MixedPairs, differs from most existing similarity methods for binary vectors in that it satisfies three key properties: it is weighted based on the frequencies p_i of shared affiliations, it gives a small positive score for every affiliation that both people lack, and it extends symmetrically to affiliations that are present in over half the data.

Empirical evaluations. We show that LR and MixedPairs each give state-of-the-art performance at detecting pairs in real data along with synthetic. In real Gaussian data, an approximation to LR called $\frac{P(d|\epsilon)}{P(m|\phi)}$ is more robust when the distances between true pairs are unknown. In real affiliation data, three methods seem to be the most reliable: the cosine of the pair's *idf*-weighted vectors; the rarity of the pair's shared affiliations, which we call SharedWeight1100; and our adaptation of Pearson correlation, which we call Weighted Correlation.

Problem space analysis. We present some of the first-ever analyses of problem spaces for ASOUND. Specifically, we look at how characteristics of the data affect the performances of various methods at detecting pairs.

Task difficulty. We find that in both Gaussian and bipartite data, true pairs become easier to distinguish—for most scoring methods—as true pairs get more similar and as the number of dimensions increases. In bipartite data, a given affiliation becomes more informative as its frequency in the data approaches one-half.

Method properties. We characterize how the interaction between properties of data sets and properties of scoring methods can allow a method to perform near-optimally in some parts of the problem space, yet poorly in others.

- In Gaussian data, the LR score performs better than using distance or rarity alone, provided the true pairs are not too close together. Distance alone suffices when true pairs are very close together, and rarity alone suffices when true pairs are as far apart as independent points.
- In bipartite data, the three key properties above divide new and existing scoring methods into similarly-performing groups. The behavior of each group can be explained in part as a function of the group's properties and the data set characteristics. For example, the top-performing methods—MixedPairs, CosineIDF, SharedWeight1100 and Weighted Correlation—all satisfy the first two properties.

Next, we elaborate on the chapter by chapter contributions. In Chapter 2, we introduce the task of detecting anomalously similar objects via a specific application: identifying "tribes" of coworkers who move from job to job together, given the employment histories of 2.5 million registered representatives in the securities industry. We frame it as a problem of identifying pairs (or small groups) of people with anomalously similar job sequences. We contend that this task differs from clustering, and that a simple measure of career similarity, such as counting the number of jobs or years working together, will not suffice. Using a Markov process to model the career trajectories of independent people, we develop a score for pairs: the rarity of the shared components of their careers. Tribes are defined as the groups of people connected by high-scoring links. In evaluations, we show that the people in these tribes (as scored by either rarity or the number of jobs) share unusual sequences of jobs, work in more zip codes together, and have higher fraud risk scores, than other individuals. This work was first published in the Proceedings of the International Conference on Knowledge Discovery and Data Mining [53].

Following that applied setting, in Chapter 3 we move to a theoretical treatment of ASOUND. We derive a domain-independent generative model in which a data set is a mixture of singletons and linked pairs, and we show that approximate inference in this model leads to a likelihood ratio (the LR of (1.1)) we can use to score each pair. Instantiating this LR in a given domain requires models of singletons and positive pairs in that domain. We develop these generative models of singletons and pairs for Gaussian-distributed points in a k-dimensional space. When the models are plugged in, the LR turns out to be a function of just two features of a pair: its distance apart (the pair's similarity) and its (midpoint's) distance from the origin (the pair's rarity). We vary the problem space via the distribution of distances between true pairs. When true pairs are very close together, the task is easy, and it suffices to measure the distance between pairs. When they are as far apart as independent points, the task is harder, since the only information comes from the pair's rarity. At values in between, the LR combines the features to give better performance than either feature alone. As the dimensionality increases, true pairs become easier to detect. Finally, in real data, we find that the LR again improves on either feature alone. We also find empirically that an approximation to the LR, notated as $\frac{P(d|\epsilon)}{P(m|\phi)}$, is more robust than it when the true parameter is unknown. This chapter's material has appeared at the International Conference on Machine Learning [54].

In Chapter 4, we return to the domain of people and affiliations seen in Chapter 2, but now without the temporal aspect. We represent a person by a binary vector describing their set of affiliations. Following the approach of Chapter 3, we develop generative models of independent and paired binary vectors. Singleton vectors are sampled from a multiple

Bernoulli distribution. To generate paired vectors, we sample one vector independently, then create the second vector as a mixture between the first vector and the independent distribution. With these models, the LR for a pair of vectors decomposes into a sum over affiliations in the data set: LR_i is a function of the frequency p_i of affiliation i and of whether neither, one, or both of the people have the affiliation (i.e., whether the pair's values are 0|0, 1/0, or 1|1). This function has three useful properties. It is higher (the pair is more likely positive) when a shared affiliation is rarer; it is symmetric over the range of p_i, so a pair receives the same score for lacking a frequent affiliation as for sharing a rare one; and although the function is highest for (rare) affiliations shared by both people (1|1), it also treats 0|0 as mildly positive evidence, and 1/0 as negative evidence. We also introduce two more scoring functions that satisfy these properties: Weighted Correlation is an adaptation of Pearson correlation, and SharedWeight1100 is related to the shared rarity score of Chapter 2. Comparing these to a number of existing similarity methods, we find no others that have all three properties; that the methods' performance in experiments relates to which properties they satisfy; and that this new group, together with CosineIDF, seems to be the most robust.

We also analyze the difficulty of detecting pairs as a function of the characteristics of this domain. Like in Chapter 3, we find that ASOUND gets easier in synthetic data when true pairs are more similar and when the dimensionality of the data (the cardinality of the affiliation set) increases. In addition, it gets easier the closer each p_i gets to 0.5. That is, it is easiest to distinguish positive from negative pairs when each affiliation is held by about half the people. To validate these properties in real data, we run experiments that truncate the data sets to use only certain subsets of affiliations. The performance mostly degrades as anticipated; however, with the subset of affiliations having p_i closest to 0.5, the performance is unexpectedly strong, still competitive with results from the full affiliation set.

CHAPTER 2
TRIBES OF COWORKERS

In this chapter, we present a family of algorithms to uncover “tribes”—groups of individuals who share unusual sequences of affiliations. Given a database of employment histories, we search for pairs, and extend them to groups, of employees who were coworkers at multiple jobs. The aim is to distinguish those who worked together intentionally from those who simply shared frequently occurring employment patterns in the industry.

We apply the algorithms to a data set consisting of millions of employment records from the Financial Industry Regulatory Authority, the private organization that regulates stock brokers in the United States. The resulting tribes contain individuals who have higher risk score for fraud, whose risk scores are correlated within their tribes, and who are geographically mobile, all at significant levels compared to random or to other sets of individuals who share jobs.

The algorithms and evaluation techniques developed here are centered around a specific data set, but one could look for “tribes” in a number of domains. The important properties in the scenario are that “individuals,” or one type of entity, are connected to “affiliations,” another type of entity, and that the connections change over time. We form a model of “typical” sequences of affiliations, which allows us to calculate the likelihood of an individual having any given sequence of affiliations. Then, for each pair of individuals, we determine the sequence they have in common, if any. To score them, we compute the likelihood of that sequence, which is proportional to the likelihood that two independent individuals would share the given sequence by chance alone.

In contrast to the more common tasks of clustering or community detection, in this work we seek to infer fine-grained, strong associations that exist among only a fraction of the individuals within a larger data set. Other tasks with this structure could include finding students that select classes together, given a table of students and their enrollments;

inferring sets of cars traveling in caravan on a highway, given sightings at different locations and times [21, 102]; and discovering family structure in animal groups, from tagged individuals frequently sighted together [11, 22]. If we remove the temporal aspect of the problem and simply require a bipartite graph of affiliations, then a generalized version of the model could identify people with unusually similar tastes in books or movies, highly related documents sharing words that rarely co-occur, or friends within an album or yearbook containing photos of large groups.

This framework is particularly suited to situations involving large organizations, where the original data does not describe detailed associations among individuals. For instance, in our employment domain from the securities industry, thousands of people often share a loose relation of working at the same branch. In such cases, we can benefit from learning a model of typical affiliation patterns. Then, against this background, small groups doing unusual things stand out in contrast.

Note that in contrast to Chapters 3 and 4, which will require explicit models of connected pairs, here we try to set up the event space so that the pairs that are low-probability under the general model are exactly the pairs we want. We assume most people are independent, so the likelihood of two people's careers is the product of the likelihoods of the individual careers. We also assume that, for a given two people, we only need to pay attention to the jobs they share. This is reasonable since, for example, we are not interested in pairs of people who happen to have low-probability careers but never work together. To score a pair of people, we compute the likelihood of their shared jobs, ignoring how that intersection relates to either person's full career.

2.1 Motivation and Data

The Financial Industry Regulatory Authority (FINRA, known through 2007 as the National Association of Securities Dealers), regulates securities firms in the United States, with responsibility for preventing and discovering misconduct among its registered representatives, or "reps." With over 600,000 reps under its jurisdiction, FINRA must focus its investigatory resources on those reps most likely to commit fraud or other violations of securities regulations. In conversations over the course of related projects [48, 95], FINRA representatives suggested that fraud is sometimes committed by colluding groups of reps that have passed together through multiple places of employment. If we could identify "tribes" of reps moving together from job to job, we could test them for elevated rates of one or more indicators of fraud risk.

Of course, in finding tribes, we will certainly also identify harmless sets of friends that worked together in the industry, perhaps recruiting one another to new jobs. Our hope is that in the discovered groups, reps will tend to be homogenous with respect to fraud risk: mostly low-risk or mostly high-risk. Such tribes could then serve as starting points for detecting new fraud rings.

Our source data is a table of employment histories: for each rep, a series of records contains the branch identifier, start date, and end date for every employment the rep has held in the securities industry. The data set is large, containing (after some preliminary cleaning) 4.8 million records describing employments of 2.5 million reps at 560,000 branch offices. The branches range in size from 1 to 35,000 employees. The branch identities themselves have been inferred, through an earlier process of link consolidation from office addresses [48], from the 22,000 firms that have ever registered with FINRA. The employment histories span the twentieth century through 2006, though most records are from 1990 or later.

Two features of the real-world data shape our approach. First, many employment histories include simultaneous, overlapping jobs or leave gaps between employments. This muddies the concept of a transition between jobs: a rep does not necessarily leave one job when starting another, nor vice versa. Overlapping jobs are too common to consider discarding from the data: 20% of employees hold more than one job at some point, and 10% even begin multiple jobs (up to 16) on the same day. With transition dates ill defined, we cannot formulate this task as a search for employees changing jobs within a common interval of time. Therefore, we direct our attention to the times and places that people have been coworkers, as opposed to the boundaries between them.

The second key feature is that mass movements of employees between jobs are common. In addition to continual flows between firms (e.g., common career paths within a given city), the businesses change: branches are closed or opened; firms merge or are acquired. Reps in

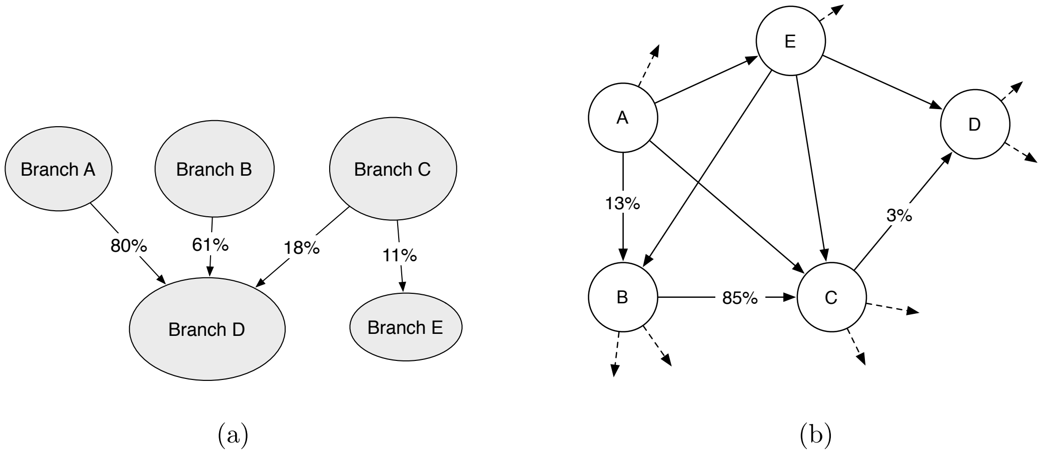


Figure 2.1: (a) Example of branch-branch transition patterns. The left-most edge indicates that 80% of the reps who ever worked at Branch A were later employed by Branch D. Only edges with high percentages are shown. (b) Schematic of Markov process for generating job sequences.

these flows could end up being colleagues at multiple organizations without even knowing each other. We can visualize such trends as transition diagrams, as in Figure 2.1a, to create a map of the whole industry. The meaning of the numbers along the edges will be discussed and refined in Section 2.3.3; roughly speaking, they indicate the percentage of employees at the source branch that later work at the destination branch.

Many of these transition percentages are high, which establishes that job movement in the industry is not random. For instance, among branches of fewer than ten employees, about 73% have some destination where at least 90% of the employees later end up. Among larger branches, 30% of the branches have some destination where at least 50% of their employees go. These figures increase slightly if we ask which transitions are common within a given year—to spotlight abrupt shifts like mergers—as opposed to throughout the lifetime of a branch office. This structured transition pattern is what we hope to factor out in order to find genuinely tight associations among individuals.

2.2 Task Definition and Approach

2.2.1 Formulation

In the most general setting, we define this task to be the identification of anomalously related entities. As input, we require a bipartite graph $G = (R \cup A, E)$ of entities $R = \{r_1, r_2, , r_n\}$ and affiliations (in this case, branches) $A = \{a_1, a_2, , a_k\}$. The entities should connect to at least several affiliations, on average, so as not to be too simply characterized. Each affiliation should attach to a large number of entities, enough so that the behavior of this set of entities can be modeled. The current formulation requires that an entity's affiliations be sequentially ordered (e.g., chronologically), while a more general extension would consider an unordered set.

The groups of entities we wish to return are those sharing unusual combinations of affiliations. Our strategy for this task revolves around developing a good definition of "unusual." For a pair or group to be considered anomalous, their shared affiliations need not individually be unusual, but the particular configuration of them should be; entities that are alike in typical ways are not part of our target. Our scores are similar in spirit to *tf-idf* weights in that they emphasize unusual shared affiliations [83]; however, our method for estimating joint likelihoods is unique. We use the co-occurrence rate (or transition rate) of each pair of affiliations to approximate a joint probability distribution over affiliations. Then, to measure the significance of a pair of entities, we compute the likelihood of their sequence of shared affiliations.

2.2.2 Tribe-Finding Process

We are given a bipartite graph $G = (R \cup A, E)$. In the FINRA data, these are reps and branches, and each edge $e \in E$ is annotated with a time interval: $e = (r_i, a_j, t\_start_{ij}, t\_end_{ij})$. The general tribe-discovery process, assuming we are given such time intervals, is summarized in Algorithm 1. It begins with listing all pairs $f_{ij} = (r_i, r_j)$ of individuals that have ever worked together. This can be a large list (2.6 billion pairs, in our case), generated by iterating through the branches and recording every pair of reps $f_{ij} = (r_i, r_j)$ whose employment stints at a branch overlap.

For each pair, we then summarize their coworker relationships, keeping track of the jobs where they coincide. We note additional information, such as the date the reps first coincide at each job and the total time spent at overlapping jobs. The algorithm stores the pairs in a new graph $H = (R, F)$, where $F = \{f_{ij}\}$, and each edge is annotated with:

$$\begin{aligned}\mathit{SharedAffilInfo}_{ij} = \{&\text{the sequence of jobs } \{a_x, a_y, \dots\} \text{ shared by } r_i \text{ and } r_j \\ &\cup \text{additional information described above}\}.\end{aligned}$$

For purposes of efficiency, we retain only the rep pairs that had at least three jobs in common. This leaves us a graph $H' = (R, F')$, with $|R| = 2.5$ million, and $|F'| =$ approximately 3 million pairs of individuals that are coworkers multiple times: the candidates for tribes.

The algorithm proceeds by identifying all significant pairs. For each edge in F', we compute a score Score-Pair($\mathit{SharedAffilInfo}_{ij}$) measuring how significant or unusual its sequence of shared jobs is. Section 2.3, which follows, discusses the choice of function to use for Score-Pair.

Once the significance scores are computed, we pick a threshold d for the scores and remove all edges f_{ij} for which $C_{ij} < d$. Then, we compute the connected components of H', which are designated the tribes. The output of the algorithm is a list of tribes: sets of reps within components of size two or higher in H'.

Computationally, Determine-Candidate-Pairs is the most expensive step. If the maximum degree of a branch is k, it requires enumerating and storing information about $O(|A|k^2)$ potential pairs. Once we have created the graph of pairs H and pruned it to a smaller H', the remaining steps are in $O(|F'|)$. Estimating the model parameters will generally require $O(|E|)$, one pass through the source data.

2.3 Scoring Methods

The choice of scoring methods constitutes the heart of the task. (We also use the term ranking method, since we use the scores only to rank the pairs.) Given a sequence of shared jobs such as in Figure 2.2a or 2.2b, we must decide whether it is unusual for a pair of coworkers to have worked together at all of these jobs. We do so for each rep pair by

Algorithm 1 Tribes algorithm.

```
function FIND-TRIBES(Reps $R$, Affils $A$, Edges $E$, Time Intervals $T$, Model Parameters
        $m$, Threshold $d$)
    /* Create graph $H \leftarrow (R, F)$ */
    ($F$, *SharedAffilInfo*) $\leftarrow$ DETERMINE-CANDIDATE-PAIRS($R, A, E, T$)

    /* Prune graph to create $H' = (R, F')$ and score edges */
    $(F', C) \leftarrow$ PRUNE-AND-SCORE-CANDIDATE-PAIRS($F$, *SharedAffilInfo*, $m$)

    return RECOVER-TRIBES($R, F', C, d$)

function DETERMINE-CANDIDATE-PAIRS(Reps $R$, Affils $A$, Edges $E$, Time Intervals $T$)
    $F \leftarrow \emptyset$, *SharedAffilInfo*$\leftarrow$ []
    for $a$ in $A$ do
        $R_a \leftarrow$ reps associated with $a$ in $E$
        for all pairs $(i, j)$ of reps in $R_a$ do
            if OVERLAP($T(a, i), T(a, j)$) then
                $F \leftarrow F \cup (i, j)$
                $SharedAffilInfo_{ij} \leftarrow SharedAffilInfo_{ij} \cup$ OVERLAPINFO($T(a, i), T(a, j)$)
    return $F$, *SharedAffilInfo*

function PRUNE-AND-SCORE-CANDIDATE-PAIRS(Rep Pairs $F$, Overlaps *SharedAffilInfo*,
        Model Parameters $m$)
    $F' \leftarrow \emptyset, C \leftarrow$ []
    for edge $f_{ij}$ in $F$ do
        if NUMBRANCHES($SharedAffilInfo_{ij}$) $\geq 3$ then
            $F' \leftarrow F' \cup f_{ij}$
            $C_{ij} \leftarrow$ SCORE-PAIR($SharedAffilInfo_{ij}, m$)
    return $F', C$

function RECOVER-TRIBES(Reps $R$, Candidate Pairs $F'$, Scores $C$, Threshold $d$)
    for edge $f_{ij}$ in $F'$ do
        if $C_{ij} < d$ then
            $F' \leftarrow F' - f_{ij}$
    return CONNECTEDCOMPONENTS($R, F'$)
```

computing a rarity score for their sequence of shared jobs. Note that we consider only the *shared* jobs, specifically those held *at the same time*, to matter in this calculation.

2.3.1 Simple Measures

Two straightforward methods for ranking the pairs are:

- JOBS = the number of jobs in the shared sequence.
- YEARS = the number of years of overlap.

Computing JOBS is a straightforward count of the job sequence. For YEARS, we add up the length of each overlap period, so that if a pair of reps works simultaneously at two branches for ten years, this counts as twenty years of overlap.

These scores are most naturally seen as raw numbers measuring rep similarity. But in the probabilistic setup that follows, we can also interpret them as measuring the likelihood that two independent reps would share this sequence of jobs.

2.3.2 Probabilistic Background Models

Our null hypothesis is that reps move independently of each other and according to some background model. The less likely a shared sequence of jobs is under the null hypothesis, the more likely it is that the rep pair is connected. Under the null hypothesis, if P(Rep 1 holds sequence s of jobs) = p, then P(Reps 1 and 2 each hold sequence s of jobs) = p^2. That term p^2 is thus what we should use for ranking the pairs. Since p and p^2 are rank-equivalent, it suffices to calculate and rank the pairs by p, the probability of a single rep holding a given (shared) sequence of jobs.

Each scoring function we discuss can be seen as arising from a different background model of movement. In this light, JOBS can be seen as stemming from a naive model of how reps choose employments. At each decision point, a rep either picks a new job, choosing among all branches with equal probability, or else stops working. Under this model, any given sequence of n jobs is equally likely and is more likely than a sequence of $n+1$ jobs.

The measure YEARS could arise from the following model: each day, a rep independently chooses a new job (which could be the same as the current job). Then, the longer the career of a rep, the less likely it is. This model is obviously not realistic, but it serves our needs with respect to scoring a pair of reps and their shared jobs: the more days a pair spends as coworkers, the larger the deviation from the model.

These simple methods treat all branches equivalently. As described earlier, however, when reps in the securities industry change jobs, they follow patterns caused by industry events and influenced by geographical and other factors. Accounting for these patterns motivates the next models.

2.3.3 Markov Model

A more complex model for how reps choose jobs is a Markov process: the flows of reps among jobs are described as transition probabilities among jobs, such as in Figure 2.1b. This inspires our third ranking function.

- PROB: Evaluate the probability of the shared jobs according to a Markov process.

If each rep held one job at a time, and changed it at each time step, we could model movement using an ordinary Markov process, as follows: each rep picks a start branch randomly. Then at each step, the rep's next branch is decided stochastically based only on the current branch. We ignore actual time spent at each job; at each step in the process, a rep either moves to a new branch or leaves the workplace. We also assume that transition probabilities are static over time.

This model satisfies a tradeoff between flexibility and performance. We want a model that flexibly mimics the characteristics of each branch without exactly reproducing the original data. In addition, the procedure must be tractable on a large data set. The process of computing all pairs of coworkers is time- and space-intensive, so it would be infeasible, for example, to generate random replicates of the network and recompute shared job sequences.

Using this model, we could estimate the probability of a rep having a job sequence such as in Figure 2.2a as

$$\begin{aligned} p &= P(\text{Branch A} \rightarrow \text{Branch B} \rightarrow \text{Branch C} \rightarrow \text{Branch D}) \\ &= p_A \cdot t_{AB} \cdot t_{BC} \cdot t_{CD}. \end{aligned} \tag{2.1}$$

The quantities in this expression are

$$\begin{aligned}
p_i &= P(\text{start at Branch } i) \\
&= (\text{\# reps ever at Branch } i)/(\text{\# reps in database}) \\
t_{ij} &= P(\text{transition from Branch } i \text{ to Branch } j \mid \text{currently at Branch } i) \\
&= (\text{\# reps who leave Branch } i \text{ and next go to Branch } j)/(\text{\# reps ever at Branch } i).
\end{aligned}$$

If job sequences in the database were as simple as Figure 2.2a, this model would be sufficient. However, Figure 2.2b is more typical of the data. The reps in this example start at the same branches, split apart for a few years, come back together, and then both begin two jobs (Branches C and D) at the same time. We would like to ignore Branches E, F, and G, which are not shared, but it is not obvious how to do this. To allow for these more complex situations, we adjust the model such that it is no longer exactly a Markov process but instead allows us to score just the four shared jobs. After this adjustment, our probability calculations (e.g., for Figure 2.2a) will remain almost the same.

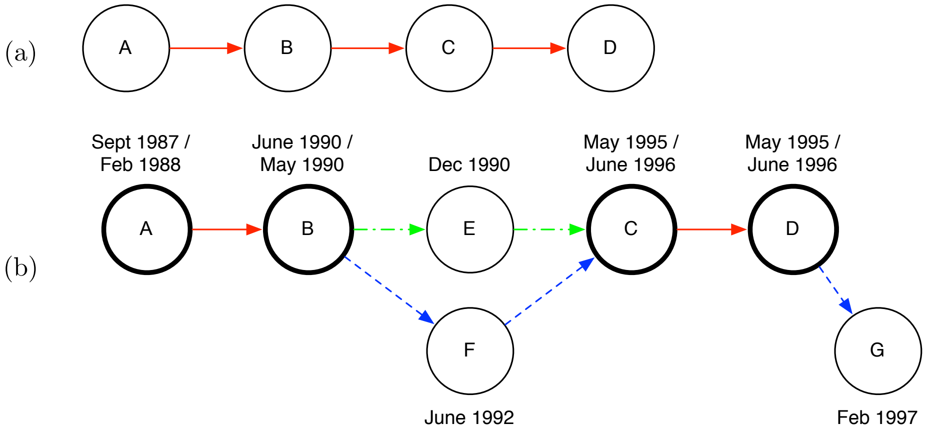


Figure 2.2: Job sequences to score. Nodes represent branches. In (b), two reps have differing trajectories, but only the shared jobs (in bold) are used for scoring. Labels show start dates for each rep at each job.

The first modification is to allow reps to have different paths between shared jobs, such as from Branch B to Branch C in Figure 2.2b. To do this, we replace the quantity t_{ij} with a new quantity v_{ij}:

$$\begin{aligned} v_{ij} &= P(\text{move to Branch } j \textit{ at any point} \text{ after Branch } i \mid \text{currently at Branch } i) \\ &= (\#\text{ reps who go to Branch } j \textit{ at any point} \text{ after working at Branch } i) \\ &\quad /(\#\text{ reps ever at Branch } i) \end{aligned}$$

Computing v_{ij}, the probability of an eventual transition, directly from the data is cleaner than an alternate approach that might attempt to sum up direct transition probabilities along all possible paths.

The other modification is to allow for simultaneous jobs. We treat the shared job sequences as if they are in a definite order, but the underlying situations can be complicated. For example, in Figure 2.2b, the reps work at Branches C and D simultaneously, not one after the other. To extend the model to handle these situations, we replace the quantity v_{ij} with a new quantity w_{ij}:

$$\begin{aligned} w_{ij} &= P(\text{move to Branch } j \text{ at any point } \textit{simultaneous to or after} \text{ Branch } i \\ &\qquad \mid \text{currently at Branch } i) \\ &= (\#\text{ reps who start at Branch } j \text{ at any point } \textit{simultaneous to or after} \\ &\qquad \text{starting at Branch } i)/(\#\text{ reps ever at Branch } i) \end{aligned}$$

Now, to score the pair shown in Figure 2.2b, we use the equation shown in Equation (2.1), but replace each t_{ij} with w_{ij}. In particular, we calculate $P(\text{Branch B } \rightarrow \text{Branch C})$ without regard for the intermediate branches. This is no longer a true, generative, Markov model; each $w_{ij} \geq v_{ij} \geq t_{ij}$, so the transition probabilities leaving a branch no longer sum to 1 ($\sum_i t_{ij} = 1, \text{but} \sum_i w_{ij} \geq 1$). Instead, it is an approximation: we factor the probability of a rep having a given sequence into these Markov-like terms. Using these estimates, we can compute an approximate score for any shared sequence of jobs, even if we have to ignore some jobs along the path.

2.3.4 Markov Models with Timing Variations

The PROB model, which includes the modifications described above, treats jobs in a sequence as being ordered by time, but it does not take into account when the transitions occur. We create two variations by changing the treatment of time.

First, we account for non-static transition probabilities. We hypothesize that the scoring will be more accurate if we can represent single-event mass movements, as well as changes in industry patterns over time. For instance, consider the case where 30% of reps at Branch A eventually move to Branch B, but in 1997 Branch A was purchased by Branch B, so 99% of the reps who were at Branch A in 1997 also worked at Branch B in 1997. To account for such variations, rather than scoring a transition based on a general probability of a rep moving from Branch A to Branch B, we describe a more specific event. Now, the rep is moving from Branch A at time X, to Branch B at time Y. (Specifically, the rep is first seen at Branch A at time X, and then first seen at Branch B at time Y which is equal to or later than time X.) Time is divided into bins, with bins representing one year or more. Each branch is given its own bin divisions, which depend on the number of employees at the branch in different years. We allocate the bins so that there are at least 10 people who worked at each branch in each bin period, provided the branch has had that many employees during its history.

The parameters needed for this new model, which we call PROB-TIMEBINS, require changing p_i and (again) w_{ij}. We now compute

$$\begin{aligned} p_{iX} &= (\#\text{ reps ever at Branch } i \text{ during time } X)/(\#\text{ reps in database}) \\ y_{iXjY} &= (\#\text{ reps ever at Branch } i \text{ during time } X \text{ and at Branch } j \text{ during time } Y, \\ &\qquad \text{where } Y \geq X)/(\#\text{ reps ever at Branch } i \text{ during time } X). \end{aligned}$$

We take the opposite extreme for the second variation. The PROB model is not very informed about time: because the w_{ij} values describe the probability of being at Branch j anytime after or simultaneous to being at Branch i, only the relative order of i and j matter. To find out how important that directionality of time is, we create a simpler model, PROB-NOTIME, which ignores even the order of job moves. For this model, we use the original p_i, with a new transition quantity z_{ij}, representing the probability that a rep works at Branch j *either before or after* they work at Branch i. However, there is an ambiguity in this formulation, because $z_{ij} \neq z_{ji}$. We can still score a *sequence* of jobs using the equivalent of Equation (2.1). Here, we keep the same temporal ordering of branches used in the other methods. But a model that disregards time should be able to score a *set* of jobs; it should be well-defined regardless of their order. This procedure is not. Nevertheless we include it here, since in the experiments, it turns out to work almost as well as PROB (see Section 2.4, next). Because a well-founded model for scoring unordered sets would allow this framework to be applied to situations without a time ordering, we return to this question in Chapter 4.

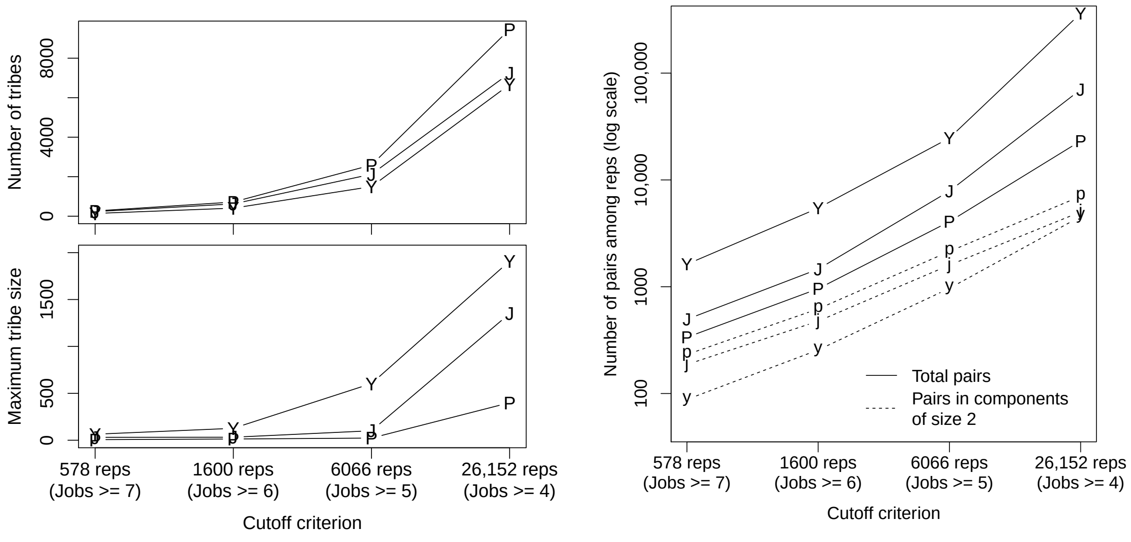
2.4 Evaluation and Results

Ideal tribes consist of reps that know each other and have coordinated their movements among jobs. Since we cannot verify the personal relationships among thousands of securities reps across the country, we evaluate our tribes using indirect measures. First, we examine structural characteristics of the tribes produced with the various scoring methods. Then, we analyze the tribes patterns of risk scores and geographic movement.

2.4.1 Tribe Structure

Using the process described in Section 2.2.2, we compile a list (the edges F') of the 3 million pairs of reps in the database that shared at least three different jobs. We rank these pairs using the five scoring functions described in Section 2.3: JOBS, YEARS, PROB, PROB-TIMEBINS, and PROB-NOTIME. All but JOBS give quasi-continuous values as scores. For these, we can choose a threshold d to keep any desired number of pairs. When we compute the connected components of these pairs, we get a set of tribes of assorted sizes and a corresponding set of reps in these tribes. For JOBS, the scores are discrete: all pairs have at least 3 jobs, and the maximum number of shared jobs is 25. To compare the various scoring functions fairly, for each continuous method we determine a cutoff d such that the resulting *number of reps in the tribes matches* (±1) the number of reps in tribes formed with JOBS. Below, we analyze the tribes corresponding to 4 different sizes of rep sets.

Figures 2.3a and 2.3b display structural characteristics of these tribes. We omit these characteristics for the variations on prob (prob-timebins and prob-notime), as they are substantially similar to those for prob. Figure 2.3a indicates that, for each matched set, prob creates more tribes, and smaller tribes, than jobs or years. Figure 2.3b further shows that the majority of pairs created by the prob ranking go into tribes of size two—pairs of associated reps. In contrast, jobs and even more so years, in order to get an equally large set of reps, provide many more pairs—edges in the graph F'—but the additional edges go to fill in the enormous components[1], instead of creating new, small groups.



(a)

(b)

Figure 2.3: Comparison of tribe structures, for equal-size sets of reps produced using jobs (J or j), prob (P or p), or years (Y or y). (a) Number of tribes and maximum tribe size. (b) Number of pairs, and number of pairs in two-person tribes.

We can see this effect from another perspective by considering the rarity of high-ranked job sequences. For jobs and prob, the scores are based solely on the job sequence; therefore, if a number of reps all share an identical job sequence, then the scores of their edges are equal. If that (shared) score passes the threshold, then the whole set of reps will be included

[1] Components with hundreds or even with dozens of nodes are unlikely to be the tightly-linked tribes we seek. In practice, we would probably disregard tribes with more than ten members. Dropping the larger tribes has little effect on the remaining analyses, so we leave them in.

in the tribes. For this reason, a ranking that scores frequent job sequences as significant will have large connected components among its tribes.

Table 2.1 shows the average, for each pair included in tribes, of the number of times its exact shared job sequence occurs among the 3 million pairs we scored. The low averages for the PROB ranking confirm that this model succeeds in scoring rare sequences as significant. JOBS also brings in fairly rare sequences. For YEARS, when one pair passes the threshold d, others with the same job sequence do not necessarily cross it, since the score depends on how long the coworkers are together. However, we see that the reps working together for the longest times tend to have common sequences of jobs. For comparison, among all 3 million pairs, job lists appear an average of 40.72 times, far fewer than among the pairs identified by YEARS.

Ranking	**Number of reps in tribes**			
	578	**1600**	**6066**	**26,152**
PROB	1.06	1.07	1.21	1.51
JOBS	1.16	1.35	2.05	4.31
YEARS	315.73	194.05	87.07	224.78

Table 2.1: Average number of times a job sequence occurs among all pairs of reps.

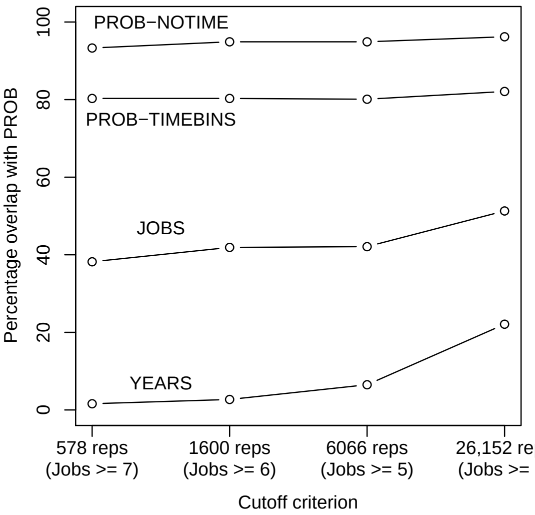


Figure 2.4: Percentage overlap of rep set with that from PROB.

Figure 2.4 gives a sense of how diverse the resulting tribes are. It shows, for several cutoffs, the percentage overlap between the set of reps produced by PROB and the equal-sized set produced by each other ranking. We see that the PROB variations, particularly PROB-NOTIME, give results fairly close to PROB. The rep sets created by JOBS are related but substantially different, while those of YEARS have almost no overlap.

2.4.2 Disclosure Scores

As part of their oversight, FINRA and other regulatory organizations require disclosures to be filed on reps for a variety of actions they commit and events that take place. These disclosures span categories such as customer complaints, bankruptcies, criminal charges, and regulatory actions; some are mundane and merely reflect administrative reporting requirements, while others represent serious breaches of trust. We can use these disclosures as assessments of past behavior or as predictors of future fraud risk. We compute a "disclosure score" for each rep as a weighted sum of their disclosures, where serious categories are weighted more highly (the weights were developed in consultation with FINRA); in this system, the vast majority of reps are assigned a score of zero.

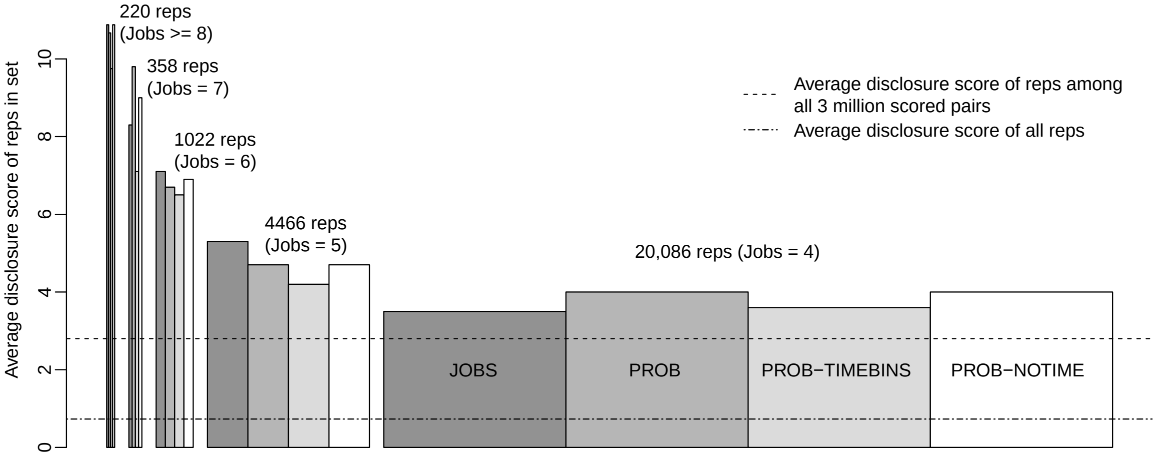


Figure 2.5: Disclosure scores of the top-ranked reps. Bar widths reflect the number of reps in each set.

When we examine the disclosure scores of reps in tribes, we find that they score well above average, and that the scores of reps at the top of the rankings are higher than those lower down. Figure 2.5 displays the average disclosure scores of various sets of reps. Similar to above, the reps are ordered and placed into bins based on which cutoff allows the rep to be included in the set of tribes. The bar widths correspond to the number of reps in the bin. For each cutoff, the four bars correspond to the top reps under JOBS, PROB, PROB-TIMEBINS, and PROB-NOTIME, respectively.

Results for YEARS are not displayed, as its scores are low: all fall below the higher dashed line. In fact, for its highest-ranked reps, the values are below the lower dashed line, and unlike with the other ranking systems, they rise as we move down the list of reps, reaching 2.4 for the largest set of reps. This implies that reps who have worked together for many years are the least likely of all to commit fraud.

One alternative explanation for the high disclosure scores seen among top reps is that these reps may simply have had longer careers than average, and so accumulated more disclosures. We tested this explanation by dividing all reps into groups based on their number of jobs held and number of years in the industry. Given a top-ranked set of reps from the tribes, we replaced the disclosure score of each rep with the average score from the rep's matched group, and recalculated the average for the set. If the matched disclosure scores were elevated, then our top-ranked reps would simply have long histories. In fact, the matched scores all give averages close to 2.8, the height of the dashed line, which means that the high scores are not caused by career length.

2.4.3 Disclosure Score Correlation within Tribes

If the tribes are of good quality and the conjecture is correct that reps at high risk of disclosures often move in tribes, then we would expect each tribe's disclosure scores to be homogenous. That is, disclosure scores of individuals within a tribe would be correlated: some tribes would have multiple members with high scores, while other tribes would have low scores. Judging tribes by the properties of their members' disclosure scores is not ideal, since the outcome depends on that second conjecture. In addition, since the frequency of disclosures is very low, under this lens only high-risk tribes look conclusively like high-quality tribes; low-risk tribes are hard to distinguish from random sets of reps. Finally, note the potential problem of incomplete information: reps that appear low-risk compared to their tribe-mates might just have evaded detection. It is precisely these individuals that the FINRA may be interested in investigating in the future.

We perform several experiments to test whether the tribes are homogenous with respect to disclosure scores. First, we examine individual pairs of reps, using a chi-square test to assess whether reps with positive disclosure scores pair with each other more often than expected at random. If we used all the pairs that formed tribes, then reps in large components would be represented more than once; to avoid this, we only perform this test on the tribes of size 2. Since the rankings all show significance at the $p \leq 10^{-7}$ level, we compare them using the phi-square statistic, which is chi-square normalized to have a maximum value of 1. By this measure, the pairs from all five rankings are about equally homogenous, as shown at the top of Figure 2.6a.

