Latent Source Models for Nonparametric Inference

by

George H. Chen

B.S. with dual majors in Electrical Engineering and Computer Sciences,
Engineering Mathematics and Statistics, UC Berkeley, May 2010

S.M. in Electrical Engineering and Computer Science, MIT, June 2012

Electrical Engineer, MIT, June 2014

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology

June 2015

Signature of Author: _____

George H. Chen
Department of Electrical Engineering and Computer Science
May 20, 2015

Certified by: _____

Polina Golland
Associate Professor of Electrical Engineering and Computer Science
Thesis Co-Supervisor

Certified by: _____

Devavrat Shah
Associate Professor of Electrical Engineering and Computer Science
Thesis Co-Supervisor

Accepted by: _____

Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Committee for Graduate Students

Latent Source Models for Nonparametric Inference

by George H. Chen

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Abstract

Nearest-neighbor inference methods have been widely and successfully used in numerous applications such as forecasting which news topics will go viral, recommending products to people in online stores, and delineating objects in images by looking at image patches. However, there is little theoretical understanding of when, why, and how well these nonparametric inference methods work in terms of key problem-specific quantities relevant to practitioners. This thesis bridges the gap between theory and practice for these methods in the three specific case studies of time series classification, online collaborative filtering, and patch-based image segmentation. To do so, for each of these problems, we prescribe a probabilistic model in which the data appear generated from unknown "latent sources" that capture salient structure in the problem. These latent source models naturally lead to nearest-neighbor or nearest-neighbor-like inference methods similar to ones already used in practice. We derive theoretical performance guarantees for these methods, relating inference quality to the amount of training data available and problem-specific structure modeled by the latent sources.

Thesis Co-Supervisor: Polina Golland
Title: Associate Professor of Electrical Engineering and Computer Science

Thesis Co-Supervisor: Devavrat Shah
Title: Associate Professor of Electrical Engineering and Computer Science

Acknowledgments

Five years previous, I embarked on an adventure that would turn out far better than I could've imagined — an adventure full of plot twists, red herrings, disasters, and triumphs culminating in the best years of my life thus far. What ultimately made this rollercoaster experience that is grad school transcend any expectation of mine wasn't really the research or the classes or the free food. No, what strung together grad school for me was my social network that kept me going every step of the way. These few lines are but a feeble attempt at a proper thank you to the incredible people who have helped me in my MIT journey.

Perhaps my grad school experience was peculiar, between switching (sub)fields twice, exploring an armada of rather time-consuming extracurricular activities, and sporadically finding time to actually do research. That I largely don't relate to PHD Comics and that I've had so much freedom to pursue my vichysoisse of random interests, inside and outside of lab, is thanks to my two stellar advisors Polina Golland and Devavrat Shah.

When I started at MIT, I knew nothing about medical image computing. Despite this, Polina offered to take me on as a student months before I moved to the colder climates of Cambridge. Over the years, she provided me with an enormous amount of applied insight, helping me think about how to debug numerous technical hurdles — often by visualizing my way out of despair — and keeping me focused on problems pertinent to practitioners. She's a straight shooter who is more than happy to explain why she disagrees with me on various courses of action but to let me still try what I wanted, whether it turned out to be a huge mistake or whether I ended up pleasantly surprising her by succeeding by the skin of my teeth. Regardless, she always has her students' best interests at heart and has been great to talk to about my interests professional or otherwise.

Whereas Polina supplied me with applied insight, Devavrat supplied me with theoretical insight. After being a teaching assistant for Devavrat and chatting to him about scoring competitive dorm events that I was organizing during my second year at MIT, I decided to ask Devavrat to be a co-advisor to stretch me in a direction that I knew very little about: tackling theoretical problems. Like Polina, undeterred by my lack of experience, Devavrat agreed to co-advise me. In the years that followed, after my usual meetings with him in which I often entered the meeting dejected after futile attempts at cracking a problem, I walked away optimistic, energized, and ready to prove the next lemma. Suffice it to say, Devavrat's enthusiasm is dangerously contagious.

Finding the third member of my thesis committee turned out to be easy. As an undergrad at UC Berkeley, I had run into then-grad-student Sahand Negahban at various social events. Apparently as a Berkeley undergrad, he had been an executive officer in the same student organization I had become heavily involved in (Eta Kappa Nu, the international electrical and computer engineering honor society). Meanwhile, he was also my teaching assistant for Berkeley's probabilistic graphical models class. Little did I know that many years later, he'd postdoc in Devavrat's group, start professing up a storm at Yale, and land on my thesis committee, pointing me to a slew of results from the statistics community that I was unaware of. Along the way, he also introduced me to biking a fixie, which remains my preferred method of commute.

My work in Polina and Devavrat's groups wouldn't have been possible without my various collaborators over the years and the members of both groups who provided feedback on my paper drafts, practice presentations, and incoherent ramblings. Specifically for the work in this thesis, Stan Nikolov ran experimental results on data from Twitter before he ventured off to work there full-time, and Guy Bresler found and resolved a few plot holes in my proofs (oops). Guy was also a source of great debate over whether a probabilistic model reflects reality, and of conversation on cycling, climbing, and skiing. I'm convinced that Guy is the Bear Grylls of MIT. I also thank my officemates and other members of CSAIL and LIDS who had to put up with my spontaneous questions and remarks, chain coffee drinking, and other antics. Adrian Dalca and Ramesh Sridharan were particularly helpful with random day-to-day issues like fighting a regularly malfunctioning file system and other computing quagmires. I also thank my labmate Dhruv Parthasarathy for hosting me in Tel Aviv during the 2014 Israel-Gaza conflict.

To stall research, I taught three semesters and one IAP session, all for classes related to statistical inference. In these endeavors, I had the fortune of teaching for Costis Daskalakis, Greg Wornell, Lizhong Zheng, and my two advisors, and for the IAP course, I co-lectured with Ramesh Sridharan. Teaching has been a tremendously rewarding experience for me, helping students learn and occasionally being the one to introduce that mind-blowing concept that might come handy in the future, whether it's understanding a fundamental result like Bayes' rule or the underlying Markov chain theory behind Google PageRank or the power of approximate inference, for example, in quickly localizing a self-driving car. Of course, teaching is also a great way to learn. Through a myriad of discussions with the rest of the teaching staff, I solidified my understanding of heaps of concepts that I didn't know as well before. Meanwhile, I purchased a tablet so that I could make Khan-Academy-style videos for students, and I thank Roger Grosse, Gauri Joshi, Ramesh Sridharan, and George Tucker for pitching in and making videos as well. Ramesh and I also developed a series of projects during MIT's inaugural offering of undergraduate class "Introduction to Inference", and we continued our collaboration for the IAP course "Statistics for Research Projects". Our efforts led to us jointly winning a departmental teaching award as well as MIT's Goodwin Medal for "conspicuously effective teaching". I want to especially thank the faculty nominators for recognizing our work.

Despite the immense amount of learning in lab and teaching, the most learning in grad school for me happened outside of academics during my year-long tenure as one of five

executive officers in the Sidney-Pacific graduate dorm alongside Jen Jarvis, Pierre-Olivier Lepage, Steve Morgan, and Steph Nam. As the Sidney-Pacific Executive Council 2012-2013, the five of us managed 50 student officers and ran a 700-resident grad dorm with the guidance of housemasters Roger and Dottie Mark, and associate housemasters Annette Kim and Roland Tang — housemasters who let us try and fail and try again, who made sure we knew what to do when we did fail, and who cared deeply about us and our leadership development. Together, we secured $140,000 of funding, programmed copious amounts of events, started a community garden, launched a new website, and overhauled resources throughout the building to better service students. We also weathered severe conflict resolution scenarios, and when the Boston Marathon bombings happened and the city shut down, our officers mobilized to feed the entire dorm and offer support. All of this (and so much more) was student-driven and on a volunteer basis. Easily this year emerged as the best and the most pivotal of my grad school career, seeing the formulation of nearly all my closest friendships here and the inception of many future opportunities.

Among the Sidney-Pacific friendships, Brian Spatocco, who had recruited me for and taught me to run competitive dorm events before he recruited me to become an executive officer for the dorm, reached out to recruit me a third time, asking me to join a rural development project with mutual friends from the dorm Kendall Nowocin and Kelli Xu. Suddenly, the four of us with disparate backgrounds combined forces to analyze satellite images, automatically identifying buildings and villages in rural India and simulating installation of solar-powered grids with the goal of helping bring electricity to those without it. We ended up winning the $10,000 grand prize at the 2014 MIT IDEAS Global Challenge, which we used to visit five companies in India and travel to extremely poor villages and slums. Needless to say, this travel experience was jarringly eye-opening, seeing the conditions of how some of the poorest people in the world live. For our travels, we added a fifth member Vivek Sakhrani to the team, now called GridForm. Then Ramesh Sridharan put me in touch with Kush Varshney who had worked on using satellite images to identify candidate villages to send direct cash transfers to. Shortly afterward, together with Kush Varshney and Brian Abelson, GridForm put forth a journal paper on the amazing potential of satellite image analysis in aiding rural development planning.

I now give special thanks to a few people (some revealed already above!) who have been central to my grad school life. I thank my housemates Steve Morgan, Javier Sanchez-Yamagishi, Brian Spatocco, and Kelli Xu for putting up with my hectic, high-octane schedule this past year and me often not hanging out as a result. I also thank Boris Braverman, Shaiyan Keshvari, Eric Trieu, and Nate Zuk for some incredible conversations about life, the universe, and everything.

It's not easy doing justice to the names already mentioned and to the ones that haven't been of many friends that I would like to thank from the many walks of my life, from those I met at Sidney-Pacific to the musical cyclists at the Biomansion to my buddies in Brain and Cognitive Sciences to my college friends from Eta Kappa Nu to the bartenders at Backbar, and beyond — a social network that has helped see me through to the end of this extraordinary chapter in the meta-thesis that is my life.

I end by thanking my brother and my parents for their support over the years.

Contents

List of Figures

Chapter 1

Introduction

The last two decades have seen an unprecedented explosion in the availability of data pertaining to virtually all avenues of human endeavor. People document their day-to-day lives on social networks. Stores collect information on customers to better recommend products. Hospitals house electronic medical records to assist medical diagnosis. In no small part due to the Internet and increasingly powerful sensors whether it be relatively inexpensive smartphone cameras to far more elaborate devices such as magnetic resonance imaging scanners and the Large Hadron Collider, we now collect and share massive troves of data. How do we turn this deluge of data into valuable insights?

A fundamental problem is that we do not know the rich structure underlying the data a priori. However, by having access to gargantuan volumes of data, practitioners have found that we can often sidestep the question of explicitly positing or learning the structure underlying the data altogether. When we posit intricate structure for data, the structure may stray from reality or otherwise not account for the full palette of possibilities in what the data look like. When we learn structure, the computational overhead and amount of data needed may dwarf what is sufficient for tackling the prediction task we aim to solve. Instead of positing or learning structure, we can instead let the data more directly drive our predictions using so-called nonparametric inference methods.

This thesis showcases three case studies of nonparametric inference. In our first case study, to forecast whether an ongoing news topic will go viral, we compare its activity to those of past news topics that we know have gone viral as well as those that did not. Along similar lines, in online stores or recommendation websites that have both huge user bases and endless items to recommend, we predict whether a user will like an item by looking at what other similar users like. For our final case study, in computer vision, to delineate where an object of interest is in an image, we compare patches of the image to patches in a training database of images for which we know where the object is.

■ 1.1 Nonparametric Inference

The nonparametric inference methods used throughout this thesis are variations of a simple approach called *weighted plurality voting*, which we describe in the context of our first case study. To predict whether the news topic "Barclays scandal" will go viral, we compare its activity to those of past news topics which we know to have gone viral or not. These labeled past news topics are called *training data*. Each of these past news topics

casts a weighted vote, where the weight depends on how similar the past news topic's activity is to that of the news topic "Barclays scandal". Summing over all weighted votes from past news topics that had gone viral results in a total vote for "Barclays scandal" going viral. Similarly, summing over all weighted votes from past news topic that did not go viral results in a total vote for "Barclays scandal" *not* going viral. Finally, weighted plurality voting declares "Barclays scandal" will go viral or not based on which of the two total votes is higher. In this case, there are two outcomes we vote on: whether news topic "Barclays scandal" goes viral or not. With only two outcomes, weighted plurality voting is called weighted *majority* voting. In general when there are more than two outcomes, weighted plurality voting chooses whichever outcome has the highest total vote, which need not be the majority.

We refer to weighted plurality voting as a nearest-neighbor-like method. In our example of forecasting whether the news topic "Barclays scandal" goes viral, the past news topics that are most similar to "Barclays scandal" contribute the most to the final prediction. These most similar training data points could be thought of as "nearest neighbors" to the news topic "Barclays scandal". Naturally, a commonly used variant of weighted plurality voting is *nearest-neighbor classification*, in which we only consider weighted (which also allows for unweighted) votes of the single most similar past news topic or a select few of the most similar past news topics.

Despite their name, nonparametric inference methods, such as weighted plurality voting, are generally not parameter-free. Rather, they make very few assumptions on the underlying model for the data. There could still be parameters that must be chosen. For example, we need to define a measure of how similar two news topics' activities are in how we described weighted plurality voting. In practice, these choices can make substantial differences in inference quality, and in this thesis we shall see how some of these choices relate to relatively weak assumptions on structure in the data.

An impetus for the popularity of nearest-neighbor-like methods lies in their efficiency of computation. Fast approximate nearest-neighbor search algorithms for high-dimensional spaces (e.g., Gionis et al. (1999); Bawa et al. (2005); Ailon and Chazelle (2006); Andoni and Indyk (2006); Datar et al. (2004); Muja and Lowe (2009); Mathy et al. (2015)) can rapidly determine which data points are close to each other, while readily parallelizing across different search queries. These methods often use locality-sensitive hashing (Gionis et al., 1999), which comes with a theoretical guarantee on its approximation accuracy for nearest-neighbor search, or randomized trees (e.g., Bawa et al. (2005); Muja and Lowe (2009); Mathy et al. (2015)), which quickly prune search spaces when the trees are sufficiently balanced. Such trees can be efficiently constructed as data stream in (Mathy et al., 2015). Even the large body of work on random decision forests, which are fast in practice, can be thought of in terms of nearest-neighbor inference (Criminisi et al., 2011). Here, each tree in the forest has leaves that are associated with different training data, and to predict a particular property of a new data point, we determine which leaf the new data point belongs to, effectively finding which training data are nearest to it. Thus, one could interpret random decision forests as learning nearest-neighbor relationships that maximize performance on an inference task such as classification or regression.

While nearest-neighbor-like methods have been widely used in practice, there is little theoretical understanding of when, why, and how well these methods work in terms of the amount of training data available and relevant structural properties in the data. This thesis aims to bridge the gap between theory and practice for nearest-neighbor-like methods. To do so, we present what we call *latent source models* for the three seemingly disparate problems of time series classification, online collaborative filtering, and patch-based image segmentation, corresponding to our three case studies. The recurring theme in how we approach each of these problems turns out to be the same. We begin with a generic model that assumes very little structure. We show how an oracle algorithm can be described by weighted plurality voting. We then approximate the oracle algorithm with a nearest-neighbor-like method akin to what's used in practice, for which we derive theoretical performance guarantees.

Existing theoretical work on nearest-neighbor methods has largely been asymptotic in nature or otherwise studied much more general settings than what we consider. As the amount of training data tends to infinity, nearest-neighbor classification has been shown to achieve a probability of error that is at worst twice the Bayes error rate, and when considering the nearest K neighbors with K allowed to grow with the amount of data, the error rate approaches the Bayes error rate (Cover and Hart, 1967). However, rather than examining the asymptotic case in which the amount of data goes to infinity, we instead pursue nonasymptotic performance guarantees in which no quantities tend to infinity. Nonasymptotic guarantees have been studied for nearest-neighbor methods in fairly general settings in order to obtain rates of convergence for classification (Cover, 1968), regression (Kpotufe, 2011), and density estimation (Dasgupta and Kpotufe, 2014). Our results are far less general and provide tighter guarantees as we instead focus on the three case studies above, each of which exhibits different structure. Guarantees on rates of convergence have been demonstrated previously for very specific settings in terms of the number of nearest neighbors considered (Hall et al., 2008). For simplicity, we will not consider choosing an appropriate number K of nearest neighbors, instead either using just the nearest neighbor ($K = 1$), using all neighbors within some ball, or using all neighbors but weighting them according to a similarity measure between data points.

■ 1.2 Latent Source Models and Theoretical Guarantees

This thesis establishes nonasymptotic performance guarantees for nearest-neighbor-like methods in the three diverse applications in terms of the available training data and structural properties specific to each application. To do so, for each case study, we treat our training data as random i.i.d. samples from an underlying probabilistic model that is intentionally chosen to be simple with few assumptions. Much like how physicists aim to better understand the world by studying simple idealized situations such as merely two particles colliding, our probabilistic modeling also, for each case study, examines simple situations, emphasizing specific structural elements to the application of interest. Each time, we provide empirical justification for why the specific structural elements we assume reasonably model data from the particular application. Of course, there could be

other more elaborate structure present in the data, but as long as the simple structure we consider is present, our theoretical results apply.

We now provide an overview of what the basic problem setups are and what structural assumptions we impose for each of the three case studies, resulting in three probabilistic models. For each case study, we informally state our main theoretical result. Note that we reuse variable names across the case studies, with the same variable name meaning something similar across applications.

Time Series Classification (Chapter 3)

Motivated by the application of forecasting which news topics will go viral on Twitter, we hypothesize that in various time series classification problems, there are not many prototypical time series relative to the number of time series we have access to. For example, we suspect that news topics only go viral on Twitter in a relatively small number k of distinct ways whereas we can collect a massive number $n \gg k$ of Twitter time series corresponding to different news topics. To operationalize this hypothesis, we propose a latent source model for time series, where there are k unknown prototypical time series referred to as latent sources, each of which has label "viral" or "not viral". A new time series is generated by randomly choosing one of these latent sources, adding noise, and then introducing a random time shift. The true unobserved label for the time series is the same as that of whichever latent source the time series is generated from. The goal is to infer what this label is, given the time series observed at time steps $1, 2, \ldots, T$.

We relate weighted majority voting and nearest-neighbor classification to an oracle MAP classifier that knows what the true latent sources are, and we provide theoretical performance guarantees for both nonparametric classifiers. Our guarantees depend on an intuitive "gap" condition that asks that any two training time series of opposite labels be sufficiently separated by squared Euclidean distance $\Omega(\sigma^2 T)$ so as to not confuse classification, where σ is a noise scale parameter. Squared Euclidean distance comes into play because we use a variant of it to define similarity between time series. If this gap condition does not hold, then the closest two training time series of opposite labels are within noise of each other! We informally summarize our theoretical guarantees in the theorem below.

Theorem 1.2.1 (Informal statement of Theorem 3.4.1)**.** *Under the latent source model for time series classification and with the gap condition satisfied, if we have* $n = \Theta(k \log k)$ *labeled training time series, then weighted majority voting and nearest-neighbor classification each correctly classify a time series with high probability after observing its first* $\Omega(\log k)$ *time steps.*

As our analysis accounts for how much of the time series we observe, our results readily apply to the "online" setting in which a time series is to be classified while it streams in, as is the case for forecasting which ongoing news topics go viral, along with the "offline" setting where we have access to the entire time series.

Why not just learn the latent sources? We answer this question by looking at a specific instantiation of our model that leads to a spherical Gaussian mixture model,

with each latent source corresponding to a mixture component. We show that existing performance guarantees on learning spherical Gaussian mixture models require more stringent conditions than what our results require, suggesting that learning the latent sources is a harder problem than the ultimate goal of classification.

Finally, we assess how good our theoretical guarantees are on nearest-neighbor and weighted majority voting time series classification. To do so, we establish a new lower bound on the probability of misclassification for *any* classifier. In other words, no classifier can achieve better misclassification rate than this lower bound. Our performance guarantees on nearest-neighbor and weighted majority voting classification are stated in terms of upper bounds on misclassification rate, and by comparing these upper bounds with our lower bound, we see when the decay rates of misclassification are similar. We also discuss a change in the nonparametric inference methods that could potentially lead to better performance both in theory and practice.