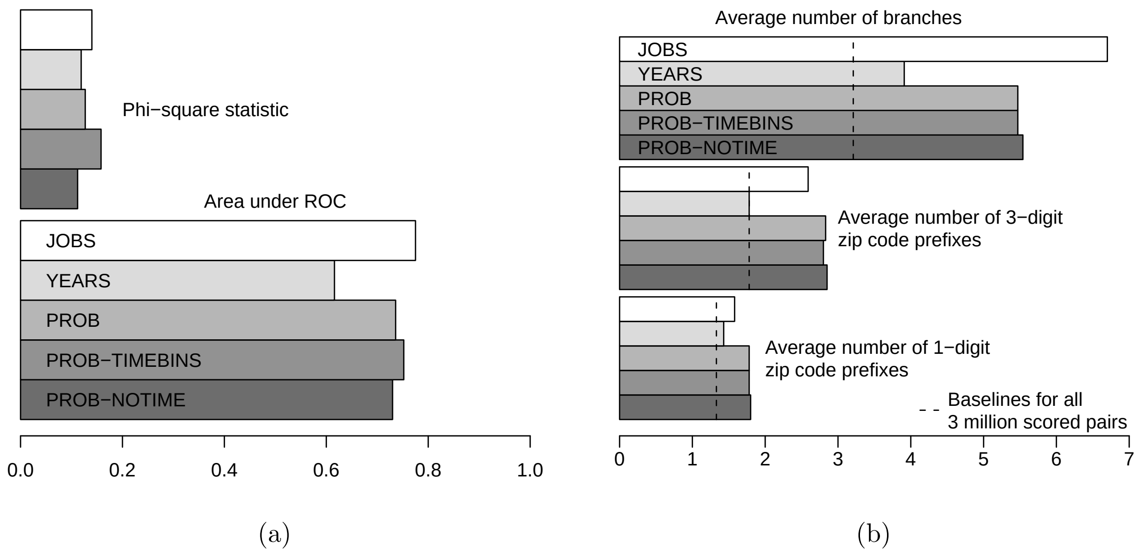


Figure 2.6: Comparison of tribe sets matched to have 1600 reps. (a) Tribe homogeneity, measured two ways. (b) Geographic mobility. For the distinct job sequences among the pairs, bars show average numbers of branches and zip code prefixes.

Next, we set up a prediction task with the tribes: we try to predict the disclosure score of each rep, using the average score of its tribe-mates. We can compute an AUC (area under the ROC curve) for these predictions if the classification task is binary. The AUC values shown are for the task "is the rep's score higher than the average among this set of reps?" By this measure, JOBS comes out a little more homogenous than PROB-TIMEBINS, followed by the other PROB rankings, and YEARS trails.

At the tribes sets shown, matched to have 1600 reps, PROB-TIMEBINS has a higher phi-square than the others, whereas JOBS gives the highest AUC. These results vary at other cutoffs. Phi-square remains highest for either PROB-TIMEBINS or JOBS, while the highest AUC alternates among JOBS and the PROB-based models.

2.4.4 Geographic Movement

The final indirect measure we use is the postal codes of the branches. If groups of reps travel geographically, particularly across large distances, this suggests they are staying together intentionally. Reps participating in the natural patterns of branch changes are less likely to move to far-off places together. We use the five-digit zip codes associated with most (96% of) branches as a way to estimate geographic movement. The first digit designates a broad region of the United States, and the first three correspond to a particular large city or local region. Counting the number of unique one-digit or three-digit zip code prefixes associated with a rep pair's list of shared branches gives an idea of the geographic mobility of the pair. As with disclosure scores, since we expect many high-quality tribes will not have geographic movement, this measure can only be used to evaluate tribes in the aggregate.

Figure 2.6b displays information about geographic movement. For each pair in the set, we calculate how many jobs they share, then how many (one-digit and three-digit) zip codes these jobs are in. The values shown are averages over the distinct shared job sequences in the set.

The pairs identified by PROB show the greatest mobility as measured by the number of zip codes covered. Even though the pairs from JOBS have the highest number of shared jobs—by the definition of that ranking—the shared jobs of PROB pairs are located in a larger number of zip codes. Pairs in the YEARS ranking move least of all, even less than the average among the 3 million scored pairs, which indicates that long-term coworkers tend to settle down. These long-term YEARS tribes—judging from their low disclosure scores, low overlap with the others, and low movement—do not seem to be the type of tribes we are looking for.

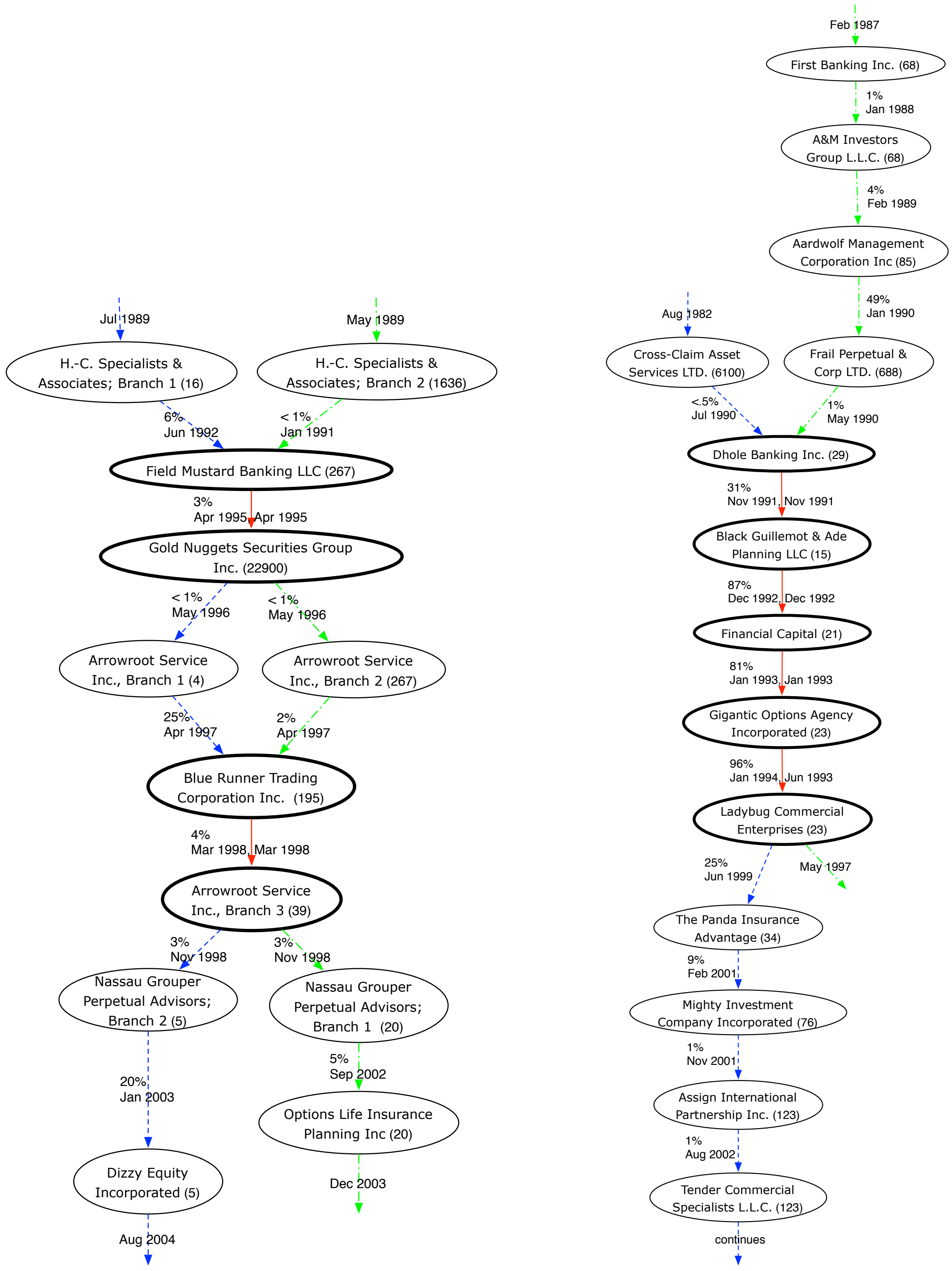
2.4.5 Discussion

To sum up, the rankings JOBS, PROB, PROB-TIMEBINS, and PROB-NOTIME create tribes whose reps have higher disclosure scores, on average, than random (Section 2.4.2). Reps with high (or non-zero) disclosure scores are associated in tribes with other such reps under these rankings (Section 2.4.3). The PROB models create tribes that visit more zip codes together, even though the JOBS tribes have more shared jobs (Section 2.4.4). The PROB models produce more individual pairs in tribes, while JOBS and YEARS produce larger connected components as tribes (Section 2.4.1).

The fact that the JOBS and PROB models perform comparably at various cutoffs, yet pick different sets of reps, suggests that there is room for improvement by combining the best of both systems. Of the tribes ranked highly by JOBS but not PROB, some, on inspection, appear to be just the types we hoped to avoid: pairs of reps taking a large number of very common transitions together. Others look like good tribes, and it appears PROB may miss them because of poor probability estimates at small branches. When both reps at a two-person branch move to the same new job, it is impossible to tell whether they moved together because their firm was bought, or because they wanted to stay together. The PROB model assumes the former, calculating the move as 100% likely to occur by chance, but this may not be the best policy. More generally, the PROB model seems to favor large firms, either because the probability estimates are more stable there, or perhaps because it is possible to create smaller transition probabilities from larger firms. We have not yet succeeded in correcting for this property, and the conclusion might be that the model is simply better suited for situations with large branches.

Qualitatively, many of the tribes look convincing when the reps' job histories are displayed together. It is a compelling feature that transition dates often coincide closely, even though the model did not use them.

As examples, Figures 2.7a and 2.7b display the career histories of two potential tribes. Each of these tribes consists of a single pair of reps. The pair in Figure 2.7a was scored by PROB as highly significant, while that in Figure 2.7b, even though it has a long history together and was ranked highly by JOBS, appears to be following typical patterns; it was scored as not significant by PROB. As it turns out, the reps from the significant pair have disclosure scores of 18 and 24, primarily since in April 1996 they were both fired (disclosures show an Internal Review and a Termination for each). One of the reps from


Jul 1989
May 1989
H.-C. Specialists & Associates; Branch 1 (16)
H.-C. Specialists & Associates; Branch 2 (1636)
6%
Jun 1992
< 1%
Jan 1991
Field Mustard Banking LLC (267)
3%
Apr 1995 Apr 1995
Gold Nuggets Securities Group Inc. (22900)
< 1%
May 1996
< 1%
May 1996
Arrowroot Service Inc., Branch 1 (4)
Arrowroot Service Inc., Branch 2 (267)
25%
Apr 1997
2%
Apr 1997
Blue Runner Trading Corporation Inc. (195)
4%
Mar 1998 Mar 1998
Arrowroot Service Inc., Branch 3 (39)
3%
Nov 1998
3%
Nov 1998
Nassau Grouper Perpetual Advisors; Branch 2 (5)
Nassau Grouper Perpetual Advisors; Branch 1 (20)
20%
Jan 2003
5%
Sep 2002
Dizzy Equity Incorporated (5)
Options Life Insurance Planning Inc (20)
Aug 2004
Dec 2003
Feb 1987
First Banking Inc. (68)
1%
Jan 1988
A&M Investors Group L.L.C. (68)
4%
Feb 1989
Aardwolf Management Corporation Inc (85)
Aug 1982
49%
Jan 1990
Cross-Claim Asset Services LTD. (6100)
Frail Perpetual & Corp LTD. (688)
<.5%
Jul 1990
1%
May 1990
Dhole Banking Inc. (29)
31%
Nov 1991 Nov 1991
Black Guillemot & Ade Planning LLC (15)
87%
Dec 1992 Dec 1992
Financial Capital (21)
81%
Jan 1993 Jan 1993
Gigantic Options Agency Incorporated (23)
96%
Jan 1994 Jun 1993
Ladybug Commercial Enterprises (23)
25%
Jun 1999
May 1997
The Panda Insurance Advantage (34)
9%
Feb 2001
Mighty Investment Company Incorporated (76)
1%
Nov 2001
Assign International Partnership Inc. (123)
1%
Aug 2002
Tender Commercial Specialists L.L.C. (123)
continues

(a) (b)

Figure 2.7: (a) Example tribe ranked highly by PROB but not by JOBS. (b) Example tribe ranked highly by JOBS but not by PROB. Nodes display branches and their sizes (in parentheses). Firm names, which are fictitious, indicate how some branches are related. Arrows leading into nodes show the starting dates of employment and the transition probabilities. Solid lines are moves executed by both reps in the pair; dashed lines are by one rep only.

the non-significant pair has no disclosures, while the other was fired in 1997 for "diversion of profitable trades to personal," for which they received a score of 12.

2.5 Conclusions and Extensions

One of the strengths of this work is that, beginning with no explicit knowledge of this industry, we can discover, model, and factor out typical job transitions, even though in real life these are caused by a combination of geography, career trends, and other factors. Moving forward, we could extend this model by incorporating external or domain-specific information. For example, we could consider relationships between reps who work in the same city but not at the same branch, or we could better handle some cases of reps having many simultaneous jobs given a better understanding of the industry and the data sources.

In this work, we had access to a complete history of employments and disclosures. In practical use, tribe identification will be an ongoing process, a situation we need to consider; it will be more difficult to recognize tribes when they have shared only a few jobs.

The most interesting aspect of this formulation, compared to related work, is our accounting for simultaneous jobs and different paths between the same jobs. We needed to allow for multiple affiliations starting and ending at arbitrary times, yet the model does not describe the network's changes day by day; instead, it observes certain discrete events (job transitions, and coworkers intersecting at a job) as time moves forward.

It may be worthwhile to incorporate more timing information, such as job durations, into the model, or other properties like the lengths of reps' non-intersecting careers. In the direction of simplifying, we plan to explore the time-oblivious version of the model (PROB-NOTIME) to see how well it can be applied to other types of tasks (see Chapter 4). In addition, we may incorporate a clustering or other dimensionality-reduction technique for the branches, either as an initial step in order to produce fewer but more robust transition probabilities, or afterwards to further analyze the resulting transition graph. We wish to investigate whether adjustments may improve the model's behavior with small branches. Finally, we hope to experiment with other domains and data sets.

CHAPTER 3
PAIRS OF POINTS IN GAUSSIAN DATA

In this chapter, we analyze the task of detecting ASOUND: detecting anomalously similar objects in unlabeled data. Variations of this task appear in such diverse areas as social network analysis, security, fraud detection, and entity resolution. To address ASOUND in a general form, we propose a simple, flexible mixture model in which most entities are generated independently from a distribution, but a small number of pairs are constrained to be similar. We predict the true pairs using a likelihood ratio that trades off the entities' similarity with their rarity. To investigate the properties of this model, we apply it to the problem of detecting linked pairs of points within a Gaussian distribution in $\mathbb{R}^k$. The likelihood ratio method always outperforms a method using only similarity; however, with certain parameter settings, similarity turns out to be surprisingly competitive. Using real data, we apply the model to detect twins given their birth weights and to re-identify cell phone users based on distinctive usage patterns.

3.1 Introduction

In Chapter 2, we saw that in determining which entities in a large data set are linked, identifying pairs with shared characteristics is only a first step. A remaining challenge is how to assess the significance of candidate pairs. We return to that question here.

Intuitively, a pair is more likely to be linked the more the entities are *similar* and the more the entities (or merely their shared aspects) are *rare*. (Pairs can also occur in dense regions, but those pairs will be less distinguishable.) Across the literature, numerous measures of pair strength have been developed. These usually describe the similarity of the entities, and sometimes also their rarity. Some measures are probabilistically based, and almost all are domain-specific.

In this chapter, we develop a framework for reasoning about pairs in arbitrary domains. In this framework, we explicitly model how both paired and non-paired entities are generated. Then we compare the paired and non-paired models using a likelihood ratio, a score that implicitly accounts for both similarity and rarity. We instantiate these models in the simplest of systems—continuous data with Gaussian distributions—in order to minimize domain-specific aspects and focus on these questions:

- Supposing we knew everything about a domain, how would ASOUND be solved optimally? (Section 3.2)
- Do we even need a model, or will a simple distance-only baseline be equally effective? If so, why and under what circumstances? (Section 3.4.3)
- As we approach realistic scenarios, in which the distance between pairs or the number of pairs is not known (Section 3.4.4), or in which the form of the model might not fit the data (Section 3.5), will this method still be feasible?

In Section 3.3, we present a generative model for continuous data in k dimensions, and for inference, a likelihood ratio score ("LR") to compute for every pair. In the synthetic data of Section 3.4, we find that one key parameter most affects performance: t, which describes how far apart the linked pairs may be. We compare LR to baseline methods that measure only similarity of pairs ("d", for distance), only rarity, or sub-optimal combinations of the two. Surprisingly, we find that d can perform almost as well as LR—that is, rarity doesn't matter—but only for the easiest problems, those with the smallest values of t. By examining the theoretical distributions of positive (i.e., linked) and negative (non-linked) pairs, we are able to explain why this happens.

Moving towards situations where parameters are unknown (and true labels might be unavailable), we examine performance when our estimate $\hat{t}$ mismatches the model and discover it governs the score's balance of similarity versus rarity. When the optimal t is unknown, the approximation we call $\frac{P(d|\epsilon)}{P(m|\phi)}$ is a robust alternative to the full LR; its numerator measures the likelihood of the distance between the pair, under the positive model, and its denominator describes the pair's rarity with respect to the data set. In Section 3.5 we apply the model to two real data sets constructed to be labeled instances of this task. As we vary $\hat{t}$, the performance trends are comparable to those in synthetic data. We find that both real data sets are in a middle range of difficulty, a range where performance is only moderate, but where LR distinctly outperforms d.

3.2 Domain-Independent Model

The model below makes the following assumptions, which are reasonable for many applications. First, the number of linked entities is low. Second, the linked entities appear only in disjoint pairs, not larger groups. Third, the non-linked entities—the vast majority—can be modeled as being independently generated from some distribution ϕ. Finally, the pairs can be modeled as being generated jointly according to some process θ. This θ should be specified, like in Section 3.3 below, in such a way as to involve ϕ but also keep pairs close together.

This framework is flexible, in that arbitrary domains and distributions can be swapped in depending on the choices of ϕ and θ. In Section 3.3 we will fully specify a model in which entities are Gaussian-distributed points, and we work with it in the remainder of this chapter. In Chapter 4, the entities will instead be binary vectors, and new distributions of ϕ and θ will be defined accordingly. We keep these models simple so that we can study the effects of parameter choices. However, more complex models could be developed to fit the needs of specific applications. Moreover, although for present purposes we want pairs to be close together, one could instead specify a θ that makes pairs be far apart or in another specific configuration.

3.2.1 Generative Process

The output will be n entities, $\mathbf{x}_1, \ldots, \mathbf{x}_n$, where some pairs are generated together. Let ϕ be the distribution of singletons. Let θ be the process for generating pairs. Two variables are unobserved: r, the actual number of pairs, and $C = \{c_{ij}\}$, a (binary) adjacency matrix describing which entities are in pairs. We control the number of pairs with the variable q, such that the expected number of pairs $E(r) = qn$.

When $c_{ij} = 1$ we say that the entities $\mathbf{x_i}$ and $\mathbf{x_j}$ form a *pair* (or a *link*), or equivalently, that the pair is *positive*. When $c_{ij} = 0$ we say that the entities are *singletons* or that the pair is *negative*.

Symbol	Meaning
n	Number of points
k	Number of dimensions
r and q	Number of positive pairs: $r \in [0, n/2]$, controlled by $q = \frac{\mathrm{E}(r)}{n}$.
$C = \{c_{ij}\}$	Adjacency matrix of true pairs. Each c_{ij} is 1 (positive) or 0 (negative).
ϕ and σ	Distribution of singletons: $\phi = \mathrm{Normal}(\boldsymbol{\mu}, \sigma^2 I)$.
ϵ and ν	Distribution of distances $\mathbf{d}_{ij}$ between positive pairs: $\epsilon = \mathrm{Normal}(\mathbf{0}, \nu^2 I)$.
θ	Joint distribution of positive pairs: $P(\mathbf{m}_{ij}, \mathbf{d}_{ij} \mid \theta) = P(\mathbf{m}_{ij} \mid \phi) P(\mathbf{d}_{ij} \mid \epsilon)$.
$t = \frac{\nu}{\sigma}$	Normalized parameter describing distance between true pairs: 0 is identical.
$m' = \frac{\lVert \mathbf{m}_{ij} - \boldsymbol{\mu} \rVert}{\sigma}$	Normalized distance from center to midpoint of a pair
$d' = \frac{\lVert \mathbf{d}_{ij} \rVert}{\sigma}$	Normalized distance between a pair

Table 3.1: Notation in this chapter.

The generative process is as follows. First, choose how many and which entities are in pairs.

1. Generate r, the number of pairs: $r \sim \mathrm{Binomial}(n/2, 2q)$. (With this proportion, $r \in [0, n/2]$, and $\mathrm{E}(r) = qn$.)
2. Generate $C = \{c_{ij}\}$ uniformly from among all adjacency matrices of r links where no point has > 1 link. Let $a_i \in \{0, 1\}$ indicate the number of links incident to point i in C.

At this stage, for each $\mathbf{x}_i$, we know whether it will be a singleton or part of a pair with $\mathbf{x}_j$.

3. Generate $\mathbf{x}_1, \ldots, \mathbf{x}_n$:
    (a) If $a_i = 0$, then generate $\mathbf{x}_i \sim \phi$.
    (b) For each pair (i, j) for which $c_{ij} = 1$, generate $(\mathbf{x}_i, \mathbf{x}_j) \sim \theta$.

This is essentially a mixture model for the data: one mixture component is a distribution of independent entities (ϕ), the other is a distribution of pairs (θ).

3.2.2 Inference

In this paper, we never explicitly infer r or C. Instead, to make inference efficient, we reason about each possible pair as if it were independent of the others.

For each individual pair $(\mathbf{x}_i, \mathbf{x}_j)$, we estimate the probability that it is positive. The quantity of interest is $p_{ij} = P(c_{ij} = 1 \mid \mathbf{x}_1, \ldots, \mathbf{x}_n, \phi, \theta, n, q)$, but it is easier to reason about it in odds form:

$$\text{PO} = \frac{p_{ij}}{1 - p_{ij}} = \frac{P(c_{ij} = 1 \mid \mathbf{x}_1, \ldots, \mathbf{x}_n, \phi, \theta, n, q)}{P(c_{ij} = 0 \mid \mathbf{x}_1, \ldots, \mathbf{x}_n, \phi, \theta, n, q)}.$$

PO, the posterior odds, is rank-equivalent to p_{ij}, and p_{ij} can be recovered at any time using $\frac{LR}{1+LR} = p_{ij} = P(c_{ij} = 1 \mid \mathbf{x})$. We will produce a probability estimate for each c_{ij}, and at evaluation time, compare these values to the true set $\{c_{ij}\}$. (From this point on, the model parameters ϕ, θ, n and q are implicit.)

We approximate, for every pair of points:

$$\begin{aligned}
\text{PO} = \frac{P(c_{ij} = 1 \mid \mathbf{x}_1, \ldots, \mathbf{x}_n)}{P(c_{ij} = 0 \mid \mathbf{x}_1, \ldots, \mathbf{x}_n)} &\approx \frac{P(c_{ij} = 1 \mid \mathbf{x}_i, \mathbf{x}_j)}{P(c_{ij} = 0 \mid \mathbf{x}_i, \mathbf{x}_j)} \\
&= \frac{P(\mathbf{x}_i, \mathbf{x}_j \mid c_{ij} = 1) P(c_{ij} = 1)}{P(\mathbf{x}_i, \mathbf{x}_j \mid c_{ij} = 0) P(c_{ij} = 0)} \qquad (3.1) \\
&= \frac{P(\mathbf{x}_i, \mathbf{x}_j \mid \theta) P(c_{ij} = 1)}{P(\mathbf{x}_i \mid \phi) P(\mathbf{x}_j \mid \phi) P(c_{ij} = 0)}. \qquad (3.2)
\end{aligned}$$

Line (3.1) is an application of Bayes' theorem. In (3.2), we use Step 3 of the generative model to write out the likelihoods for positive and negative pairs, respectively. Line (3.2) consists of a likelihood ratio (LR) term,

$$\text{LR} = \frac{P(\mathbf{x}_i, \mathbf{x}_j \mid \theta)}{P(\mathbf{x}_i \mid \phi) P(\mathbf{x}_j \mid \phi)},$$

and a prior odds term, $\frac{P(c_{ij}=1)}{P(c_{ij}=0)}$. Because the prior odds (see next paragraph) is constant throughout a data set, going forward we will focus on the likelihood ratio term.

The term for the prior $P(c_{ij} = 1)$ is r divided by the total number of pairs, which is $\frac{2r}{n(n-1)}$ when r is known. When r is unknown, we compute $P(c_{ij} = 1)$ from q by marginalizing over possible values[1] of r (see (3.3)). In (3.4), $P(r = h \mid q)$ is expanded using

[1] Note that the summation omits the term $h = 0$. Although our process can generate data sets having $r = 0$, we discard those samples because our performance measure is only defined in the presence of positive pairs.

$r \sim \text{Binomial}(n/2, 2q)$. In either case, $P(c_{ij} = 0) = 1 - P(c_{ij} = 1)$.

$$P(c_{ij} = 1 \mid q) = \sum_{h=1}^{n/2} P(r = h \mid q) P(c_{ij} = 1 \mid r = h) \tag{3.3}$$

$$= \sum_{h=1}^{n/2} \binom{n/2}{h} (2q)^h (1 - 2q)^{n/2-h} \frac{2h}{n(n-1)} \tag{3.4}$$

3.2.3 Limitations of this Inference Method

The output of inference is a list of likelihood ratios, one for each potential pair. We can convert the likelihood ratios to probability estimates, and we can turn them into a discrete set of positive pairs, if desired, by thresholding the scores.

One drawback to treating each pair as independent is that, in violation of the generative model, the resulting (thresholded) adjacency matrix $\hat{C}$ may assign some entities to more than one pair. We could remedy this situation with additional post-processing (instead of or in addition to the thresholding), keeping only the highest-probability links. Alternatively, we could reconsider the model's assumptions: if an entity is matched to more than one pair, we may have underestimated ϕ in that region or the entities may actually belong to a group of more than two. It could be seen as a strength if the method is able to detect such groups when the generative process only describes pairs.

Another way to avoid assigning any point to more than one pair would be to infer the full C: compute $P(C_l \mid \mathbf{x}_1, \ldots, \mathbf{x}_n)$ for every valid matrix C_l and choose the one with maximum likelihood. This would be computationally challenging: for a typical data set in this paper, there are more than 1.6×10^{16} such matrices.

Another simplification is that we are modeling all negative pairs as if they were formed by two singletons. In truth, of the $\frac{n(n-1)}{2} - r$ negative pairs, $2r(n - r - 1)$ of them involve at least one entity from a positive pair. As r rises from 1 to $\frac{n}{2}$, the fraction of non-modeled pairs increases from near-0 to near-all of them. In Section 3.4.4, we discuss how these non-modeled negatives can under certain circumstances affect performance.

3.3 Model for Gaussian-Distributed Data

Next, we instantiate the model of Section 3.2 for a domain of points in Euclidean space. The entities $\mathbf{x_i}$ will now be points in $\mathbb{R}^k$, and we set the distribution of independent points to be a radially symmetric normal: $\phi = \text{Normal}(\boldsymbol{\mu}, \sigma^2 I)$.

For positive pairs, we define θ to use ϕ, so that the linked entities roughly resemble singletons. To generate points $\mathbf{x_i}$ and $\mathbf{x_j}$, the process θ is:

1. Generate $\mathbf{m}_{ij} \sim \phi$.
2. Generate displacement vector $\mathbf{d}_{ij} \sim \epsilon$.
3. Set $\mathbf{x}_i = \mathbf{m}_{ij} + \mathbf{d}_{ij}$ and $\mathbf{x}_j = \mathbf{m}_{ij} - \mathbf{d}_{ij}$.

Here, θ samples from ϕ to generate the midpoint of a pair, then it displaces the individual points symmetrically about that midpoint. Within this θ, we must specify ϵ, the distribution governing the size of that displacement. We define ϵ to be another radially symmetric normal: $\epsilon = \text{Normal}(\mathbf{0}, \nu^2 I)$.

Returning to the likelihood ratio from Eq. (3.2), we can now expand its numerator:

$$\text{LR} = \frac{P(\mathbf{x}_i, \mathbf{x}_j \mid \theta)}{P(\mathbf{x}_i \mid \phi) P(\mathbf{x}_j \mid \phi)} = \frac{\frac{1}{2^k} P(\mathbf{m}_{ij} \mid \phi) P(\mathbf{d}_{ij} \mid \epsilon)}{P(\mathbf{x}_i \mid \phi) P(\mathbf{x}_j \mid \phi)}. \tag{3.5}$$

The generative process for positive pairs was described in terms of $\mathbf{m}_{ij}$ and $\mathbf{d}_{ij}$, so the most natural way to write its likelihood function would be $P(\mathbf{m}_{ij}, \mathbf{d}_{ij} \mid c_{ij} = 1) = P(\mathbf{m}_{ij} \mid \phi) P(\mathbf{d}_{ij} \mid \epsilon)$. However, since Lines (3.1) and (3.2) are written as functions of $(\mathbf{x}_i, \mathbf{x}_j)$, we have to perform a change of variables; the mapping is one-to-one but introduces the constant $\frac{1}{2^k}$ (see Lemma A.0.1 in Appendix A).

It turns out that the likelihood ratio can be simplified since the underlying distributions are normals. To see this, next, we plug in normal probability density functions for the terms involving ϕ and ϵ:

$$P(\mathbf{m}_{ij} \mid \phi) P(\mathbf{d}_{ij} \mid \epsilon) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^k e^{-\frac{\|\mathbf{m}_{ij} - \boldsymbol{\mu}\|^2}{2\sigma^2}} \left(\frac{1}{\sqrt{2\pi}\nu}\right)^k e^{-\frac{\|\mathbf{d}_{ij}\|^2}{2\nu^2}} \tag{3.6}$$

$$P(\mathbf{x}_i \mid \phi) P(\mathbf{x}_j \mid \phi) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^k e^{-\frac{\|\mathbf{x}_i - \boldsymbol{\mu}\|^2}{2\sigma^2}} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^k e^{-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}\|^2}{2\sigma^2}} \tag{3.7}$$

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{2k} e^{-\frac{m^2 + d^2}{\sigma^2}}. \tag{3.8}$$

For Eq. (3.8), we have defined $m = \|\mathbf{m}_{ij} - \boldsymbol{\mu}\| = \|\frac{(\mathbf{x}_i + \mathbf{x}_j)}{2} - \boldsymbol{\mu}\|$ and $d = \|\mathbf{d}_{ij}\| = \|\frac{(\mathbf{x}_i - \mathbf{x}_j)}{2}\|$ (dropping the subscript ij when it is clear from context) and applied Lemma A.0.2.

Substituting these expressions back into (3.5) gives:

$$\begin{aligned} \mathrm{LR} &= \frac{\frac{1}{2^k}\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^k e^{-\frac{m^2}{2\sigma^2}} \left(\frac{1}{\sqrt{2\pi}\nu}\right)^k e^{-\frac{d^2}{2\nu^2}}}{\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{2k} e^{-\frac{m^2+d^2}{\sigma^2}}} \\ &= \left(\frac{\sigma}{2\nu}\right)^k e^{\frac{1}{2}\left(\frac{m^2+2d^2}{\sigma^2} - \frac{d^2}{\nu^2}\right)}. \end{aligned} \tag{3.9}$$

The likelihood ratio in Eq. (3.9) is fairly simple: instead of depending on the full data vectors $\mathbf{x}_i$ and $\mathbf{x}_j$—$2k$ coordinates in all—it uses just two measures of the pair, m and d.

We assume (for now) that the model parameters are available at inference time. Among them, n and q (or r) affect only $\frac{P(c_{ij}=1)}{P(c_{ij}=0)}$. Changing them affects the individual scores, but not the ranking. We also need σ and ν. However, it turns out we can rewrite the score as a function of their ratio $t = \frac{\nu}{\sigma}$. Eq. (3.10) shows the final, reparametrized LR as a function of $m' = \frac{m}{\sigma}$, $d' = \frac{d}{\sigma}$, and $t = \frac{\nu}{\sigma}$ without σ:

$$\mathrm{LR} = \left(\frac{1}{2t}\right)^k e^{\frac{1}{2}\left(m'^2 + d'^2\left(2 - \frac{1}{t^2}\right)\right)}. \tag{3.10}$$

3.4 Applying the Model to Synthetic Data

In this section, we study the behavior of the algorithm when the data has been generated by the model. We will address (a) how the task's difficulty is affected by model parameters (primarily t, but also the dimensionality k, the number of points n, and the number of pairs q or r) (Section 3.4.2); (b) how the score for an individual pair varies as a function of t and its (m', d') values (Section 3.4.3); and (c) how performance is affected by changing the value $\hat{t}$ used during inference (Section 3.4.4).

3.4.1 Experimental Setup

We evaluate performance by comparing the ranked list of predicted pairs to the set of true pairs, calculating the AUC (area under the ROC curve) of the ranking. We considered other common measures of ranking such as average precision or Hand's H measure, but they were unsuitable because, unlike AUC, they fluctuate when the number of true positives or negatives does [49, 60]. In realistic scenarios, it may also be important to focus attention on the very top of the ranked list or on the individual probability estimates. These paths are left to future work.

For present purposes, the ranked list contains all pairs. In larger data sets, efficiency would become a concern, as it is in entity resolution. Existing techniques from that literature address efficiency either by making the score calculation faster or by scoring only those subsets of pairs that are judged similar according to some preliminary measure [31, 46]. See Section 5.7 for further details. A method known as "binning" can be used for continuous data: in each dimension, create overlapping bins for the data, and only consider pairs that lie within the same bin in some dimension [85]. For the data sets in this paper and practical values of parameters, applying this method, i.e., filtering out pairs having high d, would probably bring gains in efficiency at little loss to performance.

For synthetic data experiments, given any parameter setting of n, q and t, we generate 100 data sets from the model. Within each data set, we score every pair and evaluate the AUC of the ranked list compared to the true pairs. These experiments use $k = 2$ dimensions and (without loss of generality) $\sigma = 1$.

3.4.2 Performance on Synthetic Data

The likelihood ratio of Eqs. (3.2) and (3.10) is the Bayes estimate for distinguishing positive from negative pairs, so it should perform close to optimally, depending on how closely the data matches the two modeled classes. We compare it to four baseline methods.

One, d, measures only the similarity of points in a pair: it ranks by d_{ij}, the distance between the points, with smaller distance meaning more likely positive. The second, m, measures only the rarity (i.e., local sparseness) of the pair: it ranks by m_{ij}, the distance from the origin to their midpoint, with higher distance meaning more likely positive. It can be seen from Eq. (3.10) that using m (or m') is rank-equivalent to using LR if d' is held constant. Likewise, using d (or d') is rank-equivalent to using LR if m' is held constant—provided that $\frac{1}{t^2} > 2$, or $t < {}^{1}\!/\!\sqrt{2} \approx 0.71$. Generally we will use $t \ll 1$, so this will be the case.

The third baseline, called $LR[d]$, is a likelihood ratio designed to take into account only d, not m. It is computed as $\frac{P(\mathbf{d}|c_{ij}=1)}{P(\mathbf{d}|c_{ij}=0)}$. This is intended as an analogy to a Fellegi-Sunter model, in that it takes the likelihood ratio of a similarity function of a pair (see Section 5.4) [50]. For the synthetic data, the score is similar to Eq. (3.10), but the discriminant function in the exponential reduces to $d'^2\left(2 - \frac{1}{t^2}\right)$. The fourth baseline, $\frac{P(d|\epsilon)}{P(m|\phi)}$, is an intuitive if naive way to combine the the terms for similarity and for rarity. But it is actually a reasonable approximation to the full LR of Eq. (3.5) when d is small enough, because in that case $P(m \,|\, \phi) \approx P(x_i \,|\, \phi) \approx P(x_j \,|\, \phi)$ and the terms cancel out. In the synthetic data, this method is rank-equivalent to $\left(m'^2 + d'^2\left(\frac{-1}{t^2}\right)\right)$.

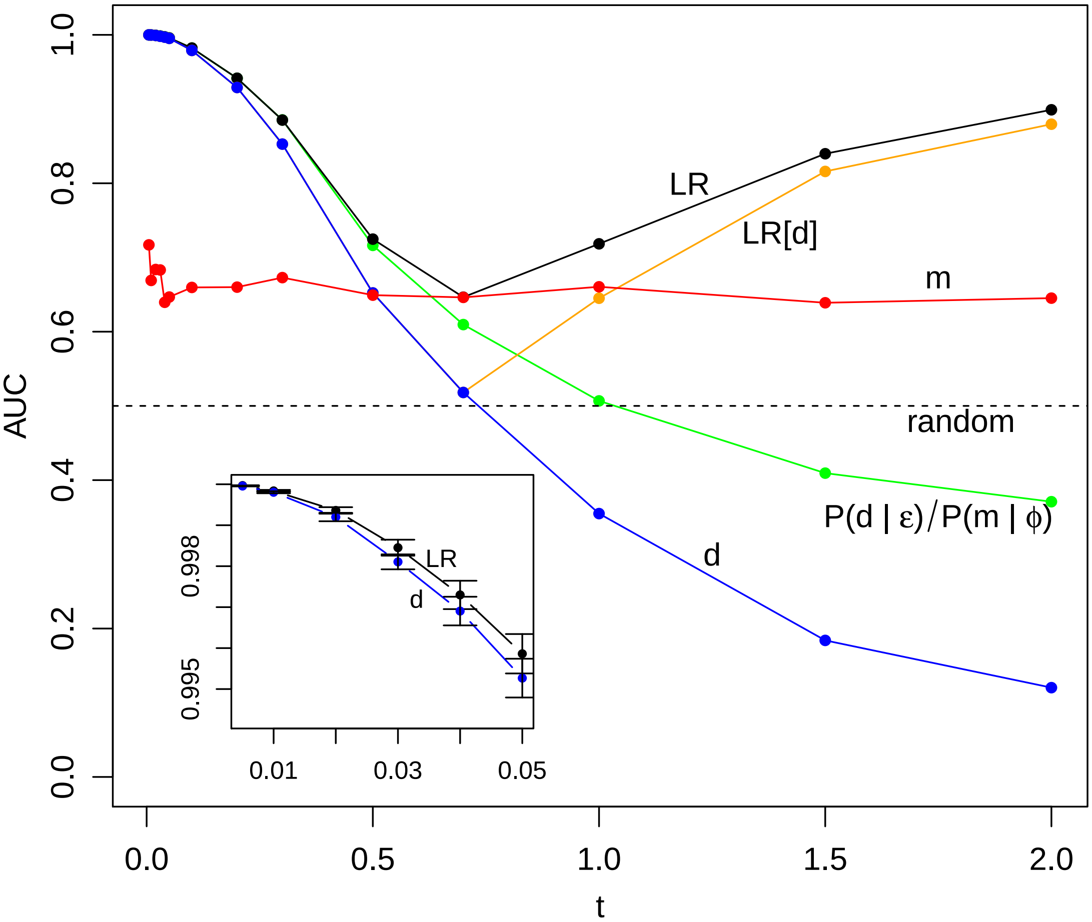


Figure 3.1: AUC as a function of t, the parameter describing distance between positive pairs, for five methods. Each point is the average of 100 trials. Inset shows a close-up of the smallest values of t, with error bars indicating 95% confidence intervals. In the inset, $P(d \,|\, \epsilon)/P(m \,|\, \phi)$ would be visually indistinguishable from LR. Parameters are $n = 200$, $E(r) = 4$, and $\sigma = 1$.

Figure 3.1 shows performance as we vary t for one setting of (n, q). (Other settings were similar.) The results can be divided into three realms. First, when t is very low (see inset), the AUCs of both LR and d are almost perfect. LR is always above d, but they are nearly indistinguishable. Next, as t approaches $1/\sqrt{2}$, both LR and d drop, and they diverge; at its minimum value, LR matches m, while d is nearly 0.5, or random. When $t > 1/\sqrt{2}$, LR increases again, while d continues to decrease, now ranking pairs in the wrong order. Meanwhile, m is much lower and steady. The third and fourth baselines each partially augment d: $LR[d]$ is identical except that it changes the direction of ranking at $1/\sqrt{2}$, and $\frac{P(d|\epsilon)}{P(m|\phi)}$ incorporates m, so it performs near optimal for low t, but it does not change direction at $1/\sqrt{2}$.

3.4.3 Understanding Performance

Conceptually, we can explain why $t = 1/\sqrt{2}$ is always a turning point, regardless of the form of ϕ. In each dimension l, $d_l = \frac{x_{il} - x_{jl}}{2}$, so for negative pairs, since x_i and x_j are independent, $\mathrm{E}(d_l|-) = 0$ and $\mathrm{Var}(d_l|-) = \frac{1}{2}\mathrm{Var}(x_l) = \frac{\sigma_l^2}{2}$. For the positive pairs, by definition $\mathrm{E}(d_l|+) = 0$ as well, while $\mathrm{Var}(d_l|+) = \nu_l^2 = (t\sigma_l)^2$. When we set $t = 1/\sqrt{2}$, the positives' $\mathrm{Var}(d_l|+) = \frac{\sigma_l^2}{2}$ matches that of the negatives. In these experiments, at $t = 1/\sqrt{2}$, not only do the means and variances of d match, but both the positive and negative distributions of d_l are normals; i.e., the distributions of d are identical for positive and negative pairs. Therefore d contains no distinguishing information, and LR is only using m. At higher t, the positives become farther apart, on average, than the negatives.