Online Recommendation Systems (Chapter 4)

Our second case study examines online recommendation systems like Netflix and Pandora, where by "online" we mean that items are recommended to users over time. For such systems, a widely used approach is collaborative filtering, which capitalizes on the basic idea that if Alice is similar to Bob and Bob likes oranges, then Alice probably likes oranges too. Despite the prevalence of collaborative filtering methods in recommendation systems, there has been little theoretical development on when, why, and how well they work as a function of system quantities including the number of users n, the number of items m, and how much time has elapsed T. Here, the training data are the ratings revealed so far. At each time step, we make recommendations to users, effectively choosing what the training data look like at the next time step. Thus, recommendations we make now can affect recommendations we make later! From this dynamical process in which the classical tradeoff between exploration and exploitation emerges, we aim to simultaneously learn about users while giving them good item recommendations. To tame this complex problem, we study a variant of a specific commonly used collaborative filtering method and anchor our analysis to a toy model of an online recommendation system.

We focus on cosine-similarity collaborative filtering, which measures how similar users are by the cosine similarity of their revealed item preferences so far. To decide what item to recommend to a user, the method finds nearest neighbors to the user in terms of cosine similarity, and computes votes for items based on revealed item ratings of the nearest neighbors. Weighted plurality voting then selects an item for recommendation. This procedure inherently *exploits* what we currently know about users. Before we have seen enough ratings though, exploitation may not work well. We include two types of exploration, one to explore the space of items, and one to explore the space of users. We call the resulting recommendation algorithm Collaborative-Greedy.

To analyze Collaborative-Greedy, we introduce a simple model of an online recommendation system in which at each time step, for each of the n users, we recommend exactly one item. We assume every user then immediately consumes and rates the item

that we recommended with one of two ratings: $+1$ (like) or -1 (dislike). Once a user consumes an item (e.g., watches a movie), we disallow the item to be recommended to the same user again. Initially, none of the users have consumed any items, and so up to time T, each user will have consumed and rated exactly T items. For simplicity, we do not assume structure over items, so the rating for each item gives us no information about the rating for any other item. We also assume item preferences to be static, i.e., a user's rating for an item does not change over time. We remark that while our setup clearly oversimplifies the rich, complex temporal dynamics in a real online recommendation system, it offers a clean baseline framework for theoretically evaluating recommendation algorithms and, despite its simplicity, highlights the crucial role collaboration plays in recommendation.

To see why collaboration is essential in our model, consider when there's only a single user. Then to learn anything about an item, we have to ask the user to rate the item, but upon doing so, we can't recommend that item again! Moreover, because the rating for an item is assumed to not provide us any information about other items (so we don't know which items are similar), we can't hope to find what the good items are except through exhaustive trial and error. However, with a pool of users, and with structure over users, we should be able to make good recommendations. A complimentary approach beyond the scope of this thesis is to exploit item similarity and recommend similar items to what a user already liked. Theoretical analysis for this alternative setup is in (Voloch, 2015).

For our setup, we impose a simple structure over users by assuming that they come from a relatively small number k of latent sources, where $k \ll n$. This assumption is similar to the structure we imposed on time series in our first case study. We remark that although our model of an online recommendation system is overly simplistic, the assumptions we place on what the actual rating data look like are quite weak as we will assume very little about the different underlying item preferences for the k different latent sources. Under this latent source model for online recommendation systems, we establish a theoretical guarantee for Collaborative-Greedy.

Theorem 1.2.2 (Informal statement of Theorem 4.4.1)**.** *Under the above latent source model, a low noise condition, and a notion of separation between the k user types, with number of users $n = \Theta(km)$, the expected fraction of likable items that* Collaborative-Greedy *recommends is essentially optimal after some initial number of time steps scaling as nearly* $\log(km)$*, where m is the total number of items.*

We supplement our theory with experimental results, where we simulate an online recommendation system and show that Collaborative-Greedy outperforms existing baseline online recommendation methods.

Patch-Based Image Segmentation (Chapter 5)

Lastly, given an image, we look at how to separate a foreground object of interest from the background, a problem referred to in computer vision as image segmentation. Our motivating application is delineating anatomical organs in medical images. Here, nearest-neighbor-like methods are, again, widely used in practice and are often very successful.

Specifically, there has been a proliferation of methods operating on small patches of images. For example, to determine whether a pixel is foreground or background, consider the patch centered at that pixel — is this patch similar to patches we have seen in training data for which we know whether their center pixels are foreground or background? If so, we can transfer labels over from these "nearest-neighbor" patches found in training data.

Naturally, as patches provide only local context, one would expect a local theory to explain when these nearest-neighbor-like patch-based image segmentation methods to work. We show that this is indeed the case. Specifically, we build on existing work that has shown that natural image patches could be very accurately modeled as a Gaussian mixture model (Zoran and Weiss, 2011, 2012) by now asking that patches centered at nearby pixels satisfy a certain local smoothness property that we introduce and that we call a *jigsaw condition*. The basic idea is that nearby patches are like puzzle pieces that need to fit to properly form an overall plausible image. Our probabilistic model, which we call a *latent source model for patch-based image segmentation*, thus has two key components: patches appear to be generated from mixture models, and that nearby mixture models are in some sense similar.

For nearest-neighbor-like methods to succeed when searching for nearby patches, we again ask for a gap condition to hold so that the closest training patches with different labels are well-separated. However, we only need to consider training patches centered at pixels that are close by, where how close by depends on the jigsaw condition. We specifically ask for this gap to be at least squared Euclidean distance $\Omega(\sigma^2 \log(|N|k \log(|\mathcal{I}|)))$, where σ is a noise parameter, k refers to the maximum number of mixture components needed in representing an image patch, $|N|$ is the maximum size of neighboring pixels that relates to the jigsaw condition, and $|\mathcal{I}|$ is the number of pixels. We now informally state our main result.

Theorem 1.2.3 (Informal statement of Theorem 5.1.1)**.** *Under the latent source model for patch-based image segmentation and with the gap condition satisfied, if we have* $n = \Theta(k \log(|\mathcal{I}|k))$ *labeled training images, then weighted majority voting and nearest-neighbor segmentation each achieve average pixel mislabeling rate that can be made arbitrarily small (but in general nonzero).*

The jigsaw condition imposes local structure and enables us to borrow the analysis techniques for the time series classification case. Going beyond this local condition, we introduce global structure to produce a more general probabilistic model for which we derive an iterative, easily parallelizable inference algorithm. Special cases of our new algorithm recover various existing patch-based segmentation algorithms. We empirically show that our new algorithm substantially outperforms the two algorithms we provide theoretical guarantees for. Establishing a theoretical performance guarantee for this new algorithm that exploits global structure is a topic for future research.

1.3 Summary of Results

Qualitatively, our theoretical performance guarantees for all three case studies rely on *separation*. For time series classification, we ask that time series corresponding to different labels be well-separated. For online collaborative filtering, we ask that users have item preferences that are well-separated across different types of users. For patch-based image segmentation, we ask that patches corresponding to different labels be well-separated. This idea of separation arises naturally since a nearest-neighbor method can be confused and make an error when two data points of different types are too close together and the nearest neighbor found is of the wrong type.

Using such a separation (also called a margin) condition is not new and has, for instance, led to fast learning rates for nonparametric classifiers in general settings (Mammen and Tsybakov, 1999; Tsybakov, 2004; Audibert and Tsybakov, 2007) and to results on how many training data are sufficient for learning Gaussian mixture models (Dasgupta and Schulman, 2007). We emphasize that we ground our analysis in three case studies and do not aim for as general a setting as possible since our goal is to justify the performance of nearest-neighbor-like methods resembling what's already used in practice for the three applications of interest. As such, certain structural elements specific to these applications appear in our results. Our analysis could help practitioners for these applications better understand specific structure in data that enable nearest-neighbor methods to succeed, while hinting at new algorithms with potentially better performance guarantees, such as our new iterative patch-based image segmentation algorithm.

1.4 Bibliographic Note

Preliminary versions of the time series classification results (Chapter 3) and online collaborative filtering results (Chapter 4) appeared in (Chen et al., 2013) and (Bresler et al., 2014), respectively. The image segmentation work (Chapter 5) will appear in (Chen et al., 2015).

Chapter 2

Preliminaries

We review relevant probability results in this chapter, most of which are used in analyzing all three case studies. A recurring theme is that data appear to come from different clusters. The nonparametric inference methods we examine work well when there are enough training data from every cluster and, moreover, the inference can distinguish between different clusters. The main machinery ensuring that these two events hold comprises concentration inequalities that provide precise statements for how fast random empirical quantities converge to deterministic population quantities, such as how fast a sample average approaches the population mean that it estimates. In turn, we can then derive rates at which probabilities of various "bad" events go to zero.

We present some concentration inequalities in Section 2.1, and then apply one such inequality to determine how many training data to collect in Section 2.2. To keep the exposition reasonably self-contained as well as to entertain the interested reader, we include proofs for all results in this chapter, relying only on elementary probability, calculus, and big O notation from introductory computer science.

■ 2.1 Concentration Inequalities

Consider a random variable X with finite mean $\mathbb{E}[X]$. With different additional assumptions on X, we can bound the probability that X deviates significantly from $\mathbb{E}[X]$:

- **Markov inequality.** Suppose that X is nonnegative. Then for any $s > 0$,

$$\mathbb{P}(X \geq s) \leq \frac{\mathbb{E}[X]}{s}.$$

  Proof. Fix $s > 0$. Note that $X \geq s\mathbb{1}\{X \geq s\}$, where $\mathbb{1}\{\cdot\}$ is the indicator function: $\mathbb{1}\{\mathcal{A}\} = 1$ if statement $\mathcal{A}$ holds, and $\mathbb{1}\{\mathcal{A}\} = 0$ otherwise. Taking the expectation of both sides, $\mathbb{E}[X] \geq \mathbb{E}[s\mathbb{1}\{X \geq s\}] = s\mathbb{E}[\mathbb{1}\{X \geq s\}] = s\mathbb{P}(X \geq s)$. ■

- **Chebyshev inequality.** Suppose that X has finite variance $\text{var}(X)$. Then for any $s > 0$,

$$\mathbb{P}(|X - \mathbb{E}[X]| \geq s) \leq \frac{\text{var}(X)}{s^2}.$$

Proof. Apply Markov's inequality to $(X - \mathbb{E}[X])^2$, which has expectation $\text{var}(X)$. ■

- **Chernoff bounds.** Let $\phi > 0$. If the expectation $\mathbb{E}[\exp(\phi X)]$ is finite, then for any $s \in \mathbb{R}$, we have the upper tail bound

$$\mathbb{P}(X \geq s) \leq \exp(-\phi s)\mathbb{E}[\exp(\phi X)].$$

Proof. Apply Markov's inequality to $\exp(\phi X)$ with s replaced by $\exp(\phi s)$. ■

The expectation $\mathbb{E}[\exp(\phi X)]$ being finite is a strong assumption on X. In particular, note that the *moment generating function* $M_X(z) \triangleq \mathbb{E}[\exp(zX)]$ of X, if it is finite, tells us all the moments of X: $\mathbb{E}[X^\ell] = \frac{d^\ell M_X}{dz^\ell}(0)$ for all $\ell \in \{0, 1, \dots\}$, from which we could compute, for example, the mean and variance of X. Where the moment generating function $M_X(z)$ of X is finite also tells us how fast a tail of the distribution of X decays. As suggested by the bound above, asking that $M_X(z)$ be finite for $z = \phi > 0$ leads to a bound on the probability of X being large, i.e., the upper tail of the distribution of X.

To bound the probability of X being small, we ask that the moment generating function $M_X(z)$ be finite for $z = -\phi < 0$. Specifically, if $\mathbb{E}[\exp(-\phi X)]$ is finite, then for any $s > 0$, we have the lower tail bound

$$\mathbb{P}(X \leq s) \leq \exp(\phi s)\mathbb{E}[\exp(-\phi X)].$$

Proof. Apply Markov's inequality to $\exp(-\phi X)$ with s replaced by $\exp(-\phi s)$. ■

- **Binomial concentration.** Let X be distributed as a binomial distribution with n trials and probability of success p, which we denote $X \sim \text{Binomial}(n, p)$. Then for $0 < s \leq np$,

$$\mathbb{P}(X \leq s) \leq \exp\Big(-\frac{1}{2np}(np - s)^2\Big).$$

Proof. We use the lower tail Chernoff bound: $\mathbb{P}(X \leq s) \leq \exp(\phi s)\mathbb{E}[\exp(-\phi X)]$. To get a handle on $\mathbb{E}[\exp(-\phi X)]$, first note that the moment generating function of $X \sim \text{Binomial}(n, p)$ is $M_X(z) = \mathbb{E}[\exp(zX)] = (1 + p(e^z - 1))^n$. Then

$$\mathbb{E}[\exp(-\phi X)] = M_X(-\phi) = (1+p(e^{-\phi}-1))^n \leq \big[\exp\big(p(e^{-\phi}-1)\big)\big]^n = \exp(np(e^{-\phi}-1)),$$

where the inequality uses the fact that $1 + x \leq e^x$ for all $x \in \mathbb{R}$. Hence,

$$\mathbb{P}(X \leq s) \leq \exp(\phi s)\exp(np(e^{-\phi} - 1)) = \exp(\phi s + np(e^{-\phi} - 1)).$$

We choose ϕ to make the right-hand side as small as possible. This is a calculus exercise of setting the derivative (with respect to ϕ) of the exponent in the right-hand side to 0. Doing so, we find the optimal choice to be $\phi = \log(\frac{np}{s})$. Substituting this back into the above bound,

$$\mathbb{P}(X \leq s) \leq \exp\left(s \log \frac{np}{s} - np + s\right) = \left(\frac{np}{s}\right)^s \exp(-(np-s)) = \frac{\exp(-(np-s))}{(\frac{s}{np})^{\frac{s}{np} \cdot np}}.$$

Next, noting that $x^x \geq \exp(-(1-x) + (1-x)^2/2)$ for $x \in (0,1)$, and that $\frac{s}{np} \in (0,1)$,

$$\begin{aligned}\mathbb{P}(X \leq s) &\leq \frac{\exp(-(np-s))}{(\frac{s}{np})^{\frac{s}{np} \cdot np}} \leq \frac{\exp(-(np-s))}{(\exp(-(1-\frac{s}{np}) + (1-\frac{s}{np})^2/2))^{np}} \\ &= \exp\left(-\frac{1}{2np}(np-s)^2\right),\end{aligned}$$

where the last step is due to a bit of algebra. ■

- **Sub-Gaussian concentration.** If the distribution of X has tails that decay at least as fast as that of a Gaussian, then it is called sub-Gaussian (Buldygin and Kozachenko, 1980). For example, any Gaussian random variable is sub-Gaussian. Any random variable with finite support, such as a uniform distribution over a finite interval, is also sub-Gaussian. Formally, X is sub-Gaussian with parameter $\sigma > 0$ if for every $\phi \in \mathbb{R}$,

$$\mathbb{E}\big[\exp\left(\phi(X - \mathbb{E}[X])\right)\big] \leq \exp\left(\frac{\phi^2\sigma^2}{2}\right).$$

In other words, the centered random variable $X - \mathbb{E}[X]$ has moment generating function $\mathbb{E}[\exp(\phi(X-\mathbb{E}[X])]$ that is always upper-bounded by the moment generating function of a zero-mean Gaussian with variance σ^2. As discussed previously, the moment generating function relates to how fast the tails of a distribution decay.

Before presenting tail probability bounds for sub-Gaussian random variables, we note an important property that summing two independent sub-Gaussian random variables W_1 and W_2 with parameters σ_1 and σ_2 results in another sub-Gaussian random variable, now with parameter $\sqrt{\sigma_1^2 + \sigma_2^2}$. This property follows from independence and the definition of a sub-Gaussian random variable:

$$\begin{aligned}&\mathbb{E}\big[\exp\left(\phi(W_1 + W_2 - \mathbb{E}[W_1 - W_2])\right)\big] \\ &= \mathbb{E}\big[\exp\left(\phi(W_1 - \mathbb{E}[W_1])\right)\big]\mathbb{E}\big[\exp\left(\phi(W_2 - \mathbb{E}[W_2])\right)\big] \\ &\leq \exp\left(\frac{\phi^2\sigma_1^2}{2}\right)\exp\left(\frac{\phi^2\sigma_2^2}{2}\right) = \exp\left(\frac{\phi^2(\sigma_1^2+\sigma_2^2)}{2}\right).\end{aligned}$$

From this result, one can readily show that if $W_1, \ldots, W_n$ are independent and each is sub-Gaussian with parameter σ, then defining random vector $W = (W_1, \ldots, W_n)$ and letting $a = (a_1, \ldots, a_n) \in \mathbb{R}^n$ be an arbitrary vector of deterministic constants,

the dot product $\langle a, W\rangle = \sum_{u=1}^{n} a_u W_u$ is sub-Gaussian with parameter $\|a\|\sigma$, where $\|\cdot\|$ denotes the Euclidean norm of a vector.

If X is sub-Gaussian with parameter σ, then for any $s > 0$, the following bounds hold:

$$\mathbb{P}(X \geq \mathbb{E}[X] + s) \leq \exp\Big(-\frac{s^2}{2\sigma^2}\Big),$$
$$\mathbb{P}(X \leq \mathbb{E}[X] - s) \leq \exp\Big(-\frac{s^2}{2\sigma^2}\Big),$$
$$\mathbb{P}(|X - \mathbb{E}[X]| \geq s) \leq 2\exp\Big(-\frac{s^2}{2\sigma^2}\Big).$$

Proof. The first two inequalities use Chernoff bounds. Plugging in centered random variable $X - \mathbb{E}[X]$ into the upper tail Chernoff bound, and using the definition of a sub-Gaussian random variable, we obtain

$$\begin{aligned}\mathbb{P}(X \geq \mathbb{E}[X] + s) &= \mathbb{P}(X - \mathbb{E}[X] \geq s)\\ &\leq \exp(-\phi s)\mathbb{E}[\exp(\phi(X - \mathbb{E}[X]))]\\ &\leq \exp(-\phi s)\exp\Big(\frac{\phi^2\sigma^2}{2}\Big) = \exp\Big(-\phi s + \frac{\phi^2\sigma^2}{2}\Big).\end{aligned}$$

We optimize ϕ to make the right-hand side as small as possible, amounting to setting the derivative (with respect to ϕ) of the exponent in the right-hand side to 0. Doing so, we find the optimal choice to be $\phi = s/\sigma^2$, which yields the first inequality

$$\mathbb{P}(X \geq \mathbb{E}[X] + s) \leq \exp\Big(-\frac{s^2}{2\sigma^2}\Big).$$

When X is sub-Gaussian with parameter σ, then so is $-X$, and by similar reasoning as above, one could establish the second inequality

$$\mathbb{P}(X \leq \mathbb{E}[X] - s) \leq \exp\Big(-\frac{s^2}{2\sigma^2}\Big).$$

The last inequality follows from union-bounding the first two:

$$\begin{aligned}\mathbb{P}(|X - \mathbb{E}[X]| \geq s) &= \mathbb{P}\big(\{X \geq \mathbb{E}[X] + s\} \cup \{X \leq \mathbb{E}[X] - s\}\big)\\ &\leq \mathbb{P}(X \geq \mathbb{E}[X] + s) + \mathbb{P}(X \leq \mathbb{E}[X] - s) \leq 2\exp\Big(-\frac{s^2}{2\sigma^2}\Big).\end{aligned}$$ ■

A basic proof technique for analyzing an ensemble of random variables $X_1, \ldots, X_n$ is to first show that the sum $\sum_{u=1}^{n} X_u$ satisfies one of the above conditions, upon which we can bound the probability that it deviates from its expected value $\mathbb{E}[\sum_{u=1}^{n} X_u]$. For example, this approach enables us to examine how fast an empirical average $\frac{1}{n}\sum_{u=1}^{n} X_u$ converges to the true population mean $\mathbb{E}[\frac{1}{n}\sum_{u=1}^{n} X_u]$.

■ 2.2 Collecting Enough Training Data

Suppose there are k clusters that occur with probabilities $\pi_1, \dots, \pi_k$. Thus, when data are generated i.i.d. from these clusters, each data point belongs to cluster $g \in \{1, \dots, k\}$ with probability π_g. How many samples are needed before we see all k clusters?

With equal cluster probabilities $\pi_1 = \cdots = \pi_k = 1/k$, the question reduces to the classical *coupon collector's problem*, where the samples are thought of as coupons of which there are k types. In this case, it turns out that after collecting $\mathcal{O}(k \log k)$ coupons, a coupon collector will have seen all k coupon types with high probability. Of course, in the settings we are interested in, each coupon is a training data point.

We present the solution to the more general problem with potentially unequal cluster probabilities in the form of two lemmas. The first lemma establishes how many samples are sufficient so that we will, with high probability, draw a sample from a *particular* cluster.