We next examine how the LR score of an individual pair combines the two measures of it, m' and d'. Figure 3.2 shows that the score increases when m' increases; for the boxes in which $t < 1/\sqrt{2}$, the score increases when d' decreases, and when $t > 1/\sqrt{2}$, the score increases when d' increases, as discussed above. At $t \approx 1/\sqrt{2}$ the contour lines are vertical, which shows visually that the only information is contained in m. Now, consider the smallest setting of t, in which empirically d performs almost as well as LR. The contour lines in the first box are almost horizontal, indicating that d' contains almost all the information (in the LR score, $\frac{d'^2}{t^2} \gg m^2$). This dominance of d' explains why the two methods are almost indistinguishably strong.

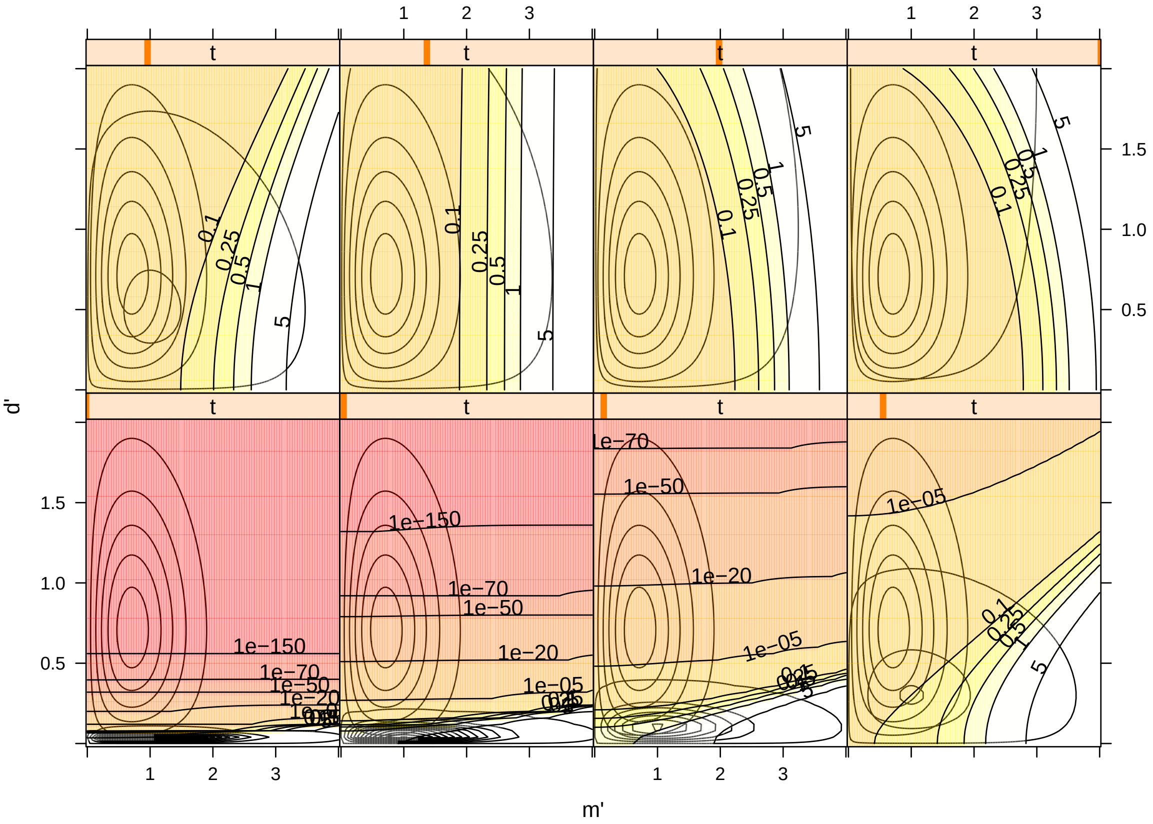


Figure 3.2: Color and labeled contour lines: likelihood ratio assigned as a function of (m', d') when $n = 25$, $\mathrm{E}(r) = 10$. Higher $P(c_{ij} = 1 \mid m', d')$ is whiter. Within each box, left contour lines: density function for negative pairs; bottom/middle contour lines: density function for positive pairs. Top orange bar: relative values of t (from bottom left) across (0.02, 0.05, 0.1, 0.3, 0.5, 0.7, 1, 2).

Figure 3.2 becomes more informative once we know not only what score is assigned to a given position, but also the distributions of positive and negative pairs along these axes. It turns out that with normal distributions for ϕ and ϵ in $\mathbb{R}^k$, the distributions of positive and negative pairs have closed forms (full derivations are in Section A). Each distribution is a product of two independent χ_k distributions, one describing m', one describing d':

$$P(m' \mid \phi)P(d' \mid \epsilon) = \left(\frac{1}{t}\right) \chi_k(m')\chi_k\left(\frac{d'}{t}\right) \tag{3.11}$$

$$P(m' \mid \phi)P(d' \mid \phi) = 2\chi_k(m'\sqrt{2})\chi_k(d'\sqrt{2}). \tag{3.12}$$

The peak of χ_k is at $\sqrt{k-1}$. Since $k = 2$ here, that peak is at $(1, t)$ for the positive pairs and $({}^{1}\!/\!\sqrt{2}, {}^{1}\!/\!\sqrt{2})$ for the negatives. As t changes, the only effect is on the d' dimension of the positives. Visually, it is clear that the distributions are well separated at small t and begin to overlap as t grows. In higher dimensions, the distributions become better separated (see Appendix B), so the task should become easier as k increases.

3.4.4 Sensitivity to Parameters and to Assumptions

When n increases or q decreases, intuition suggests that since true pairs are less frequent, the problem might get harder. However, since AUC is unaffected by changes to class proportions, a glance at the class distributions of Figure 3.2 should help solidify the (more relevant) intuition that changing the number of positives or negatives will not affect the separation between the classes. At inference time, if we mis-guess q, the probability estimates for pairs change, but the ranking of LR scores does not.

At data generation time, the situation is more subtle. For a given n, as the number of pairs increases towards $n/2$, the performance of LR can actually decrease—but only for large $t > {}^{1}\!/\!\sqrt{2}$. This is due to interference of the non-modeled pairs described in Section 3.2.3: when t is large, the positive points no longer resemble the singletons, and when enough points become part of positive pairs, the majority of negatives no longer resemble the modeled negatives. However, we observe no such performance effects with smaller t.

In many realistic problem scenarios, we will not know q nor, more importantly, t. Figure 3.3 shows how performance degrades when using an incorrect value $\hat{t}$ for inference. For

LR, $\hat{t}$ determines the balance between d' and m', and the direction of d''s effect. When $\hat{t}$ approaches 0, LR approaches d; when $\hat{t}$ reaches $1/\sqrt{2}$, LR matches m, then continues to drop; and the optimal is in between, at the true t. For $\frac{P(d|\epsilon)}{P(m|\phi)}$, performance is surprisingly robust: when $\hat{t}$ is underestimated, performance drops just like LR's, but when $\hat{t}$ is overestimated, $\frac{P(d|\epsilon)}{P(m|\phi)}$ remains high. This is because $\frac{P(d|\epsilon)}{P(m|\phi)}$ has no turning point in its use of d: as $\hat{t} \to \infty$, $\frac{P(d|\epsilon)}{P(m|\phi)}$ merely puts less weight on d and eventually converges to m. Meanwhile, $LR[d]$ simply matches d, and its AUC flips to $(1-d)$ when $\hat{t} > 1/\sqrt{2}$.

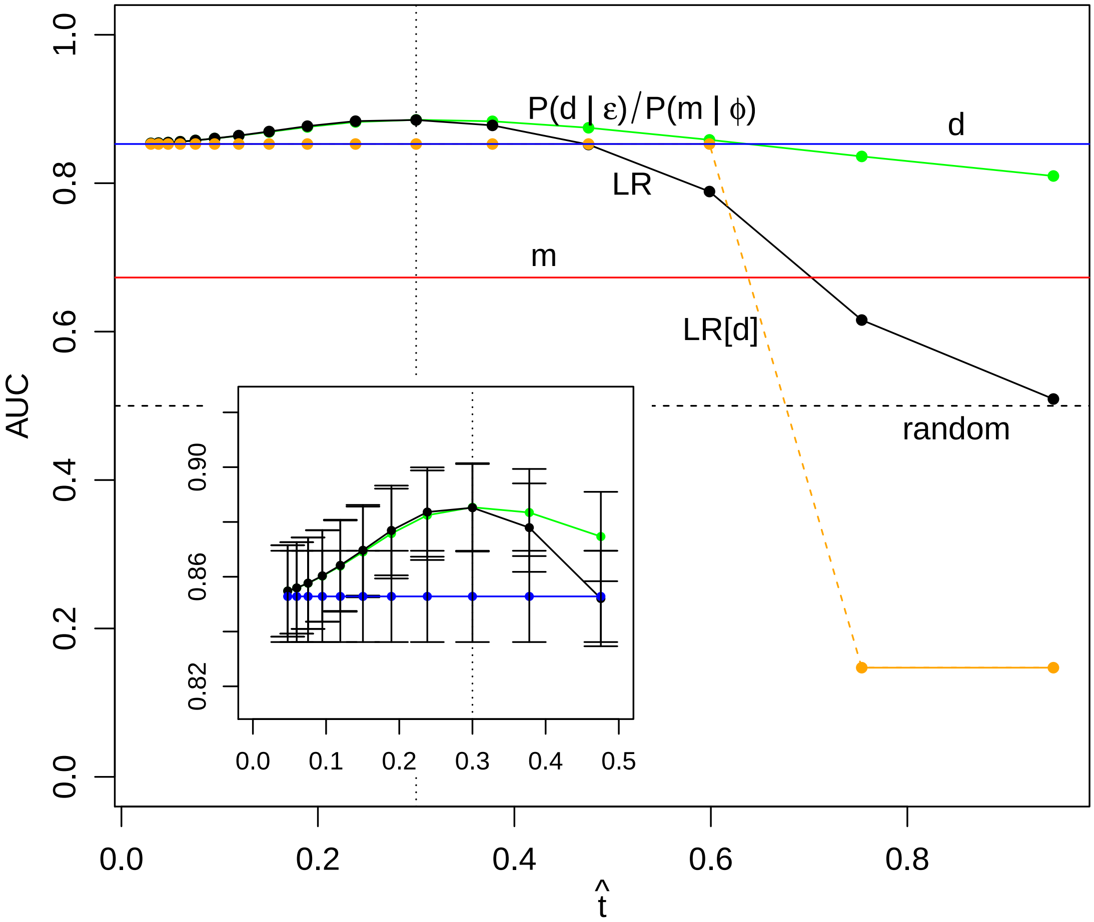


Figure 3.3: Performance as $\hat{t}$ varies. True parameters are $t = 0.3$ (vertical dotted line), $n = 200$, and $E(r) = 4$.

The implications for data sets with unknown parameters can be summarized as follows. Mis-guessing q does not affect the ranking, and our inference methods seem to work well even when the data contains a large number of pairs, as long as $t < 1/\sqrt{2}$. As long as we know positive pairs are closer together than negative pairs, then when using LR, we should always use a $\hat{t}$ less than $1/\sqrt{2}$. Finally, mis-guessing t can be harmful, but there are several options for avoiding the performance drop-off: (a) use d, which is parameter-free and often performs well, (b) underestimate t, rather than overestimate it, to ensure performance will not drop below d, or (c) use $\frac{P(d|\epsilon)}{P(m|\phi)}$, which is more robust to overestimates of t.

3.5 Applying the Model to Real Data

To apply this model to an arbitrary data set in $\mathbb{R}^k$, we need to specify several parameters. The distribution of singletons is straightforward: estimate ϕ (which can be of any desired form) from the entire data set. For positive pairs, we preserve the generative process θ in which $\mathbf{m} \sim \phi$ and $\mathbf{d} \sim \epsilon$. We let ϵ remain a normal, but it should no longer be radially symmetric, since the variables might be at different scales. We define a vector version of $\mathbf{t}$ such that $t_l = \frac{\nu_l}{\hat{\sigma_l}}$ in each dimension l, where $\hat{\sigma_l}$ is the (empirical) estimate of the variance of the negatives. Then we can write $\mathbf{d} \sim \epsilon$ =Normal$(0, \mathbf{t}'\hat{\Sigma}^{-1}\mathbf{t})$, where $\hat{\Sigma}$ is a diagonal covariance matrix estimated from the data. As before, the key parameter to specify is $\mathbf{t}$, which describes the distance between the positive pairs. That distance will match the negative pairs when $\mathbf{t} = 1/\sqrt{2}(1, 1, \ldots, 1)$.

The baseline methods d and m can be generalized as $P(\mathbf{d} \,|\, \epsilon)$ and $\frac{1}{P(\mathbf{m} \,|\, \phi)}$, respectively. When all the components of $\mathbf{t}$ are equal, $P(\mathbf{d} \,|\, \epsilon)$ becomes rank-equivalent to a natural k-dimensional measure called scaled Euclidean distance. The method $LR[d]$ requires an estimate of $P(\mathbf{d} \,|\, c_{ij} = 0)$; for this, we fit a normal to the set of all pairwise displacement vectors $\mathbf{d}$.

3.5.1 Data Sets

The Matched Multiple Birth Data from the National Center for Health Statistics (2000) contains infant birth and mortality data for all twins and larger multiples born in the U.S. from 1995–2000. In this data, two variables could potentially serve to re-identify paired infants: birthweight (grams) and Apgar score (a 0–10 assessment of newborn baby health). True pairs of twins might be expected to have one baby larger and healthier than the other. Yet tests of a sample of twins show the pairs' values are correlated (with a Pearson correlation of 0.79 for weight, 0.44 for Apgar), so there is at least some signal for the algorithm to work with.

The second data set is derived from the Reality Mining data, cell phone data collected from 94 students and faculty over a nine-month period [44]. Our task instances address the question "Is an individual's phone usage pattern distinctive enough to identify them?" We summarize each user's weekly behavior with seven aggregate features: total communication events; number of distinct contacts; number of calls made, received, and missed; number of SMS's received and sent. Each such person-week becomes a point in a data set, and the pairs are defined as instances of the same individual in two different weeks.

From each data source, we construct 100 labeled instances of the pair detection task. An instance of twins data consists of five pairs of twins and 90 singleton babies. An instance of cell phone data consists of five pairs of person-weeks and 75 singletons. In the experiments below, ϕ is always a normal distribution with diagonal covariance.

3.5.2 Experiments and Results

In contrast to typical usage scenarios, where it might be difficult to validate the results of a run, here we have ground truth labels and can experiment with different values of $\hat{\mathbf{t}}$. It has one component for each variable, and for these domains all we know in advance is that pairs should be "close together"—i.e., each component is in the range $(0, {}^1\!/\!\sqrt{2})$. For the two-variable twins data, we explore a grid of possible values. For the seven-variable cell phone data, the exponential state space becomes a problem, so we restrict $\hat{\mathbf{t}}$ to the form $a \cdot (1, 1, \ldots, 1)$ for some constant a.

Figure 3.4 shows that the methods behave very much the same way on real data as they do on synthetic. As before, Best-$LR > d > m$, and $\frac{P(d|\epsilon)}{P(m|\phi)}$ is an excellent alternative when $\hat{\mathbf{t}}$ is unknown.

The grid search on twins data reveals that when we vary the individual components of $\hat{\mathbf{t}}$, this affects the relative strengths of the variables. For instance, setting $\hat{t}_{weight} = 0.001$ (stringently small) but leaving $\hat{t}_{apgar} = 0.7$ (flexible) is almost equivalent to ranking only by d_{weight}. For a fixed ratio among the components of $\hat{\mathbf{t}}$, the relative strengths of the variables are held constant, and only the balance with m will vary.

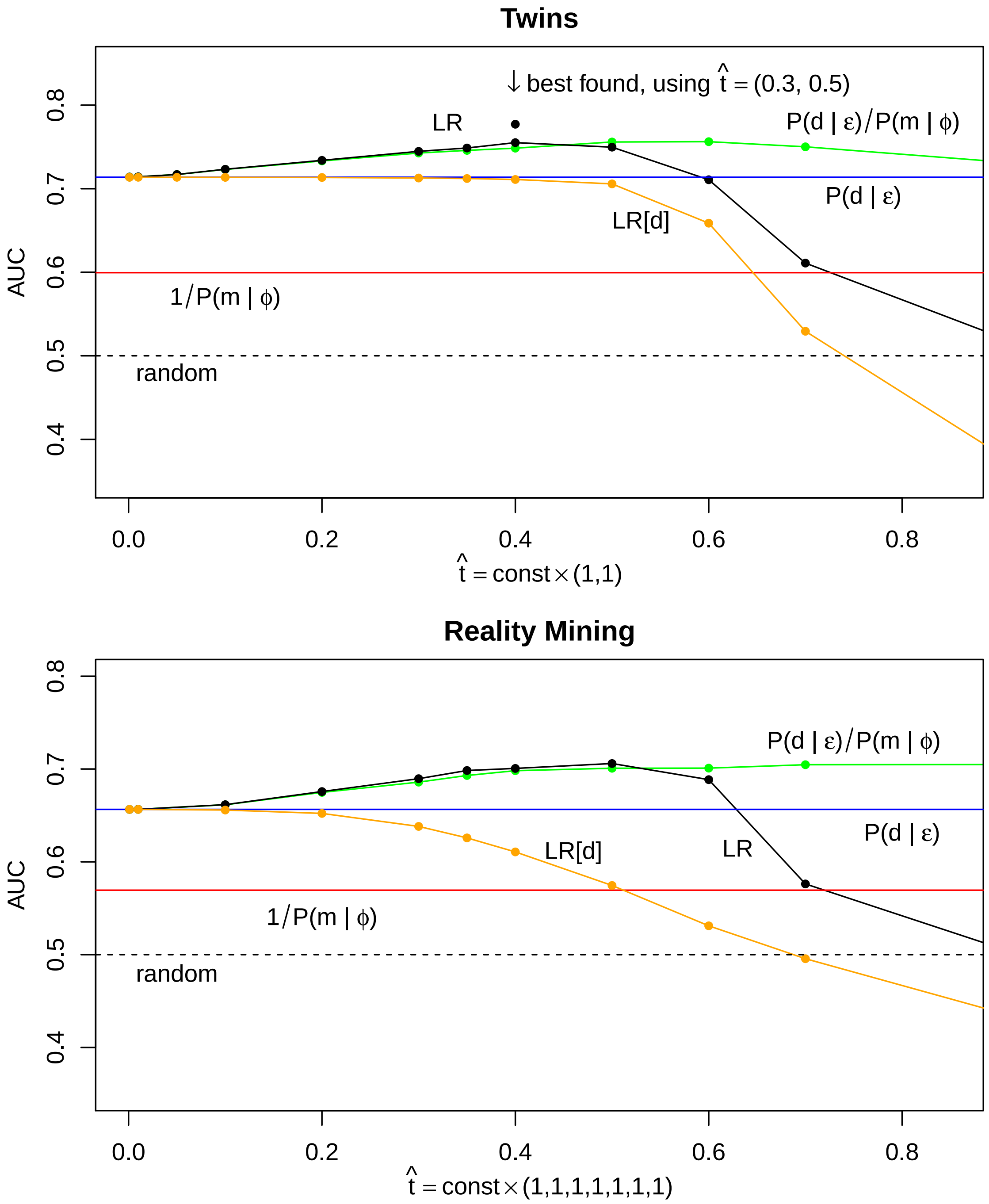


Figure 3.4: Results (average AUC) on real data sets as $\hat{\mathbf{t}}$ varies.

As a comparison, we also estimate a best fit $\mathbf{t}$ from a large sample of twins: that $(t_{weight}, t_{apgar}) = (0.33, 0.57)$ is not far from the $\hat{\mathbf{t}} = (0.3, 0.5)$ found by searching. Separate experiments with single variables show that for twins, weight is a strong feature, but Apgar is not. With Reality Mining, the strongest individual features are number of SMS's sent and number of contacts.

It is not surprising that both these tasks turn out to be difficult given their respective feature sets; in particular, it has been noted that for the Reality Mining data, phone communication is not nearly as consistent as proximity patterns [45]. If the trends of Figure 3.1 generalize to here, then the relatively low AUCs may go hand in hand with the high values of $\hat{\mathbf{t}}$ and the performance boost of LR over $P(\mathbf{d} \mid \epsilon)$.

3.6 Conclusions

This chapter introduces a simple model for ASOUND, the task of distinguishing tightly linked pairs from singleton points, given a mixture of both. This task has not been previously described in a general form, although specific instances have been studied in numerous contexts. From the generative model, we derive a likelihood ratio incorporating both the similarity and rarity of the pairs. A single parameter t describing the distances between pairs turns out to govern the task's difficulty; at inference time, this same parameter describes how to trade off a pair's similarity with its rarity. This LR method always outperforms using only similarity, but in a certain parameter range, similarity turns out to be surprisingly competitive. We demonstrate how to apply the model to real-world data sets having unknown parameters. In the next chapter, we build a different version of this model for a more complex domain.

CHAPTER 4
LATENT TIES IN AFFILIATION DATA

In this chapter, we further study methods for identifying unusually similar pairs of entities within a data set, or ASOUND. However, now each entity (or, "item") is no longer a point, but a binary vector representing, for example, one person's set of affiliations. The pairs of interest are items with a connection, such as a latent social tie, causing them to be more similar than independent pairs would be. Below, we present a simple, analytically tractable model, named MixedPairs, for generating and scoring pairs with a given degree of correlation. Within synthetic data, where MixedPairs is optimal, we can both predict its performance and characterize which features of pairs and data sets cause pairs to be easy to identify. We then transfer these insights to real data and to other similarity methods. While an individual pair stands out if it shares many rare affiliations, we find that overall, a data set's pairs become more conspicuous as the total number of affiliations increases and when each affiliation is present in about half the population. We show how the relative performance of similarity methods varies depending on the affiliation frequencies and whether the methods capture the information present for those frequencies. We find that a group of four similarity methods—three of which are first introduced here—satisfies key properties present in MixedPairs and performs the most robustly across data sets.

4.1 Introduction

As we discussed in Chapter 2, the problem of detecting pairs of people from their shared affiliations would be more widely applicable if we could handle affiliations that were static, as opposed to dynamically changing. In this chapter, we take on that more general domain: identifying anomalously similar entities (ASOUND) within a bipartite graph. A number of recent projects have demonstrated inferring ties among people who are unusually similar—if not based on their job histories, then based on online behavior or physical proximity [1, 40, 45, 86]. Problems of this flavor are a natural fit to ASOUND, provided we can construct suitable representations and models for the data.

In this chapter, we assume the data provided can be described as an affiliation network: a bipartite graph consisting of "items" (e.g., people) connected to "affiliations" (e.g., events attended or purchases). As before, we define the task as computing, in an unsupervised manner, a similarity score for all pairs of items. Again we assume that most items are independent. Since each item can be described by its binary vector of affiliations, the methods within this chapter are equally applicable to detecting pairs within a collection of binary vectors.

This version of ASOUND resembles—and could potentially inform or be informed by—several others that identify linked entities or compute similarities between nodes in a graph. These problems include resolving entities [46], predicting links in unipartite networks [76], detecting collusion [143], calculating document similarity [83], and re-identifying an individual from two samples of their data [93].

The most closely matching problem is probably that of inferring latent social ties, mentioned above. People who know each other often behave similarly—whether due to homophily, imitation or intentional coordination—and it is interesting to ask how much information about the social network is recoverable from data about individuals. Diverse methods have been used in that literature, with methods usually tailored to specific data sets. However, there has been less theoretical work addressing the principles behind the methods. Two recent papers seek to bridge this gap by making clear assumptions about how ties form in social networks and demonstrating how popular measures perform well under, or fail to satisfy, those assumptions [58, 110]. Our work proceeds in the vein of these papers, beginning with a formal model and examining the scoring methods it leads to.

Our high level goals are twofold. First, we want a principled method for assigning probabilities to item pairs, one that describes whether the items are more similar than would be expected by chance alone. Second, we seek an understanding of how the size and distribution of the data (numbers and node degrees both of items and affiliations) affects the statistical power to identify pairs within a given data set.

In this chapter, we present such a method, a generative probabilistic model of affiliation data sets that contain latent pairs of items. The method extends the framework of Section 3.2 by specifying models for singletons and pairs in affiliation data. The resulting model, which we call MixedPairs, is simple enough to be understandable and to permit efficient inference. Doing inference within the model produces a likelihood ratio (LR)—a similarity measure between pairs that describes their probability of being connected. The LR is a readily interpretable function, a sum across the vector components of the pair (Sections 4.2.2 and 4.2.3).

The MixedPairs model also gives us an ability to simulate data in order to explore properties of the problem space. In synthetic data, we are able to predict the task's difficulty as a function of parameters (Section 4.3.2). The ability to identify pairs within a data set depends, not surprisingly, on how similar the latent pairs are. It also turns out to be related to the entropy of the data set: the task gets easier as the dimensionality increases, and as the frequency of each affiliation approaches 0.5.

After analyzing the model's behavior in synthetic data, we widen the scope to other similarity methods and real data sets. We show that the MixedPairs LR satisfies certain desirable properties, properties shared by the other top-performing similarity methods (Sections 4.5 and 4.6.2). Two such methods (Weighted Correlation and SharedWeight1100) are derived in this chapter (Section 4.4.1), and we recommend them as parameter-free alternatives to MixedPairs. We find that in their experimental performance, the comparison methods cluster into groups depending on which of the properties they satisfy. We also find that, while results vary widely across data sets and methods, one entropy-like property generally holds. This property tells us that within a given data set, the affiliations most informative for detecting pairs are those with frequencies closest to 0.5 (Section 4.6.3).

4.2 The MixedPairs Model

The input data is a $n \times k$ binary matrix, the adjacency matrix of a bipartite graph connecting n items (e.g., people or records) to k affiliations (e.g., events or binary features). We describe each item by its row in the matrix, that is, a binary vector of k components. We assume that most items are drawn independently from the same joint distribution of affiliations. Each item can be modeled as having its set of affiliations drawn from some distribution ϕ that can be estimated from the data set as a whole. In this chapter, ϕ is constrained to a simple form, a multiple Bernoulli distribution over affiliations.

We assume that the data set also contains a relatively small number of linked pairs, which we wish to identify. Linked pairs are those that, for some reason—such as a common origin, a copying event or a social tie—are generated dependently in a way that gives them unusually similar sets of affiliations. Examples of linked pairs in our experiments include two documents that share a block of text (and therefore have similar word vectors), two colleagues who work together closely (and therefore display similar voting behavior), or two samples of data taken from the same person's cell phone in different weeks (that therefore record proximity to similar sets of cell towers). By ignoring domain-specific features such as word order or cell phone timestamps, we lose predictive ability for any individual task, but gain the ability to study how algorithms generalize across problem instances.

Although in this research we keep the data sets small for experimental purposes, a typical real-world problem setting might be a database of thousands or more people (e.g., website users) and their recorded behavior, in which we conjecture there are interesting patterns of correlated social behavior, but for which we have few or no labeled examples. To address that scenario, we prefer an unsupervised method, in which we can assert merely that linked pairs are more similar than they would be by chance (MixedPairs has one parameter, and the other methods have none). We also need methods that are efficient; scoring all pairs in a large data set is enough of a computational challenge [10] that we avoid methods with latent variables, for example.

4.2.1 Generative Process

The MixedPairs model specifies how a data set of linked pairs and singletons is generated. The inputs to the model are the total number n of items to generate; the number r of pairs among them; the size k of the set of affiliations (that is, the length of each vector); a similarity parameter $s \in [0, 1]$ for the pairs; and the distribution ϕ of the singletons parametrized by k affiliation frequencies $(p_1, p_2, \ldots, p_k)$, with each $p_i \in [0, 1]$.

The data set itself will be a mixture of singletons and pairs. The generative process for them is as follows:

1. Generate $n - 2r$ singletons. Each singleton X is drawn independently from ϕ. That means $X = (x_1, \ldots, x_k)$, where for each $i \in \{1, \ldots, k\}$, $x_i \sim \text{Bernoulli}(p_i)$.
2. Generate r pairs. Each pair (X_1, X_2) is generated as:

$$\begin{aligned} &X_1 \sim \phi \\ &X_2 \sim sX_1 + (1-s)\phi \end{aligned}$$

   That is, X_2 is generated from a mixture of ϕ and X_1. For each component x_{2i} of X_2, with probability s, x_{2i} is forced to match x_{1i}, while with probability $(1 - s)$, x_{2i} is drawn from $\text{Bernoulli}(p_i)$.

If the similarity parameter s is 0, then the pairs become independent. At $s = 1$, they are forced to be identical. In general it describes the proportion of components that are forced to match, a process that increases similarity beyond the proportion that already match by chance.

An alternative process for pairs that yields the same likelihood function: draw both X_1 and X_2 from ϕ. Then generate a vector f describing which components will be flipped to match each other (if they don't already). Each component f_i is drawn i.i.d. $\sim \text{Bernoulli}(s)$. Wherever $f_i = 1$ and $x_{1i} \neq x_{2i}$, flip one of x_{1i} and x_{2i}, deciding between them with probability 0.5.

The items generated as pairs are individually indistinguishable from singletons. For X_1, this is true because it is sampled directly from ϕ. For X_2, the marginal probability of each component matches ϕ: $P(x_{2i} = 1) = sP(x_{1i} = 1) + (1-s)p_i = (s+1-s)p_i = p_i$. Since both X_2 and ϕ are component-wise independent, this shows that X_2 is distributed according to ϕ when considered by itself. Having pairs match singletons is an important property for the model to satisfy, because it assures us that detection of pairs will be based exclusively on their correlation, not some other incidental property.

Note that this generative process derives directly from that of Section 3.2. Section 3.2 is explicit about certain aspects we skip here, such as the adjacency matrix C and uncertainty

Symbol	Meaning
n	Number of items
k	Number of affiliations
r	Number of positive pairs
s	Similarity parameter of positive pairs: 0 (independent) to 1 (identical)
c	True label of a pair: 1 (positive) or 0 (negative)
p_i	Affiliation frequency, for $i \in (1, \dots, k)$
ϕ	Distribution of singletons: $\phi = \text{Multiple-Bernoulli}(p_1, \dots, p_k)$
b_i	One vector component of a pair: $1\|1$, $1/0$, or $0\|0$
$\text{LLR}(b_i)$	Log likelihood ratio assigned by MixedPairs; see Table 4.2.

Table 4.1: Notation in this chapter.

over the number of pairs r; for a fuller treatment, refer back to that section. Below, the likelihood ratio from (4.1) used in scoring is equivalent to that from (3.2). The concerns regarding treating all pairs as independent, first discussed in Section 3.2.3, are reviewed below.

4.2.2 Likelihood Functions and Inference

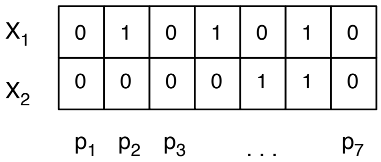
To detect pairs within a data set, we compute a score for every possible pair of items, estimating its probability of being positive (a true pair) as opposed to negative (composed of two singletons). Evaluation will compare these scores to the ground truth labels. In this work, we ignore the issue of joint inference (that according to the generative model, if (X_1, X_2) is a pair then (X_1, X_3) cannot be). By ignoring that constraint, we can score every pair independently—which is much more computationally tractable—and we can deploy the scoring function in a broader class of situations, as a general-purpose similarity measure.

Note that the data will also include negative pairs not formed from two singletons. These are negatives which combine a point from a true pair together with a singleton or a point from a different true pair. Since the individual points from true pairs look just like singletons, as we showed above, these "different" negative pairs can be accurately modeled just like the others, as two independent singletons.

We compute the score in odds form. Letting $c \in \{0 \text{ (negative)}, 1 \text{ (positive)}\}$ be the class label for an arbitrary pair (X_1, X_2), we estimate the posterior odds ("PO"):

$b_i = (x_{1i}, x_{2i})$	$P(b_i \mid c=0, p_i)$	$P(b_i \mid c=1, p_i, s)$	$\text{LR}(b_i \mid p_i, s)$	$\text{WC}(b_i \mid p_i)$ $= \frac{1}{s}(\text{LR}(b_i \mid p_i, s) - 1)$
1 1	p_i^2	$p_i(s + (1-s)p_i)$	$\frac{s+(1-s)p_i}{p_i}$	$\frac{1-p_i}{p_i}$
1 0	$p_i(1-p_i)$	$p_i(1-s)(1-p_i)$	$1-s$	-1
0 1	$(1-p_i)p_i$	$(1-p_i)(1-s)p_i$	$1-s$	-1
0 0	$(1-p_i)^2$	$(1-p_i)(s+(1-s)(1-p_i))$ $= (1-p_i)(1-p_i+p_is)$	$\frac{1-p_i+p_is}{1-p_i}$	$\frac{p_i}{1-p_i}$

Table 4.2: For a single component of a pair, likelihood functions and ratio under MixedPairs; score from Weighted Correlation.



X_1	0	1	0	1	0	1	0
X_2	0	0	0	0	1	1	0
	p_1	p_2	p_3		. . .		p_7

Figure 4.1: In this example pair, the likelihood ratio comes out to $\text{LR}(X_1, X_2) = \left(\prod_{i \in \{1,3,7\}} \frac{1-p_i+p_is}{1-p_i}\right)\left(\frac{s+(1-s)p_6}{p_6}\right)(1-s)^3$.

$$\frac{P(c=1 \mid X_1, X_2)}{P(c=0 \mid X_1, X_2)} = \frac{P(c=1)}{P(c=0)} \times \frac{P(X_1, X_2 \mid c=1)}{P(X_1, X_2 \mid c=0)}. \tag{4.1}$$

The first term on the right, the prior odds, is constant across all pairs in a data set. The second term, the likelihood ratio $\text{LR}(X_1, X_2)$, determines the ranking induced by the scores, which is our primary focus. When an estimate is available for the prior (e.g., under the model, r positive pairs over $\left(\frac{n(n+1)}{2} - r\right)$ negative pairs), the posterior odds can be converted to a probability using $P(c=1 \mid X_1, X_2) = \frac{\text{PO}}{1+\text{PO}}$.

Since for both positive and negative pairs each vector component is generated independently, their likelihood ratio can be factored into products over the components. The notation b_i is used to represent one component of the pair (x_{1i}, x_{2i}), $\text{LR}(b_i)$ (shorthand for $\text{LR}(b_i \mid p_i, s)$) is the likelihood ratio for that component, and LLR is the log likelihood ratio (of a component or pair, respectively). Then we can write the score for a pair as:

$$\text{LR}(X_1, X_2) = \frac{\prod_i P(x_{1i}, x_{2i} \mid c=1)}{\prod_i P(x_{1i}, x_{2i} \mid c=0)} = \prod_{i=1}^{k} \text{LR}(b_i)$$

or

$$\text{LLR}(X_1, X_2) = \sum_{i=1}^{k} \log \text{LR}(b_i).$$

Figure 4.1 shows an example pair of points to score. At each vector component, there is an associated p_i from ϕ, and there are four possible configurations of the pair: $b_i \in \{1|1, 1|0, 0|1, 0|0\}$. Table 4.2 shows the likelihood functions for each component, and their ratio $\text{LR}(b_i)$, as a function of b_i, p_i and s.

Notice that $b_i = 1|0$ and $b_i = 0|1$ have the same likelihoods. This means scoring is symmetric with respect to X_1 versus X_2; there is no need to designate one point as X_1. Furthermore, we can define the event $b_i = 1/0$ to mean $b_i \in \{1|0\} \cup \{0|1\}$, which reduces b_i to three possible configurations: $\{1|1, 1/0, 0|0\}$. The combined event $1/0$ has probabilities $P(1/0) = 2P(1|0)$ for positive or negative pairs and $\text{LR}(1/0) = \text{LR}(1|0) = 1 - s$.

4.2.3 Understanding MixedPairs Scores

As we have seen, given a pair, MixedPairs produces a score, $\text{LLR}(X_1, X_2) = \sum_i \text{LLR}(b_i \mid p_i, s)$, that is a sum over components. The top left of Figure 4.2 illustrates how the function $\text{LLR}(b_i \mid p_i, s)$ works per component. First, notice that the value is always positive when items match (when $b_i = 1|1$ or $0|0$). The value is highest when the pair matches on an unexpected event (for $b_i = 1|1$, highest with p_i near 0, and for $b_i = 0|0$, highest with p_i near 1). It decreases towards 0 as the matched event becomes more common, with the curves for $b_i = 1|1$ and $0|0$ intersecting at $p_i = 0.5$. The score for a mismatch ($b_i = 1/0$) is negative and constant with respect to p_i. Second, notice the symmetry with respect to $p_i = 0.5$: $\text{LLR}(1|1 \mid p_i, s) = \text{LLR}(0|0 \mid (1 - p_i), s)$. That means that a $1|1$ for a rare affiliation (e.g., where $p_i = 0.1$) is treated just like a $0|0$ for a common affiliation (e.g., $p_i = 0.9$).

As s increases (Fig. 4.2 middle), the absolute values of all scores increase. At $s = 0$, true pairs are independent; for all values of b_i and p_i, $\text{LR}(b_i \mid p_i, 0) = 1$, so $\text{LLR}(b_i \mid p_i, 0) = 0$. At $s = 1$, true pairs are required to be exact duplicates, so $\text{LR}(1/0) = 0$ and $\text{LLR}(1/0) = -\infty$.

We will examine the per-component curves of other scoring methods later in Section 4.5. Below, we take advantage of MixedPairs's independence across components to analyze its behavior across the parameter space.

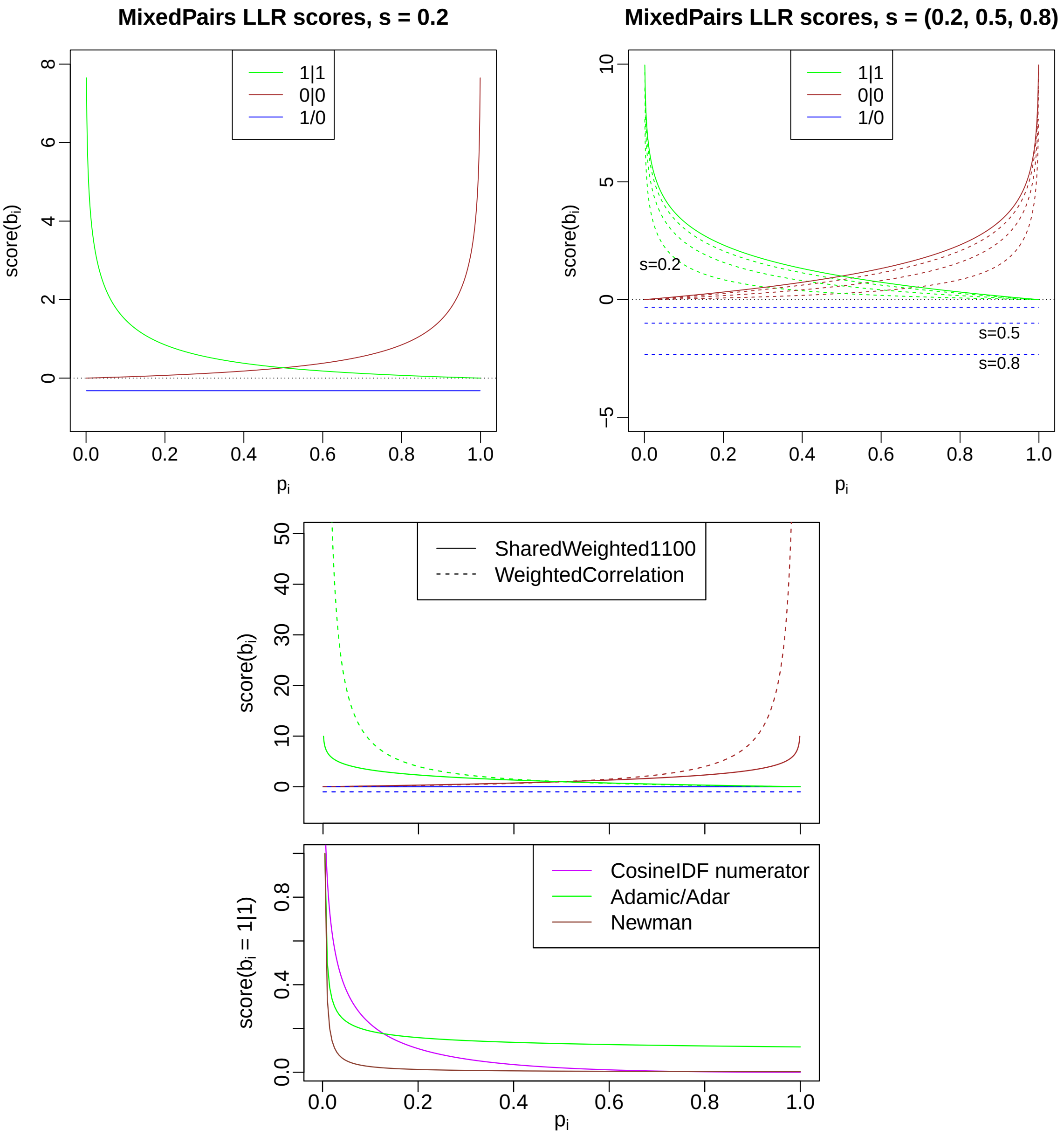


Figure 4.2: LLR$(b_i \mid p_i, s)$: log likelihood ratio for a component having a given p_i, if the pair's configuration is 1|1 (green line), 0|0 (brown), or 0|1 (blue), respectively. At top left, for $s = 0.2$; top right, for several values of s. In the top right, the solid lines indicate the limits, at $s = 1$, for LLR$(1|1) = \frac{1}{p_i}$ and LLR$(0|0) = \frac{1}{1-p_i}$; LLR$(1/0) = -\infty$ cannot be displayed. SharedWeight1100 matches the solid lines, but for it, Score$(1/0) = 0$. SharedWeight11 uses the same green 1|1 line but also assigns Score$(0|0) = 0$. Center, per-component scores for SharedWeight1100 again (solid lines, now a different scale) and Weighted Correlation (dashed lines). At bottom, Score(1|1) for the numerator of CosineIDF, for Adamic/Adar, and for Newman, using $n = 400$ for the latter two.

4.3 Analysis of MixedPairs Behavior

Within the system defined by MixedPairs's generative models of positive and negative pairs, we can fully describe the scores that will be assigned to pairs. Depending on properties of the data set, positive pairs will be easier or harder to detect. We characterize these properties first for individual pairs, then for the overall distribution of pairs.

4.3.1 Scores for Pairs: When is a Point Worth Checking?

MixedPairs produces an estimate $\mathrm{P}(c = 1 \mid \text{pair})$ for every pair in the data. If we are only interested in high-probability pairs, we could ask "what's the highest score item X could receive?" as a means towards discarding some items for efficiency purposes. Intuitively, we should expect true pairs in sparse regions of the probability space to be easier to identify—i.e., to receive higher scores—than equally similar pairs in dense regions.

The highest score for any item X_1 occurs at an exact duplicate. That is, $\arg\max_{X_2} \mathrm{P}(c = 1 \mid (X_1, X_2), s, \phi) = X_1$. This is easy to see, since given each x_{1i}, $\mathrm{LR}(b_i)$ is always highest when $x_{1i} = x_{2i}$. The highest score for an exact duplicate occurs when $s = 1$: $\arg\max_s \mathrm{P}(c = 1 \mid (X_1, X_1), s, \phi) = 1$. This is because increasing s increases both $\mathrm{LR}(1|1)$ and $\mathrm{LR}(0|0)$. Finally, regardless of s, the highest scoring pair would always be an exact duplicate of the lowest-probability item in the space: $\arg\max_{X_1} \mathrm{P}(\mathrm{c} = 1 \mid (X_1, X_1), s, \phi)$ has $x_{1i} = 1$ when $p_i < 0.5$ and $x_{1i} = 0$ when $p_i > 0.5$ (allowing either value at $p_i = 0.5$). That rule maximizes the score because it sets $b_i = 1|1$ whenever $\mathrm{LR}(1|1) > \mathrm{LR}(0|0)$, and vice versa.

In contrast, a good starting candidate for items to discard would be the highest-probability item in the space—that having $x_{1i} = 0$ where $p_i \leq 0.5$ and vice versa. Even if it has an exact duplicate, their pair's $\mathrm{P}(c = 1 \mid (X_1, X_1), s, \phi)$ might fall below some threshold of practical utility. Notice that the densest region of ϕ comprises items that are mostly or all zeros, assuming all $p_i < 0.5$. In varied domains, these items might correspond to "records with little data," "short documents," or "people with low activity levels." Informally, such items are often discarded or found difficult to match [40, 86, 111] from such "sparse" information, where "sparse" refers to the number of 1's or the amount of surprising information. In the current analysis, we would describe these items as too dense, or high-probability, for the true pairs among them to receive high scores.

Thus the highest scores for pairs occur when pairs are exact duplicates. These scores are higher when the model requires true pairs to be more similar, and when more components of a duplicate are rare events according to ϕ. But rare items are, by definition, rare, and we are also interested in understanding whether a given data set on the whole is likely to have its positive pairs be identifiable. As we show next, this property is related to the entropy of the data distribution: data sets in which more items are unusual, on average—that is, those with high entropy—turn out to be the data sets in which positive pairs can be more easily identified, on average.

4.3.2 In Which Data Sets are Pairs Detectable?

Because MixedPairs is generative, we can reason about the distributions of scores it will assign to the positive and negative pairs it produces. In the following discussion, assume s is fixed and known.

Let $B_i \in \{1|1, 1/0, 0|0\}$ be a random variable describing component i of a pair, and let $V_i = \mathrm{LLR}(B_i \mid p_i, s)$ be the score assigned to that component. V_i can be seen as a random variable having three possible values. Those values $\mathrm{LLR}(B_i \mid p_i, s)$ and their associated probabilities $P(B_i \mid p_i)$ (for positive or negative pairs, respectively) can be read off of Table 4.2. The full score assigned to a pair is $V = \mathrm{Score}(X_1, X_2) = \sum_{i=1}^{k} V_i$. We wish to describe the distribution of V. (We use the more general notation $\mathrm{Score}(X_1, X_2)$ instead of $\mathrm{LLR}(X_1, X_2)$, to point out that this analysis can be extended to any scoring method whose scores are component-wise independent and additive. See Appendix C for results with other methods.)

In the special case where p_i is a constant across all values of i, every V_i follows the same distribution, so the Central Limit Theorem applies. It tells us that $V \sim \mathrm{Normal}(k\mathrm{E}(V_i), k\mathrm{Var}(V_i))$. For positive pairs, those component-wise $\mathrm{E}(V_i)$ and $\mathrm{Var}(V_i)$ are $\mathrm{E}_{\mathrm{pos}}(V_i) = \sum_{b_i} \mathrm{P}(b_i \mid \mathrm{pos})\mathrm{Score}(b_i)$ and $\mathrm{Var}_{\mathrm{pos}}(V_i) = \sum_{b_i} \mathrm{P}(b_i \mid \mathrm{pos})\left(\mathrm{Score}(b_i) - \mathrm{E}_{\mathrm{pos}}(V_i)\right)^2$, and similarly for negative pairs. Thus if p_i is constant and we know p_i and s, the distributions of scores for both positive and negative pairs are normals whose mean and variance are easily computed.

Figure 4.3 shows actual MixedPairs LLR scores from a synthetically generated data set. If $s = 0$, the distributions overlap exactly, with a mean of 0. As s increases, $\mathrm{E}_{\mathrm{pos}}(LLR)$

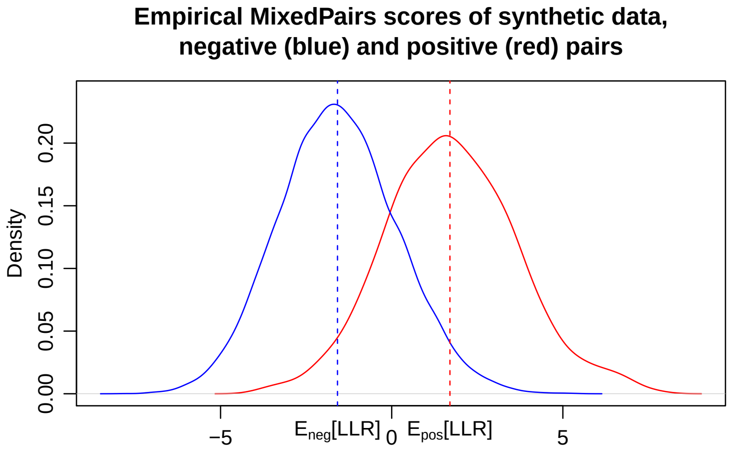


Figure 4.3: Empirical log likelihood ratio scores from MixedPairs, with predicted means indicated. (Data is sampled from Figure 4.6's ϕ setting 49, labeled "B.")

increases and $\mathrm{E}_{\mathrm{neg}}(LLR)$ decreases. At $s = 1$ exactly, there is a discontinuity: $\mathrm{E}_{\mathrm{neg}}(LLR) = -\infty$ because of the scores for non-duplicate pairs. At $s = 1$, for both the positive pairs and the vanishingly[1] small fraction of negatives that are duplicates, $\mathrm{E}(LLR) = H(\phi)$, the entropy of the distribution. If desired, we could convert $\mathrm{E}_{\mathrm{pos}}(LLR)$ and $\mathrm{E}_{\mathrm{neg}}(LLR)$ to probabilities and interpret them as the median probabilities anticipated for positive and negative pairs under this model.

One way to evaluate the separation between the distributions would be to compute the distance between the two means: $\mathrm{E}_{\mathrm{pos}}(V) - \mathrm{E}_{\mathrm{neg}}(V)$, which (for MixedPairs) is equivalent to the Jeffreys divergence between the distributions of positive and negative pairs [71]. However, what we use instead, throughout this work, is AUC (area under the ROC curve), which is sensitive not only to the means of the scores but also their variances. The AUC of two normal distributions can be approximated using [47]:

$$\mathrm{AUC} = \Phi\left(\frac{\mathrm{E}_{\mathrm{pos}} - \mathrm{E}_{\mathrm{neg}}}{\sqrt{\mathrm{Var}_{\mathrm{pos}} + \mathrm{Var}_{\mathrm{neg}}}}\right). \tag{4.2}$$

[1] The fraction of negative pairs that are duplicates is also related to entropy: $\mathrm{P}(\mathrm{dup} \mid \mathrm{neg}) = \sum_{x\sim\phi} \mathrm{P}(x \mid \phi)^2 = e^{-H_2(\phi)}$, where $H_2(\phi)$ is the collision entropy of ϕ. As $H_2(\phi)$ increases, the number of negative candidates for scoring (at $s = 1$) decreases, here too making the task easier when ϕ has higher entropy.

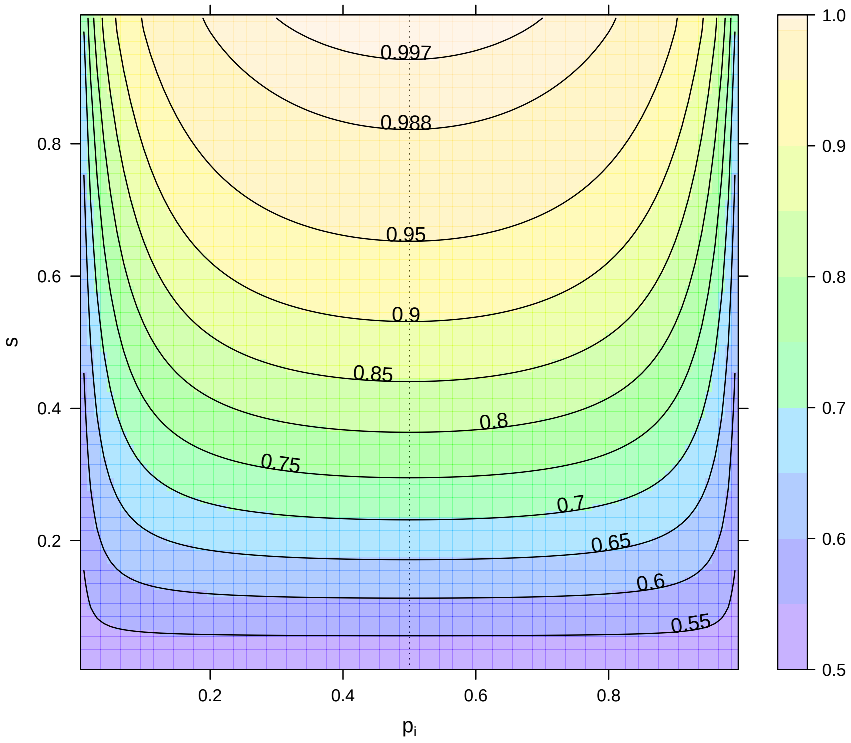


Figure 4.4: AUC predicted for MixedPairs as a function of p_i and s, using $k = 10$.

We use the parameters of the normal distributions under constant p_i and the AUC formula above to estimate AUC as a function of p_i, k, and s, for data generated and scored by MixedPairs. Figure 4.4 displays those estimates at $k = 10$. From the figure, we can see that AUC increases overall as s increases; that is, true pairs are easier to identify when they are more closely correlated. As k increases (see Figure C.1a), the contour lines keep the same shape, while performance increases. That is, additional affiliations always help. As a function of p_i, performance is highest when $p_i = 0.5$ and decreases symmetrically as p_i moves towards 0 or 1. This maximum at $p_i = 0.5$ contrasts with the focus earlier on how more extreme values of p_i can yield higher scores for individual pairs. What this tells us is that on average across a data set, the information content per component is highest for components with p_i at 0.5. These are the affiliations for which we don't know whether to expect a 0 or 1; either outcome is informative, and for components that match, both LLR(1|1) and LLR(0|0) provide non-negligible (and equal) contributions to the overall score.

So far, in order to have scores be normally distributed, we have been limited to data sets in which ϕ has the same p_i for every affiliation. This would be a strong restriction, but we find that in practice, even across diverse collections of affiliation frequencies $\{p_i\}$, the assumption of normality works reasonably well for synthetically generated data. When p_i varies, we can apply the same AUC estimate by assuming that $V \sim \mathrm{Normal}(\sum_i \mathrm{E}(V_i), \sum_i \mathrm{Var}(V_i))$. In fact, the roughly normal densities shown in Figure 4.3 come from data in which the p_i are not constant, but uniformly distributed. Figure 4.5 shows both this predicted AUC and the performance of MixedPairs on 69 different settings of ϕ (see Section 4.6 for experimental details). As we can see, they agree well.

We have seen that a paired item X will receive a higher score if $\mathrm{P}(X \mid \phi)$ is low, yet that on average, the AUC of a data set is highest when all $p_i = 0.5$, which is when all values of $\mathrm{P}(X \mid \phi)$ are equal. This property is reminiscent of entropy. The entropy of a distribution, $H(\phi) = -\sum_{x\sim\phi} \mathrm{P}(x \mid \phi) \log \mathrm{P}(x \mid \phi)$, is maximized when ϕ is uniform across a domain, even though individual contributions of $-\log \mathrm{P}(x \mid \phi)$ could be higher by making $\mathrm{P}(x \mid \phi)$ lower. A natural question is whether the AUC predicted for detecting pairs in a data set is simply a function of $H(\phi)$. We did not find a direct mapping between the quantities, but it seems clear they are related. The red line in Figure 4.5 shows the entropy for each ϕ (see scale on right axis). The x axis is ordered by the predicted AUC, and the entropy mostly increases in tandem. Both are maximized at the uniform distribution (i.e., when all $p_i = 0.5$). In our domain of binary vectors, the entropy of that distribution is simply k, the number of affiliations. Regarding the distributions of LLR scores, we mentioned above that $\mathrm{E}_{pos}[LLR]$ is maximized when $s = 1$, in which case $\mathrm{E}[LLR] = H(\phi)$ for the duplicates. The upper bound for this value is thus k, achieved when ϕ is uniform. In that setting, the LLR of every duplicate pair is k.

4.4 Alternative Methods

Before proceeding to experiments, we introduce two alternative similarity methods, Weighted Correlation and SharedWeight, which are closely related to MixedPairs. Following that, we examine the properties of a number of methods in Section 4.5.

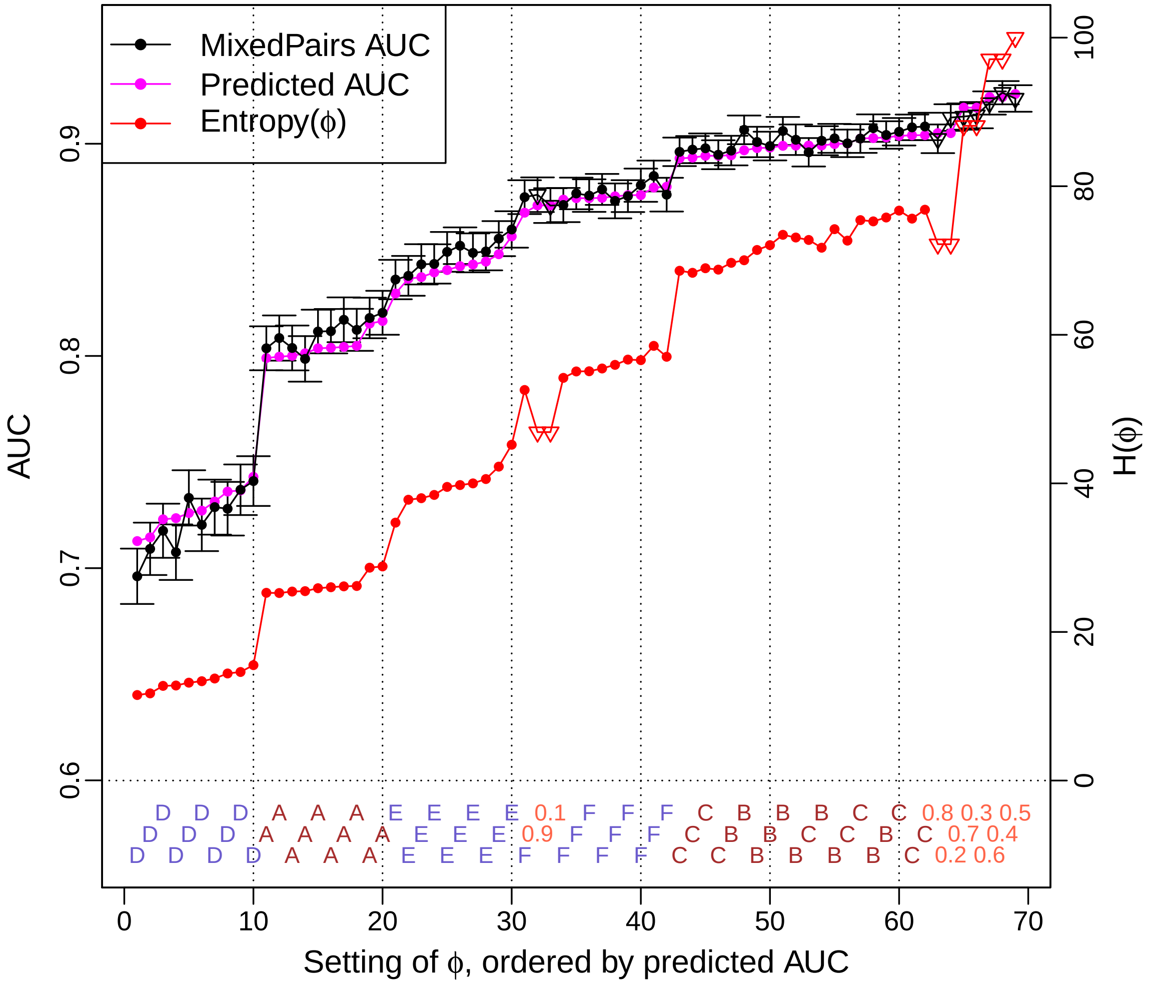


Figure 4.5: AUC predicted for MixedPairs in the synthetic experiments, actual AUC, and entropy of each ϕ. Each position on the x-axis corresponds to one setting of ϕ, with $k = 100$ and $s = 0.2$. Magenta line: AUC predicted using Eq. 4.2. Black line: average AUC over 400 experimental trials, which matches the predictions; same data as top line of Figure 4.6. Error bars are 95% confidence intervals. Red line (right axis): $H(\phi)$, which correlates with AUC.

4.4.1 Weighted Correlation

Traditional Pearson correlation is defined as the covariance of distributions X and Y, divided by the product of their standard deviations:

$$\mathrm{Corr}(X,Y) = \frac{\mathrm{E}[(X-\mathrm{E}(X))(Y-\mathrm{E}(Y))]}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}.$$

The denominator normalizes the value to lie between -1 and 1. This calculation is based on the premise that for all i, $X_i \sim X$ and $Y_i \sim Y$ for some distributions X and Y. Correlation is invariant to linear transformations of the variables, so it can also be written as the covariance of the standardized versions of the variables, X' and Y':

$$\begin{aligned}\mathrm{Corr}(X,Y) &= \mathrm{Covar}(X',Y') = \mathrm{E}[X'Y'] \\ &= \frac{1}{k}\sum_{i=1}^{k}\left(\frac{X_i-\mathrm{E}(X)}{\sqrt{\mathrm{Var}(X)}}\right)\left(\frac{Y_i-\mathrm{E}(Y)}{\sqrt{\mathrm{Var}(Y)}}\right).\end{aligned}$$

In our problem setting, where X and Y are items, it is not the case that each X_i or Y_i comes from a single distribution. Rather, each component comes from a different distribution, but the distributions of X_i and Y_i match: $\forall i,\ X_i, Y_i \sim \mathrm{Bernoulli}(p_i)$. So it makes more sense here to standardize each component separately, using our outside knowledge of p_i: define $X''_i = \frac{X_i-\mathrm{E}(X_i)}{\sqrt{\mathrm{Var}(X_i)}} = \frac{X_i-p_i}{\sqrt{p_i(1-p_i)}}$, and $Y''_i = \frac{Y_i-\mathrm{E}(Y_i)}{\sqrt{\mathrm{Var}(Y_i)}} = \frac{Y_i-p_i}{\sqrt{p_i(1-p_i)}}$.

We define Weighted Correlation to be much like the covariance of X'' and Y'':

$$\begin{aligned}\mathrm{WC}(X,Y) &= \mathrm{E}[X''Y''] \\ &= \frac{1}{k}\sum_{i=1}^{k}\left(\frac{X''_i-\mathrm{E}(X''_i)}{\sqrt{\mathrm{Var}(X''_i)}}\right)\left(\frac{Y''_i-\mathrm{E}(Y''_i)}{\sqrt{\mathrm{Var}(Y''_i)}}\right) \\ &= \frac{1}{k}\sum_{i=1}^{k}\left(\frac{(X_i-p_i)(Y_i-p_i)}{p_i(1-p_i)}\right) \\ &= \frac{1}{k}\sum_{i=1}^{k}\mathrm{WC}(b_i \mid p_i),\end{aligned}$$

$$\text{where } \mathrm{WC}(b_i \mid p_i) = \begin{cases} b_i = 1|1: & \frac{(1-p_i)^2}{p_i(1-p_i)} & = & \frac{1-p_i}{p_i} \\ b_i = 1/0: & \frac{-p_i(1-p_i)}{p_i(1-p_i)} & = & -1 \\ b_i = 0|0: & \frac{(-p_i)^2}{p_i(1-p_i)} & = & \frac{p_i}{1-p_i} \end{cases} .$$

This function differs from $\mathrm{Covar}(X'', Y'')$ in that it contains no terms like $\mathrm{E}(X'')$ or $\mathrm{Var}(X'')$ that summarize X'' across $i = 1$ to k. As a result, each component's contribution is independent of the others. Since each component has been mean-shifted and standardized with respect to ϕ, each $\mathrm{E}(X_i'') = 0$ and $\mathrm{E}[(X_i'')^2] = 1$ when the expectation is taken across all items or all pairs in the data set. However, within the sample of k components, those expressions will not necessarily hold. As a consequence, $\mathrm{WC}(X, Y)$ can produce values outside of the interval $[-1, 1]$.

Like Pearson correlation, Weighted Correlation is unchanged when computed based on the original vectors, rather than the standardized ones:

$$\begin{aligned} \mathrm{WC}(X, Y) &= \frac{1}{k} \sum_{i=1}^{k} \left(\frac{X_i - \mathrm{E}(X_i)}{\sqrt{\mathrm{Var}(X_i)}} \right) \left(\frac{Y_i - \mathrm{E}(Y_i)}{\sqrt{\mathrm{Var}(Y_i)}} \right) \\ &= \frac{1}{k} \sum_{i=1}^{k} \left(\frac{(X_i - p_i)(Y_i - p_i)}{p_i(1-p_i)} \right) . \end{aligned}$$

The derivation above is shown via X'' and Y'' because the process of standardizing the variables highlights how, in the current context, $\mathrm{E}(X)$ and $\mathrm{Var}(X)$ are not the right choices for the moments of X_i.

4.4.2 Relationship between Weighted Correlation and MixedPairs

The far right column of Table 4.2 demonstrates an unexpected connection between the per-component scores of Weighted Correlation and MixedPairs: $\mathrm{WC}(b_i) = \frac{1}{s}(\mathrm{LR}(b_i) - 1)$, or $\mathrm{LR}(b_i) = s\mathrm{WC}(b_i) + 1$. They are simple linear transformations of each other. $\mathrm{WC}(b_i)$ cleverly removes the s from the expression $\mathrm{LR}(b_i)$ (since Weighted Correlation doesn't use s), and $\mathrm{WC}(b_i)$ is shifted lower by 1. For instance, for independent pairs, $\mathrm{LR}(b_i)$ will be about 1, and $\mathrm{WC}(b_i)$ will be about 0.

At first glance, this connection between the methods may seem baffling. How can Weighted Correlation perfectly remove the s from MixedPairs? And how could the methods give similar experimental results (as Section 4.6 will show), if Weighted Correlation takes a sum of its $\mathrm{WC}(b_i)$ terms, while MixedPairs takes the sum of their logs?

In answer to the first question, MixedPairs uses the parameter s since it models both positive and negative pairs. Weighted Correlation, in contrast, only reports the extent of deviation from the negative pairs model. If we estimate the distributions of Weighted Correlation scores under the MixedPairs model, it turns out that $\mathrm{E}_{\mathrm{neg}}(\mathrm{WC}) = 0$ always (unlike $\mathrm{E}_{\mathrm{neg}}(\mathrm{LLR})$, which varies according to s). For positive pairs, $\mathrm{E}_{\mathrm{pos}}(\mathrm{WC}) = s$. That is, the s of MixedPairs positives reappears in the correlation coefficient measured by Weighted Correlation.

As for the second question, it turns out that Weighted Correlation computes an arithmetic mean (AM) of its terms, while the log likelihood score of MixedPairs can be seen as a geometric mean (GM):

$$\begin{aligned}
\mathrm{WC}(X,Y) &= \frac{1}{k}\sum_i \mathrm{WC}(b_i) = \mathrm{AM}(\mathrm{WC_i}) \\
\mathrm{LLR}(X,Y) &= \sum_i \log \mathrm{LR}(b_i) = \log\left(\prod_i LR_i\right) \\
&= k\log\left(\prod_i LR_i\right)^{\frac{1}{k}} = k\log \mathrm{GM}(\mathrm{LR_i}).
\end{aligned}$$

The arithmetic mean is appropriate for Weighted Correlation since covariances and correlations are defined as expected values. MixedPairs is a ratio, with its “uninformative” score at 1, and the geometric mean is appropriate for it.

Moreover, at the very low values of s seen in the experiments, MixedPairs approximates Weighted Correlation. This happens because at low values of x, $\log(1+x) \approx x$. At these s, $\sum_i \log \mathrm{LR}(b_i) = \sum_i \log\left(s\mathrm{WC}(b_i) + 1\right) \approx \sum_i s\mathrm{WC}(b_i)$. In this situation, the scores of the methods match up to a constant factor—$\mathrm{LLR}(X,Y) \approx ks\mathrm{WC}(X,Y)$—so the two rankings become equivalent.

4.4.3 Shared Weight

An alternative scoring method, which we motivate on intuitive grounds, is to look at just the shared components y of the vectors, and compute $\frac{1}{P(y|\phi_y)}$ (i.e., use ϕ restricted to the components in question). This has the appealing properties that $\frac{1}{P(y|\phi_y)}$ increases both as y expands to more components and as the shared components become more rare. We can define the y to include only 1|1s, yielding SharedWeight11, or both 1|1s and 0|0s, yielding SharedWeight1100.

We can write down a (non-generative) likelihood function for positive pairs—the numerator of the expression below—such that, when we take the likelihood ratio, we get SharedWeight1100.

$$\begin{aligned}\frac{P(X_1, X_2 \mid \text{positive})}{P(X_1, X_2 \mid \text{negative})} &= \frac{P(X_1 \mid \phi)\left(\prod_{i\in 1/0} P(x_{2i} \mid \phi_i)\right)\left(\prod_{i\in\{1|1,0|0\}} 1\right)}{P(X_1 \mid \phi)\left(\prod_{i\in\{1/0,1|1,0|0\}} P(x_{2i} \mid \phi_i)\right)} \\ &= \frac{1}{\prod_{i\in\{1|1,0|0\}} P(x_i \mid \phi_i)} = \frac{1}{P(y \mid \phi_y)}.\end{aligned}$$

(The notation "$i \in 1|1$" is shorthand for "$i : b_i = 1|1$.") The likelihood function can be interpreted as saying that for the components where X_2 matches X_1, it copied them with probability 1, but where they don't match, X_2 drew its components from ϕ. For the method SharedWeight11, we change the indices of the products so that 0|0 gets treated (i.e., ignored) like 1/0. As we can see from the middle of Figure 4.2, above, and again in Table 4.3, the SharedWeight methods, where they are non-zero, match the score MixedPairs gives at $s = 1$.

4.5 Comparisons Among Methods

In experiments, we will compare the methods derived above—MixedPairs, Weighted Correlation, and SharedWeight—to a selection of similarity measures that would typically be candidates for this task [74, Ch. 3.5], [83, Ch. 6]. The full list is shown with their definitions in Table 4.3. We are concerned primarily with the ranking induced by the scores, so we gloss over rank-equivalent transformations such as taking the log.

The simplest methods for comparing two sets of affiliations are to count the number of shared affiliations (*SharedSize*) or count the differences between them (*Hamming* distance). These are not equivalent: SharedSize counts the 1|1s in a pair, whereas Hamming distance (multiplied by −1 so as to act as a similarity measure) counts the 1/0s in a pair. If we count both the 1|1s and 0|0s in a pair, as we do in *SharedSize1100*, we find this measure is rank-equivalent to Hamming: $\#(1|1) + \#(0|0) = k - \#(1/0)$. Other classic methods that treat the affiliations as interchangeable include *Jaccard* similarity, *Cosine* similarity, and *Pearson* correlation. Euclidean distance is omitted here because for binary vectors, it is rank-equivalent to Hamming distance.

Some similarity methods have been designed to vary as a function of p_i, so that a pair's score is higher when the affiliation they share is rarer. What we refer to as *CosineIDF* is a familiar method for text documents, in which each affiliation is weighted by the log of its inverse document frequency $\frac{1}{p_i}$ before the cosine of the vectors is computed [83]. (Of the *tf-idf* weighting commonly used in information retrieval, the *tf* portion is not meaningful for binary vectors.) *Adamic/Adar* was developed for an instance of the current task: predicting friendships among people given shared affiliations (mailing list memberships and data from their personal homepages) [1]. *Newman* was proposed as a measure of collaboration strength in affiliation data (among scientists, based on the papers they coauthored) [96].

It turns out that in all of these methods, either $\text{Score}(X_1, X_2)$ or $\log(\text{Score}(X_1, X_2))$ can be rewritten as a sum of per-component scores, just as MixedPairs was in Section 4.2. These component-wise decompositions, shown in Table 4.3, provide a common basis for comparing the functions. The methods at the top of the table are the simple counts of components, while most in the bottom half are functions of p_i. The methods Jaccard, Cosine, CosineIDF, and Pearson normalize the scores based on the lengths (the total number or weight of 1s) of each individual vector. Since their denominators use the item lengths, these methods are not component-wise independent like the others. On the contrary, these give a score of 1 to any exact duplicate pair, whether in a dense or a sparse region. Adamic/Adar and Newman are the only measures that vary with m, the number of items in the data set. Notice all the zeros in the middle of the table: many measures sum only over components of type $b_i = 1|1$. When a method gives the same score (e.g., 0) to two types of components, generally that means it cannot distinguish between them. However, the normalized methods work differently: even though $\text{Score}(0|0) = \text{Score}(1/0) = 0$, the 1/0s increase the denominator, so they decrease the overall score, unlike the 0|0s.

Name	Definition[a]	Additive score per component			$f(p_i)$?	Distinguishes 3 values of b_i?	Symmetric about $p_i = 0.5$?
		per $1\vert 1$	per $0\vert 0$	per $1/0$			
SharedSize	$\#(1\vert 1)$	1	0	0			
Hamming	$-\#(1/0)$	0	0	-1			✓
(= SharedSize1100)	$\#(1\vert 1) + \#(0\vert 0)$	1	1	0			✓
Jaccard	$\frac{\#(1\vert 1)}{\#(1\vert 1)+\#(1/0)}$	$\frac{1}{\#(1\vert 1)+\#(1/0)}$	0	0		✓	
Cosine	$\frac{X_1 \cdot X_2}{\lVert X_1\rVert \lVert X_2\rVert}$	$\frac{1}{\lVert X_1\rVert \lVert X_2\rVert}$	0	0		✓[c]	
CosineIDF	$\frac{Y_1 \cdot Y_2}{\lVert Y_1\rVert \lVert Y_2\rVert}$	$\frac{(\log p_i)^2}{\lVert Y_1\rVert \lVert Y_2\rVert}$	0	0	✓	✓[c]	
Pearson	$\frac{1}{k}\sum_i \frac{(x_{1i}-q_1)(x_{2i}-q_2)}{\sqrt{q_1(1-q_1)q_2(1-q_2)}}$	$\frac{1}{k}\sqrt{\frac{(1-q_1)(1-q_2)}{q_1 q_2}}$	$\frac{1}{k}\sqrt{\frac{q_1 q_2}{(1-q_1)(1-q_2)}}$	$-\frac{1}{k}\sqrt{\frac{q_1(1-q_2)}{(1-q_1)q_2}}$ *		✓[c]	
Weighted Correlation	$\frac{1}{k}\sum_i \frac{(x_{1i}-p_i)(x_{2i}-p_i)}{p_i(1-p_i)}$	$\frac{1}{k}(\frac{1-p_i}{p_i})$	$\frac{1}{k}(\frac{p_i}{1-p_i})$	$\frac{1}{k}(-1)$	✓	✓	✓
Adamic/Adar[b]	$\sum_{i \in 1\vert 1} \frac{1}{\log(np_i)}$	$\frac{1}{\log(np_i)}$	0	0	✓		
Newman[b]	$\sum_{i \in 1\vert 1} \frac{1}{np_i - 1}$	$\frac{1}{np_i-1}$	0	0	✓		
SharedWeight11	$\sum_{i \in 1\vert 1} \log \frac{1}{P(x_i\vert\phi_i)}$	$\log \frac{1}{p_i}$	0	0	✓		
SharedWeight1100	$\sum_{i \in \{1\vert 1,0\vert 0\}} \log \frac{1}{P(x_i\vert\phi_i)}$	$\log \frac{1}{p_i}$	$\log \frac{1}{1-p_i}$	0	✓	✓	✓
MixedPairs	$\sum_{i=1}^{a} \log \mathrm{LR}(b_i \mid p_i, s)$	$\log \frac{s+(1-s)p_i}{p_i}$	$\log \frac{1-p_i+p_i s}{1-p_i}$	$\log(1-s)$	✓	✓	✓

Table 4.3: Similarity methods for vector pairs. Comparison of scores per component and method properties.

[a] The notation $\#(b_i)$ refers to the number of components of type b_i in the pair, and "$i \in 1|1$" is shorthand for "$i : b_i = 1|1$." For the cosine methods, $\|X\|$ represents the Euclidean norm of X. In CosineIDF, Y is the IDF-weighted version of X: for $j \in \{1,2\}$, $Y_j = X_j \cdot (\log \frac{1}{p_1}, \ldots, \log \frac{1}{p_k})$. In Pearson, q_j represents $\frac{1}{k}$ times the number of 1s in item X_j. When $X_1 = 0$ and/or $X_2 = 0$, which makes some functions above undefined, we assign a score of 0. [b] In the formulas for Adamic/Adar and Newman, we substitute np_i as an equivalent to the usual description: "the number of times affiliation i appears in the data set." In situations where p_i is provided theoretically, we use $\max(np_i, 2)$ in order to keep these functions well-behaved. [c] These methods further distinguish $b_i = 1|0$ from $b_i = 0|1$.

* The score for $b_i = 0|1$ is shown; for $b_i = 1|0$, q_1 and q_2 switch.

A few trends are visible. First, Score(1|1) is almost always positive, and if it varies with p_i, it decreases as p_i increases. If 1/0 has a non-zero effect, it decreases the score, and this decrease almost never depends on p_i. If $p_i < 0.5$, then Score(1|1) $\gg$ Score(0|0) ≥ 0. In Figure 4.2 earlier, we examined the per-component scores for MixedPairs. The bottom plots show these scores for the other functions that vary with p_i. Although their scales differ, the functions are roughly the same shape.

These diverse functions can be categorized according to which of three properties they satisfy (see right side of Table 4.3).

P1 Scores are weighted based on p_i. Specifically, Score(1|1) is a decreasing function of p_i.

P2 Scores are sensitive to all three values of b_i. That is, Score(X_1, X_2) changes if any b_i changes (among values $\{1|1, 0|0, 1/0\}$).

P3 Scoring is symmetric about $p_i = 0.5$. If $p_i > 0.5$, then the usual roles of 0 and 1 are switched: 0 becomes the more surprising event, and Score(0|0) > Score(1|1). Formally, for all x_{1i}, x_{2i}, and p_i, $\text{Score}(X_1, X_2 \mid x_{1i}, x_{2i}, p_i) = \text{Score}(X_1, X_2 \mid (1 - x_{1i}), (1 - x_{2i}), (1 - p_i))$.

The experiments will show that each of these properties is beneficial. Regarding P3, we find that it is important to correctly handle affiliations in which $p_i > 0.5$. We will show that when even small numbers of such affiliations are present, non-symmetric methods can sometimes be helped by a "flipping" operation. As for relative performance, we will see that the methods fall into four groups, depending on which combination of P1 and P2 they satisfy. Group 1 methods, which satisfy both, are generally the strongest, and include MixedPairs, CosineIDF, Weighted Correlation, and SharedWeight1100. Group 2 comprises the unweighted normalized methods, which satisfy P2 but not P1: Jaccard, Cosine and Pearson. Group 3 consists of the 1|1-based methods, satisfying P1 but not P2: Adamic/Adar, Newman, and SharedWeight11. Group 4 contains the simplest methods, SharedSize and Hamming. These do not perform as well, nor similarly to each other, but they can help us understand the others.

	Synthetic						Newsgroups	Reality Mining (Day; Week)			Congress
	D	A	E	F	B	C		Blue	Cell	Apps	
Num affiliations	100						734 to 1355	590; 1799	1263; 4031	44; 67	1074 to 1773
Median p_i	.01	.04	.06	.16	.22	.47	.01			.03; .07	.65 to .89
Fraction of $p_i > 0.5$	0	0	0	.08	0	.46	.002 to .009	0 to .001		.15; .13	.55 to .82
Best $\hat{s}$	.2 (true value given)						.001				(.01, .001, .001, .001; .4 .1, .4, .1)

Table 4.4: Data set properties (averages in trials).

4.6 Experiments

We run experiments using affiliation data in which the true pairs are known. From each data source, we construct small data sets containing 75 items each: 65 singletons and 5 (disjoint) true pairs. Each experimental trial consists of scoring all $\binom{75}{2}$ pairs, then computing the AUC of the true labels against the ranking induced by the scores. Plots display the average AUCs across 400 trials. One reason we evaluate using AUC is that it is unaffected by the proportion of positives to negatives in a data set; rather, it measures the separation between the two distributions of scores, as we noted in Section 4.3.2. In this task, the negative pairs will always greatly outnumber the positives, and the class skew increases with k. In order to get enough positives to estimate the AUC, it is more efficient to use many small data sets, as we do here, than fewer larger ones.

4.6.1 Data Sets

We generate synthetic data from the MixedPairs model using $s = 0.2$ for true pairs, $k = 100$ affiliations, and 69 different settings of $\phi = \text{MultipleBernoulli}(p_1, p_2, \ldots, p_n)$. In nine of those settings, p_i is a constant $\in \{0.1, 0.2, ..., 0.9\}$. In the other 60 settings, the p_i vectors are randomly sampled from either uniform distributions (labels A, B and C in the plots) or exponential distributions (labels D, E, and F) having means of 0.05, 0.25, 0.5, and 0.02, 0.1, 0.2, respectively.

The first real-world data sets are constructed from the 20 Newsgroups data, which contains 1000 articles posted to each of 20 Usenet newsgroups [105]. Within each newsgroup, true pairs are defined as articles that quote each other—that is, articles sharing a block of at least 10 consecutive words. This task is meant to evoke plagiarism detection, but in a setting in which the true label is easy to determine. In this setup, the articles are our items, the words present in an article are our affiliations, and the challenge is for the methods to detect the quoted articles using only a binary vector (word presence/absence) representation. Headers and punctuation are removed during preprocessing. Even with the binary representation, this task is easy when using the full vocabulary; to avoid ceiling effects, these experiments only use every fourth word in the vocabulary.

Next, we construct instances of the task from the Reality Mining data, collected from the cell phones of 94 individuals over a period of nine months [45]. In this scenario, the goal is to re-identify individuals: we define true pairs to be samples from the same phone during different time windows. For an individual's affiliations, we use either the set of applications run on the phone, the IDs of bluetooth devices scanned nearby, or the IDs of the cell towers providing service. For the size of the time window, we use either one day or one week.

Finally, we look at voting and sponsorship data from the U.S. House of Representatives [122]. The idea is to detect pairs of Congress members whose voting profiles are unusually similar compared to others in their party. Such pairs might be closely aligned in ideology, or one member might be guiding or imitating the other's votes. We validate the pairs against relationships that are more easily observed, those of cosponsoring numerous bills together [52].

For each of Congresses 110–113 (covering 2007–2014) and each party, we examine the affiliation network of members and the bills they voted in favor of. Ground truth labels are derived from the bills each member either sponsored or cosponsored (among all House Resolutions and House Joint Resolutions introduced that session): true pairs were those in the top 1% of number of shared cosponsorships[2].

In the synthetic data experiments, the methods are given access to the true values of ϕ and s. In the real data experiments, in each separate trial, ϕ is estimated from the (small)

[2] The sponsorship data is itself another affiliation network, so there is a potential circularity in choosing a measure of legislator tie strength to define ground truth. We use SharedSize of cosponsorships without flipping the high p_is.

data set at hand. Since the optimal s for MixedPairs is unknown in real data, we try a range of values and display the best result for each data set. In most of the real data, MixedPairs improves as long as $\hat{s}$ decreases, and we use $\hat{s} = 0.001$. At that parameter value, MixedPairs converges to WeightedCorrelation, as discussed in section 4.4.2.

Table 4.4 summarizes properties of the real data sets. While the synthetic data covers a range of values of p_i, the affiliation frequencies are quite low in Newsgroups and two of the Reality Mining sets; the majority of their affiliations are seen only once per small data set. In Reality Mining Apps, the p_i are moderate, and in Congress, more than half of them are above 0.5, which reflects the fact that bills are brought to a vote in Congress only when they are likely to pass. When scoring a data set, we drop affiliations with p_i of exactly 0 or 1 because they do not affect the methods' rankings (with the single exception of Pearson), and because intuitively speaking, they provide no signal.