Lemma 2.2.1. *With n i.i.d. samples where each sample belongs to cluster $g \in \{1, \dots, k\}$ with probability π_g,*

$$\mathbb{P}\big(\textit{cluster } g \textit{ has } \leq \frac{1}{2} n\pi_g \textit{ occurrences}\big) \leq \exp\Big(-\frac{n\pi_g}{8}\Big).$$

Proof. This result follows from a Chernoff bound applied to the binomial distribution, described in the previous section on concentration inequalities: Note that the g-th cluster occurs $A \sim \text{Binomial}(n, \pi_g)$ times, so

$$\mathbb{P}\big(A \leq \frac{1}{2} n\pi_g\big) \leq \exp\Big(-\frac{1}{2n\pi_g}(n\pi_g - \frac{1}{2}n\pi_g)^2\Big) = \exp\Big(-\frac{n\pi_g}{8}\Big). \qquad \blacksquare$$

This lemma says that with n samples, then with probability at least $1 - \exp(-n\pi_g/8)$, there are more than $n\pi_g/2$ samples from the g-th cluster.

We next ensure that we see *every* cluster with high probability.

Lemma 2.2.2. *With n i.i.d. samples where each sample belongs to cluster $g \in \{1, \dots, k\}$ with probability π_g,*

$$\mathbb{P}\big(\exists \textit{ a cluster with } \leq \frac{1}{2} n\pi_{\min} \textit{ occurrences}\big) \leq k \exp\Big(-\frac{n\pi_{\min}}{8}\Big),$$

where $\pi_{\min} \triangleq \min\{\pi_1, \dots, \pi_k\}$.

Proof. This result follows from a union bound:

$$\begin{aligned}
&\mathbb{P}\Big(\bigcup_{g=1}^{k} \{\text{cluster } g \text{ has } \leq \frac{1}{2} n\pi_{\min} \text{ occurrences}\}\Big) \\
&\leq \sum_{g=1}^{k} \mathbb{P}\Big(\text{cluster } g \text{ has } \leq \frac{1}{2} n\pi_{\min} \text{ occurrences}\Big)
\end{aligned}$$

$$\leq \sum_{g=1}^{k} \mathbb{P}\Big(\text{cluster } g \text{ has } \leq \frac{1}{2} n\pi_g \text{ occurrences}\Big)$$

$$\leq \sum_{g=1}^{k} \exp\Big(-\frac{n\pi_g}{8}\Big) \leq \sum_{g=1}^{k} \exp\Big(-\frac{n\pi_{\min}}{8}\Big) = k \exp\Big(-\frac{n\pi_{\min}}{8}\Big). \qquad \blacksquare$$

Hence, with probability at least $1-k\exp(-n\pi_{\min}/8)$, there are more than $n\pi_{\min}/2$ samples from every cluster. In particular, with any choice of $\delta \in (0,1)$, if we have $n \geq \frac{8}{\pi_{\min}} \log(k/\delta)$ samples, then we will see $n\pi_{\min}/2 > 0$ samples from every cluster with probability at least $1-\delta$. Returning to the coupon collector's problem where $\pi_{\min} = 1/k$, we see that collecting $n = \lceil 8k \log(k/\delta) \rceil = \mathcal{O}(k \log k)$ coupons is sufficient to see all k types of coupons with high probability.

Chapter 3

Time Series Classification

A key task in mining temporal data is being able to identify anomalous events, often before they happen. As a running example used throughout this chapter, consider a time series that tracks how much activity there is for a particular news topic on Twitter. Given the time series up to present time, will this news topic go viral? Borrowing Twitter's terminology, we label the time series a "trend" and call its corresponding news topic a "trending topic" if the news topic goes viral. Otherwise, the time series is labeled "not trend". We seek to forecast whether a news topic will become a trend *before* it is declared a trend (or not) by Twitter, amounting to what's called a binary time series classification problem: can we correctly classify a time series as one of two labels?

Importantly, we skirt the discussion of what makes a topic considered trending. While it is not public knowledge how Twitter defines a topic to be a trending topic, Twitter does provide information for which topics are trending topics. We take these labels to be ground truth, effectively treating how a topic goes viral to be a black box supplied by Twitter. Thus, our goal of forecasting trending news topics could more generally be described as predicting whether a news topic will be a trend before it is declared to be a trend by any third party that we can collect ground truth labels from. Existing work that identify trends on Twitter (Cataldi et al., 2010; Mathioudakis and Koudas, 2010; Becker et al., 2011) instead, as part of their trend detection, define models for what trends are. The same could be said of previous work on novel document detection (Kasiviswanathan et al., 2011, 2012). In contrast, we neither define a model for what trends are nor do we assume we know such a definition. Our only way of indirectly probing a definition is through collecting ground truth labels of news topics from a third party like Twitter.

For this problem of forecasting news trends on Twitter, Nikolov (2012) has shown that weighted majority voting can predict whether a news topic will be a trend in advance of Twitter 79% of the time, with a mean early advantage of 1 hour and 26 minutes, a true positive rate of 95%, and a false positive rate of 4%. We summarize this result later in this chapter.

The success of nearest-neighbor or nearest-neighbor-like methods in time series classification is hardly limited to forecasting which news topics will go viral. Such methods have also been used, for example, to detect abnormal brain activity in EEG recordings (Chaovalitwongse et al., 2007), classify protein sequences (Kaján et al., 2006), and predict whether a cell phone subscriber will switch providers (Lee et al., 2012). In fact, while numerous standard classification methods have been tailored to classify time series, a

simple nearest-neighbor approach has been found to be hard to beat in terms of classification performance on a variety of datasets (Xi et al., 2006), with results competitive to or better than various other more elaborate methods such as neural networks (Nanopoulos et al., 2001), decision trees (Rodríguez and Alonso, 2004), and support vector machines (Wu and Chang, 2004). More recently, researchers have examined which distance to use with nearest-neighbor classification (Batista et al., 2011; Ding et al., 2008; Weinberger and Saul, 2009) or how to boost classification performance by applying different transformations to the time series before using nearest-neighbor classification (Bagnall et al., 2012). These existing results are mostly experimental, lacking theoretical justification for both when nearest-neighbor-like time series classifiers should be expected to perform well and how well.

In this chapter, we establish theoretical performance guarantees for nearest-neighbor and weighted majority voting time series classification in terms of available training data. To focus the exposition on high-level ideas and intuition, we defer all proofs to Section 3.7. We begin in Section 3.1 by precisely stating the nonparametric inference methods we shall analyze. As we view data as random, our theoretical guarantees rely on an underlying probabilistic model for time series. We present such a model in Section 3.2, which we call a *latent source model for time series classification*. Our model is guided by the hypothesis that there are only a small number of unknown latent sources from which time series are generated.

Turning toward analysis, in Section 3.3, we show how weighted majority voting approximates an oracle maximum a posteriori (MAP) classifier that knows the latent sources. Naturally, we can't hope to outperform this oracle MAP classifier. However, if the number of training data grows large, we would expect weighted majority voting to better approximate the oracle classifier. With this high-level intuition in mind, we then present our main theoretical result in Section 3.4, stating nonasymptotic performance guarantees for nearest-neighbor and weighted majority voting time series classification. Note that neither method actually estimates what the latent sources are.

To compare our work to existing results that can learn the latent sources, and also to examine how far our performance guarantees are from what is optimal, in Sections 3.5 and 3.6, we consider a specific instantiation of our model that makes it into a spherical Gaussian mixture model. In this special case, the latent sources are precisely the mixture model components, so we can compare our guarantees to existing guarantees for algorithms that learn spherical Gaussian mixture models. As we show in Section 3.5, the theoretical guarantees for these existing learning algorithms either require more stringent conditions on the data or require more training data than what we show is sufficient for time series classification. Then in Section 3.5, we establish a lower bound on the misclassification rate for *any* classifier and compare this bound with our upper bounds on the misclassification rates of nearest-neighbor and weighted majority voting classification.

Finally, we present experimental results in Section 3.8. Using synthetic data, we find weighted majority voting to outperform nearest-neighbor classification early on when we observe very little of the time series to be classified. This suggests weighted majority voting to be better suited for forecasting anomalous events than nearest-neighbor clas-

sification. Returning to our running example of forecasting news trends on Twitter, we summarize the weighted majority voting results by Nikolov (2012) and show that the real time series data in this case do exhibit clustering behavior. The cluster centers could be thought of as the underlying latent sources.

■ 3.1 Nonparametric Time Series Classifiers

We represent each time series as a function mapping $\mathbb{Z}$ to $\mathbb{R}$. Given time series Y observed at time steps $1, 2, \ldots, T$, we want to classify it as having either label $+1$ ("trend") or -1 ("not trend"). To do so, we have access to labeled training data consisting of time series $Y_1, \ldots, Y_n$ with corresponding labels $L_1, \ldots, L_n \in \{+1, -1\}$. We now present three simple nonparametric inference methods for time series classification.

Nearest-neighbor classifier. Let $\widehat{u} = \operatorname{argmin}_{u \in \{1,\ldots,n\}} \mathbf{d}^{(T)}(Y_u, Y)$ be the index of the nearest training data point to Y, where $\mathbf{d}^{(T)}(Y_u, Y)$ is some "distance" between time series Y_u and Y, superscript (T) indicates that we are only allowed to look at the first T time steps (i.e., time steps $1, 2, \ldots, T$) of Y (but we're allowed to look outside of these time steps for each training time series Y_u for $u \in \{1, \ldots, n\}$). Then we declare the label for Y to be the same as that of the $\widehat{u}$-th (i.e., the nearest) training data point:

$$\widehat{L}^{(T)}_{\mathrm{NN}}(Y) = L_{\widehat{u}}. \tag{3.1}$$

Distance $\mathbf{d}^{(T)}(\cdot, \cdot)$ could, for example, be squared Euclidean distance: $\mathbf{d}^{(T)}(Y_u, Y) = \sum_{t=1}^{T} (Y_u(t) - Y(t))^2 \triangleq \|Y_u - Y\|_T^2$. However, this distance only looks at the first T time steps of training time series Y_u. Since time series in our training data are fully known, we need not restrict our attention to their first T time steps. Thus, we instead use the following distance function throughout the rest of this chapter:

$$\mathbf{d}^{(T)}(Y_u, Y) = \min_{\Delta \in \{-\Delta_{\max}, \ldots, 0, \ldots, \Delta_{\max}\}} \sum_{t=1}^{T} (Y_u(t+\Delta) - Y(t))^2 = \min_{\Delta \in \{-\Delta_{\max}, \ldots, 0, \ldots, \Delta_{\max}\}} \|Y_u \circledast \Delta - Y\|_T^2,$$

where we minimize over integer time shifts with a pre-specified maximum allowed shift $\Delta_{\max} \geq 0$, and we use $A \circledast \Delta$ to denote time series A advanced by Δ time steps, i.e., $(A \circledast \Delta)(t) = A(t + \Delta)$.