4.6.2 Main Results

The synthetic experiment results, Figure 4.6, show how all 12 methods generally rise and fall together as a function of ϕ. The x-axis is ordered by the predicted (and roughly, actual) AUC of MixedPairs, as in Figure 4.5. On the far right are the settings with p_i closest to 0.5 (p_i constant at 0.5, followed by 0.4 and 0.6, 0.3 and 0.7, etc.); on the far left are those with the lowest p_i. In between, each ϕ label (A, B, C, etc.) ends up grouped together, with the labels ordered by mean(p_i). Labels B and C are indistinguishable to MixedPairs by symmetry about 0.5 (though see the discussion of P3 below). In the settings where p_i is constant (plotted with triangles), some weighted and unweighted methods converge.

Naturally, MixedPairs is optimal on its own data; the other top methods are Weighted Correlation, SharedWeight1100, and Pearson. As p_i increases, the groups of methods described earlier differentiate into clusters: with some exceptions, Group 1 (plotted with ●) > Group 2 (unweighted normalized, plotted with $\oplus$) > Group 3 (1|1-based and weighted, plotted with ■).

Group 4 (□), SharedSize and Hamming, is worth understanding. Notice that the lines cross: SharedSize is always one of the weakest methods, while Hamming goes from the worst, at low p_i, to the best, at $p_i = 0.5$. When p_i is near zero, most b_is in the data

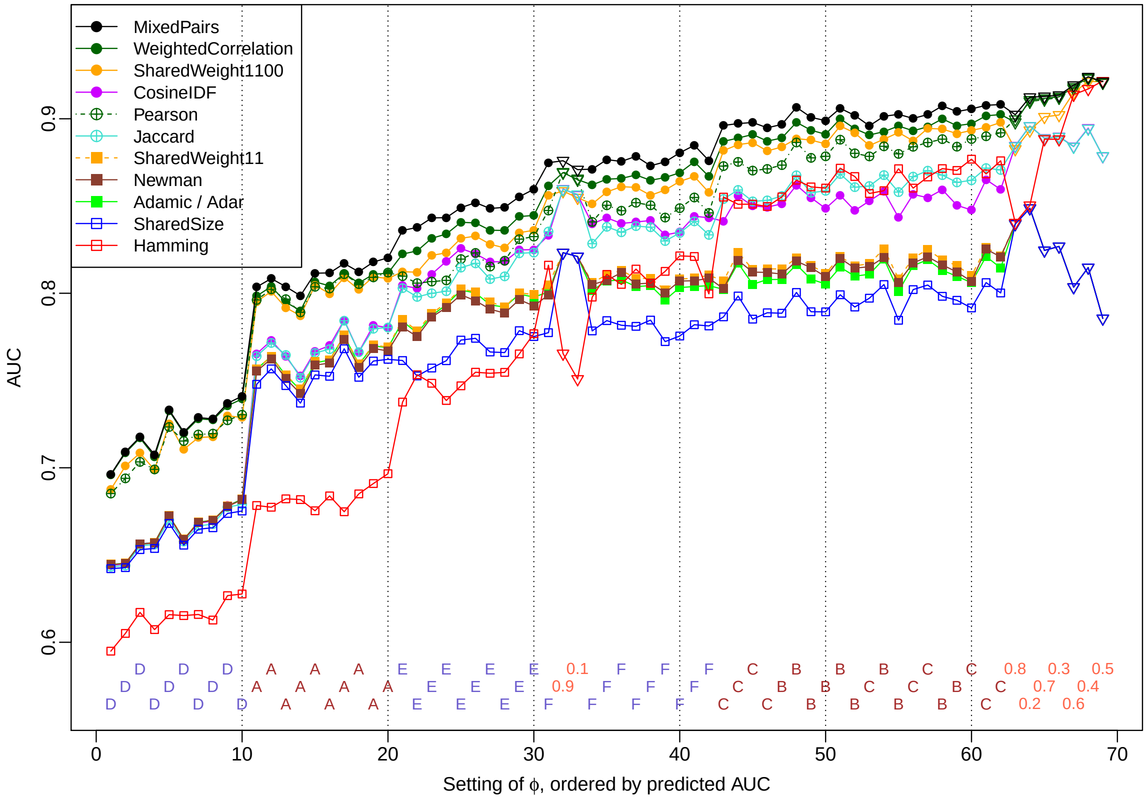


Figure 4.6: Synthetic data experiments. Each position on the x-axis corresponds to one setting of ϕ. In each trial, a data set of 75 items are generated, all pairs are scored, and the AUC of the ranking is computed with respect to the true set of 5 pairs. Each point shows the average of 400 trials for the setting; 95% confidence intervals are approximately +/-0.008, and they overlap between neighboring methods. During inference, the true ϕ and s are provided, and affiliations with $p_i > 0.5$ are flipped. (Compare to Figure D.1 to see the effects of flipping.) Cosine is omitted due to being visually indistinguishable from Jaccard.

are $0|0$ or $1/0$, but the occasional $1|1$s are extremely informative for detecting positives. Hamming cannot distinguish $1|1$ from $0|0$, so at low p_i, it is almost useless. At the other end, when $p_i = 0.5$, b_is of $1|1$ and $0|0$ are equally likely. There, the important distinction is the one Hamming can make—whether $b_i = 1/0$ or not—so it performs as well as optimal. Meanwhile, at the lowest p_i, SharedSize can identify the $1|1$s, and it even performs on a par with Group 3—perhaps there are so few $1|1$s that it suffices to count them—but it cannot keep up when the p_is (and their range) increase. At $p_i = 0.5$, the inability to distinguish $0|0$ from $1/0$ makes both SharedSize and Group 3 fall off.

In the real data sets (Figure 4.7), the results vary widely, but a few trends can be observed. First, for the most part, the methods perform in line with others of their group. In Newsgroups, Group 1 $\geq$ Group 3 $\geq$ Group 2. In Reality Mining and Congress, generally Group 1 $\geq$ Group 2 $\geq$ Group 3. SharedSize is usually below these groups. Second, the relative performance of the methods seems to be affected, like in synthetic data, by whether the p_i are low or high (see Table 4.4; Figure 4.7 roughly orders the real data sets by p_i, except for Reality Mining Apps). We can certainly see this with Hamming, despite all other the differences one would expect across domains. In Newsgroups and Reality Mining Bluetooth and Cell towers, where the median p_i is only 0.01, Hamming is far below SharedSize. In Reality Mining Apps and Congress, where p_is are higher, Hamming moves to the middle of the pack. There is also some indication that Group 2 pulls above Group 3 only at these higher p_i, where distinguishing between $0|0$ and $1/0$ becomes more important, and where the weights assigned by weighted methods vary less.

We can see the effect of property P1, weighting scores based on p_i, by examining methods that differ only in that property. SharedSize is the unweighted version of the Group 3 methods, and as we have seen, it usually performs worse than them. Also, CosineIDF is the weighted version of Cosine, and CosineIDF generally performs the better of the two—in the real data, if not in synthetic. Property P2, distinguishing among all three values of b_i, is supported by the comparison of SharedWeight1100 to SharedWeight11. Their only difference is in whether a $0|0$ receives a small positive score versus a score of 0, like $1/0$ does. In almost every example, SharedWeight1100 achieves a substantially higher AUC.

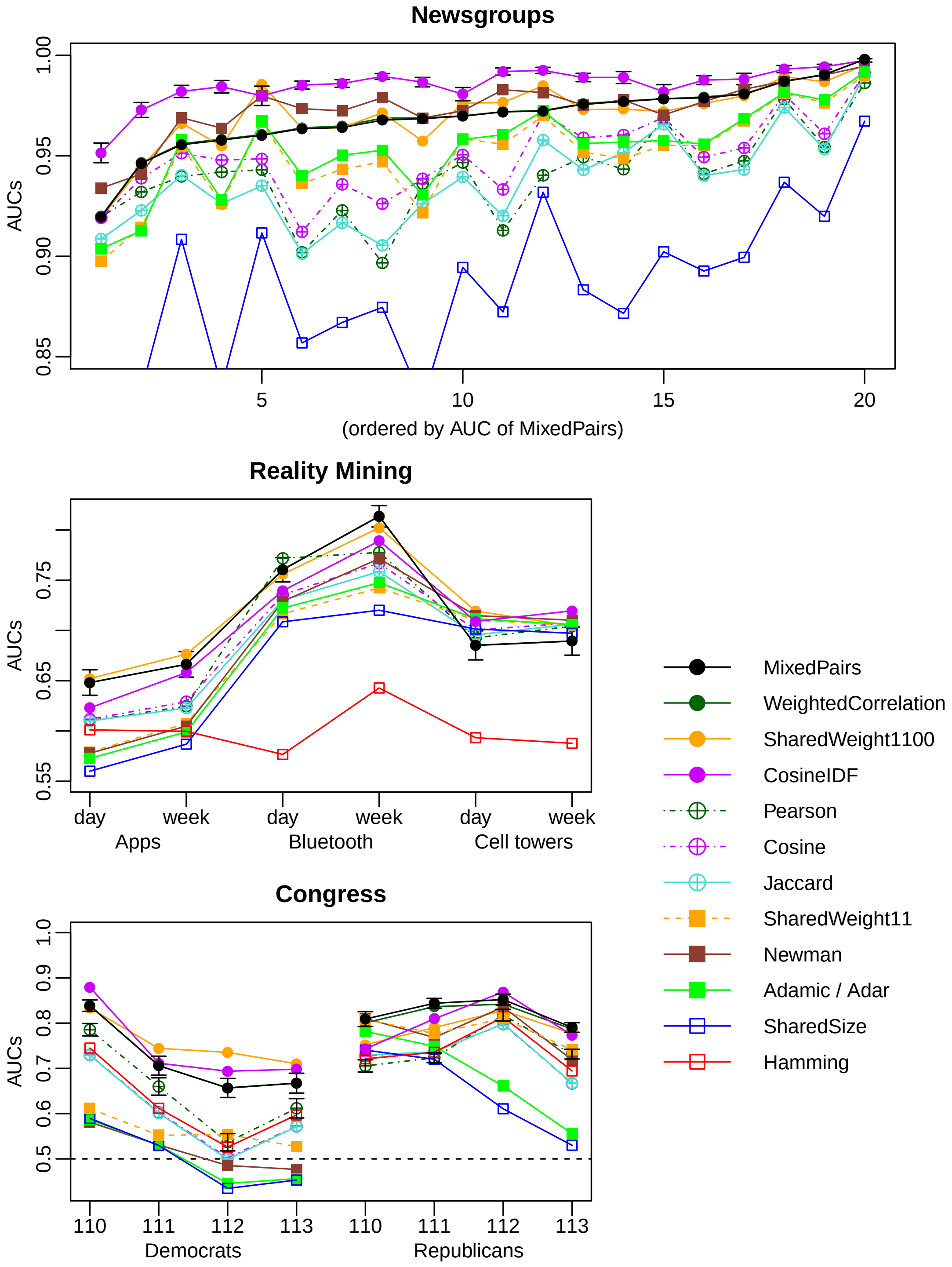


Figure 4.7: Experimental results on real data, for Newsgroups, Reality Mining and Congress. Affiliations are not flipped. (See Figure D.2 for flipped version.) Error bars show 95% confidence intervals. Weighted Correlation coincides almost exactly with MixedPairs; in Congress, Cosine coincides almost exactly with Jaccard. In Newsgroups, Hamming, not plotted, never exceeds 0.70.

Finally, we examine the role of P3, symmetry about $p_i = 0.5$ in scores assigned. Methods lacking this symmetry may have been intended only for situations where $p_i < 0.5$, and they generally have impaired performance in the presence of affiliations for which $p_i > 0.5$. To help them compensate, we tried an approach of "flipping" the corresponding affiliation bits: swapping the 0s and 1s in that component, throughout the data set, such that the new $p'_i < 0.5$. For methods that already satisfy P3, this operation has no effect.

Overall, the impact of flipping is substantial yet inconsistent, so it complicates the interpretation of experimental results. It helps more often than it hurts; it changes the relative ordering of methods; and it provides evidence for the groupings we have described, as methods tend to move up or down together with their group.

In the synthetic data, flipping is so unequivocally helpful to the non-symmetric methods, that for the sake of readability, this is what Figure 4.6 shows. Without it, in some high-p_i settings, the AUCs of some methods otherwise drop by up to 0.15 (see Figure D.1 of Appendix D).

In real data, the effects of flipping can again be dramatic, but they are also mixed. Figure D.2 shows the full results, and Figure 4.8 summarizes the effects for all three real data sets. (Reality Mining Bluetooth and Cell towers are omitted because they have almost no high-frequency affiliations to flip.) For the 1|1-based methods (Group 3 and SharedSize), flipping helps almost uniformly. For the normalized methods of Group 2, the direction of change is inconsistent; CosineIDF generally moves in the same direction as them when flipped, but not nearly as much. It is interesting that such noticeable effects can result when just a tiny proportion of affiliations are flipped: in Newsgroups and Reality Mining Apps, an average of 5–7 affiliations have $p_i > 0.5$, out of around 1000 total, or 50, respectively. All in all, the operation of flipping seems useful to explore and to better understand for methods lacking symmetry. Of course, this "fix" is entirely unneeded for MixedPairs, Weighed Correlation or SharedWeight1100.

The Group 1 methods, which satisfy all three properties, are consistently among the top performers. Of these four methods, MixedPairs is the least practically convenient, because it requires a parameter. Weighted Correlation and SharedWeight1100, both derived in Section 4.4, are easy to explain and to use. CosineIDF is more typically used for text data than for

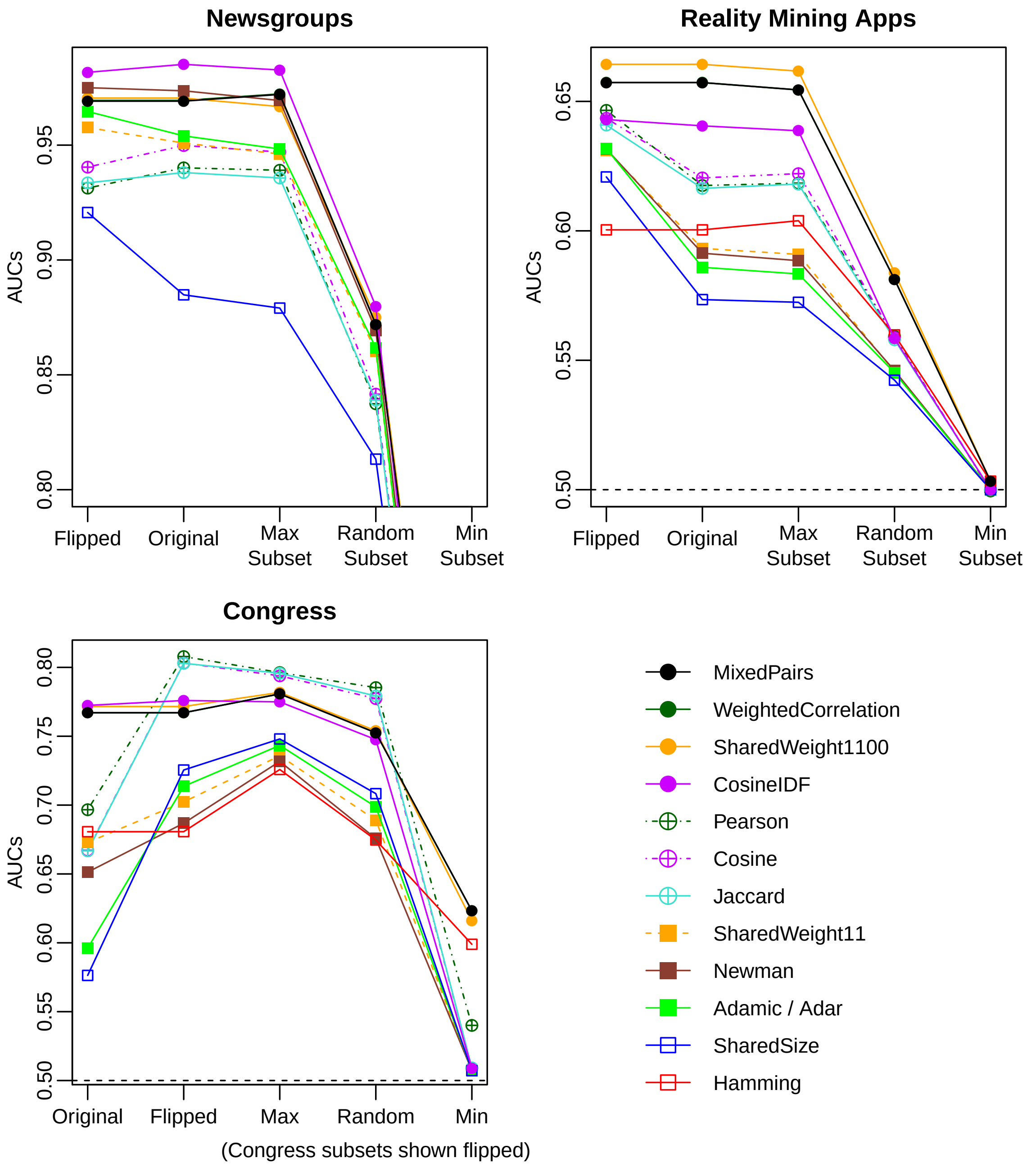


Figure 4.8: Average AUCs across all experiments and trials for Newsgroups, Reality Mining (Apps only), and Congress. Left side of each figure shows the effect of flipping affiliations having $p_i > 0.5$; note the different x-axis for Congress. Right side shows performance using only one quarter of the affiliations, with subsets chosen three ways.

affiliation data, and its particularly high performance in Newsgroups suggests there may be properties of text domains that it is the best suited to handle.

4.6.3 Varying the Distribution of Affiliations

Section 4.3.2 showed how, at least with MixedPairs and synthetic data, the overall ability to detect pairs in a data set is related to the entropy of the affiliation probabilities, and in particular, performance increases as the p_is approach 0.5. If such properties hold in real data as well, we could use them to select or manipulate data sets, to make pairs either more or less easily identifiable. In the real data, we do not find a relationship between $H(\phi)$ and performance, but we do find that the affiliations nearest 0.5 contain the most information, as we demonstrate next. According to theory, if we use only a subset of a data set's affiliations, the AUC should always decrease compared to the original. If we take a subset of affiliations with values of p_i near 0.5, the AUC should be higher than if we take a subset with values of p_i closer to 0 or 1.[3]

We test this theory by rerunning inference in the real data sets, but with access only to subsets of the affiliations. In the Random setting, we choose a random subset of one quarter of the affiliations. In the Max setting, we choose the quarter of affiliations with values closest to 0.5, and in the Min setting, those with values farthest from 0.5. The subsets are chosen independently for each small data set. (The calculations of Max and Min are based on the p_i estimated across all items, before the small data sets were constructed, but these subsets vary from trial to trial due to tied values of p_i.)

The anticipated behavior is that, for each trial, the affiliation subsets should perform worse than the original full set of affiliations, and that among the affiliation subsets of equal size, Max > Random > Min. Figure 4.9 shows representative examples of results in the Congress data, for two methods. In Figure 4.8, the right side of each graph displays the full averages for these experiments. In the Reality Mining data that the plot omits, affiliation

[3] The plots shown in Appendix C suggest that for non-symmetric methods, the optimal values of p_i may be much closer to 0. While determining these values is left to future work, we point out that for the lower-p_i data sets here, the affiliations nearest 0.5 are mostly also below 0.1; therefore, Max might simultaneously be the set nearest other, lower, optimal values.

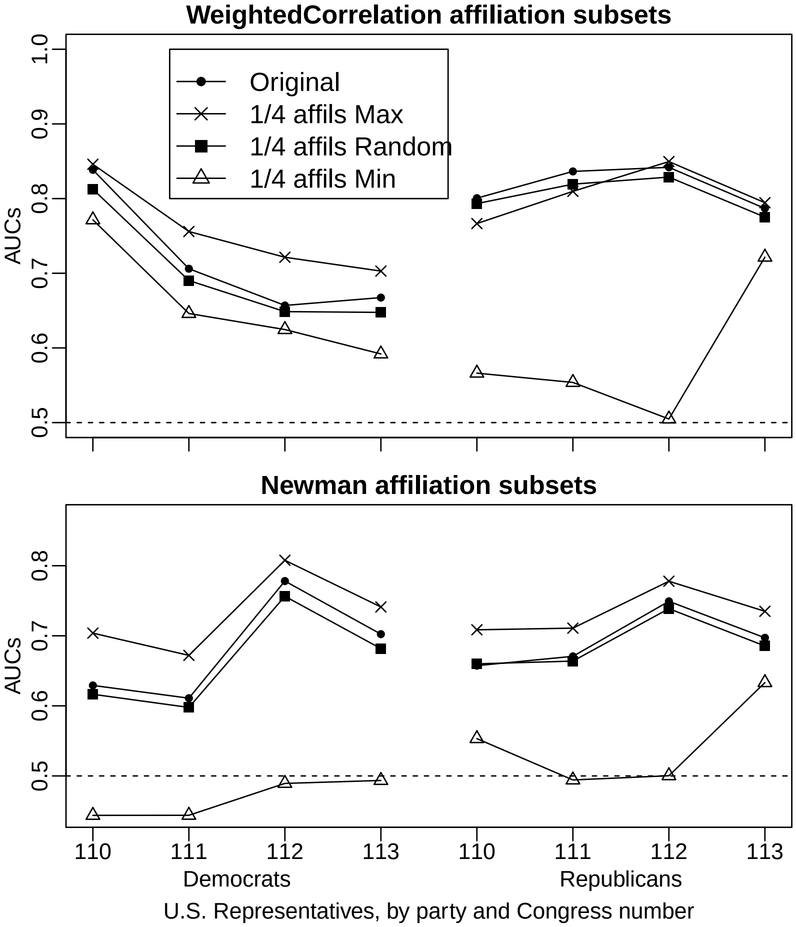


Figure 4.9: How performance degrades depending on which 1/4 of affiliations are left available. Example results from Congress data, with high p_is flipped.

subset results are consistent with the others. The Congress experiments are shown with affiliations flipped because this condition comes out more cleanly.

Across all data sets and methods, the results are fairly consistent, but they are not quite as expected. Among the affiliation subsets, Max > Random > Min; however, often Max $\geq$ Original as well. Min often gives near-random performance because, due to the sampling process, the resulting data set ends up as all zeros. The fact that Max performs so well, while using only a quarter of the affiliations, suggests opportunities for reducing storage or computational costs with minimal loss of AUC, simply by ignoring rarer (and also extremely frequent) affiliations.

4.7 Discussion

The main takeaways of Sections 4.5 and 4.6 are how component-wise analyses both of scoring methods and data sets are fruitful for understanding experimental performance. Rewriting familiar methods as their per-component functions calls attention to their respective commonalities and weaknesses (Table 4.3). Against the backdrop of the "ideal" LLR curves from MixedPairs, we can see how methods lacking symmetry about p_i would break down when the true $p_i > 0.5$; how methods that do not weight based on affiliation frequencies would be sub-optimal, especially when the ideal weights vary widely; and how methods that cannot distinguish all three values of b_i would do poorly whenever the distinction they lack is important (at high or low p_i, respectively). In both the real and synthetic data, we have seen evidence of how the p_i vector of the data set affects performance: when p_i approaches 0.5, detecting pairs gets easier—that is, there is more signal to work with, for methods that can use it. This is because the distribution ϕ becomes more uniform, as discussed in Section 4.3.2.

In terms of the best methods to use for this task, we cannot offer a complete characterization of what causes relative differences among the methods across the data sets above. However, for this task and evaluation measure, Group 1 seems robust—CosineIDF, Weighted Correlation, SharedWeight1100, and MixedPairs. They satisfy the three key properties (except for CosineIDF's lack of symmetry, which seems not to hurt it in the real data), and they are at the top of the overall averages (Figure 4.8, Original conditions).

As a scoring method, MixedPairs itself has room to improve. In most real data sets, the best $\hat{s}$ we found is 0.001, which seems unrealistic. If the data were generated according to the model, in our data sets of about 1000 affiliations, this would correspond to positives sharing not even 1 more affiliation than negatives. On one hand, this $\hat{s}$ makes the practical recommendation easy: use Weighted Correlation instead, which produces the same scores (up to rank equivalence) and needs no parameter. On the other hand, it suggests a mismatch between the model's assumptions and real data. Two aspects worth reconsidering are the multiple Bernoulli model, which implies that affiliations are independent (unlikely in real data), and the positive pairs model, which lets X_2 copy 0s from X_1 (implausible in some domains).

A third aspect seems the most likely culprit: the implicit assumption that all items have roughly the same number of 1s. Under the model, the distribution of 1s per item is a narrow normal; in real data, this distribution, which could also be described as the distribution of item lengths or of affiliations per person, is far wider. When two items differ in length by d, they are forced to have at least d 0/1s. Under MixedPairs, 0/1s produce negative scores; minimizing their effect, which depends on s, could be what causes the best $\hat{s}$ to be so low. In information retrieval, document modeling has progressed away from multiple Bernoulli models, over the years, towards models that normalize based on document length, such as CosineIDF does, as well as towards multinomials for representing count data [69, 83]. Extending MixedPairs in these directions could prove beneficial.

Regardless of these issues, MixedPairs has great value as a tool. Its simplicity, and in particular its component-wise independence, are what make the rest of the story visible. The curves of Figure 4.2 are easy to interpret, and they furnish the non-obvious suggestion that putting a small positive score on $0|0$ can add up to a large difference in performance (as we see when comparing SharedWeight11 to SharedWeight1100). Since MixedPairs is generative, we can use it to create synthetic data. Since we can also model the distributions of scores for the data it generates, we can explore the effects of parameters even without running experiments.

Through these approaches, we have been able to analyze factors that contribute to the comparative performance of many methods and address theoretical questions that are not usually asked about similarity scores. These questions include which items are most conspicuous (if paired), and how changes in the overall data distribution, including the number and frequencies of affiliations, affect the ability of optimal methods (or other methods) to detect pairs. Finally, we put forth the four methods listed above as recommendations for this task or to use as baselines in more complex domains.

CHAPTER 5

RELATED WORK

In this chapter, we return to discussing ASOUND, the detection of unusually similar objects in unlabeled data, from a domain-agnostic point of view. Special attention is paid to the domains from Chapters 2–4. In the first section, we review technical approaches to the problem. After that, we survey a number of research areas containing similar problems.

Among these areas, we turn first to social network analysis, the source of many of our problem instances. It offers relatively few methods that satisfy our requirements, so we then cast a wider net. We find closely related models in information retrieval, entity resolution and forensic science—likelihood ratio methods that have been influential in these fields. Some of these areas provide guidance regarding the task's computation issues, discussed in the final section.

5.1 Approaches for Identifying Similar Entities

Open any textbook on analyzing data in a domain, and it will describe measures of distance for objects in that domain. Distance measures can be turned into similarity measures, and vice versa, by inverting or negating them or subtracting from their maximum. In many problem-solving contexts, basic similarity methods are what people use and all they need; the preceding chapters illustrated how Euclidean distance, Hamming distance, and set intersection, respectively, can be sufficient for ASOUND under certain conditions. Some standard similarity or distance measures for points also include Manhattan and Mahalanobis distances; for sets or binary vectors, set intersection or distance, Jaccard index, Sørensen-Dice index, or cosine similarity (see also Section 4.5); for strings, Levenshtein edit distance or Jaro-Winkler similarity; for graph nodes, shortest path distance between them, or similarity between their sets of neighbors [19, 25, 46, 74, 76]. However, since we have formulated the task with particular constraints—namely, that we identify entities more similar than chance would account for, and that we use the full data set as a source of information about independent entities—we prefer hypothesis-driven tools to these general-purpose methods.

Some tools sound applicable but are not. One common task that ASOUND resembles is unsupervised clustering: partitioning data instances into related groups, often with a goal of understanding the data's latent structure [74]. However, ASOUND differs because we expect our "clusters" (pairs) to be small and rare, and the majority of the entities to be drawn from a single general population. If the data do contain large-scale clusters, we want to make sure they are described by the general model, so that we can recognize deviations from them. Another related framework is spatial point processes, which describe patterns among points distributed in space, such as the locations of trees in a forest. The Strauss process can model situations where points attract each other to form local clusters [6]. However, as with clustering, point process models are better suited for globally describing a data set.

Almost all approaches we review are pair-based, focusing on a symmetric score to assign independently to each pair. This focus contrasts with other possibilities such as treating a pair asymmetrically, choosing the best-matching pairs from among mutually exclusive candidates, or reasoning about the structure or size of groups connected by high-scoring links.

For scoring pairs, another near-miss task is anomaly detection: identifying points (or here, pairs) that are outliers from the rest of the data. While we do wish to detect pairs that are near each other—nearer than most other pairs in the data set—we are not strictly looking for "anomalously similar" pairs; rather, we must also take into account the pairs' rarity. Once we have a scoring measure, we hope extreme values of the score will signify true pairs, but this is not anomaly detection in its classic sense. ASOUND has more in common with significance testing: we want to distinguish true pairs from singletons that are close together by chance. Yet we do not want to merely test significance against a null hypothesis of independence; that approach would return pairs composed of unusual points in any configuration. To specify how the positive pairs should look, we introduce a second model.

Probably the simplest statistical method for distinguishing among two hypotheses, and the basis for our work, is the likelihood ratio. In classical hypothesis testing, the Neyman-Pearson lemma states that in order to distinguish between two fully specified models H_0 and H_1 for data D, the most powerful test statistic is the likelihood ratio $\frac{P(D|H_0)}{P(D|H_1)}$ [24]. In Bayesian statistics, when there are no model parameters to integrate over, the same quantity is used to compare the two models and here is known as the Bayes factor [67]. In three of the areas surveyed below, foundational work is based on likelihood ratios: see Robertson and Jones [106] for information retrieval, Fellegi and Sunter [50] for entity resolution, and Lindley [79] for forensic science.

If the models are not known, but labeled examples are available from each class, then the standard machinery of supervised learning can be applied: choose features to construct, train a classifier on the labeled examples, and use it to classify the pairs in question [62]. Finally, when the data are unlabeled and can be modeled as having latent structure, an expectation-maximization (EM) algorithm may be suitable [24]. EM is used to infer latent variables in situations where a likelihood model is provided but some variables are unobserved. For instance, one could specify that there are two classes of data, each of a known form but with unknown parameters. To maximize the full joint likelihood, the algorithm alternately assigns points to classes and updates the parameter estimates of each class. Since EM is often applied to mixture models, it might be possible to use it with our models of Chapters 3 and 4 in order to infer the parameters of the positive model and the class labels simultaneously. Such models have been widely applied in entity resolution [12, 63, 139].

5.2 Social Network Analysis

This dissertation, particularly Chapters 2 and 4 with their affiliation data, has a natural home within the social networks research community. With that term, we are referring broadly to work by computer scientists, physicists and social scientists studying human social networks, online social systems, and models and algorithms for graphs (or synonymously, networks) in general. Depending on the community, these efforts variously focus on modeling the global structure of graphs, understanding the characteristics of a particular data source or its underlying social processes, or developing predictive algorithms applicable to graph-structured data.

When we emphasize the bipartite graph representation of its input data, ASOUND resembles link prediction among the "people" (item nodes) in its graph (see Section 5.2.2 below). When, like in Chapter 2, we further aggregate the inferred pairs into clusters, the work also relates to projects identifying small groups, often with a fraud-detection angle (see Section 5.6).

5.2.1 Models of Graphs

Any project in which graph structure plays a role should consider the applicability of existing models of graphs. In the networks literature, one major thrust is designing models of large-scale graph structure to adequately capture properties seen in real-world data. Many of these properties are described in a review by Newman [97], and Newman et al. offer a collection of key articles in the developing field [98]. Some influential paradigms include preferential attachment models, which are generative processes that replicate distributions of node degrees and other commonly observed properties [7]; exponential random graph models, which explain edges based on frequencies of local graph structures [133]; stochastic blockmodels, which explain edges based on latent communities [116]; and latent space models, which explain edges based on their nodes' proximity in a latent coordinate system [64]. Goldenberg et al. present a comprehensive survey of these global network models [56]. Some more recent innovations include mixed membership stochastic blockmodels, in which each node belongs to multiple latent groups and links are explained by group-specific connection probabilities [3], and Kronecker graph models, which can generate a number of properties observed in real-world networks [73]. As is typical in networks research, most of these models are initially developed for unipartite graphs, but some are later extended to more complex data such as graphs with multiple types of nodes or edges, graphs with node attributes or edge weights, or dynamic graphs.

A few such models have been developed specifically for bipartite graphs. Skvoretz and Faust were the first to formulate exponential random graph models for this setting, and Wang et al. summarize the current inference methods and local graph features available for them [114, 131]. For situations involving bipartite graphs augmented by edges within one node type (e.g., people connected both to affiliations and to each other), Chang and Blei offer a mixed membership, latent variable model in which words are generated by latent topics and the similarity of topic vectors determines which documents are linked [26]. To add time dynamics to these social-plus-affiliation graphs, Zheleva et al. propose a generative model in which nodes and links of both types are added progressively [148]. More recently, Snijders et al. have introduced an alternative dynamic model for these graphs, a stochastic actor-oriented model in which the node set remains constant, but edges come and go [117].

Finally, dimensionality reduction techniques—applicable not only to bipartite adjacency matrices, but to any matrix—have been used widely on graph-structured data. In information retrieval, they are used to model collections of documents connected to words [16, 43], and in collaborative filtering, to model users connected to ratings [121]. Of particular interest, singular value decomposition creates a lower-dimensionality description of each item [43, 121], and latent Dirichlet allocation, or more broadly, topic modeling, uses hierarchical Bayesian models to infer a latent mixture of topics for each item [16].

In ASOUND, it is not clear that fitting one of these models to the bipartite graph structure, which is known and fixed, would help predict the other type of edge—between people—of which we have no examples. Some large-scale graph models have indeed been used to predict edges, for instance by choosing the highest-probability edges from a set not seen by the model [3, 33]. In Chang and Blei's relational topic model, the bipartite structure helps predict the other type of edge; however, in that situation, both types of edges are present in the input data, and the model learns their joint structure [26]. Similarly, from early work on link prediction, Taskar et al.'s Relational Markov Network learns a global model of edge structure plus node and edge attributes, but it too needs training data [127]. For ASOUND, it might be possible to leverage node attributes inferred from global models to characterize the similarity of two nodes. Our approach in Chapter 4 is essentially a simple global model of the bipartite structure, one in which, importantly, entities are either independent or have a latent tie, and links are conditionally independent given the node properties. But since the goal in ASOUND is to predict links, we look next at that literature, with its more local models.

5.2.2 Predicting Links

In concert with the rapid expansion of interest in network data over the last fifteen years, the subject of link prediction has emerged out of a few exploratory papers into a widespread publication topic. Lichtenwalter et al. suggest why: “Put simply, any environment that naturally maps to a network probably has an equally coherent mapping from link prediction in that network back to an important question in the environment” [77]. A recent survey by Wang et al. gives a comprehensive overview of the topic [132].

The “canonical” link prediction problem is to predict future edges in a unipartite graph. Many early papers addressed the prediction of new social collaborations [76, 77, 100, 129]: given a set of collaborative events, such as papers written during a given time period, a unipartite graph is created by linking all pairs of people who share an event (i.e., as a projection of an original bipartite graph), and the task is to predict which edges will newly form in the next time period. Liben-Nowell and Kleinberg’s empirical study of about a dozen graph-topological features across five physics bibliographic datasets has served as a starting point from which many other features have been developed [76]. Of these features, one major class is based on the immediate graph neighborhood of a pair, such as common neighbors (i.e., SharedSize), Jaccard and Adamic/Adar, while another class characterizes the graph connectivity between the pair, such as shortest path length, Katz score, and random walk scores such as personalized PageRank [34, 76, 132]. Liben-Nowell and Kleinberg also experimented with dimensionality reduction techniques. They reported surprisingly good performance from their simplest method, common neighbors. In experiments, it usually beat Jaccard, and Adamic/Adar was often best of all, along with variations of Katz score.

In parallel with the development of these features, other work introduced a supervised approach to link prediction. The basic supervised approach uses a classifier to combine topological features from above with content-based features, such as similarity of the authors’ interests [61, 100, 127, 129]. O’Madadhain et al. described how training and applying a classifier to individual edges independently is a more scalable approach than previous alternatives, such as the global model presented by Taskar et al. (who also implemented local models for comparison) [100, 127]. Most link prediction work today uses this supervised approach, fitting a model to a set of training edges (and non-edges). The models themselves come in numerous forms and degrees of complexity. In support of this approach, Lichtenwalter et al. argue that supervised learning has strong advantages over unsupervised methods for link prediction; for instance, it can flexibly model the target concepts across different problem settings and incorporate ensemble methods to improve performance [77]. In ASOUND, the task is by definition unsupervised, so supervised approaches are not directly applicable. However, given all the newly developed topological features, it may be interesting to examine them for use in future models.

The power that comes with using a graph abstraction to represent data can also serve as a weakness: everything looks the same as a graph. That is, differing needs and assumptions among problem domains are sometimes overlooked. Link prediction's early scenario of predicting new collaborations, for example, is a poor match to the link prediction needed in ASOUND. Apart from the issue of supervised versus unsupervised tasks, classic link prediction expects a unipartite graph as input, while ASOUND needs a full (unprojected) bipartite graph. Moreover, ASOUND's true pairs generally share affiliations, which means they are *already* linked in the unipartite projection of the data.

As link prediction has grown in popularity, however, a multitude of variations have been introduced. In addition to predicting the future state of a graph, link prediction is used to recommend people to follow on social media [8], to recognize the same person in different graphs [146], and to de-noise input data by adding or removing links [3]. Wang et al. discuss work predicting repetition of existing links over time, link disappearance, inactive links, link reciprocity and triadic closure, tie polarity, tie strength, links in bipartite graphs (e.g., collaborative filtering), and links in heterogeneous graphs (those having multiple types of nodes or edges) [132]. One line of research relates most closely to ASOUND: inferring social ties from similar online activity.

5.2.3 Inferring Social Ties

The explosion of online data with a social network component has made it possible to study the interplay of social ties and behavioral data in numerous domains, and of particular interest to us, to predict or infer social ties from the behavioral data. In contrast to the abstract formulations of some link prediction problems, work in this area is quite data centric. The literature differentiates among types of behavioral data, and there is an emphasis on understanding the constraints and human processes that generate each type. As such, many of these papers contain descriptive and exploratory analyses of new data sources; these analyses in turn guide new models and features to construct. Most of the papers below contain a link prediction component—whether for inferring existing social ties among users [2, 39, 41, 86, 101, 109, 142], predicting in-person interactions [45, 149], or recommending items to users [81, 142]. The most popular link prediction technique is supervised learning, but some papers use unsupervised methods [2, 58, 104]. A minority present more complex models, such as models of multiple link types or joint models of attributes plus structure [109, 142].

One area of active interest is geographic mobility patterns and the extent to which we can infer friendships from similarity in people's spatio-temporal movement. Cho et al. offer a model of human movement as a function of previous locations, time of day, and locations of one's friends, applying it to data from cell phone location traces and to the location-based social networks Gowalla and Brightkite [28]. Others predict social ties using either cell phone location traces [41, 45, 104], check-ins to location-based social networks [101], or spatio-temporal co-occurrences of photographs posted to Flickr [39].

Among these, the method most similar to ours is that by Crandall et al. [40], in which pairs of friends travel to locations together to take pictures. This paper derives a likelihood ratio calculation from generative models in which positive pairs travel either independently or together, depending on the day. The work by Provost et al. [104] is interesting in that it uses unsupervised methods to compute geographic similarity among users, then—lacking ground truth social ties—evaluates whether the methods succeed at linking different traces from the same person. Also of note is Eagle et al. [45], the source of the Reality Mining data we use in Chapters 3 and 4. This project uses mobile phones to collect location and proximity data among users, then compares how the behavioral data matches self-reports of friendships. The paper examines features such as the amount of time spent together during or outside of work hours, phone calls, and having friends at work as they relate to reported proximity, reported friendships, and job satisfaction.

To differentiate from location-based social networks, Liu et al. introduce the term "event-based networks" for platforms such as Meetup, in which users form social communities online with the explicit purpose of meeting in groups together offline [81]. In the former, offline co-occurrences with friends take place only rarely; in the latter, all offline meetings are with friends, as users do not record other (individual) movements in the system. Zhuang et al. in turn use the term "ephemeral social networks" to describe the quick-forming social structures of conferences or other temporary, immersive events; they try to predict who will interact in person, using people's attributes and pre-existing social ties [149]. Given a bipartite graph of people and events, Gupte and Eliassi-Rad present an axiomatic approach to inferring tie strength; see Section 5.2.4 below for more details [58].

In some settings, we are given observations of people interacting, but not all interactions are strong or relevant. One of the earliest empirical comparisons of social ties and shared affiliations, by Kossinets and Watts [68], looks at predictors of new email relationships, finding that shared (and strong) mutual friendships are the most important, followed by shared classes, while shared attributes have little effect. In other work with emails, De Choudhury et al. question the operational definition and practical meaning of observed relationships [42]. They show that very different "true" networks result when they vary the threshold on email volume that defines a link, and they argue that the right network to use is simply the one that performs best on the subsequent task of interest. Finally, Merritt et al. present a paper that is more typical of this genre: using a massive data set about gamers playing *Halo: Reach* online, they construct features of player pairs given their observed matches and train a supervised model to predict friendships [86].

Another category of online behavioral data is produced by friends without them necessarily interacting. Mitzlaff et al. discuss how the concept of homophily—similarity among friends—can be extended to similarity among the actions of friends, in what they call the "social distribution hypothesis" [91]. They demonstrate how the interaction strength between pairs of users on Twitter, Flicker or Bibsonomy correlates with similarity in the pairs' hashtag and folksonomy vocabularies, as well as the pairs' geographic proximities. In other work with similarity of vocabularies, Aiello et al. focus on social annotation sites in which users post and annotate items such as URLs or songs [2]. They develop a null model to check whether pairs' activities are more similar than would be expected by chance, and they experiment with a number of ways to aggregate the $(user, item, tag)$ triples into similarity scores in order to predict links between users. Using graphical models, Sadilek et al. exploit textual similarity on Twitter, along with geotagged co-locations and an observed portion of the social network, to infer the remaining social edges [109]. In another more algorithmic contribution, Yang et al. develop a joint probabilistic model of friendships and interests, which they apply to a Yahoo! social network to predict (or recommend) both social ties and applications installed [142].

Back in the offline world, as Chapter 2 mentions, animal biology has a related concept called "association indices" for measuring the strength of association between individuals. Given a number of sightings of herds or schools of animals at different times and locations, these measures are used to estimate family ties by examining which individuals are present at each sighting. When individuals frequently co-occur, they are determined to be linked; and then these links are used for social network analysis of the animal populations [11, 22, 70, 137]. Regarding Chapter 2's dynamic networks, a few other papers bear mentioning. Magdon-Ismail et al. [82], searching for hidden groups in a unipartite social network, propose a Markov chain model of how individuals' group affiliations change over time, and Baumes et al. [9] follow up with a more tractable model. The caravan identification task mentioned in that chapter's introduction has a realistic motivation from the military: using airborne video surveillance data to detect convoys moving on the ground or infer other vehicle activities [21, 102]. The image data available, however, is not nearly clear enough to distinguish the identities of different vehicles.

5.2.4 Other Methods

In being unsupervised yet driven theoretically, the ASOUND problem makes different assumptions than most of the link prediction work discussed above. One theoretical paper on unipartite link prediction, by Sarkar et al. [110], strikes a similar balance. It assumes a latent space model, in which links form between nearby nodes, and it provides results on why, under that model, certain similarity measures (such as SharedSize and Adamic/Adar) are likely to perform well, while others (such as those based on long paths in the graph) are not.

For affiliation data, recent work by Gupte and Eliassi-Rad appears to address exactly our task of latent tie prediction given a bipartite graph: specifically, inferring tie strength among pairs of people based on events of different sizes they attended [58]. This work formalizes desirable properties of similarity measures into a set of axioms, characterizes the form of measures that satisfy these axioms, and shows how various common measures either fit this form or fail to satisfy some of the axioms. Our MixedPairs and other models violate several of their axioms—for example, the requirement that that only 1|1s be used as evidence for ties. The conflict seems to be that their axioms describe properties only of ties *caused* by people attending the observed events together. In MixedPairs, since ties exist before affiliations are chosen, they can result in both shared 1|1s and 0|0s; for instance, friends might jointly decide to stay home instead of going to an event.

When picking comparison similarity measures for affiliation data, we could only sample from the vast set that have been proposed across fields [19, 25]. MixedPairs is useful as a reference method because it is backed by a specific model and satisfies desirable properties we enumerated. Some popular methods obviously lack one of the properties, and so we can conjecture how they might behave within our framework (subject, of course, to experimental validation). For instance, Dice coefficient ($\frac{2\cdot\#(1|1)}{2\cdot\#(1|1)+\#(1/0)}$ in our notation) [25] is unweighted, so it might resemble Jaccard, and the Delta and Linear methods discussed by Gupte and Eliassi-Rad [58] are 1|1-based, so they might behave like Adamic-Adar and Newman.

Yet it would be interesting to take a closer look at certain probability-based methods. One such example is Lin's information-theoretic, axiomatic definition of similarity between objects of any type, provided they can be described with probability distributions [78]. This measure would allow putting weights on affiliations, but it is also normalized such that the similarity of any object with itself is 1—properties that serve well in CosineIDF. Another example is a method by Tang and Srihari which takes $\frac{P(d|\epsilon)}{P(m|\phi)}$, an approximation of the likelihood ratio that is valid when detecting pairs within normally-distributed data (as we saw in Chapter 3), and extends it to arbitrary distributions [125]. The approximation requires a distance measure between objects and a rarity measure. The appeal of the method is that it tries to incorporate the information found in a likelihood ratio without the need to develop a full generative model for pairs of objects in each new domain. We discuss this paper further at the end of Section 5.5.

5.3 Matching Text Documents

Information retrieval has a rich literature centered around ranking the best-matching documents in a corpus, given a short query. Documents and queries are often represented as bags of words, that is, vectors of counts of each word in the vocabulary. If we can overlook the distinction between binary vectors and count vectors, this research area has much of relevance to our task, including methods for computing similarity for items within a larger corpus. Researchers dealing with documents and words—including for text classification and other tasks beyond retrieval—have experience contending with the large n of numbers of documents, the large k of vocabulary size, and data sparsity in the resulting high-dimensional space.

5.3.1 Document Retrieval

In document retrieval, some methods to draw from include relevance models, intended to distinguish relevant from non-relevant documents given a query; *tf-idf* and other term weighting systems for computing retrieval scores as a function of shared words; and statistical language models, such as those based on query or document likelihood, cross entropy, or KL divergence, which compare the query and document based on models estimated from them [83]. Advances over the years have been in areas such as optimizing the formulas to use for term weights in the score; normalizing to handle documents of different lengths; and moving from Bernoulli vector models to multinomial models [69, 75, 83, 84]. Other directions of interest include modeling correlations among words and reducing the dimensionality of the vocabulary, for instance through latent semantic indexing or probabilistic topic modeling, to alleviate data sparsity and dependencies [16, 43, 88].

However, one characteristic limits the adaption of retrieval models to ASOUND: while we treat the items in a pair symmetrically, retrieval models treat the query and the document as fundamentally different types of objects. Kraaij discusses this asymmetry and some possible workarounds in the context of a need for a document-document similarity measure: since most models only need to produce the best ranking of documents for a given query, their scores are incomparable across different queries [69]. Aside from the issue of symmetry, the sheer diversity of probabilistic models that have been developed in information retrieval reflects the challenge of choosing an event space that represents all the properties generally expected to matter [84, 107].

The multiple Bernoulli models we use in Chapter 4 recall the classic Binary Independence Model (BIM) developed in the late 1970s by Robertson and Spärck Jones among others [83, 106]. Both models score pairs using likelihood ratios that compare matches to non-matches. Table 5.1 shows the relationship between the derivations. MixedPairs is a symmetric version that describes the joint likelihoods of X_1 and X_2 (under matching and non-matching conditions). BIM describes the conditional likelihoods of the document conditioned on the query (under matching and non-matching conditions) and then makes the assumption that only words present in the query matter. MixedPairs is in some ways more general than BIM, since it models both types of items, and in other ways more specific, as it assumes a particular generative model for matches. BIM gives non-zero weights only to $1|1$ and $Q_i = 1|D_i = 0$ terms, or in its rearranged form (valid only when holding the query constant), only to $1|1$ terms. It is worth noting that the BIM formula, with certain added assumptions, is commonly cited as a theoretical justification for IDF weights, $(\log \frac{1}{p_1}, \ldots, \log \frac{1}{p_n})$ [83, 107]. In that IDF form, $\sum_{i \in 1|1} \log \frac{1}{p_i}$, the score is what we called SharedWeight11. The method we call CosineIDF normalizes this quantity by the (weighted) lengths of the documents, which is a common way to use IDF weights (see the Vector Space Model of retrieval and Salton's SMART system) [69, 83].

5.3.2 Near-Duplicate Detection

Aside from document retrieval, the information retrieval community also has research on identifying distinctive textual content as it reappears across documents. The term *near-duplicate detection* encompasses tasks across a "similarity spectrum," from pruning a corpus of web documents of duplicates that differ only in minor presentation artifacts, to clustering the messages received in an email campaign around the form letters they originate with,

RSJ Model	**MixedPairs Model**
(Writing $c \in \{0,1\}$ in place of $R \in \{0,1\}$.)	(Writing (Q, D) in place of (X_1, X_2).)
$\frac{P(c=1 \mid D,Q)}{P(c=0 \mid D,Q)} =$	$\frac{P(c=1 \mid D,Q)}{P(c=0 \mid D,Q)} =$
$= \underbrace{\frac{P(c=1 \mid Q)}{P(c=0 \mid Q)}}_{\text{constant for a given } Q} \cdot \frac{P(D \mid Q, c=1)}{P(D \mid Q, c=0)}$	$= \underbrace{\frac{P(c=1)}{P(c=0)}}_{\substack{\text{constant across} \\ \text{all } D \text{ and } Q}} \cdot \frac{P(Q,D \mid c=1)}{P(Q,D \mid c=0)}$
	Expand $P(Q, D)$ using joint likelihoods for positive and negative pairs.
$\overset{\text{rank}}{=} \frac{P(D \mid Q, c=1)}{P(D \mid Q, c=0)}$	$\overset{\text{rank}}{=} \frac{P(Q \mid \phi)P(D \mid Q, c=1)}{P(Q \mid \phi)P(D \mid \phi)}$
Assume the set of non-relevant documents can be approximated by the entire corpus.	
$= \frac{P(D \mid Q, c=1)}{P(D \mid \phi)}$	$= \frac{P(D \mid Q, c=1)}{P(D \mid \phi)}$
$= \prod_{i=1}^{n} \frac{P(D_i \mid Q_i, c=1)}{P(D_i \mid \phi)}$	$= \prod_{i=1}^{n} \frac{P(D_i \mid Q_i, c=1)}{P(D_i \mid \phi)}$
	$= \prod_{i=1}^{n} \mathrm{LR}(b_i)$ (quantities from Table 4.2)
$= \prod_{i=1}^{n} \begin{cases} Q_i = 1, D_i = 1 : \frac{r_i}{p_i} \\ Q_i = 1, D_i = 0 : \frac{1-r_i}{1-p_i} \\ Q_i = 0 : 1 \\ \text{(Assumes words not in the query are equally probable in relevant and non-relevant documents)} \end{cases}$	$= \prod_{i=1}^{n} \begin{cases} Q_i = 1, D_i = 1 : & \frac{s+(1-s)p_i}{p_i} \\ Q_i = 1, D_i = 0 & \\ \quad \text{or} \quad : & 1-s \\ Q_i = 0, D_1 = 1 & \\ Q_i = 0, D_i = 0 : & \frac{1-p_i+p_i s}{1-p_i} \end{cases}$

Table 5.1: Comparison of MixedPairs with the Binary Independence Model (BIM). Derivation of their respective likelihood ratio scores. MixedPairs models both the "query" and "document," while BIM models the document conditioned on its relevance to the query. (Continues below.)

RSJ Model	
For a given Q, $\overset{\text{rank}}{=} \sum_{i \in 1\|1}^{n} \log \frac{r_i(1-p_i)}{p_i(1-r_i)}$ With some approximations, can derive IDF weights: $\ldots \quad \approx \sum_{i \in 1\|1}^{n} \log \frac{1}{p_i}$ This is exactly the SharedWeight11 method of Chapter 4.	

Table 5.1 continued.

detecting plagiarism, or identifying information reuse at the level of sentences that are "difficult to believe [. . .] were written independently of one another" (quoted phrases are from Metzler et al. [89]) [59, 89, 103, 141]. Recent papers have examined patterns of text reuse as a means for understanding the spread of ideas, for instance by tracking variations of direct quotes to reveal the dynamics of breaking news stories, detecting reprints of 19th century news stories to study the geography and speed by which they "went viral," or identifying copied sections of Congressional bills to expose the flow of ideas through the lawmaking process [72, 115].

Methods for near-duplicate detection generally address one or both of two concerns: efficiently indexing the corpus to reduce the number of pair comparisons needed, and developing measures of similarity, which depend on the size, amount of variation, and other characteristics anticipated in the copied portions. Useful overviews of these techniques can be found in Yang and Callan [141] and Hajishirzi et al. [59]. These ideas have been adapted for detecting near-duplicate images as well [32]. For text, documents are typically represented either as vectors of words or vectors of k-grams. Broadly, methods vary in the choice of k-grams to construct, the choice of vector components (whether words or k-grams) to ignore for efficiency purposes, whether to weight the vector components, and the choice of similarity method to use on the resulting vectors. One family of techniques uses hash functions to create "fingerprints" of a document, then computes similarity at the fingerprint level. These techniques include shingling [20], I-Match [30], and more generally, locality-sensitive hash functions, which are designed to efficiently identify the closest pairs of binary vectors within a large collection under a given distance measure [55, 74]. In other methods, Hajishirzi et al. [59] adapts a supervised method from Yih [144], which first learns term-weighting functions for the vector components, then computes either the cosine or extended Jaccard similarity of the weighted vectors. Metzler et al. [89] and Yang and Callan [141] use similarity and weighting measures from document retrieval.

I-Match is of particular interest with respect to our affiliation subset experiments (Section 4.6.3): to detect near-duplicate documents, it takes the unique set of terms left in a document after a filtering step; computes a hash value; and looks for collisions. In the paper that introduces the technique, Chowdhury et al. [30] experiment with various filtering criteria as a function of IDF term weights. They remove terms either in the low range of values, high, middle, or both edges of the distribution, and they find the best performance with a filter that retains the 10% of terms with the highest IDF values. Since IDF percentiles are identical to percentiles of what we call p_i, their preference for retaining the rarest words is at odds with our preference for retaining those with p_i nearest 0.5. The discrepancy could be because their true pairs are nearly identical, so a nuanced similarity measure is not needed; instead, almost all documents contain at least some of the rarest 10% of words, which provide the strongest signal of a match. Conversely, it is possible that for us, using the subset of affiliations with p_i nearest 0.05, 0.1 or some other non-zero value would perform better—especially for certain (non-symmetric about $p_i = 0.5$) similarity measures—than those with p_i nearest 0.5 or 0, the only options we explored.

5.3.3 Plagiarism Detection

Plagiarism detection draws on the same near-duplicate detection techniques, but with some twists. A plagiarized document contains material closely borrowed from other sources, but the borrowed material may consist entirely of short passages, in any order, possibly from multiple source documents. Recent work is reviewed by Meuschke and Gipp [90] as well as in reports from the PAN annual competition and workshop [103]. Prior to applying near-duplicate detection strategies, a suspicious document may first be subdivided into chunks of a size expected to match a single source closely. After candidate matches are found, further steps are required, such as aligning the document texts, to assess which passages were plagiarized and from which sources.

The adversarial component of plagiarism is challenging: while existing methods perform well with exact duplicates, they are not very robust to obfuscation strategies, which range from mild rewording and synonym replacement to substituting look-alike letters from different font families and embedding hidden text—the last examples being intended to combat automatic plagiarism detection systems. Some techniques rely on non-traditional features anticipated to be invariant when copied, such as the sequence of stopwords (after content words are removed), which tends to be preserved when a sentence structure is, or the sequence of citations within an academic text, which is likely to be preserved even if the text is translated to a different language. It is worth noting that for plagiarism detection, and in fact for most tasks involving matching text documents, the final call on match quality always comes down to human judgment.

Separate from plagiarism of text documents, another research area has emerged around detecting plagiarism of software, whether in academic or commercial settings [80, 134]. As a recent example, Wang et al. [130] developed a system to detect illegal clones of Android apps on marketplaces, a task which is challenging because the code may have been obfuscated. There is also a large body of work on code cloning, a more general term that encompasses the detection of benign copy-and-paste passages within large code bases for purposes of software analysis [108]. These techniques are diverse, compared to those for text duplicates, because of the logical structures that can be leveraged in computer code.

5.4 Matching Database Records

Another closely related research area is entity resolution, also known as record linkage or data de-duplication, in which a database contains multiple records pertaining to the same real-world object, and the task is to correctly match and consolidate them [31, 46, 63].

Database records generally contain text fields such as names and addresses, and methods for handling them cannot necessarily be adapted for continuous or binary data. For instance, we can safely ignore the numerous approximate string similarity functions, such as Levenshtein edit distance, Monge-Elkan, or Jaro-Winkler. All the same, the scoring methods warrant careful examination, and many additional issues treated are quite relevant.

5.4.1 Fellegi-Sunter Model

Much work in what is called "probabilistic record matching" stems from the Fellegi-Sunter model, first published in 1969 [50]. This model ranks pairs using a likelihood ratio, $\frac{P(\gamma|c=1)}{P(\gamma|c=0)}$, where c is our usual label for matches or non-matches, and γ is some function of the pair—a "comparison vector," or what we might call a feature vector. In typical usage, the components of the feature vector are assumed to be independent, so $\log \frac{P(\gamma|c=1)}{P(\gamma|c=0)} = \sum_i \log \frac{P(\gamma_i|c=1)}{P(\gamma_i|c=0)}$. So far, this resembles our models from Chapters 3 and 4. In the simplest model, each feature $\gamma_i = \gamma_i(X_{1i}, X_{2i})$ describes whether a pair matches in one (or more) fields; that is, it is a binary value. The model has also been extended to allow γ_i to be a normalized score from a string similarity function [140]. This likelihood ratio differs from those we use in that it measures similarity, including the probability of the similarity vector occurring by chance, but not the rarity of a pair's values; sharing a rare value yields the same $\gamma_i = 1$ as sharing a frequent value. More broadly, while the LR from our generative models is the ratio of joint likelihoods of the data itself, the Fellegi-Sunter LR is the ratio of likelihoods of some feature vector of the pair. Of course, no feature vector can contain more information than the original data, so when a joint likelihood function is known, it should be preferred.

To incorporate rarity, Fellegi and Sunter described an alternative version of γ_i. In this version, the function's range of values is no longer $\{0, 1\}$, but a set such as {non-match, matches on "Abadi", matches on "Arnold", matches on "Aronson", . . . , matches on "Zychowski", matches on something else}, where the enumerated names might be just a subset [50, 63]. With this representation, the LR is higher for rare matches. Outside of likelihood ratios, another way to account for rarity is by capturing it within the string similarity measure; for instance, borrowing from information retrieval, Cohen weights each word (or substring) by its *tf-idf* value, then takes the cosine similarity of the resulting vectors [35, 37]. Alternatively, rarity can be ensured at a post-processing stage: in one method, regardless of the distance measure, the final clusters are required to be both close together and in sparse regions [27].

Using the Fellegi-Sunter model requires estimates of $P(\gamma_i \mid c = 1)$ and $P(\gamma_i \mid c = 0)$, known in the literature as $m(\gamma_i)$ (for matched pairs) and $u(\gamma_i)$ (for unmatched pairs), respectively. In the absence of labeled examples, $P(\gamma_i \mid c = 0)$ for unmatched pairs can be approximated as two random draws from the data set, just as in our models. (Most often in this literature, all entities A from one data set are considered as candidate matches to all entities B from another. But these assumptions can be revised for different scenarios.) For matched pairs, $P(\gamma_i \mid c = 1)$, the probability of agreement in the ith component, is traditionally modeled as the probability of one random draw from the data set yielding some value j, times a fixed probability of the second entity's value matching the first [36, 50, 63]. Regardless of the particular models, the EM algorithm has proven useful for inferring their parameters, along with the cutoff threshold between positive and negative pairs, in tandem with the pairs' class labels [12, 63, 139].

Cohen et al. have explored connections between Fellegi-Sunter and information retrieval models, pointing out that if word frequencies are equal among matched and unmatched pairs, then $\log \frac{P(\gamma_i|c=1)}{P(\gamma_i|c=0)}$ reduces to exactly $\log \frac{1}{p_i}$, the component's IDF weight [36, 37]. As we noted in discussing the Binary Independence Model (Section 5.3.1), that IDF weight also appears in our methods SharedWeight11 and SharedWeight1100. Cohen et al. also observe that when a pair mismatches, the log odds term in Fellegi-Sunter is a negative constant that is independent of the mismatching word(s). While our model MixedPairs and the Hit-Miss model (below) produce negative constants for mismatches as well, Cohen et al. consider this property intuitively undesirable and adjust their score to remove it [37].

5.4.2 Broader Matching Process

Apart from pairwise scoring, the entity resolution literature has addressed other problems that need to be handled when detecting duplicates, in or out of databases. A number of entity resolution issues are important in typical databases but outside our scope. Prior to the automated matching, much effort is required to clean and standardize data fields, correct systematic errors, and otherwise preprocess records so as to make matches perceivable [63]. It can be non-trivial to subdivide free text fields, such as full names or addresses, into sub-components for matching, such as first name or street number—a task called segmentation [31]. Then, once two records are declared to match, a method is needed for merging their data into one canonical record. When databases hold information about multiple types of objects—whether explicitly, through separate tables, or implicitly, such as when variant spellings of cities reappear within string fields—information from consolidating each type of entity should, ideally, be able to help with the others [14, 31].

Certain issues are entirely applicable to problems such as ASOUND. One inevitable concern is how to avoid the quadratic increase in the number of pairs to score as the data set grows; we describe blocking and related techniques in Section 5.7 below. Another is how the training sets, probability estimates and constraints on valid link configurations must be modified when detecting duplicates within one database versus across two or more (including possible constraints about which databases can be assumed already free of duplicates) [31]. Yet another relevant issue is how to handle the resulting graph of linked records, including possibly inconsistent scores (that is, scores violating transitive closure) among related pairs. One approach is to apply a graph clustering method as a post-processing step [27, 31]. Conversely, one can treat the full configuration of true pairs as a latent variable to infer, thereby rendering the scores of adjacent pairs dependent. This is an approach we dismiss, due to efficiency concerns, in Section 3.2.3, but Steorts et al. have recently proposed a tractable approach worth considering [118].

5.4.3 Hit-Miss Model

Some of the closest models to Chapters 3 and 4 can be found in a 1990 paper by Copas and Hilton [38], along with follow-up publications in statistics and computer science [51, 99, 123, 124]. Copas and Hilton depart from Fellegi-Sunter by modeling the full joint likelihoods of matching and non-matching pairs. That is, they use a likelihood ratio equivalent to our Eq. (1.1) ($\frac{P(X_1,X_2|\text{positive})}{P(X_1|\phi)P(X_2|\phi)}$), rather than Fellegi-Sunter's $\frac{P(\gamma|c=1)}{P(\gamma|c=0)}$. They present a generative model for pairs in Gaussian-distributed data, and another, called the Hit-Miss model, for categorical data. They even analyze the predicted distributions of scores under these models much like we do in Figure 4.3, choosing to measure class separation using the symmetrized (or Jeffreys) divergence—that is, the distance between the two class means. For one-dimensional Gaussian data, the analysis shows that scoring with distance only (more precisely, with what we call $LR[d]$ in Chapter 3) yields a symmetrized divergence that approaches that of their LR as the distance between pairs approaches zero, and it is always at least half that of the full LR. For categorical data, they find that using their LR is better than using a binary agreement vector (γ), with equality when the values are uniformly distributed and a growing benefit as the values' "incidence rates" become more skewed.

The generative models we develop in Chapters 3 and 4 are almost, but not exactly, special cases of those presented by Copas and Hilton [38]. Their models have a latent true value for each entity and a noise process. Singletons are noisy observations of their true values, while pairs are independent noisy observations of the same true value. The Hit-Miss model specifies a frequency for each true value in a categorical distribution, a noise probability a, and a probability b of the observed field being blank. When an observation is noisy, its observed value is sampled from the original categorical distribution. Given a MixedPairs model with parameter s, we can produce a Hit-Miss model with identical likelihood ratios by setting $b = 0$, $a = 1 - \sqrt{s}$, and allowing only the values $\{0, 1\}$ in each component Semantically, $s = (1 - a)^2$ is the probability in MixedPairs that one entity copies the other, or the probability in Hit-Miss that both entities copy the original value. The difference is that our models lack the notion of an unobserved true value for pairs to copy. Copas and Hilton's models are elegant, yet so far they have been applied only to record linkage. Our work takes these ideas in new directions by applying them to different problems and broader definitions of linked entities; by narrowing in on the case of bipartite graph data with no blanks; and by exploring the parameter space to understand strengths and weaknesses of other methods.

Norén et al. apply the Hit-Miss model to detect duplicate reports of the same event within an adverse drug reaction database [99]. They offer two extensions to the model: for numeric fields, a model in which the observed value can either hit, miss (draw randomly), or deviate (a small amount) from the original value; and for databases with correlated fields, a method to adjust for pairwise correlations. One point of interest in this study is an efficiency trick like the one we suggest in Section 4.3.1: Norén et al. compute a cutoff threshold, then drop all records which, even if perfectly duplicated, would have scores below the threshold. These "unmatchable" records—containing mostly blanks and common field values—constitute about 30% of the data, with proportions varying widely by country.

Other work that builds on the models of Copas and Hilton [38] moves towards jointly inferring all matches between two databases. Fortini et al. extend the Gaussian model of pairs to a hierarchical Bayesian model whose likelihood function encompasses all singletons and pairs in two databases, the number of pairs, and their adjacency matrix, and they perform inference through MCMC [51]. Tancredi and Liseo similarly extend Hit-Miss to a Bayesian model for categorical data [123, 124]. In their work, the goal is not merely to infer the links between databases, but to use the posterior distribution to inform further analyses of the matched data. In a related piece (also mentioned earlier), Steorts et al. present another model for categorical data, one in which the unobserved linkage structure can flexibly represent match possibilities among one, two, or arbitrarily many databases.

5.5 Forensic Science

In forensic science, an important question is whether material found at a crime scene matches that taken from a suspect—that is, whether the two samples originate from the same source or different sources. Likelihood likelihood ratio methods, first introduced to the field by Lindley in 1977 [79, 126], are the modern approach to quantifying the "strength of evidence" towards either hypothesis. Since the probability estimates affect the outcomes of criminal trials, they need to be as accurate as possible. To this end, work in forensics centers around developing features and likelihood functions to fit particular materials. There are lines of research specializing in glass shards, paint flakes, MDMA tablets, handwriting, footwear, fingerprints, DNA samples, and voice recordings, among others [17, 92, 120, 125, 126, 145]. The problem setup can vary based on whether either item of the pair has been measured just once or several times, and whether the sample from the suspect is being compared to a previously-studied source (e.g., one or more types of glass with known properties) or to a sample whose background is unknown.

The seminal paper by Lindley [79] explores a scenario closely related to Chapter 3 (and also independently addressed by Copas and Hilton [38]): whether two real numbers are independent samples from a general, known, Gaussian distribution, or whether they are both sampled from some tighter Gaussian distribution with a different mean. Lindley assumes that rather than just two points, we are given two clusters of m and n points, representing measurements taken from a crime scene and a suspect, respectively. Working in the one-dimensional case only, he shows the LR to simplify to an expression similar, though not identical, to our Equation 3.9, dependent on what we would call $\frac{m^2}{\sigma^2}$ and $-\frac{d^2}{\nu^2}$. He also examines the behavior of the LR, calculating its value as a function of parameters, much like in our Figure 3.2, and calculating the expected probabilities of error as a function of hypothetical cutoff thresholds. Later work with continuous data has extended this model to the multivariate case [4] and to fit arbitrary background distributions using kernel density estimators, undirected graphical models, or Gaussian mixture models [5, 17, 92].

A recent, relevant paper by Bolck et al. addresses the benefits of what they term "feature-based" versus "score-based" likelihood ratio models [18]. Feature-based models are those in which, like ours, a pair's likelihood function is its joint density function. Score-based models are closer to Fellegi-Sunter models: a univariate distance measure $\gamma(X, Y)$ is computed between the items in a pair, and the resulting value is compared to distributions from matching and non-matching pairs, using the likelihood ratio $\frac{P(\gamma|c=1)}{P(\gamma|c=0)}$. These are like Fellegi-Sunter models except that these reduce a multivariate input to a single continuous distance value, rather than one distance value per component. Score-based models require a training set of positive and negative pairs, unlike feature-based models, but after the model is learned, they are simpler to use, only needing a single distance computation to produce the LR for a new pair.

Bolck et al. find that compared to feature-based models, score-based models produce LRs of smaller magnitude, and this is because they use less information: they reduce the multivariate signal to a single variable and ignore information about rarity. Yet they also discuss some advantages to score-based models: they are less sensitive to variations in data, do not require complex models, and are more robust than feature-based models to retaining their scores when the number of available features is decreased. This paper is distinctive in that it also examines the effect of changing the background distribution, called "Z." The authors test the scores' robustness—via the magnitude of the LRs—when using fewer samples or fewer features. To select the least informative features to remove, they use principal component analysis to identify features with the lowest variance in Z. In Section 4.6.3, our justification is different, but the features we remove, those with the most extreme p_i values, likewise are those with the lowest variance.

Tang and Srihari offer a different perspective on the "feature-based" versus "score-based" dichotomy, extending it with two more options [125]. First, they point out that score-based models can either reduce the input pair to a scalar distance value, as above, or to one value per component—a "vector distance" that retains more information, like Fellegi-Sunter models. (In Chapter 3, our $LR[d]$ is a vector distance model.) Second, they focus on the decomposition of the likelihood ratio, in Lindley's model for Gaussians, into one term for the distance between the items $(-\frac{d^2}{\nu^2})$ and another for the rarity of the pair $(\frac{m^2}{\sigma^2})$. Extrapolating from that model, Tang and Srihari propose constructing likelihood ratios for any type of items as the product of a distance term and a rarity term. Although these LR approximations are justified mainly by analogy and leave many choices up to the modeler, the experiments show reasonably good performance over a broad range of data types, including binary vectors, categorical-valued vectors, and attributed graphs. In addition, in work that complements and supports our Section 3.4, Tang and Srihari offer some theoretical and synthetic comparisons of the full joint LR for Gaussians to its scalar distance-based alternative. Measuring performance using accuracy and KL divergence, in place of our AUC, they find that the scalar distance-based method is close to optimal only when true pairs are very close together and the dimensionality is low.

5.6 Fraud and Security

In some scenarios, there is a fraud or security interest in detecting suspiciously similar people or objects. Chapter 2 described how unusual patterns of sharing multiple jobs over time may be a sign of collusion among stockbrokers. Likewise, in online auctions, unusual patterns of participating in (or avoiding) the same auctions may be a sign of collusion among shill bidders. Trevathan and Read describe several strategies used by shill bidders to drive up prices and present algorithms that detect them in simulated data [128]. More widespread than these are the problems caused by individuals controlling large groups of fake or compromised accounts or machines on the internet, whether in the context of online social networks, advertising click fraud, or botnets, as we describe below. In each case, the groups achieve their goals through coordinated, often synchronized behavior, which it is in the interests of the network operators to detect.

Online social networks are often infiltrated by Sybil (spam) accounts. One detection strategy addresses accounts that simulate human behavior by copying it from legitimate accounts. For example, Bilge et al. outline a profile cloning attack in which a spammer duplicates a profile and uses the stolen identity to acquire network ties with the real person's friends [15]. Similarly, on Weibo, a popular Chinese microblogging service, Zhang et al. identify Sybil users whose tweets are mostly copied from others (excluding explicit retweets) [147]. In this case, similarity between tweets is defined as the Jaccard similarity between their sets of words, and locality-sensitive hashing is used to make the all-pairs computation feasible at scale.

Another detection strategy is to look for groups displaying coordinated behavior. Yang et al. examine a labeled set of Sybil accounts from Renren and find that the vast majority are connected to other Sybils both in content (measured via Jaccard similarity among the links they post) and in timing, even if they are rarely connected in the social graph [143]. Beutel et al.'s CopyCatch detects spam accounts on Facebook by looking for bipartite cores in which at least n users "Liked" m Pages within a given time window [13]. These cores of interest can also be defined as dense blocks within the adjacency matrix of users and actions.

Jiang et al. extend the idea of looking for dense blocks to arbitrary matrices and tensors [65]. They formulate a null model of matrix density, against which they can assign a suspiciousness score to any block as a function of its dimensions and density. This approach is of the same form as Chapter 2's prob model, which measures the value of a test statistic against a null model describing the rest of the data. In follow-on work, the same authors introduce a scoring function that relies on concepts similar to Chapter 3's "similarity" and "rarity." Under this method, called CatchSync, an account in the unipartite social network of Twitter or Weibo is scrutinized based on the accounts it follows. It is considered a spammer if these accounts are both similar to each other (high "synchronicity") and unlike the majority (low "normality") [66]. Here, similarity between accounts is defined as a function of their graph-based feature vectors. Note that although these measures capture useful intuitions, they are not derived from an explicit probability model.

A different extension of CopyCatch, called SynchroTrap, loosens the assumption that spam users must create a dense block of actions [23]. Instead of requiring that users do m identical actions within a given time window, it measures the Jaccard similarity of pairs of users, where identical actions are considered to match if they occur within a time window of each other. Then, like in Chapter 2, it chooses a cutoff threshold and returns the connected components of that graph as groups. Cao et al. describe the deployment of this system at Facebook and Instagram, focusing on techniques for efficient implementation [23].

In online advertising, fraudulent or accidental traffic to webpages, just like intentional traffic, causes ads to be served, possibly clicked on, and charged for. One source of such traffic is coordinated bots. Metwally et al. describe a problem in which fraudsters wish to generate clicks on specific webpages. Moreover, since it is straightforward to detect and block malicious traffic coming from a single machine, they address a scenario in which coalitions of fraudsters obscure their activities by sharing and distributing them across many machines [87]. In this case, traffic analysis can reveal these groups based on the similarity of the sets of sites at which they generate traffic. The proposed algorithm uses locality-sensitive hashing and sampling techniques to efficiently search for the closest pairs of machines according to their Jaccard similarity. Stitelman et al. address advertising fraud from a different angle. As they explain, "Some websites or clusters of websites are apparently created for the sole purpose of perpetrating [fraud] by using non-genuine traffic to extract payments by advertisers for advertisements that no user ever sees" [119]. They describe methods to detect not the traffic creators, but websites themselves that have high rates of unintentional traffic. In this case, the incriminating evidence is pairs (or sets) of websites visited by anomalously similar sets of browsers (as identified by cookies).

Compared to the above cases, botnets, or large groups of computers infected with malware and controlled remotely, are the greatest threat to security. Several methods to detect them focus on the groups' coordinated behavior as detected through network traffic monitoring. BotMiner [57] and BotGAD [29] use unsupervised clustering to extract groups of hosts or domains after constructing complex feature vectors of their traffic patterns. BotGAD is of particular interest in that it employs a statistical technique called a sequential likelihood ratio test as a similarity measure; this is a likelihood ratio test that outputs a 0/1 decision only once enough evidence has accumulated over time.

5.7 Computational Efficiency

Both the information retrieval and database communities have done extensive work on indexing large data sets to make them efficiently searchable. For all-pairs problems like near-duplicate detection, the key idea in computational efficiency is to reduce the number of pairs under consideration, to combat the $O(n^2)$ blow-up. Reducing the cost of each individual score is helpful but secondary. Almost every method uses a cheap approximate similarity measure as a first pass, builds an indexing structure to make this "cheap" computation feasible at large scale, and applies a more accurate measure only to the candidate pairs output by this process. Since our work focuses on that more accurate similarity measure, we can leverage existing techniques for scaling up, provided we want the same general set of candidates.

In entity resolution, the first pass measure hinges on choosing database fields that we expect to match in true pairs [31, 46]. At this step, high recall is important. In the simplest case, if we expect just one field to match perfectly—say, zip code—we use this field as a "blocking key:" for each zip code, output as candidates all pairs of records sharing that value. Blocking approaches are useful as long as the derived blocks are not too small to contain the true matches, and not too big, since within every block, we do enumerate all pairs. For certain kinds of fields, such as name or height, we can instead use the field as a "sorting key:" sort all records based on the field, then iterate through the sorted list, outputting as candidates all pairs of records within a window of a fixed size or whose field values are within a fixed distance.

If no single field is certain to match, we can repeat these processes with multiple key fields to generate larger lists of candidates. If the existing fields are not expected to match closely enough, then indexing functions come into play. An indexing function maps a field to one or more values which can then be used as blocking keys. Examples include binning, for continuous values, and for strings, phonetic encoding, suffixes of varying lengths, character n-grams or individual words. The indexed values can be used as discrete blocks, or alternatively, they can be used as inputs to canopy clustering, which assigns records into overlapping clusters. Canopy methods cheaply compute a similarity function between records and output near neighbors under that function as the candidates for further scoring [85].

Much of the recent work on near-duplicate detection of text documents, as well as of bot-controlled accounts or machines, uses hashing and fingerprinting techniques, particularly locality-sensitive hashing [55, 59, 74]. Fingerprinting is somewhat similar to indexing: both extract values, such as substrings, from documents or records, to make them easier to compare. However, in databases, the aim generally seems to be to expand the set of possible matches for a short record, while in documents, the focus is more on choosing which substrings are worth keeping. With fingerprinting, the resulting values are put through a hash function, and the document is represented as a binary vector of values.

Locality-sensitive hash (LSH) functions are a family of randomized algorithms for solving the nearest neighbors problem: given a set of points, build an indexing structure so that, given an arbitrary query point, we can quickly retrieve all neighbors within distance R [112, Ch. 1]. They are particularly suited for high dimensional spaces. LSH functions work by mapping the input data to a lower-dimensional space of “signatures” in which, for any two points, we can approximate the distance between them by the fraction of their components that agree. This representation can be useful in itself. Then, to retrieve all pairs within distance R of each other, we compute signatures of a given length—multiple times—and return points whose signatures exactly match at least once. These methods have now been developed for a number of distance measures, including Hamming, Euclidean, Jaccard, and cosine.

The only large-scale data set in the dissertation is in Chapter 2. There, to keep the number of pairs tractable, we used each branch office in turn as a blocking key to enumerate candidate pairs of reps, also requiring an overlap in the time intervals the reps worked there. While aggregating that list of pairs, we further filtered it by retaining only pairs that appeared on it three or more times. Going forward, that technique might continue to suffice, with the right infrastructure. For entities with continuous values, we could use binning, as mentioned in Section 3.4.

Experimenting with LSH-based methods is appealing, especially for dynamic and static affiliation data. The ideal, of course, would be to have an LSH method for the exact score we wish to use. Since the LSH method for cosine accepts real-valued vectors, it can be used to approximate CosineIDF directly. An alternative would be to run LSH with any distance measure under a fairly tolerant threshold, and use this as a first-pass filter. With this or any other first-pass method, it is important to think through the method's possible weaknesses. Whether the method is appropriate for a given data set—that is, whether it achieves high recall for the more accurate scoring measure while producing a small enough set of pairs—will depend on specific properties such as the number of items, the number of dimensions, the distribution of data within each dimension (e.g., p_i), and also the evaluation measure of interest. For now, these concerns need to be assessed on a case by case basis.

CHAPTER 6

CONCLUSION

In this thesis, we started with a question that seems eminently answerable using statistical methods, but lacked an obvious solution: "Given two entities within a larger collection, is their similarity best explained as the result of a link between them?" Of course there are ways to measure similarity between entities; and alternatively, if we had access to labeled training data, we could learn a classifier to distinguish pairs from non-pairs. For certain real-world problems, questions like this have been explored extensively: how to judge whether two documents contain text from the same source, two database records refer to the same entity, or two fragments of glass come from the same manufacturer. But for two people and their careers, two people and their sets of affiliations, or merely two points within a distribution, there was no established approach for determining whether they are more similar than we should expect by chance.

This gap was not a result of too little prior work on the subject, but rather, an excess of options, all addressing slightly different questions or adapted to different domains. As we discussed in Chapter 5, the fields of information retrieval and entity resolution have sophisticated methods for estimating probabilities of feature matches in text fields or documents, given unlabeled data, and they have complex indexing and hashing methods for efficiently detecting duplicates within large corpora. Forensic science builds detailed generative models of specific types of objects, such as fingerprints or samples of glass, in order to minimize the probability of error when a particular case comes up for expert judgement. Social networks research infers friendship ties by building models of human mobility patterns and interaction dynamics. Yet when it comes to determining the similarity of two nodes in a graph—a question that arises frequently—probabilistic models are lacking, and people often fall back upon rule of thumb measures, such as "number of neighbors shared" or the Adamic/Adar measure. In this dissertation, we have built new models consistent with existing literature, for domains that are general enough to be widely applicable, with a goal of better understanding the problem space (of ASOUND) that all these fields share.

After demonstrating that detecting pairs can be useful in a real-world social network inference task, and that there is room for improvement over a naive measure of similarity, we formalized the problem of detecting ASOUND, first in a setting of Gaussian-distributed points, then in a bipartite graph. We took a theoretical perspective, investigating, first, how to derive a method for scoring pairs, second, how the ability to detect pairs within a data set varies as a function of the data's distribution, and third, how the score we derive compares to baseline measures, both in form and in performance as a function of the data's distribution. Our general approach, of a symmetric score for every pair in the form of a likelihood ratio, is familiar and reasonable for problems like this, as is the assumption that negative pairs can be modeled as two independent samples from the data. Our choice to use generative models, the specific forms of these models, and the analyses that the models enabled, were unique. More concretely, we have introduced new similarity measures in each domain. For the bipartite graph domain in particular, since many problems can be rewritten in this form, we hope to see these new measures be further tested and adopted.

6.1 Review of Contributions

In Chapter 2, we began with a fraud detection problem, a large database of employee job histories, and a conjecture that groups of people who "move together" through their careers, in tribes, might be higher fraud risks than the general population. We framed this as a task of detecting "tribes" moving together, defined as connected components formed from pairs of reps whose job histories were unusually similar. The scoring measure for pairs was defined as the rarity of their shared job sequence—much like the measure we later called SharedWeight11. Since reps move among jobs over time, we used a Markov process to model independent reps. Though we lacked ground truth labels, evaluations showed that the inferred tribes were smaller, had higher risk scores, were more homogenous in risk scores, shared rarer sequences of jobs, and were more geographically mobile than comparison groups.

Chapter 3 examined ASOUND among points in Gaussian-distributed data. We presented a generative model (for any domain) of data consisting of true pairs plus singletons and showed how we could use a pair-wise likelihood ratio to infer the true pairs, provided we are willing to treat all candidate pairs as independent. For Gaussian data, we created a model of true pairs, derived its associated likelihood ratio (LR) for scoring, and showed how the LR combines d, the distance between the pair, m, the rarity of the pair's midpoint, and t, parametrizing the distance between true pairs. Examining the theoretical distributions of synthetic positive and negative pairs, we saw that the distributions become more distinct when true pairs are closer together and as the dimensionality increases. In both real and synthetic data, we showed that LR, and an approximation $\frac{P(d|\epsilon)}{P(m|\phi)}$ that is more robust when t is unknown, can improve over distance-only and rarity-only baselines whenever both these features carry information.