Weighted majority voting. Each training time series Y_u casts a weighted vote $e^{-\theta \mathbf{d}^{(T)}(Y_u, Y)}$ for whether time series Y has label L_u, where constant $\theta \geq 0$ is a scaling parameter that determines the "sphere of influence" of each training data point. The total votes for label $+1$ and -1 are given by

$$V_{+1}(Y; \theta) = \sum_{u=1}^{n} \exp\big(-\theta \mathbf{d}^{(T)}(Y_u, Y)\big) \mathbb{1}\{L_u = +1\},$$

$$V_{-1}(Y; \theta) = \sum_{u=1}^{n} \exp\big(-\theta \mathbf{d}^{(T)}(Y_u, Y)\big) \mathbb{1}\{L_u = -1\}.$$

The label with the majority of overall weighted votes is declared as the label for Y:

$$\widehat{L}^{(T)}_{\mathrm{WMV}}(Y;\theta) = \begin{cases} +1 & \text{if } V_{+1}(Y;\theta) \geq V_{-1}(Y;\theta), \\ -1 & \text{otherwise.} \end{cases} \tag{3.2}$$

Using a larger time window size T corresponds to waiting longer before we make a prediction. We need to trade off how long we wait and how accurate we want our prediction. When $\theta \to \infty$, we obtain nearest-neighbor classification.

Generalized weighted majority voting. Lastly, applications may call for trading off true and false positive rates. One way to do this is to bias the prediction toward one of the labels, such as scaling the total vote for label -1 by factor $\tau > 0$. The resulting decision rule, which we refer to as *generalized weighted majority voting*, is thus:

$$\widehat{L}^{(T)}_{\tau}(Y;\theta) = \begin{cases} +1 & \text{if } V_{+1}(Y;\theta) \geq \tau V_{-1}(Y;\theta), \\ -1 & \text{otherwise,} \end{cases} \tag{3.3}$$

where setting $\tau = 1$ recovers the usual weighted majority voting (3.2). This modification to the classifier can be thought of as adjusting the priors on the relative sizes of the two classes. Our theoretical results to follow actually cover this more general case rather than only that of $\tau = 1$.

■ 3.2 A Latent Source Model

We assume there to be k distinct prototypical time series $\mu_1, \ldots, \mu_k$ with corresponding labels $\lambda_1, \ldots, \lambda_k \in \{+1, -1\}$ that are not all the same. These labeled time series occur with strictly positive probabilities $\pi_1, \ldots, \pi_k$ and are referred to as latent sources. For inference, we will not know what the prototypical time series are, how many there are of them, or what probabilities they occur with. A new time series is generated as follows:

1. Sample latent source index $G \in \{1, \ldots, k\}$ so that $G = g$ with probability π_g.
2. Sample integer time shift Δ uniformly from $\{0, 1, \ldots, \Delta_{\max}\}$. (For a technical reason, we restrict to nonnegative shifts here, whereas in our distance function $\mathbf{d}^{(T)}$ from before, we allow for negative shifts as well. As we discuss in Section 3.7.1, the generative model could indeed allow for shifts from $\{-\Delta_{\max}, \ldots, \Delta_{\max}\}$, in which case for our proof techniques to work, the distance function we use needs to look at shifts up to magnitude $2\Delta_{\max}$.)
3. Output time series Y to be latent source μ_G advanced by Δ time steps, followed by adding noise time series W, i.e., $Y(t) = \mu_G(t + \Delta) + W(t)$. Entries of noise W are i.i.d. zero-mean sub-Gaussian with parameter σ. The true label for Y is assigned to be $L = \lambda_G$.

The above generative process defines our latent source model for time series classification. Importantly, we make no assumptions about what the latent sources actually are aside from them being distinct and that there is at least one latent source per label.

■ 3.3 Approximating an Oracle Classifier

If we knew the latent sources and if noise entries $W(t)$ were i.i.d. $\mathcal{N}(0, \frac{1}{2\theta})$ across time indices t, then the *maximum a posteriori* (MAP) estimate for label L given the first T time steps of time series Y is

$$\widehat{L}^{(T)}_{\mathrm{MAP}}(Y;\theta) = \begin{cases} +1 & \text{if } R^{(T)}_{\mathrm{MAP}}(Y;\theta) \geq 1, \\ -1 & \text{otherwise,} \end{cases} \tag{3.4}$$

where

$$R^{(T)}_{\mathrm{MAP}}(Y;\theta) \triangleq \frac{\sum_{g\in\{1,\ldots,k\} \text{ s.t. } \lambda_g=+1} \pi_g \sum_{\Delta\in\mathcal{D}_+} \exp\left(-\theta\|\mu_g \circledast \Delta - Y\|^2_T\right)}{\sum_{g\in\{1,\ldots,k\} \text{ s.t. } \lambda_g=-1} \pi_g \sum_{\Delta\in\mathcal{D}_+} \exp\left(-\theta\|\mu_g \circledast \Delta - Y\|^2_T\right)}, \tag{3.5}$$

and $\mathcal{D}_+ \triangleq \{0,\ldots,\Delta_{\max}\}$. Note that in the ratio above, the numerator is a sum of weighted votes for label $+1$, and the denominator is a sum of weighted votes for label -1.

However, we do not know the latent sources, nor do we know if the noise is i.i.d. Gaussian. We assume that we have access to n training data sampled i.i.d. from the latent source model, where we have access to all time steps of each training time series, as well as every training time series' label. Denote $\mathcal{D} \triangleq \{-\Delta_{\max},\ldots,0,\ldots,\Delta_{\max}\}$. Then we approximate the MAP classifier by using training data as proxies for the latent sources. Specifically, we take ratio (3.5), replace the inner sum by a minimum in the exponent, replace each latent source time series with training time series, drop the proportions $\pi_1,\ldots,\pi_k$ that we do not know, and replace $\mathcal{D}_+$ by $\mathcal{D}$ to obtain the ratio

$$R^{(T)}(Y;\theta) \triangleq \frac{\sum_{u\in\{1,\ldots,n\} \text{ s.t. } L_u=+1} \exp\left(-\theta\left(\min_{\Delta\in\mathcal{D}} \|Y_u \circledast \Delta - Y\|^2_T\right)\right)}{\sum_{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1} \exp\left(-\theta\left(\min_{\Delta\in\mathcal{D}} \|Y_u \circledast \Delta - Y\|^2_T\right)\right)} = \frac{V_{+1}(Y;\theta)}{V_{-1}(Y;\theta)}. \tag{3.6}$$

Plugging $R^{(T)}$ in place of $R^{(T)}_{\mathrm{MAP}}$ in classification rule (3.4) yields weighted majority voting (3.2), which as we have already discussed becomes nearest-neighbor classification when $\theta \to \infty$. That nearest-neighbor classification and weighted majority voting approximate the oracle MAP classifier suggest that they should perform better when this approximation improves, which should happen with more training data and not too much noise as to muddle where the true decision boundaries are between labels $+1$ and -1. Also, note that measuring similarity between time series in terms of squared Euclidean distances naturally comes out of using Gaussian noise. Squared Euclidean distance will thus appear in how we define the separation gap between time series of opposite labels. Using different noise model would change this gap condition.

As a technical remark, if we didn't replace the summations over time shifts with minimums in the exponent, then we have a kernel density estimate in the numerator and in the denominator (Fukunaga, 1990, Chapter 7), where the kernel is Gaussian, and our main theoretical result for weighted majority voting to follow would still hold using the same proof. We use a minimum rather a summation over time shifts to make the method more similar to existing time series classification work (e.g., Xi et al. (2006)), which minimize over nonlinear time warpings rather than simple shifts.

■ 3.4 Theoretical Performance Guarantees

We now present our main theoretical result of this chapter which provides performance guarantees for nearest-neighbor classification (3.1) and generalized weighted majority voting (3.3), accounting for the number of training data n and the number of time steps T that we observe of the time series to be classified. This result depends on the following separation gap, which depends on time horizon T:

$$\mathcal{G}^{(T)} \triangleq \min_{\substack{u,v\in\{1,\dots,n\} \text{ s.t. } L_u\neq L_v,\\ \Delta,\Delta'\in\mathcal{D}}} \|Y_u \circledast \Delta - Y_v \circledast \Delta'\|_T^2.$$

This quantity measures how far apart the two different label classes $+1$ and -1 are if we only look at length-T chunks of each time series and allow all shifts of at most $\Delta_{\max}$ time steps in either direction. We're now in a position to state this chapter's main result.

Theorem 3.4.1. *Let* $\pi_{\min} \triangleq \min\{\pi_1,\dots,\pi_k\}$, $\pi_{+1} \triangleq \mathbb{P}(L=+1) = \sum_{g=1}^k \pi_g \mathbb{1}\{\lambda_g = +1\}$, *and* $\pi_{-1} \triangleq \mathbb{P}(L=-1) = \sum_{g=1}^k \pi_g \mathbb{1}\{\lambda_g = -1\}$. *Under the latent source model for time series classification with n training data points:*

(a) The probability that nearest-neighbor classification misclassifies time series Y with label L satisfies the bound

$$\mathbb{P}(\widehat{L}_{NN}^{(T)}(Y) \neq L) \leq k \exp\Big(-\frac{n\pi_{\min}}{8}\Big) + (2\Delta_{\max}+1)n \exp\Big(-\frac{1}{16\sigma^2}\mathcal{G}^{(T)}\Big).$$

(b) The probability that generalized weighted majority voting (with parameter θ) misclassifies time series Y with label L satisfies the bound

$$\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) \neq L)$$
$$\leq k \exp\Big(-\frac{n\pi_{\min}}{8}\Big) + \Big(\tau\pi_{+1} + \frac{1}{\tau}\pi_{-1}\Big)(2\Delta_{\max}+1)n \exp\big(-(\theta - 4\sigma^2\theta^2)\mathcal{G}^{(T)}\big).$$

To interpret this theorem, first note that the two upper bounds actually match when, for generalized weighted majority voting, we choose $\tau = 1$ (corresponding to regular weighted majority voting) and $\theta = \frac{1}{8\sigma^2}$. This suggests the two methods to have similar behavior when the gap grows with the number of time steps T that we see of time series Y. Thus, we shall just examine the upper bound for nearest-neighbor classification.