In Chapter 4, we used a model of bipartite graphs (or, equivalently, binary vectors) in which most items are independent. For each item, every affiliation is chosen independently according to some affiliation frequency p_i. True pairs choose a fraction s of their affiliations jointly, and the remaining $(1-s)$ of affiliations independently. In the likelihood ratio derived from this model, the score for a pair of items can be written as a sum over all affiliations. The score per affiliation is a constant negative value for every mismatch (1/0), and a positive value for every match (1|1 or 0|0), with that value depending on the affiliation's frequency (vs. rarity) in the population. We showed that in synthetic data, positive and negative pairs become more distinguishable as s, the similarity of true pairs, increases, as the number of affiliations grows, and as affiliation frequencies approach 0.5. The latter two properties were testable in real data and found to hold, except that removing the least informative affiliations sometimes actually increases performance.

Besides the MixedPairs scoring method described above, we introduced a weighted version of Pearson correlation, and two SharedWeight methods that use the rarity of the pairs' shared matches (like in Chapter 2). Comparing these methods to eight others, we rewrote them all in the form of sums across affiliations. We found that analyzing these component-wise functions helped explain the methods' empirical performance in synthetic and real data sets. Namely, the most robust scoring methods have per-component functions that vary with p_i; that treat $1|1$, $0|0$, and $1/0$ as distinct configurations of a pair; and that treat the presence of a rare affiliation as equivalent to the absence of a high-frequency one. Lack of any one of these properties, particularly in cases where a data set's p_i vector suggests the property would be important, seems to impair performance.

6.2 Future Directions

The models we developed have answered some questions and, of course, opened up new ones. Below, we discuss avenues for future work in this area.

In affiliation data, the new methods seem promising. Some next steps will be to validate the properties we have described in additional and larger data sets and look for situations in which a change of methods could offer strong improvement. It may also be possible to adapt these methods for link prediction in unipartite graphs. We showed that the method Weighted Correlation, which we derived by analogy to Pearson correlation, is related to MixedPairs, for instance in that it computes a "correlation coefficient" of s for positive pairs generated with that parameter. Whether there is a more formal relationship among these three is worth exploring.

In terms of improving the fit of the MixedPairs model to real data sets, we see potential in two directions. The first is adjusting the form of ϕ so that it accounts for the varying node degrees of not only affiliations, but also items. The second is moving ϕ away from a multiple Bernoulli to a lower-dimensional representation, possibly using topic models, which could allow for dependencies among affiliations and even for clusters among singletons. Finally, an obvious possibility for extending this work is to shift from domains of binary vectors to count vectors. Information retrieval already has numerous methods for count vectors, however, so a pertinent question might be whether a pair-detection model would simply reduce to a pre-existing method.

The domain of dynamic affiliations, which started off this project, remains as complex as ever. In order to extend the full likelihood ratio approach to this (or any new) domain, we would need to develop a generative model for positive pairs in the domain, which is not necessarily straightforward. This challenge brings up the question of whether a full generative model is essential for good performance at the task, beyond its utility for understanding the parameter space. Another potential path could be to use manually constructed feature vectors of a pair, in combination with an EM method, to learn a distinction between positive and negative pairs. An interesting variation of the task, for complex domains especially, would be to build generative models describing alternative behavior patterns for pairs. Such pairs might avoid each other, have one follow the other after a time lag, or closely share certain types of behaviors but never others.

We have left open the causal origin of the correlation between true pairs, describing the ties as representing a "shared origin" or "channel of influence," and implying that they exist prior to the generation of the observed data. This is a useful simplifying assumption, since the nature of the ties will vary across domains. It is difficult to distinguish among causal explanations for shared behavior; in social networks, this is sometimes referred to as the question of "homophily versus social influence" [113]. However, for social network data with a temporal component, it might be useful for a generative process to distinguish between pre-existing ties and others formed as a result of shared behavior—for instance, by people attending a small event together. In the job history data, for example, if two people work at a company of only five employees, our existing models count that as strong evidence towards connections among them (which might as well be present in advance), but do not take into account that they almost certainly know each other going forward. Whether the new ties have any effect on future behavior, or whether they are of intrinsic interest to the project, will vary. But in general, domains in which we expect people to form ties as a result of their observed shared behavior might benefit from explicitly modeling such tie formation.

Regarding the modeling approach of Chapters 3 and 4, the most useful algorithmic extensions would be, first, to make the parameters of the positive models learnable, through an EM algorithm or small amounts of training data, and second, to work out indexing and comparison strategies to keep computational costs low. Meanwhile, some interesting theoretical questions with practical implications remain open. For instance, in affiliation data, we showed that entropy, or a measure related to it, plays a role in the difficulty of pair detection. Intuitively, this is because the score of a pair is a function of its rarity, and the entropy of the data set summarizes how often pairs can look rare. It would be interesting to look for this property in continuous data, and to investigate what the entropy of a data set implies for the detectability of pairs (e.g., it might induce an upper bound). In operational terms, thinking about entropy led us to experiment with the subsets of high or low frequency affiliations, manipulations which could be useful towards goals of efficiency (while preserving pair identifiability) or anonymity of a data set (while still releasing a certain number of attributes). A related operation to explore is removing certain entities from the data set: either those in dense regions that could never reach a high probability match, so could safely be pruned, or those deemed too re-identifiable, in sparse regions. Perhaps these operations could be combined iteratively to optimize a given data set for either efficiency or anonymity.

If we reflect back on the opening examples, we find that this work has addressed them mostly, but not fully. One remaining source of incompleteness stems from the task definition. The initial examples asked about the probability assigned to a particular pair within the context of a data set, while in our research, we have scored all pairs in a data set, independently, and evaluated performance based on the ranking. Changing this setup would force us to focus on different aspects of the problem. For instance, evaluating based on the accuracy of the probability estimates would emphasize the role of the class distribution, whereas evaluating after fully scoring only a limited budget of pairs would focus our attention on characterizing the most likely-looking candidate entities and pairs. The fact that we score all pairs independently has a large impact. For instance, if we doubled the size of an existing data set by sampling additional singletons (without changing the class distribution), the likelihood ratio scores of existing pairs would be unchanged. However, if we search a database for the match to a single query entity such as a fingerprint, finding a second equally good candidate match would require us to roughly halve the probabilities assigned to either pair—unless we expect the database itself to contain duplicates. These types of constraints have been addressed in the entity resolution literature—under names such as one-to-one matching, one-to-many, or many-to-many—but working out how they affect the parameter spaces we have explored remains an open problem.

APPENDIX A

LEMMAS AND DERIVATIONS FOR CONTINUOUS DATA

Lemmas

Lemma A.0.1. *Suppose a probability distribution* $P(\mathbf{x}_i, \mathbf{x}_y) = f(\mathbf{x}_i, \mathbf{x}_j)$ *is defined in terms of* $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ik}) \in \mathbb{R}^k$ *and* $\mathbf{x}_j = (x_{j1}, x_{j2}, \ldots, x_{jk}) \in \mathbb{R}^k$*. Let* $\mathbf{m} = \frac{\mathbf{x}_i + \mathbf{x}_j}{2}$ *be the midpoint of* $\mathbf{x}_i$ *and* $\mathbf{x}_j$*, and let* $\mathbf{d} = \frac{\mathbf{x}_i - \mathbf{x}_j}{2}$ *be the points' (symmetric) displacement from the midpoint. Then* $P(\mathbf{m}, \mathbf{d}) = 2^k f(\mathbf{x}_i, \mathbf{x}_j)$*.*

Likewise, the factor is $\frac{1}{2^k}$ for the inverse transformation: $P(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2^k} P(\mathbf{m}, \mathbf{d})$.

Proof. The transformation is one-to-one. For each vector component l, the transforms and their inverses are:

$$\begin{aligned} m_l &= \frac{x_{il} + x_{jl}}{2} & \qquad x_{il} &= m_l + d_l \\ d_l &= \frac{x_{il} - x_{jl}}{2} & \qquad x_{jl} &= m_l - d_l \end{aligned}$$

We can show by induction that in k dimensions, the determinant of the Jacobian, $|A_k|$, is -2^k. Then, its absolute value 2^k is the factor used in the transformation.

For a single dimension, $|A_1|$ is

$$\begin{vmatrix} \frac{\partial x_{il}}{\partial m_l} = 1 & \frac{\partial x_{il}}{\partial d_l} = 1 \\ \frac{\partial x_{jl}}{\partial m_l} = 1 & \frac{\partial x_{jl}}{\partial d_l} = -1 \end{vmatrix} = -2 = -2^1.$$

In general,

$$A_k = \begin{bmatrix} \frac{\partial x_{i1}}{\partial m_1} = 1 & \frac{\partial x_{i1}}{\partial d_1} = 1 & \frac{\partial x_{i1}}{\partial m_2} = 0 & \frac{\partial x_{i1}}{\partial d_2} = 0 & \cdots & \frac{\partial x_{i1}}{\partial m_k} = 0 & \frac{\partial x_{i1}}{\partial d_k} = 0 \\ \frac{\partial x_{j1}}{\partial m_1} = 1 & \frac{\partial x_{j1}}{\partial d_1} = -1 & \frac{\partial x_{j1}}{\partial m_2} = 0 & \frac{\partial x_{j1}}{\partial d_2} = 0 & \cdots & \frac{\partial x_{j1}}{\partial m_k} = 0 & \frac{\partial x_{j1}}{\partial d_k} = 0 \\ \frac{\partial x_{i2}}{\partial m_1} = 0 & \frac{\partial x_{i2}}{\partial d_1} = 0 & \frac{\partial x_{i2}}{\partial m_2} = 1 & \frac{\partial x_{i2}}{\partial d_2} = 1 & \cdots & \frac{\partial x_{i2}}{\partial m_k} = 0 & \frac{\partial x_{i2}}{\partial d_k} = 0 \\ \frac{\partial x_{j2}}{\partial m_1} = 0 & \frac{\partial x_{j2}}{\partial d_1} = 0 & \frac{\partial x_{j2}}{\partial m_2} = 1 & \frac{\partial x_{j2}}{\partial d_2} = -1 & \cdots & \frac{\partial x_{j2}}{\partial m_k} = 0 & \frac{\partial x_{j2}}{\partial d_k} = 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial x_{ik}}{\partial m_1} = 0 & \frac{\partial x_{ik}}{\partial d_1} = 0 & \frac{\partial x_{ik}}{\partial m_2} = 0 & \frac{\partial x_{ik}}{\partial d_2} = 0 & \cdots & \frac{\partial x_{ik}}{\partial m_k} = 1 & \frac{\partial x_{ik}}{\partial d_k} = 1 \\ \frac{\partial x_{jk}}{\partial m_1} = 0 & \frac{\partial x_{jk}}{\partial d_1} = 0 & \frac{\partial x_{jk}}{\partial m_2} = 0 & \frac{\partial x_{jk}}{\partial d_2} = 0 & \cdots & \frac{\partial x_{jk}}{\partial m_k} = 1 & \frac{\partial x_{j1}}{\partial d_k} = -1 \end{bmatrix}$$

$$\left|A_k\right| = \begin{vmatrix} 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & -1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & & & & & \\ 0 & 0 & & & & & \\ \vdots & \vdots & & & \Big[A_{k-1}\Big] & & \\ 0 & 0 & & & & & \\ 0 & 0 & & & & & \end{vmatrix} = (-1)\left|A_{k-1}\right| - (1)\left|A_{k-1}\right| = -2(-2^{k-1}) = -2^k$$

□

Lemma A.0.2. *Given points $\mathbf{x}_i$ and $\mathbf{x}_j \in \mathbb{R}^k$, with midpoint $\mathbf{m}_{ij}$ and displacement vector $\mathbf{d}_{ij}$ from the midpoint. Define $m = \|\mathbf{m}_{ij} - \boldsymbol{\mu}\|$ and $d = \|\mathbf{d}_{ij}\|$. Then $\|\mathbf{x}_i - \boldsymbol{\mu}\|^2 + \|\mathbf{x}_j - \boldsymbol{\mu}\|^2 = 2(m^2 + d^2)$.*

It then follows (see main text, Eqs. (3.7) and (3.8)) that if $\phi \sim$ Normal$(\boldsymbol{\mu}, \sigma^2 I)$, then $P(\mathbf{x}_i \mid \phi)P(\mathbf{x}_j \mid \phi)$ depends on $\mathbf{x}_i$ and $\mathbf{x}_j$ only through m and d.

Proof. The vectors $(\boldsymbol{\mu}, \mathbf{x}_i)$ and $(\boldsymbol{\mu}, \mathbf{x}_j)$ define a plane, which contains $\mathbf{m}_{ij}$. In that plane, Figure A.1 is as shown. The law of cosines tells us that $\|\mathbf{x}_i - \boldsymbol{\mu}\|^2 = m^2 + d^2 - 2md\cos f$, and that $\|\mathbf{x}_j - \boldsymbol{\mu}\|^2 = m^2 + d^2 - 2md\cos(\pi - f)$. Since $\cos(\pi - f) = -\cos f$, we can rewrite $\|\mathbf{x}_j - \boldsymbol{\mu}\|^2 = m^2 + d^2 + 2md\cos f$. This yields $\|\mathbf{x}_i - \boldsymbol{\mu}\|^2 + \|\mathbf{x}_j - \boldsymbol{\mu}\|^2 = 2(m^2 + d^2)$.

□

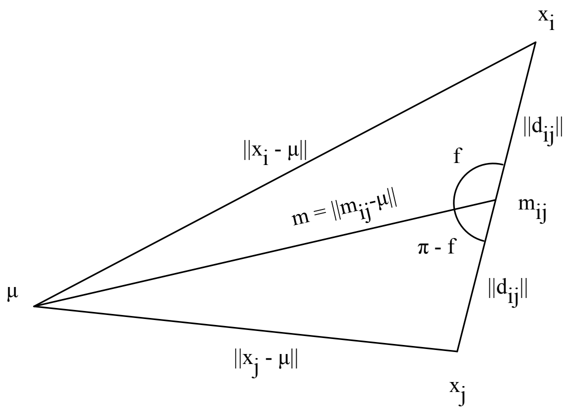


Figure A.1: Triangle formed by $\mathbf{x}_i$, $\mathbf{x}_j$, and $\boldsymbol{\mu}$.

Lemma A.0.3. *Given $\mathbf{z} \in \mathbb{R}^k$ with magnitude $z = \|\mathbf{z}\|$, and with a probability density that depends only on that magnitude: $f_{\mathbf{z}}(\mathbf{z}) = g(z)$. Then, changing variables to write the density as a function of z gives $f_z(z) = g(z)\frac{2z^{k-1}\pi^{\frac{k}{2}}}{\Gamma(\frac{k}{2})}$.*

Explanation. To transform, we would write $\mathbf{z}$ in spherical coordinates ($\mathbf{z} = (z, \alpha_1, \alpha_2, \ldots, \alpha_{k-1})$, with $\alpha_1, \ldots, \alpha_{k-2} \in [0, \pi]$ and $\alpha_{k-1} \in [0, 2\pi)$), then integrate out the angles. The factor introduced by this process is $S_k = \frac{2z^{k-1}\pi^{\frac{k}{2}}}{\Gamma(\frac{k}{2})}$, the surface area of a hypersphere of radius z in $\mathbb{R}^k$ [135, 138].

Lemma A.0.4. *If $f_y(y) = \frac{2^{1-\frac{k}{2}}}{\Gamma(\frac{k}{2})}\frac{y^{k-1}}{a^k}e^{\frac{-y^2}{2a^2}}$, defined on $y \geq 0$ with $a \in \mathbb{R}$ and $k \in \mathbb{Z}, k > 0$, then $z = \frac{y}{a} \sim \chi_k$.*

Proof. Perform the change of variables: $z = \frac{y}{a}$, so $y = az$ and $\frac{dy}{dz} = a$.

$$\begin{aligned} f_y(y) &= \frac{2^{1-\frac{k}{2}}}{\Gamma(\frac{k}{2})}\frac{y^{k-1}}{a^k}e^{\frac{-y^2}{2a^2}} \\ f_z(z) &= \frac{2^{1-\frac{k}{2}}}{\Gamma(\frac{k}{2})}z^{k-1}\left(\frac{1}{a}\right)e^{\frac{-z^2}{2}}|\frac{dy}{dz}| \\ &= \frac{2^{1-\frac{k}{2}}}{\Gamma(\frac{k}{2})}z^{k-1}e^{\frac{-z^2}{2}} \end{aligned}$$

Equation (3.6) is exactly the density function for the distribution χ_k. We will also use the notation $\chi_k(z)$ to represent that density function, so we would write above that $f_z(z) = \chi_k(z)$ and also that $f_y(y) = \chi_k(\frac{y}{a})(\frac{1}{a})$. □

Distributions for Positive and Negative Pairs

Beginning with the expression from (3.6) for the density function for positive pairs,

$$P(\mathbf{m}_{ij} \mid \phi)P(\mathbf{d}_{ij} \mid \epsilon) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^k e^{-\frac{m^2}{2\sigma^2}} \left(\frac{1}{\sqrt{2\pi}\nu}\right)^k e^{-\frac{d_{ij}^2}{2\nu^2}}, \tag{3.6}$$

we apply Lemma A.0.3 to the first and second parts, respectively, to get

$$\begin{aligned} P(m \mid \phi)P(d \mid \epsilon) &= \left(\frac{2m^{k-1}\pi^{\frac{k}{2}}}{\Gamma(\frac{k}{2})}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^k e^{-\frac{m^2}{2\sigma^2}} \left(\frac{2d^{k-1}\pi^{\frac{k}{2}}}{\Gamma(\frac{k}{2})}\right)\left(\frac{1}{\sqrt{2\pi}\nu}\right)^k e^{-\frac{d_{ij}^2}{2\nu^2}} \\ &= \left(\frac{2^{1-\frac{k}{2}}}{\Gamma(\frac{k}{2})}\right)\frac{m^{k-1}}{\sigma^k}e^{-\frac{m^2}{2\sigma^2}} \left(\frac{2^{1-\frac{k}{2}}}{\Gamma(\frac{k}{2})}\right)\frac{d^{k-1}}{\nu^k}e^{-\frac{d^2}{2\nu^2}}. \end{aligned}$$

With the use of Lemma A.0.4, we can recognize this as the product of two χ_k distributions: $\frac{m}{\sigma} = m' \sim \chi_k$, and $\frac{d}{\nu} = \frac{d'}{t} \sim \chi_k$. Rewriting in terms of m', d' and t, and using the notation $\chi_k(z)$ to represent the density function for χ_k:

$$P(m' \mid \phi)P(d' \mid \epsilon) = \chi_k(m')\chi_k\left(\frac{d'}{t}\right)\left(\frac{1}{t}\right).$$

This is intuitively reasonable because χ_k is known to describe the distance from the origin to points that are distributed as k-dimensional normals, which is exactly where $\mathbf{m}_{ij}$ and $\mathbf{d}_{ij}$ came from.

For the negative density, we take the expression from Equation (3.8) and transform from coordinates $(\mathbf{x}_i, \mathbf{x}_j)$ to $(\mathbf{m}_{ij}, \mathbf{d}_{ij})$ to (m, d).

$$\begin{aligned} P(\mathbf{x}_i \mid \phi)P(\mathbf{x}_j \mid \phi) &= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{2k} e^{-\frac{m^2+d^2}{\sigma^2}} \qquad (3.8) \\ P(\mathbf{m}_{ij} \mid \phi)P(\mathbf{d}_{jj} \mid \phi) &= 2^k\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^k e^{\frac{-m^2}{\sigma^2}} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^k e^{\frac{-d^2}{\sigma^2}} \end{aligned}$$

$$\begin{aligned} P(m \mid \phi)P(d \mid \phi) &= \left(\frac{2m^{k-1}\pi^{\frac{k}{2}}}{\Gamma(\frac{k}{2})}\right)\left(\frac{1}{\sqrt{\pi}\sigma}\right)^{k} e^{\frac{-m^2}{\sigma^2}} \left(\frac{2d^{k-1}\pi^{\frac{k}{2}}}{\Gamma(\frac{k}{2})}\right)\left(\frac{1}{\sqrt{\pi}\sigma}\right)^{k} e^{\frac{-d^2}{\sigma^2}} \\ &= \left(\frac{2^{1-\frac{k}{2}}}{\Gamma(\frac{k}{2})}\right)\left(\frac{\sqrt{2}}{\sigma}\right)^{k} m^{k-1} e^{\frac{-m^2}{\sigma^2}} \left(\frac{2^{1-\frac{k}{2}}}{\Gamma(\frac{k}{2})}\right)\left(\frac{\sqrt{2}}{\sigma}\right)^{k} d^{k-1} e^{\frac{-d^2}{\sigma^2}} \end{aligned}$$

Lemma A.0.4 now applies, to show this term is also a product of two χ_k distributions: $\frac{m\sqrt{2}}{\sigma} \sim \chi_k$, and $\frac{d\sqrt{2}}{\sigma} \sim \chi_k$, or in terms of m' and d':

$$P(m' \mid \phi)P(d' \mid \phi) = 2\chi_k(m'\sqrt{2})\chi_k(d'\sqrt{2}).$$

APPENDIX B

DENSITY PLOTS FOR GAUSSIAN DATA IN HIGHER DIMENSIONS

In Chapter 3, Figure 3.2 shows the distributions of positive and negative pairs and the resulting likelihood ratio, as a function of m' and d', for 2-dimensional data. Figure B.1 displays these plots for additional values of k. In k dimensions, for a given value of t, the LR's contour lines have roughly the same shape as before, but as k grows, the distributions end up better separated.

To see why, recall that the distributions of positive and negative pairs, respectively, are as follows:

$$P(m' \mid \phi)P(d' \mid \epsilon) = \left(\frac{1}{t}\right)\chi_k(m')\chi_k\left(\frac{d'}{t}\right) \tag{3.11}$$

$$P(m' \mid \phi)P(d' \mid \phi) = 2\chi_k(m'\sqrt{2})\chi_k(d'\sqrt{2}). \tag{3.12}$$

Since the peak of χ_k is at $\sqrt{k-1}$, the peak of the negatives is always at $(\frac{\sqrt{k-1}}{\sqrt{2}}, \frac{\sqrt{k-1}}{\sqrt{2}})$, and that of the positives is at $(\sqrt{k-1}, t\sqrt{k-1})$. As the dimensionality varies, the relative positions of the peaks are the same up to a scale factor $\sqrt{k-1}$. However, the variances of the distributions do not scale as fast. So, compared to their positions, the distributions get proportionally narrower and easier to distinguish.

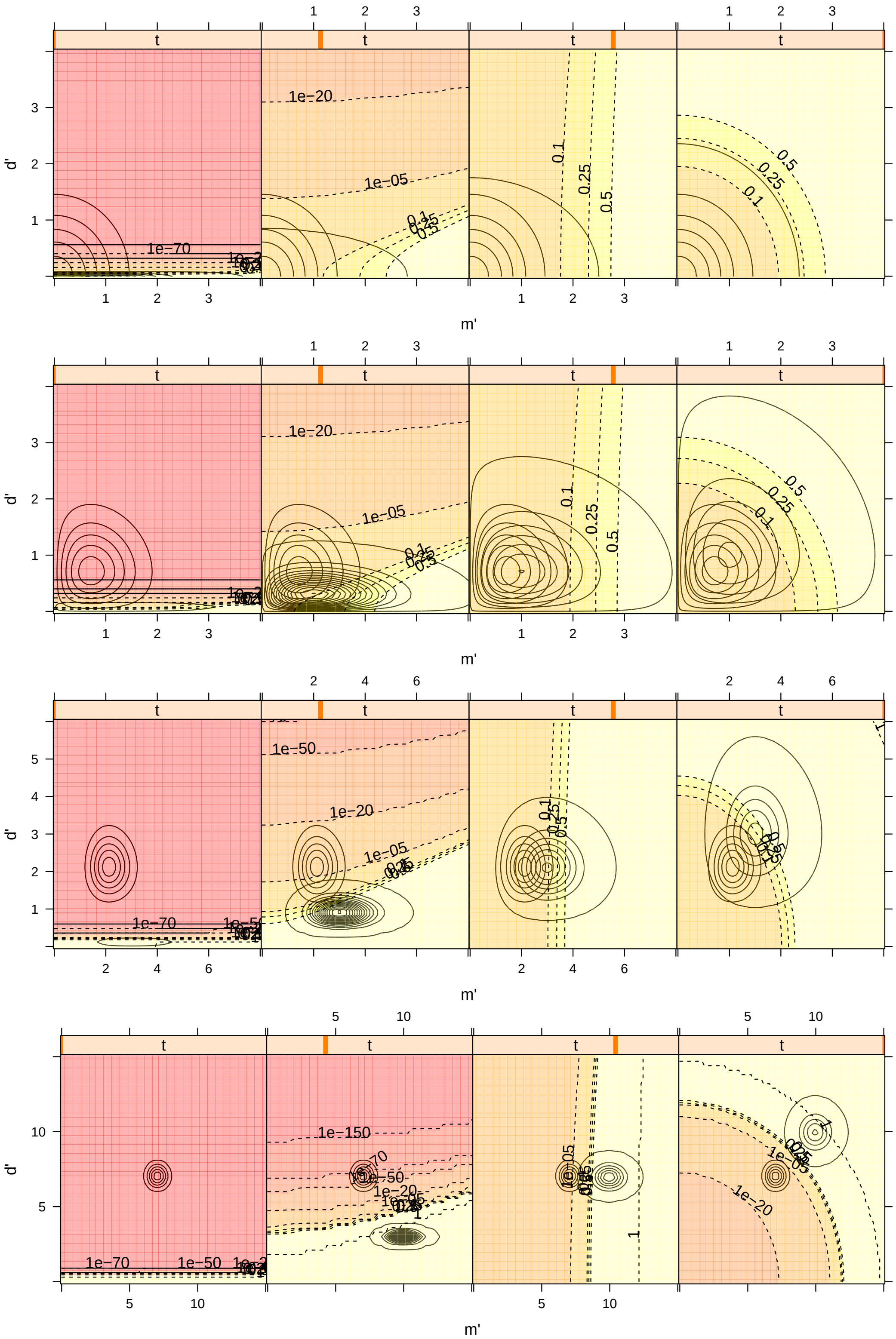
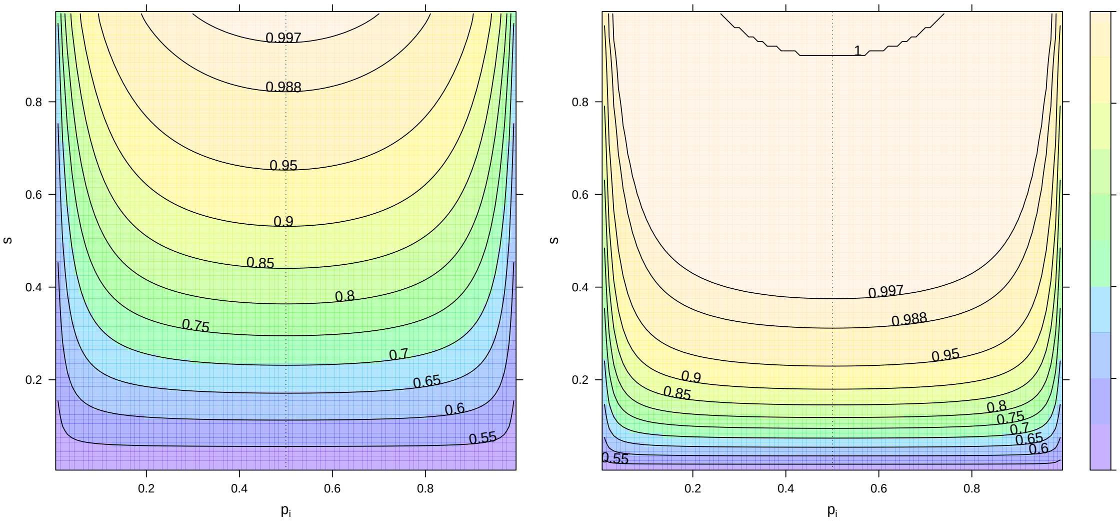


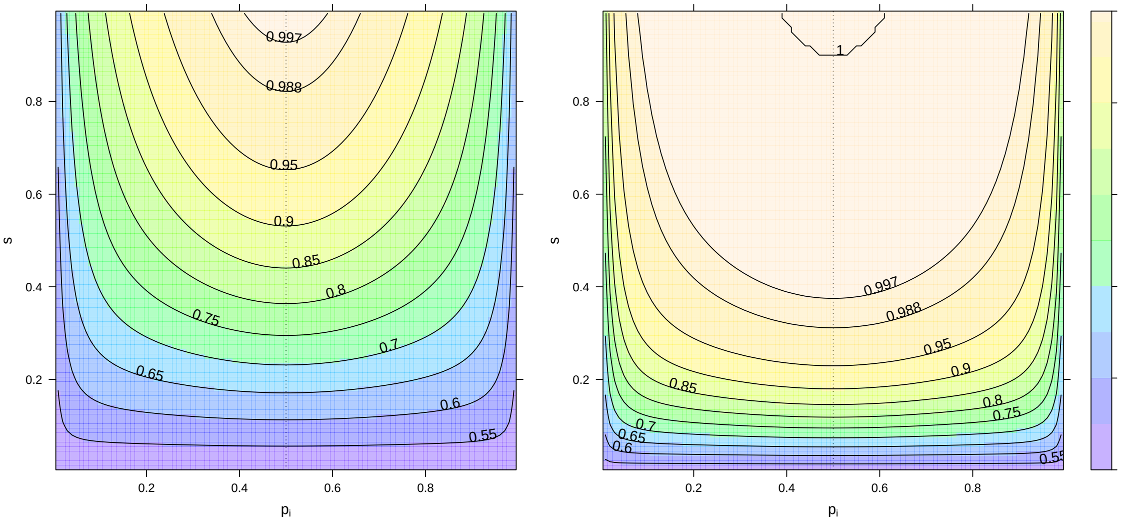
Figure B.1: Theoretical distributions of positive and negative pairs, and likelihood ratio score assigned, as a function of (m', d') when $n = 25$, $\mathrm{E}(r) = 10$. From left to right, t takes on values (.02, .3, .7, 1). From top to bottom, number of dimensions $k = 1, 2, 10, 100$. Note that the scales change for the bottom plots.

APPENDIX C

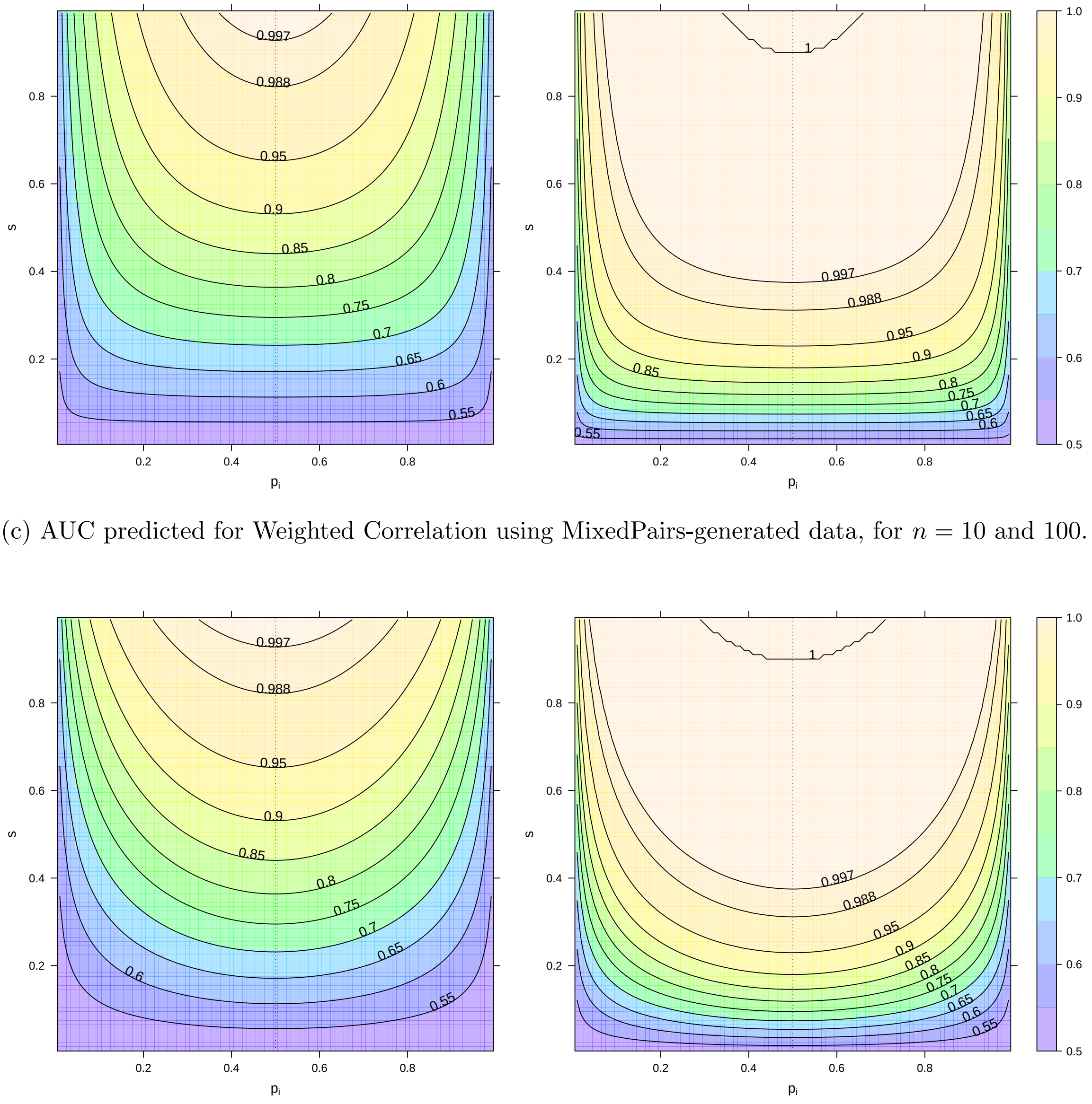
AUCS FOR COMPARISON METHODS USING MIXEDPAIRS DATA



(a) AUC predicted for MixedPairs as a function of p_i and s, using $n = 10$ (left) and $n = 100$ (right) with constant p_i. (Left side repeats Figure 4.4.)



(b) AUC predicted for SharedWeight1100 using MixedPairs-generated data, for $n = 10$ and 100.



(c) AUC predicted for Weighted Correlation using MixedPairs-generated data, for $n = 10$ and 100.

(d) AUC predicted for Hamming using MixedPairs-generated data, for $n = 10$ and 100.

Figure C.1: Theoretical AUCs for methods symmetric about $p_i = 0.5$. Notice that the settings where $n = 100$ and $s = 0.2$, which can be examined within the right-hand column, correspond to the experiment from Figure 4.6 (specifically, its constant p_i settings). These methods differ only subtly in these plots, in the curvatures of their contour lines as p_i moves away from 0.5.

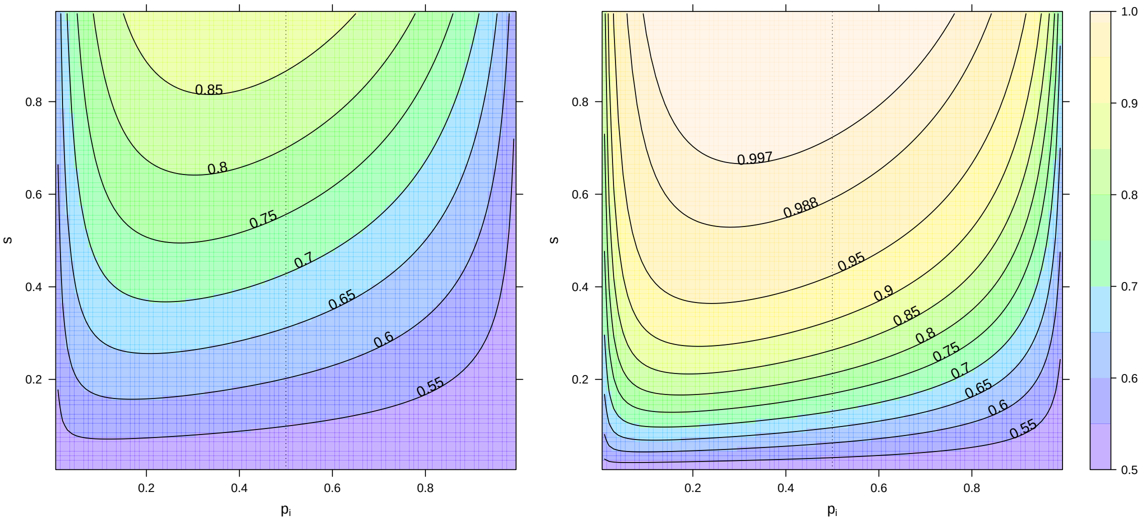


Figure C.2: AUC predicted for SharedSize using MixedPairs-generated data, for $n = 10$ and 100. When p_i is constant, all other 1|1-based methods (Adamic/Adar, Newman and SharedWeight11) are identical. Notice how, unlike with the symmetric methods, the best p_i depends on s and n and is less than 0.5.

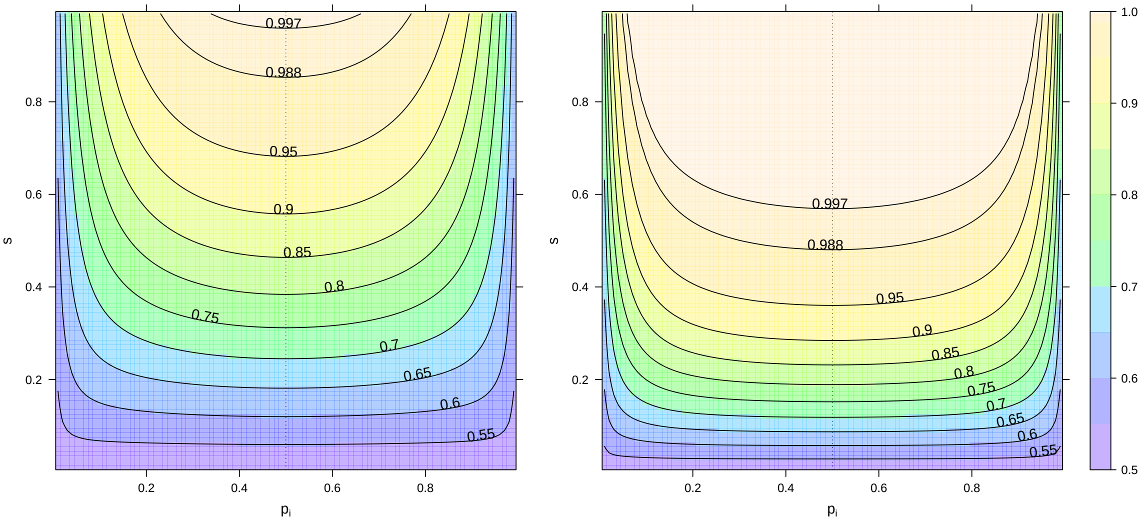
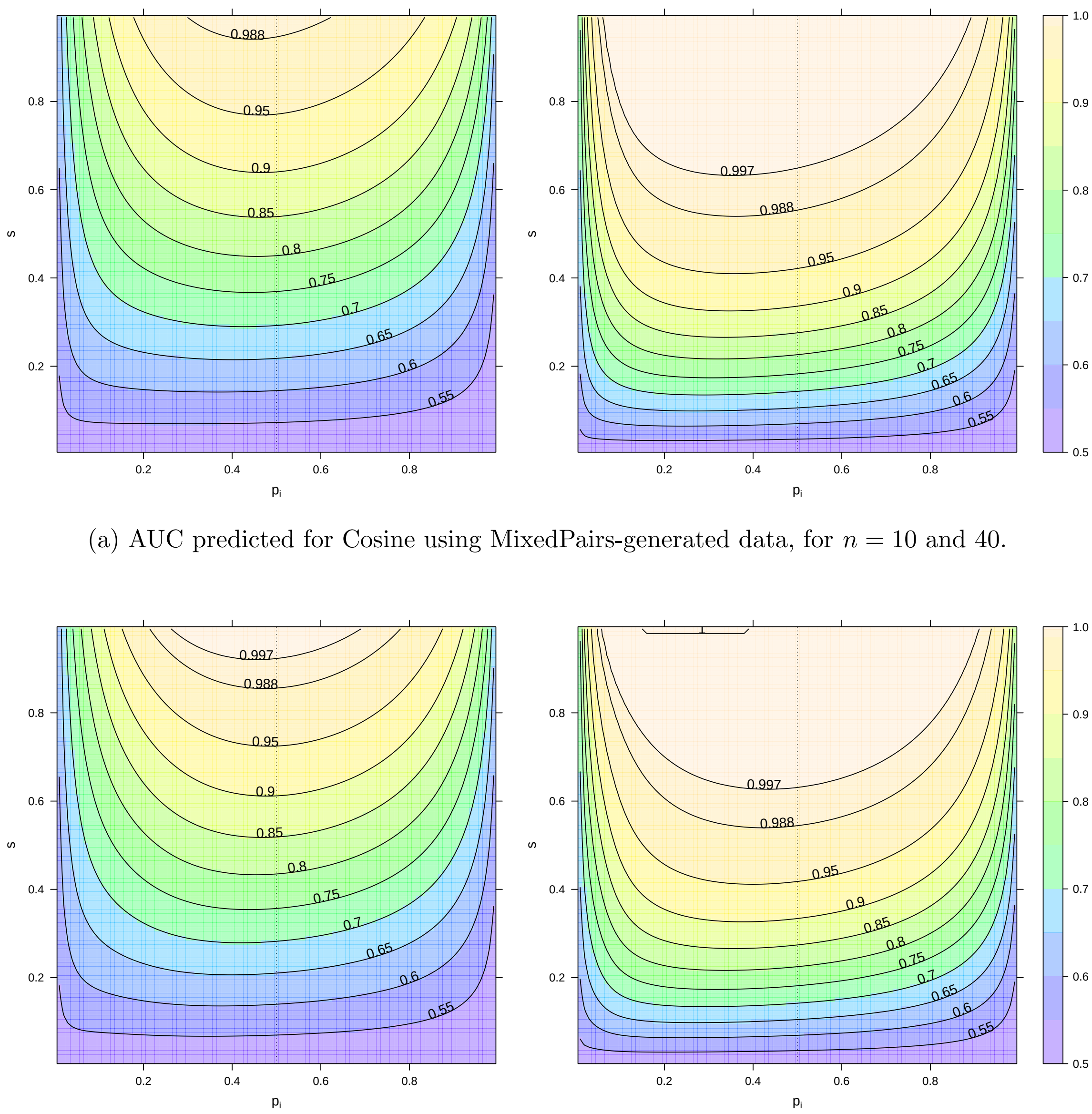


Figure C.3: Theoretical AUCs for Pearson correlation using MixedPairs-generated data, for $n = 10$ and 40. When p_i is constant, this method is symmetric. For Pearson and the other methods that are not pairwise independent, computing AUC is computationally intensive, so we only increase n to 40.