We consider sufficient conditions that ensure that the nearest-neighbor classification upper bound is at most $\delta \in (0,1)$. Specifically, we look at when each of the two terms on the right-hand side is at most $\delta/2$. For the first term, the number of training data n should be sufficiently large so that we see all the different latent sources: $n \geq \frac{8}{\pi_{\min}}\log(2k/\delta)$. For the second term, the gap $\mathcal{G}^{(T)}$ should be sufficiently large so that the nearest training time series found has the correct label: $\mathcal{G}^{(T)} \geq 16\sigma^2 \log\big(2(2\Delta_{\max}+1)n/\delta\big)$. There are different ways to change the gap, such as increasing how many time steps T we get to observe of time series Y, and changing what quantity the time series are tracking.

The linear dependence in the second term for both upper bounds in the Theorem result from a worst-case analysis in which only one training time series comes from the same latent source as the time series to be classified, and the other $n-1 \leq n$ training time series have the wrong label. If we have some estimates or bounds on k, $\pi_{\min}$, $\Delta_{\max}$, and σ^2, then one way to prevent the linear scaling with n is to randomly subsample our training data. Specifically, if we have access to a large enough pool of labeled time series, i.e., the pool has $\Omega(\frac{8}{\pi_{\min}} \log \frac{k}{\delta})$ time series, then we can subsample $n = \Theta(\frac{8}{\pi_{\min}} \log \frac{k}{\delta})$ of them to use as training data, in which case nearest-neighbor classification (3.1) correctly classifies a new time series Y with probability at least $1-\delta$ if the gap grows as

$$\mathcal{G}^{(T)} = \Omega\left(\sigma^2\left(\log(2\Delta_{\max}+1) + \log\left(\frac{1}{\delta\pi_{\min}}\log\frac{k}{\delta}\right)\right)\right).$$

For example, consider when the latent sources occur with equal probability, so $\pi_{\min} = 1/k$. Then so long as the gap grows as

$$\mathcal{G}^{(T)} = \Omega\left(\sigma^2\left(\log(2\Delta_{\max}+1) + \log\left(\frac{k}{\delta}\log\frac{k}{\delta}\right)\right)\right) = \Omega\left(\sigma^2\left(\log(2\Delta_{\max}+1) + \log\frac{k}{\delta}\right)\right),$$

i.e., logarithmic in the number of latent sources k, then nearest-neighbor classification is correct with probability at least $1-\delta$. If, furthermore, the gap grows as $\Omega(\sigma^2 T)$ (which is a reasonable growth rate since otherwise, the closest two training time series of opposite labels are within noise of each other), then observing the first $T = \Omega(\log(2\Delta_{\max}+1) + \log\frac{k}{\delta})$ time steps from the time series is sufficient to classify it correctly with probability at least $1-\delta$. This corresponds to the informal statement of Theorem 1.2.1.

Although our performance guarantees for the two nonparametric methods are comparable, in our experimental results in Section 3.8, we find weighted majority voting to outperform nearest-neighbor classification when T is small, and then as T grows large, the two methods exhibit similar performance in agreement with our theoretical analysis. For small T, it could still be fairly likely that the nearest neighbor found has the wrong label, dooming the nearest-neighbor classifier to failure. Weighted majority voting, on the other hand, can recover from this situation as there may be enough correctly labeled training time series close by that contribute to a higher overall vote for the correct class. This robustness of weighted majority voting makes it favorable in the online setting where we want to make a prediction as early as possible.

■ 3.5 Learning the Latent Sources

If we can estimate the latent sources accurately, then we could plug these estimates in place of the true latent sources in the MAP classifier and achieve classification performance close to optimal. If we restrict the noise to be Gaussian and assume $\Delta_{\max} = 0$, then the latent source model corresponds to a spherical Gaussian mixture model. To simplify discussion in this section, we assume latent sources to occur with equal probability $1/k$. We could learn a spherical Gaussian mixture model using Dasgupta and

Schulman's modified EM algorithm (Dasgupta and Schulman, 2007). Their theoretical guarantee depends on the true separation between the closest two latent sources, namely

$$\mathcal{G}^{(T)*} \triangleq \min_{g,h\in\{1,\dots,k\} \text{ s.t. } g\neq h} \|\mu_g - \mu_h\|_T^2,$$

which needs to satisfy $\mathcal{G}^{(T)*} \gg \sigma^2\sqrt{T}$. Then with number of training time series $n = \Omega(\max\{1, \frac{\sigma^2 T}{\mathcal{G}^{(T)*}}\}k \log \frac{k}{\delta})$, gap $\mathcal{G}^{(T)*} = \Omega(\sigma^2 \log \frac{k}{\varepsilon})$, and number of initial time steps observed

$$T = \Omega\left(\max\left\{1, \frac{\sigma^4 T^2}{(\mathcal{G}^{(T)*})^2}\right\} \log\left[\frac{k}{\delta} \max\left\{1, \frac{\sigma^4 T^2}{(\mathcal{G}^{(T)*})^2}\right\}\right]\right),$$

their algorithm achieves, with probability at least $1-\delta$, an additive $\varepsilon\sigma\sqrt{T}$ error (in Euclidean distance) close to optimal in estimating every latent source. In contrast, our result is in terms of gap $\mathcal{G}^{(T)}$ that depends not on the true separation between two latent sources but instead on the minimum observed separation in the training data between two time series of opposite labels. In fact, our gap, in their setting, grows as $\Omega(\sigma^2 T)$ even when their gap $\mathcal{G}^{(T)*}$ grows sublinear in T.

In particular, while their result cannot handle the regime where $\mathcal{O}(\sigma^2 \log \frac{k}{\delta}) \leq \mathcal{G}^{(T)*} \leq \sigma^2\sqrt{T}$, ours can, using $n = \Theta(k \log \frac{k}{\delta})$ training time series and observing the first $T = \Omega(\log \frac{k}{\delta})$ time steps to classify a time series correctly with probability at least $1-\delta$. This follows from our theorem below, which specializes Theorem 3.4.1 to the Gaussian setting with no time shifts and uses gap $\mathcal{G}^{(T)*}$ instead of $\mathcal{G}^{(T)}$. We also present an accompanying corollary to interpret the theorem. Both our theorem and corollary to follow still hold if gap $\mathcal{G}^{(T)*}$ were instead replaced by the squared Euclidean distance between the closest two latent sources with opposite labels:

$$\mathcal{G}^{(T)\dagger} \triangleq \min_{g,h\in\{1,\dots,k\} \text{ s.t. } \lambda_g\neq\lambda_h} \|\mu_g - \mu_h\|_T^2.$$

Intuitively, if latent sources of the same label are extremely close by (so that $\mathcal{G}^{(T)*}$ is small) yet latent sources of opposite labels are far away (so that $\mathcal{G}^{(T)\dagger}$ is large), then we should expect the classification problem to be relatively easy compared to learning the latent sources because the latter still needs to tease apart the different latent sources that are extremely close by.

Theorem 3.5.1. *Let $s_1 > 0$ and $s_2 > 0$. Suppose that $\mathcal{G}^{(T)*} \geq s_1^2$. Under the latent source model for time series with Gaussian noise, no time shifts, and n training data points:*

(a) The probability that nearest-neighbor classification misclassifies time series Y with label L satisfies the bound

$$\mathbb{P}(\widehat{L}_{NN}^{(T)}(Y) \neq L) \leq k \exp\left(-\frac{n\pi_{\min}}{8}\right) + \frac{n^2}{4}\exp\left(-\frac{s_1^2}{4\sigma^2}\right) + \frac{n^2}{4}\exp(-s_2) + n\exp\left(-\frac{1}{16\sigma^2}\left(\mathcal{G}^{(T)*} - 2s_1\sqrt{\mathcal{G}^{(T)*}} + 2\sigma^2 T - 4\sigma^2\sqrt{T s_2}\right)\right)$$

(b) The probability that generalized weighted majority voting misclassifies time series Y with label L satisfies the bound

$$\begin{aligned}&\mathbb{P}(\widehat{L}_\tau^{(T)}(Y;\theta)\neq L)\\&\leq k\exp\Big(-\frac{n\pi_{\min}}{8}\Big)+\frac{n^2}{4}\exp\Big(-\frac{s_1^2}{4\sigma^2}\Big)+\frac{n^2}{4}\exp(-s_2)\\&\quad+\Big(\tau\pi_{+1}+\frac{1}{\tau}\pi_{-1}\Big)n\exp\Big(-(\theta-4\sigma^2\theta)^2\big(\mathcal{G}^{(T)*}-2s_1\sqrt{\mathcal{G}^{(T)*}}+2\sigma^2T-4\sigma^2\sqrt{Ts_2}\big)\Big).\end{aligned}$$

Both of these guarantees still hold with $\mathcal{G}^{(T)*}$ replaced by $\mathcal{G}^{(T)\dagger}$.

As with Theorem 3.4.1, the two upper bounds are comparable and can be made to match by choosing $\tau = 1$ and $\theta = \frac{1}{8\sigma^2}$ for generalized weighted majority voting. To interpret this theorem, we choose specific values for s_1 and s_2 and consider the worst case tolerated by the theorem in which $\mathcal{G}^{(T)*} = s_1^2$, arriving at the following corollary.

Corollary 3.5.1. *Let $\delta \in (0,1)$. Under the latent source model for time series classification with Gaussian noise, no time shifts, and $n \geq \frac{8}{\pi_{\min}}\log\frac{4k}{\delta}$ training data points, if*

$$\mathcal{G}^{(T)*} \geq 4\sigma^2\log\frac{n^2}{\delta},$$
$$T \geq 4\log\frac{n^2}{\delta}+8\log\frac{4n}{\delta}+2\sqrt{\Big(3\log\frac{n^2}{\delta}+8\log\frac{4n}{\delta}\Big)\log\frac{n^2}{\delta}},$$

then nearest-neighbor classification and generalized weighted majority voting (with $\tau = 1, \theta = \frac{1}{8\sigma^2}$) each classify a new time series correctly with probability at least $1-\delta$. This statement still holds if $\mathcal{G}^{(T)*}$ is replaced by $\mathcal{G}^{(T)\dagger}$.