(a) AUC predicted for Cosine using MixedPairs-generated data, for $n = 10$ and 40.

(b) AUC predicted for Jaccard using MixedPairs-generated data, for $n = 10$ and 40.

Figure C.4: Theoretical AUCs for the unweighted normalized methods. For constant p_i, CosineIDF is equivalent to Cosine. Like with SharedSize, the p_i that maximizes AUC for these methods varies as a function of s and n. In these plots, the best p_i is less than 0.5, although this property varies at lower values of n.

Note: although normally Cosine and Pearson are undefined if either item in the pair is all 0s, we define them to be zero. The justification is that it lets these plots, along with the synthetic experiments, be smooth (and for Pearson, symmetric). (The alternative, of discarding these pairs for *all* methods, gives consistent experimental results, but makes interpretation more difficult because all AUCs then rise as p_i approaches 0.)

APPENDIX D

ADDITIONAL FIGURES SHOWING EFFECTS OF FLIPPING

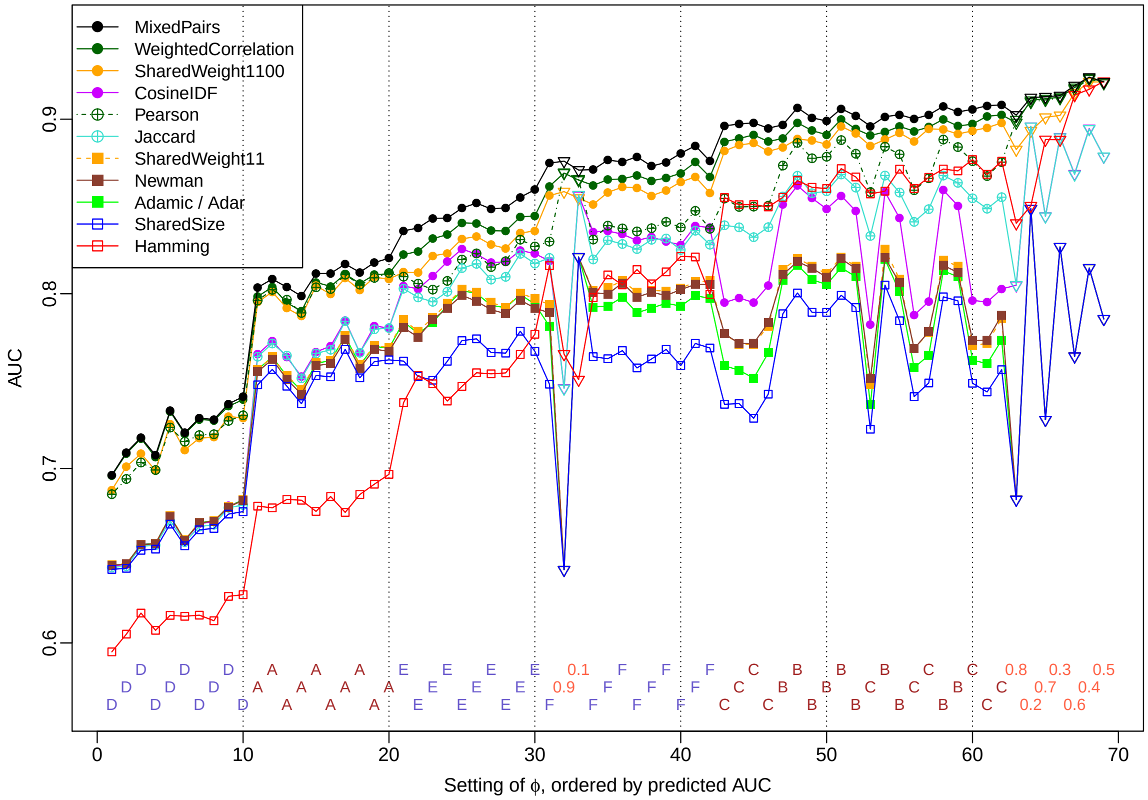


Figure D.1: Synthetic data experiments, with affiliations not flipped. Compared with Figure 4.6, the non-symmetric methods drop dramatically in some high-p_i settings, especially those labeled “C” or having a constant $p_i > 0.5$.

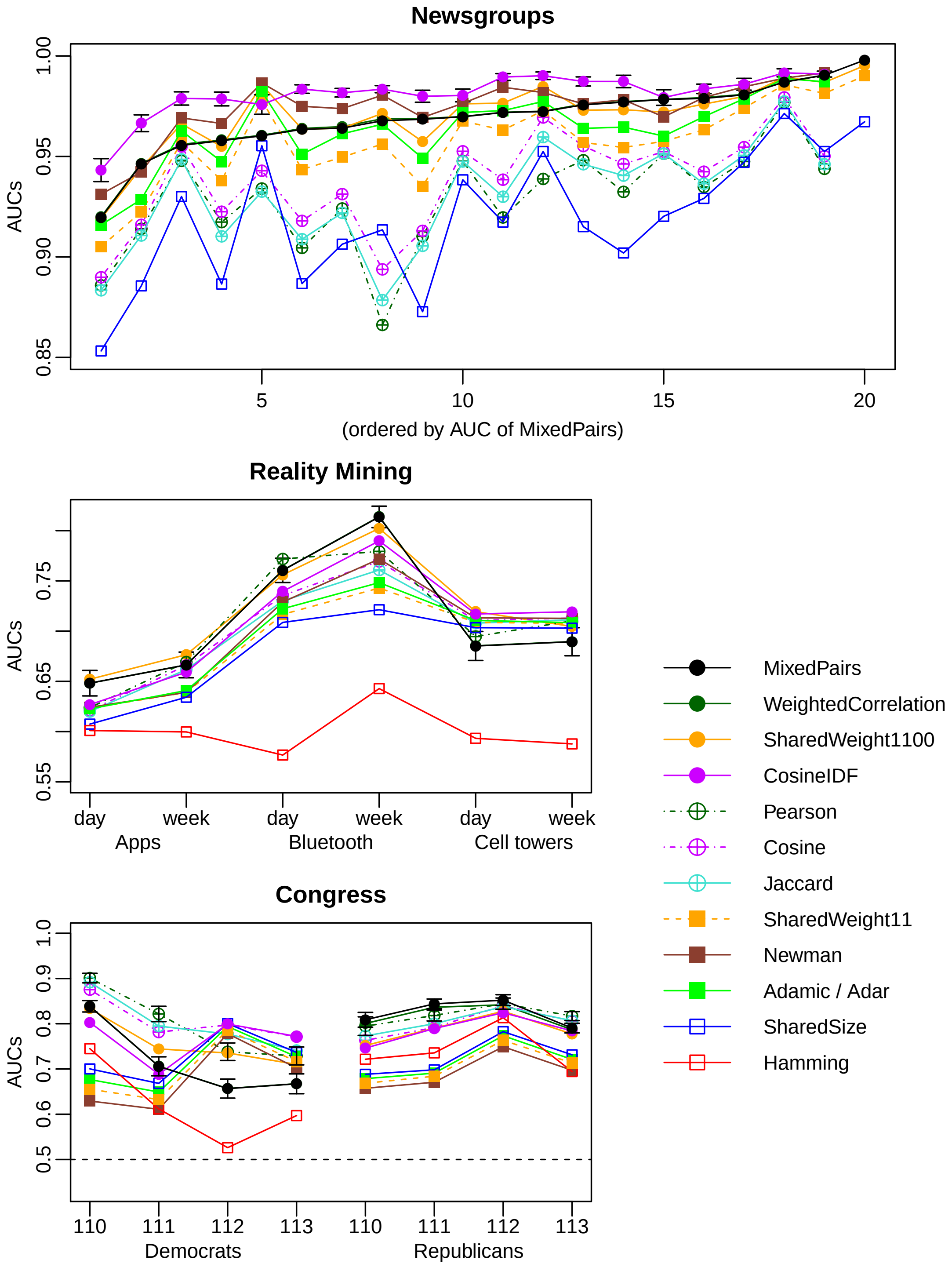


Figure D.2: Experimental results on real data, with high-p_i affiliations flipped. (Compare with Figure 4.7.)

BIBLIOGRAPHY

[1] L. A. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, *25*(3) (2003), pp. 211–230. DOI: `10.1016/S0378-8733(03)00009-1`.

[2] L. M. Aiello, A. Barrat, R. Schifanella, C. Cattuto, B. Markines, and F. Menczer. Friendship prediction and homophily in social media. *ACM Transactions on the Web*, *6*(2) (June 2012). DOI: `10.1145/2180861.2180866`.

[3] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, *9* (June 2008), pp. 1981–2014. URL: `http://portal.acm.org/citation.cfm?id=1442798`.

[4] C. G. G. Aitken and D. Lucy. Evaluation of trace evidence in the form of multivariate data. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *53*(1) (Jan. 2004), pp. 109–122. DOI: `10.1046/j.0035-9254.2003.05271.x`.

[5] C. G. G. Aitken, G. Zadora, and D. Lucy. A two-level model for evidence evaluation. *Journal of Forensic Sciences*, *52*(2) (2007), pp. 412–419. DOI: `10.1111/j.1556-4029.2006.00358.x`.

[6] A. Baddeley. Spatial point processes and their applications. In *Stochastic Geometry. Lectures Given At the C.I.M.E. Summer School* (2007), Springer Lecture Notes in Mathematics 1892, pp. 1–75. DOI: `10.1007/978-3-540-38175-4_1`.

[7] A. Barabási. Emergence of scaling in random networks. *Science*, *286*(5439) (Oct. 1999), pp. 509–512. DOI: `10.1126/science.286.5439.509`.

[8] N. Barbieri, F. Bonchi, and G. Manco. Who to follow and why: Link prediction with explanations. In *KDD '14: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2014), pp. 1266–1275. DOI: `10.1145/2623330.2623733`.

[9] J. Baumes, M. Goldberg, M. Magdon-Ismail, and W. A. Wallace. Discovering hidden groups in communication networks. In *ISI: Proceedings of the 2nd Symposium on Intelligence and Security Informatics* (Jan. 2004), Springer LNCS 3073, pp. 378–389. DOI: `10.1007/b98042`.

[10] R. J. Bayardo, Y. Ma, and R. Srikant. Scaling up all pairs similarity search. In *WWW '07: Proceedings of the 16th International Conference on World Wide Web* (2007), pp. 131–140. URL: `http://dl.acm.org/citation.cfm?id=1242591`.

[11] L. Bejder, D. Fletcher, and S. Bräger. A method for testing association patterns of social animals. *Animal Behaviour*, *56*(3) (1998), pp. 719–725.

[12] T. R. Belin and D. B. Rubin. A method for calibrating false-match rates in record linkage. *Journal of the American Statistical Association*, *90*(430) (June 1995), pp. 694–707. DOI: 10.1080/01621459.1995.10476563.

[13] A. Beutel, W. Xu, V. Guruswami, C. Palow, and C. Faloutsos. CopyCatch: Stopping group attacks by spotting lockstep behavior in social networks. In *WWW '13: Proceedings of the 22nd International Conference on World Wide Web* (Rio de Janeiro, Brazil, 2013), pp. 119–130. DOI: 10.1145/2488388.2488400.

[14] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data*, *1*(1) (Mar. 2007). DOI: 10.1145/1217299.1217304.

[15] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda. All your contacts are belong to us: Automated identity theft attacks on social networks. In *WWW '09: Proceedings of the 18th International Conference on World Wide Web* (Madrid, Spain, 2009), pp. 551–560. DOI: 10.1145/1526709.1526784.

[16] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, *3* (2003), pp. 993–1022. URL: http://www.jmlr.org/papers/v3/blei03a.html.

[17] A. Bolck, C. Weyermann, L. Dujourdy, P. Esseiva, and J. van den Berg. Different likelihood ratio approaches to evaluate the strength of evidence of MDMA tablet comparisons. *Forensic Science International*, *191*(1-3) (Oct. 2009), pp. 42–51. DOI: 10.1016/j.forsciint.2009.06.006.

[18] A. Bolck, H. Ni, and M. Lopatka. Evaluating score- and feature-based likelihood ratio models for multivariate continuous data: Applied to forensic MDMA comparison. *Law, Probability and Risk*, *14*(3) (Sept. 2015), pp. 243–266. DOI: 10.1093/lpr/mgv009.

[19] S. Boriah, V. Chandola, and V. Kumar. Similarity measures for categorical data: A comparative evaluation. In *Proceedings of the 2008 SIAM International Conference on Data Mining* (Apr. 2008), pp. 243–254. DOI: 10.1137/1.9781611972788.22.

[20] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, *29*(8-13) (Sept. 1997), pp. 1157–1166. DOI: 10.1016/s0169-7552(97)00031-7.

[21] J. B. Burns, C. I. Connolly, J. F. Thomere, and M. J. Wolverton. Event recognition in airborne motion imagery. In *Capturing and Using Patterns for Evidence Detection*, Papers from the AAAI Fall Symposium (2006).

[22] S. J. Cairns and S. J. Schwager. A comparison of association indices. *Animal Behaviour*, *35*(5) (1987), pp. 1454–1469. DOI: 10.1016/S0003-3472(87)80018-0.

[23] Q. Cao, X. Yang, J. Yu, and C. Palow. Uncovering large groups of active malicious accounts in online social networks. In *CCS '14: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* (2014), pp. 477–488. DOI: 10.1145/2660267.2660269.

[24] G. Casella and R. L. Berger. Statistical Inference. 2nd ed. Thomson Learning, June 2002. URL: www.worldcat.org/isbn/0534243126.

[25] S. H. Cha. Taxonomy of nominal type histogram distance measures. In *MATH '08: Proceedings of the American Conference on Applied Mathematics* (Cambridge, Massachusetts, 2008), pp. 325–330. URL: http://portal.acm.org/citation.cfm?id=1415640.

[26] J. Chang and D. M. Blei. Hierarchical relational models for document networks. *The Annals of Applied Statistics*, *4*(1) (Mar. 2010), pp. 124–150. DOI: 10.1214/09-aoas309.

[27] S. Chaudhuri, V. Ganti, and R. Motwani. Robust identification of fuzzy duplicates. In *ICDE 2005: Proceedings of the 21st International Conference on Data Engineering* (Apr. 2005), pp. 865–876. DOI: 10.1109/icde.2005.125.

[28] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: User movement in location-based social networks. In *KDD '11: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Diego, California, USA, 2011), pp. 1082–1090. DOI: 10.1145/2020408.2020579.

[29] H. Choi and H. Lee. Identifying botnets by capturing group activities in DNS traffic. *Computer Networks*, *56*(1) (Jan. 2012), pp. 20–33. DOI: 10.1016/j.comnet.2011.07.018.

[30] A. Chowdhury, O. Frieder, D. Grossman, and M. C. McCabe. Collection statistics for fast duplicate document detection. *ACM Transactions on Information Systems*, *20*(2) (Apr. 2002), pp. 171–191. DOI: 10.1145/506309.506311.

[31] P. Christen. Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Data-Centric Systems and Applications. Springer, July 2012. DOI: 10.1007/978-3-642-31164-2.

[32] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-Hash and tf-idf weighting. In *Proceedings of the British Machine Vision Conference* (2008), pp. 50.1–50.10. DOI: 10.5244/C.22.50.

[33] A. Clauset, C. Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, *453*(7191) (May 2008), pp. 98–101. DOI: 10.1038/nature06830.

[34] S. Cohen, B. Kimelfeld, and G. Koutrika. A survey on proximity measures for social networks. In *Search Computing* (2012), Springer LNCS 7538, pp. 191–206. DOI: 10.1007/978-3-642-34213-4_13.

[35] W. W. Cohen. Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems*, *18*(3) (July 2000), pp. 288–321. DOI: 10.1145/352595.352598.

[36] W. W. Cohen. Probabilistic record linkage: A short tutorial. Slides. Accessed May 2013. c. 2003. URL: http://www.cs.cmu.edu/~wcohen/Matching-1.ppt.

[37] W. W. Cohen, P. D. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb-03: Proceedings of IJCAI-03 Workshop on Information Integration on the Web* (Acapulco, Mexico, 2003), pp. 73–78. URL: http://www.isi.edu/info-agents/workshops/ijcai03/papers/Cohen-p.pdf.

[38] J. B. Copas and F. J. Hilton. Record linkage: Statistical models for matching computer records. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, *153*(3) (1990), pp. 287–320. DOI: 10.2307/2982975.

[39] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri. Feedback effects between similarity and social influence in online communities. In *KDD '08: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada, USA, 2008), pp. 160–168. DOI: 10.1145/1401890.1401914.

[40] D. J. Crandall, L. Backstrom, D. Cosley, S. Suri, D. Huttenlocher, and J. Kleinberg. Inferring social ties from geographic coincidences. *Proceedings of the National Academy of Sciences*, *107*(52) (Dec. 2010), pp. 22436–22441. DOI: 10.1073/pnas.1006155107.

[41] J. Cranshaw, E. Toch, J. Hong, A. Kittur, and N. Sadeh. Bridging the gap between physical location and online social networks. In *Ubicomp '10: Proceedings of the 12th ACM International Conference on Ubiquitous Computing* (2010), pp. 119–128. DOI: 10.1145/1864349.1864380.

[42] M. De Choudhury, W. A. Mason, J. M. Hofman, and D. J. Watts. Inferring relevant social networks from interpersonal communication. In *WWW '10: Proceedings of the 19th International Conference on World Wide Web* (Raleigh, North Carolina, USA, 2010), pp. 301–310. DOI: 10.1145/1772690.1772722.

[43] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6) (1990), pp. 391–407.

[44] N. Eagle and A. Pentland. Reality mining: Sensing complex social systems. *Personal and Ubiquitous Computing*, *10*(4) (2006), pp. 255–268. DOI: 10.1007/s00779-005-0046-3.

[45] N. Eagle, A. S. Pentland, and D. Lazer. Inferring friendship network structure by using mobile phone data. *Proceedings of the National Academy of Sciences*, *106*(36) (2009), pp. 15274–15278. DOI: 10.1073/pnas.0900282106.

[46] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, *19*(1) (2007), pp. 1–16. DOI: 10.1109/TKDE.2007.9.

[47] D. Faraggi and B. Reiser. Estimation of the area under the ROC curve. *Statistics in Medicine*, *21*(20) (2002), pp. 3093–3106. DOI: 10.1002/sim.1228.

[48] A. Fast, L. Friedland, M. Maier, B. Taylor, D. Jensen, H. G. Goldberg, and J. Komoroske. Relational data pre-processing techniques for improved securities fraud detection. In *KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2007), pp. 941–949. DOI: 10.1145/1281192.1281293.

[49] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8) (June 2006), pp. 861–874. DOI: 10.1016/j.patrec.2005.10.010.

[50] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, *64*(328) (Dec. 1969), pp. 1183–1210. DOI: 10.1080/01621459.1969.10501049.

[51] M. Fortini, A. Nuccitelli, B. Liseo, and M. Scanu. Modelling issues in record linkage: A Bayesian perspective. In *JSM Proceedings, Survey Research Methods Section* (2002). American Statistical Association, pp. 1008–1013.

[52] J. H. Fowler. Connecting the Congress: A study of cosponsorship networks. *Political Analysis*, *14*(4) (2006), pp. 456–487. DOI: 10.1093/pan/mpl002.

[53] L. Friedland and D. Jensen. Finding tribes: Identifying close-knit individuals from employment patterns. In *KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2007), pp. 290–299. DOI: 10.1145/1281192.1281226.

[54] L. Friedland, D. Jensen, and M. Lavine. Copy or coincidence? A model for detecting social influence and duplication events. In *ICML '13: Proceedings of The 30th International Conference on Machine Learning* (2013), JMLR W&CP 28(1), pp. 1175–1183. URL: http://jmlr.org/proceedings/papers/v28/friedland13.html.

[55] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *VLDB '99: Proceedings of the 25th International Conference on Very Large Data Bases* (1999), pp. 518–529. URL: http://portal.acm.org/citation.cfm?id=671516.

[56] A. Goldenberg, A. X. Zheng, S. E. Fienberg, and E. M. Airoldi. A survey of statistical network models. *Foundations and Trends in Machine Learning*, *2*(2) (Feb. 2010), pp. 129–233. DOI: 10.1561/2200000005.

[57] G. Gu, R. Perdisci, J. Zhang, and W. Lee. BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of the 17th USENIX Security Symposium* (2008), pp. 139–154. URL: `https://www.usenix.org/legacy/event/sec08/tech/`.

[58] M. Gupte and T. Eliassi-Rad. Measuring tie strength in implicit social networks. In *WebSci '12: Proceedings of the 4th Annual ACM Web Science Conference* (2012), pp. 109–118. DOI: `10.1145/2380718.2380734`.

[59] H. Hajishirzi, W. T. Yih, and A. Kolcz. Adaptive near-duplicate detection via similarity learning. In *SIGIR '10: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2010), pp. 419–426. DOI: `10.1145/1835449.1835520`.

[60] D. J. Hand. Measuring classifier performance: A coherent alternative to the area under the ROC curve. *Machine Learning*, *77*(1) (Oct. 2009), pp. 103–123. DOI: `10.1007/s10994-009-5119-5`.

[61] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *Proceedings of the Fourth Workshop on Link Analysis, Counterterrorism and Security* (Apr. 2006). Sixth SIAM International Conference on Data Mining. URL: `http://www.cs.rpi.edu/~zaki/PS/LINK06.pdf`.

[62] T. Hastie, R. Tibshirani, and J. Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed. Springer, 2009. URL: `http://www.worldcat.org/isbn/0387848576`.

[63] T. N. Herzog, F. J. Scheuren, and W. E. Winkler. Data Quality and Record Linkage Techniques. Springer, May 2007. URL: `http://www.worldcat.org/isbn/0387695028`.

[64] P. D. Hoff, A. E. Raftery, and M. S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, *97*(460) (Dec. 2002), pp. 1090–1098. DOI: `10.1198/016214502388618906`.

[65] M. Jiang, A. Beutel, P. Cui, B. Hooi, S. Yang, and C. Faloutsos. A general suspiciousness metric for dense blocks in multimodal data. In *ICDM: 2015 IEEE International Conference on Data Mining* (Nov. 2015), pp. 781–786. DOI: `10.1109/icdm.2015.61`.

[66] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang. Catching synchronized behaviors in large networks: A graph mining approach. *ACM Transactions on Knowledge Discovery from Data*, *10*(4) (June 2016). DOI: `10.1145/2746403`.

[67] R. E. Kass and A. E. Raftery. Bayes factors. *Journal of the American Statistical Association*, *90*(430) (June 1995), pp. 773–795. DOI: `10.1080/01621459.1995.10476572`.

[68] G. Kossinets and D. J. Watts. Empirical analysis of an evolving social network. *Science*, *311*(5757) (Jan. 2006), pp. 88–90. DOI: `10.1126/science.1116869`.

[69] W. Kraaij. Variations on language modeling for information retrieval. PhD thesis. Enschede: University of Twente, 2004. URL: http://eprints.eemcs.utwente.nl/6571/.

[70] J. Krause, D. Lusseau, and R. James. Animal social networks: An introduction. *Behavioral Ecology and Sociobiology*, *63*(7) (2009), pp. 967–973. DOI: 10.1007/s00265-009-0747-0.

[71] S. Kullback. Information Theory and Statistics. Dover, 1968. Google Books: 05LwShwkhFYC.

[72] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2009), pp. 497–506. DOI: 10.1145/1557019.1557077.

[73] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research*, *11* (Mar. 2010), pp. 985–1042. URL: http://portal.acm.org/citation.cfm?id=1756039.

[74] J. Leskovec, A. Rajaraman, and J. D. Ullman. Mining of Massive Datasets. Cambridge University Press, 2014. URL: http://mmds.org.

[75] D. D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *ECML-98: Proceedings of the 10th European Conference on Machine Learning* (1998), Springer LNCS 1398, pp. 4–15. DOI: 10.1007/bfb0026666.

[76] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, *58*(7) (2007), pp. 1019–1031. DOI: 10.1002/asi.20591.

[77] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla. New perspectives and methods in link prediction. In *KDD '10: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2010), pp. 243–252. DOI: 10.1145/1835804.1835837.

[78] D. Lin. An information-theoretic definition of similarity. In *ICML '98: Proceedings of the 15th International Conference on Machine Learning* (1998), pp. 296–304. URL: http://dl.acm.org/citation.cfm?id=645527.657297.

[79] D. V. Lindley. A problem in forensic science. *Biometrika*, *64*(2) (Aug. 1977), pp. 207–213. DOI: 10.2307/2335686.

[80] C. Liu, C. Chen, J. Han, and P. S. Yu. GPLAG: Detection of software plagiarism by program dependence graph analysis. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2006), pp. 872–881. DOI: 10.1145/1150402.1150522.

[81] X. Liu, Q. He, Y. Tian, W. C. Lee, J. McPherson, and J. Han. Event-based social networks: Linking the online and offline social worlds. In *KDD '12: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Beijing, China, 2012), pp. 1032–1040. DOI: 10.1145/2339530.2339693.

[82] M. Magdon-Ismail, M. Goldberg, W. Wallace, and D. Siebecker. Locating hidden groups in communication networks using hidden Markov models. In *Intelligence and Security Informatics* (2003), Springer LNCS 2665, pp. 126–137. DOI: 10.1007/3-540-44853-5_10.

[83] C. D. Manning, P. Raghavan, and H. Schütze. Introduction to Information Retrieval. Cambridge University Press, 2008. URL: http://informationretrieval.org/.

[84] A. McCallum and K. Nigam. A comparison of event models for Naive Bayes text classification. In *Proc. AAAI-98 Workshop on Learning for Text Categorization* (1998), pp. 41–48. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.46.1529.

[85] A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *KDD '00: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2000), pp. 169–178. DOI: 10.1145/347090.347123.

[86] S. Merritt, A. Jacobs, W. Mason, and A. Clauset. Detecting friendship within dynamic online interaction networks. In *ICWSM '13: International AAAI Conference on Weblogs and Social Media* (2013). URL: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM13/paper/view/6049.

[87] A. Metwally, D. Agrawal, and A. E. Abbadi. DETECTIVES: DETEcting Coalition hiT inflation attacks in adVertising nEtworks Streams. In *WWW '07: Proceedings of the 16th International Conference on World Wide Web* (2007), pp. 241–250. DOI: 10.1145/1242572.1242606.

[88] D. Metzler. Beyond bags of words: Effectively modeling dependence and features in information retrieval. PhD thesis. University of Massachusetts Amherst, 2007. URL: http://www.isi.edu/~metzler/papers/ir-625.pdf.

[89] D. Metzler, Y. Bernstein, W. B. Croft, A. Moffat, and J. Zobel. Similarity measures for tracking information flow. In *CIKM '05: Proceedings of the 14th ACM International Conference on Information and Knowledge Management* (2005), pp. 517–524. DOI: 10.1145/1099554.1099695.

[90] N. Meuschke and B. Gipp. State-of-the-art in detecting academic plagiarism. *International Journal for Academic Integrity*, *9*(1) (June 2013), pp. 50–71. URL: http://ojs.unisa.edu.au/index.php/IJEI/article/view/847.

[91] F. Mitzlaff, M. Atzmueller, A. Hotho, and G. Stumme. The social distributional hypothesis: A pragmatic proxy for homophily in online social networks. *Social Network Analysis and Mining*, *4* (2014), p. 216. DOI: 10.1007/s13278-014-0216-2.

[92] G. S. Morrison. A comparison of procedures for the calculation of forensic likelihood ratios from acoustic-phonetic data: Multivariate kernel density (MVKD) versus Gaussian mixture model-universal background model (GMM-UBM). *Speech Communication*, *53*(2) (Feb. 2011), pp. 242–256. DOI: 10.1016/j.specom.2010.09.005.

[93] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy* (2008), pp. 111–125. DOI: 10.1109/SP.2008.33.

[94] National Center for Health Statistics. Matched Multiple Birth Data, 1995–2000. Public-use data file and documentation. 2000. URL: http://ftp.cdc.gov/pub/Health_Statistics/NCHS/Datasets/mmb2/.

[95] J. Neville, O. Şimşek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg. Using relational knowledge discovery to prevent securities fraud. In *KDD '05: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* (2005), pp. 449–458. DOI: 10.1145/1081870.1081922.

[96] M. E. J. Newman. Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality. *Physical Review E*, *64*(1) (June 2001), p. 016132. DOI: 10.1103/PhysRevE.64.016132.

[97] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, *45* (2003), pp. 167–256. URL: http://dx.doi.org/10.1137/S003614450342480.

[98] M. E. J. Newman, A.-L. Barabási, and D. J. Watts. The structure and dynamics of networks. 2006. URL: http://www.worldcat.org/isbn/9781400841356.

[99] G. N. Norén, R. Orre, A. Bate, and I. R. Edwards. Duplicate detection in adverse drug reaction surveillance. *Data Mining and Knowledge Discovery*, *14*(3) (June 2007), pp. 305–328. DOI: 10.1007/s10618-006-0052-8.

[100] J. O'Madadhain, J. Hutchins, and P. Smyth. Prediction and ranking algorithms for event-based network data. *SIGKDD Explorations Newsletter*, *7*(2) (Dec. 2005), pp. 23–30. DOI: 10.1145/1117454.1117458.

[101] H. Pham, C. Shahabi, and Y. Liu. EBM: an entropy-based model to infer social strength from spatiotemporal data. In *SIGMOD '13: Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data* (2013), pp. 265–276. DOI: 10.1145/2463676.2465301.

[102] R. Porter, C. Ruggiero, and J. D. Morrison. A framework for activity detection in wide-area motion imagery. In *Visual Information Processing XVIII* (Orlando, Florida, USA, Apr. 2009), 73410O+. DOI: 10.1117/12.818629.

[103] M. Potthast, M. Hagen, A. Beyer, M. Busse, M. Tippmann, P. Rosso, and B. Stein. Overview of the 6th international competition on plagiarism detection. In *Working Notes for CLEF 2014 Conference* (Sheffield, UK, 2014), pp. 845–876. URL: http://ceur-ws.org/Vol-1180/.

[104] F. Provost, D. Martens, and A. Murray. Finding similar mobile consumers with a privacy-friendly geosocial design. *Information Systems Research*, *26*(2) (June 2015), pp. 243–265. DOI: `10.1287/isre.2015.0576`.

[105] J. Rennie. Home Page for 20 Newsgroups Data Set. Accessed 2013-07-26. URL: `http://qwone.com/~jason/20Newsgroups/`.

[106] S. E. Robertson and K. S. Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, *27*(3) (May 1976), pp. 129–146.

[107] S. Robertson. Understanding inverse document frequency: On theoretical arguments for IDF. *Journal of Documentation*, *60*(5) (2004), pp. 503–520. DOI: `10.1108/00220410410560582`.

[108] C. K. Roy, J. R. Cordy, and R. Koschke. Comparison and evaluation of code clone detection techniques and tools: A qualitative approach. *Science of Computer Programming*, *74*(7) (May 2009), pp. 470–495. DOI: `10.1016/j.scico.2009.02.007`.

[109] A. Sadilek, H. Kautz, and J. P. Bigham. Finding your friends and following them to where you are. In *WSDM '12: Proceedings of the 5th ACM International Conference on Web Search and Data Mining* (2012), pp. 723–732. DOI: `10.1145/2124295.2124380`.

[110] P. Sarkar, D. Chakrabarti, and A. W. Moore. Theoretical justification of popular link prediction heuristics. In *IJCAI'11: Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, Vol. 3 (2011), pp. 2722–2727. DOI: `10.5591/978-1-57735-516-8/IJCAI11-453`.

[111] R. Schifanella, A. Barrat, C. Cattuto, B. Markines, and F. Menczer. Folks in folksonomies: Social link prediction from shared metadata. In *WSDM '10: Proceedings of the 3rd ACM International Conference on Web Search and Data Mining* (Mar. 2010), pp. 271–280. DOI: `10.1145/1718487.1718521`.

[112] G. Shakhnarovich, T. Darrell, and P. Indyk, eds. Nearest-Neighbor Methods in Learning and Vision. *Theory and Practice.* MIT Press, 2005. URL: `http://www.worldcat.org/isbn/9780262195478`.

[113] C. R. Shalizi and A. C. Thomas. Homophily and contagion are generically confounded in observational social network studies. *Sociological Methods & Research*, *40*(2) (Nov. 2010), pp. 211–239. DOI: `10.1177/0049124111404820`.

[114] J. Skvoretz and K. Faust. Logit models for affiliation networks. *Sociological Methodology*, *29*(1) (Jan. 1999), pp. 253–280. DOI: `10.1111/0081-1750.00066`.

[115] D. A. Smith, R. Cordell, E. M. Dillon, N. Stramp, and J. Wilkerson. Detecting and modeling local text reuse. In *JCDL '14: Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries* (2014), pp. 183–192. URL: `http://portal.acm.org/citation.cfm?id=2740800`.

[116] T. A. B. Snijders and K. Nowicki. Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of Classification*, *14*(1) (Jan. 1997), pp. 75–100. DOI: `10.1007/s003579900004`.

[117] T. A. B. Snijders, A. Lomi, and V. J. Torló. A model for the multiplex dynamics of two-mode and one-mode networks, with an application to employment preference, friendship, and advice. *Social Networks*, *35*(2) (May 2013), pp. 265–276. DOI: `10.1016/j.socnet.2012.05.005`.

[118] R. C. Steorts, R. Hall, and S. E. Fienberg. A Bayesian approach to graphical record linkage and de-duplication. *Journal of the American Statistical Association* (Oct. 2015). DOI: `10.1080/01621459.2015.1105807`.

[119] O. Stitelman, C. Perlich, B. Dalessandro, R. Hook, T. Raeder, and F. Provost. Using co-visitation networks for detecting large scale online display advertising exchange fraud. In *KDD '13: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Chicago, Illinois, USA, 2013), pp. 1240–1248. DOI: `10.1145/2487575.2488207`.

[120] C. Su and S. N. Srihari. Evaluation of rarity of fingerprints in forensics. In *NIPS 2010: Advances in Neural Information Processing Systems 23* (2010), pp. 1207–1215. URL: `http://books.nips.cc/papers/files/nips23/NIPS2010_0981.pdf`.

[121] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* (Jan. 2009), pp. 1–19. DOI: `10.1155/2009/421425`.

[122] Sunlight Foundation and GovTrack.us. unitedstates/congress · GitHub. Accessed 2015-05-05. URL: `https://github.com/unitedstates/congress`.

[123] A. Tancredi and B. Liseo. A hierarchical Bayesian approach to record linkage and population size problems. *The Annals of Applied Statistics*, *5*(2B) (June 2011), pp. 1553–1585. DOI: `10.1214/10-aoas447`.

[124] A. Tancredi and B. Liseo. Regression analysis with linked data: Problems and possible solutions. *Statistica*, *75*(1) (2015), pp. 19–35. DOI: `10.6092/issn.1973-2201/5821`.

[125] Y. Tang and S. N. Srihari. Likelihood ratio estimation in forensic identification using similarity and rarity. *Pattern Recognition*, *47*(3) (Mar. 2014), pp. 945–958. DOI: `10.1016/j.patcog.2013.07.014`.

[126] F. Taroni, S. Bozza, A. Biedermann, P. Garbolino, and C. Aitken. Data analysis in forensic science. *A Bayesian decision perspective*. John Wiley and Sons, 2010. Google Books: `yAqNDAEACAAJ`.

[127] B. Taskar, M.-F. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *NIPS 2003: Advances in Neural Information Processing Systems 16* (2003). URL: `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.119.8942`.

[128] J. Trevathan and W. Read. Detecting collusive shill bidding. In *ITNG '07: Fourth International Conference on Information Technology* (Apr. 2007), pp. 799–808. DOI: 10.1109/itng.2007.74.

[129] C. Wang, V. Satuluri, and S. Parthasarathy. Local probabilistic models for link prediction. In *ICDM '07: Proceedings of the 2007 Seventh IEEE International Conference on Data Mining* (2007), pp. 322–331. DOI: 10.1109/icdm.2007.108.

[130] H. Wang, Y. Guo, Z. Ma, and X. Chen. WuKong: A scalable and accurate two-phase approach to Android app clone detection. In *ISSTA 2015: Proceedings of the 2015 International Symposium on Software Testing and Analysis* (2015), pp. 71–82. DOI: 10.1145/2771783.2771795.

[131] P. Wang, P. Pattison, and G. Robins. Exponential random graph model specifications for bipartite networks—A dependence hierarchy. *Social Networks*, *35*(2) (May 2013), pp. 211–222. DOI: 10.1016/j.socnet.2011.12.004.

[132] P. Wang, B. Xu, Y. Wu, and X. Zhou. Link prediction in social networks: The state-of-the-art. *Science China Information Sciences*, *58*(1) (2015), pp. 1–38. DOI: 10.1007/s11432-014-5237-y.

[133] S. Wasserman and P. Pattison. Logit models and logistic regressions for social networks: I. An introduction to Markov graphs and p*. *Psychometrika*, *61*(3) (Sept. 1996), pp. 401–425. DOI: 10.1007/bf02294547.

[134] D. Weber-Wulff, K. Köhler, and C. Möller. Collusion Detection System Test Report 2012. Nov. 2012. URL: http://plagiat.htw-berlin.de/collusion-test-2012/.

[135] E. W. Weisstein. Hypersphere — from Wolfram Mathworld. Accessed 2015-11-28. URL: http://mathworld.wolfram.com/Hypersphere.html.

[136] S. Whang and H. Garcia-Molina. Managing information leakage. In *CIDR 2011: Fifth Biennial Conference on Innovative Data Systems Research* (Asilomar, CA, USA, 2011), pp. 79–84.

[137] H. Whitehead. Analyzing Animal Societies: Quantitative Methods for Vertebrate Social Analysis. University Of Chicago Press, 2008. Google Books: KDrwLgEACAAJ.

[138] Wikipedia. N-sphere — Wikipedia, The Free Encyclopedia. Accessed 2015-11-28. 2015. URL: https://en.wikipedia.org/wiki/N-sphere.

[139] W. E. Winkler. Using the EM algorithm for weight computation in the Fellegi-Sunter model of record linkage. In *JSM Proceedings, Survey Research Methods Section* (1988). American Statistical Association, pp. 667–671.

[140] W. E. Winkler. String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In *JSM Proceedings, Survey Research Methods Section* (1990). American Statistical Association, pp. 354–359.

[141] H. Yang and J. Callan. Near-duplicate detection by instance-level constrained clustering. In *SIGIR '06: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (2006), pp. 421–428. DOI: `10.1145/1148170.1148243`.

[142] S. H. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha. Like like alike: Joint friendship and interest propagation in social networks. In *WWW '11: Proceedings of the 20th International Conference on World Wide Web* (Hyderabad, India, 2011), pp. 537–546. DOI: `10.1145/1963405.1963481`.

[143] Z. Yang, C. Wilson, X. Wang, T. Gao, B. Y. Zhao, and Y. Dai. Uncovering social network Sybils in the wild. *ACM Transactions on Knowledge Discovery from Data*, *8*(1) (Feb. 2014). DOI: `10.1145/2556609`.

[144] W. T. Yih. Learning term-weighting functions for similarity measures. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Vol. 2 (2009), pp. 793–802. URL: `http://portal.acm.org/citation.cfm?id=1699616`.

[145] G. Zadora and T. Neocleous. Likelihood ratio model for classification of forensic evidence. *Analytica Chimica Acta*, *642*(1-2) (May 2009), pp. 266–278. DOI: `10.1016/j.aca.2008.12.013`.

[146] R. Zafarani, L. Tang, and H. Liu. User identification across social media. *ACM Transactions on Knowledge Discovery from Data*, *10*(2) (Oct. 2015). DOI: `10.1145/2747880`.

[147] Q. Zhang, H. Ma, W. Qian, and A. Zhou. Duplicate detection for identifying social spam in microblogs. In *BigData Congress: 2013 IEEE International Congress on Big Data* (June 2013), pp. 141–148. DOI: `10.1109/bigdata.congress.2013.27`.

[148] E. Zheleva, H. Sharara, and L. Getoor. Co-evolution of social and affiliation networks. In *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2009), pp. 1007–1016. DOI: `10.1145/1557019.1557128`.

[149] H. Zhuang, A. Chin, S. Wu, W. Wang, X. Wang, and J. Tang. Inferring geographic coincidence in ephemeral social networks. In *Machine Learning and Knowledge Discovery in Databases* (2012), Springer LNCS 7524, pp. 613–628. DOI: `10.1007/978-3-642-33486-3\_39`.