Hence, when the latent sources occur with equal probability (so $\pi_{\min} = 1/k$), then with $n = \Theta(\sigma^2\log\frac{k}{\delta})$ training data and so long as the gap grows as $G^{(T)*} = \Omega(\sigma^2\log\frac{k}{\delta})$, after observing $T = \Omega(\log\frac{k}{\delta})$ time steps of Y, the two nonparametric inference algorithms correctly classify Y with probability at least $1-\delta$.

Vempala and Wang (2004) have a spectral method for learning Gaussian mixture models that can handle smaller $\mathcal{G}^{(T)*}$ than Dasgupta and Schulman's approach but requires $n = \widetilde{\Omega}(T^3k^2)$ training data, where we've hidden the dependence on σ^2 and other variables of interest for clarity of presentation. Hsu and Kakade (2013) have a moment-based estimator that doesn't use a gap condition but, under a different non-degeneracy condition, requires substantially more samples for our problem setup, i.e., $n = \Omega((k^{14}+Tk^{11})/\varepsilon^2)$ to achieve an ε approximation of the mixture components. These results need substantially more training data than what we've shown is sufficient for classification.

To fit a Gaussian mixture model to massive training datasets, in practice, using all the training data could be prohibitively expensive. In such scenarios, one could instead nonuniformly subsample $\mathcal{O}(Tk^3/\varepsilon^2)$ time series from the training data using the procedure given in (Feldman et al., 2011) and then feed the resulting smaller dataset, referred to as a (k,ε)-*coreset*, to the EM algorithm for learning the latent sources. This procedure still requires more training time series than needed for classification and lacks a guarantee that the estimated latent sources will be close to the true latent sources.

3.6 Lower Bound on Misclassification Rate

To understand how good our theoretical performance guarantees for nearest-neighbor and weighted majority voting time series classification are, we establish a lower bound on the misclassification rate for *any* classifier under the Gaussian noise setting with no time shifts as in the previous section. This lower bound depends on the true gap between latent sources of opposite labels, namely $\mathcal{G}^{(T)\dagger}$.

Theorem 3.6.1. *Under the latent source model for time series classification with Gaussian noise and no time shifts, the probability of misclassifying time series using any classifier satisfies the bound*

$$\mathbb{P}(\textit{misclassify}) \geq \pi_{+1}\pi_{-1}\pi_{\min}^2 \exp\left(-\frac{1}{2\sigma^2}\mathcal{G}^{(T)\dagger} - \frac{\Gamma((T+1)/2)}{\sigma\Gamma(T/2)}\sqrt{2\mathcal{G}^{(T)\dagger}} - \frac{T}{2}\right),$$

where Γ is the Gamma function: $\Gamma(z) \triangleq \int_0^\infty x^{z-1}e^{-x}dx$ defined for $z > 0$. Note that $\frac{\Gamma((T+1)/2)}{\Gamma(T/2)}$ grows sublinear in T.

We can compare this result to the misclassification rate upper bound of nearest-neighbor classification in Theorem 3.5.1. While this upper bound does not match the lower bound, its fourth and final term decays exponentially with gap $\mathcal{G}^{(T)\dagger}$ as well as time horizon T, similar to the only term in our lower bound. The other three terms in the upper bound could be made arbitrarily small but in doing so slows down how fast the fourth term decays. We suspect our upper bound to be loose as our analysis is worst-case. Even so, it's possible that nearest-neighbor and weighted majority voting simply aren't optimal.

3.7 Proofs

We now present the proofs of Theorems 3.4.1, 3.5.1, and 3.6.1, along with Corollary 3.5.1. Since we look at time series in length-T windows, we equivalently view these windows as T-dimensional vectors.

3.7.1 Proof of Theorem 3.4.1

Let $\mathcal{E}_{\text{all-sources}}$ denote the event that the training data consists of more than $n\pi_{\min}/2$ points from every latent source. By Lemma 2.2.2, this event happens with probability at least $1 - k\exp(-n\pi_{\min}/8)$. For the remainder of the proof, we assume event $\mathcal{E}_{\text{all-sources}}$ to hold.

Under the latent source model, the observed time series Y is generated from some latent source λ_{g^*} for $g^* \in \{1, \ldots, k\}$, with time shift $\Delta_1 \in \mathcal{D}_+ = \{0, 1, \ldots, \Delta_{\max}\}$ and noise signal W_1:

$$Y = \mu_{g^*} \circledast \Delta_1 + W_1.$$

Since event $\mathcal{E}_{\text{all-sources}}$ holds, there exists some training time series Y_{u^*}, for $u^* \in \{1, \ldots, n\}$, that is also generated from latent source λ_{g^*}, but with different shift $\Delta_2 \in \mathcal{D}_+$ and noise signal W_2:

$$Y_{u^*} = \mu_{g^*} \circledast \Delta_2 + W_2.$$

Therefore, we can rewrite Y in terms of Y_{u^*} as follows:

$$\begin{aligned} Y &= \mu_{g^*} \circledast \Delta_1 + W_1 \\ &= \big((Y_{u^*} - W_2) \circledast (-\Delta_2)\big) \circledast \Delta_1 + W_1 \\ &= Y_{u^*} \circledast (\Delta_1 - \Delta_2) + (W_1 - W_2 \circledast (\Delta_1 - \Delta_2)) \\ &= Y_{u^*} \circledast \Delta^* + W, \end{aligned} \tag{3.7}$$

where $\Delta^* \triangleq \Delta_1 - \Delta_2 \in \mathcal{D} = \{-\Delta_{\max}, \ldots, -1, 0, 1, \ldots, \Delta_{\max}\}$ (note the change from $\mathcal{D}_+$ to $\mathcal{D}$) and $W = W_1 - W_2 \circledast \Delta$. Since noise signals W_1 and W_2 are i.i.d. over time and sub-Gaussian with parameter σ, noise signal W is also i.i.d. over time and sub-Gaussian, now with parameter $\sqrt{2}\sigma$.

We remark that the generative model could allow for time shifts sampled uniformly at random from some arbitrary finite set $\mathcal{D}' \subset \mathbb{Z}$ that need not equal $\mathcal{D}_+$, but our analysis to follow will require nearest-neighbor classification to look at training time series shifted by amount Δ^* of the form $\Delta_1 - \Delta_2$ with $\Delta_1, \Delta_2 \in \mathcal{D}'$. When $\mathcal{D}' = \mathcal{D}_+$, then $\Delta^* \in \mathcal{D}$.

Performance Guarantee for Nearest-Neighbor Classification

Given event $\mathcal{E}_{\text{all-sources}}$, the probability of error for nearest-neighbor classification is

$$\begin{aligned} &\mathbb{P}(\widehat{L}_{\text{NN}}^{(T)}(Y) \neq L \mid \mathcal{E}_{\text{all-sources}}) \\ &= \mathbb{P}(L = +1 \mid \mathcal{E}_{\text{all-sources}})\mathbb{P}(\widehat{L}_{\text{NN}}^{(T)}(Y) = -1 \mid \mathcal{E}_{\text{all-sources}}, L = +1) \\ &\quad + \mathbb{P}(L = -1 \mid \mathcal{E}_{\text{all-sources}})\mathbb{P}(\widehat{L}_{\text{NN}}^{(T)}(Y) = +1 \mid \mathcal{E}_{\text{all-sources}}, L = -1). \end{aligned} \tag{3.8}$$

The rest of the proof is mainly on upper-bounding $\mathbb{P}(\widehat{L}_{\text{NN}}^{(T)}(Y) = -1 \mid \mathcal{E}_{\text{all-sources}}, L = +1)$. Bounding $\mathbb{P}(\widehat{L}_{\text{NN}}^{(T)}(Y) = +1 \mid \mathcal{E}_{\text{all-sources}}, L = -1)$ is similar. Suppose $L = +1$. Nearest-neighbor classification makes an error and declares $\widehat{L}_{\text{NN}}^{(T)}(Y) = -1$ when the nearest neighbor $Y_{\widehat{u}}$ found has label $L_{\widehat{u}} = -1$, where

$$(\widehat{u}, \widehat{\Delta}) = \operatorname*{argmin}_{u \in \{1,\ldots,n\}, \Delta \in \mathcal{D}} \|Y_u \circledast \Delta - Y\|_T^2 = \operatorname*{argmin}_{u \in \{1,\ldots,n\}, \Delta \in \mathcal{D}} \|Y_u \circledast \Delta - (Y_{u^*} \circledast \Delta^* + W)\|_T^2,$$

where the second equality uses equation (3.7). By optimality of $(\widehat{u}, \widehat{\Delta})$ for the above optimization problem, we have

$$\|Y_u \circledast \Delta - (Y_{u^*} \circledast \Delta^* + W)\|_T^2 \geq \|Y_{\widehat{u}} \circledast \widehat{\Delta} - (Y_{u^*} \circledast \Delta^* + W)\|_T^2 \quad \text{for all } u \in \{1, \ldots, n\}, \Delta \in \mathcal{D}.$$

Plugging in $u = u^*$ and $\Delta = \Delta^*$, we obtain

$$\begin{aligned} \|W\|_T^2 &\geq \|Y_{\widehat{u}} \circledast \widehat{\Delta} - (Y_{u^*} \circledast \Delta^* + W)\|_T^2 \\ &= \|(Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^*) - W\|_T^2 \\ &= \|Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^*\|_T^2 - 2\langle Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^*, W\rangle_T + \|W\|_T^2, \end{aligned}$$

where $\langle Q, Q' \rangle_T \triangleq \sum_{t=1}^{T} Q(t)Q'(t)$ for time series Q and Q'. The above inequality can be equivalently written as

$$2\langle Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^*, W \rangle_T \geq \| Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^* \|_T^2 \tag{3.9}$$

Since the true label is $L = +1$, which is the same label as training time series Y_{u^*}, then nearest-neighbor classification making an error implies the existence of a training time series $Y_{\widehat{u}}$ with label $L_{\widehat{u}} = -1$ that satisfies optimality condition (3.9). Therefore,

$$\begin{aligned}
&\mathbb{P}(\widehat{L}_{\mathrm{NN}}^{(T)}(Y) = -1 \mid \mathcal{E}_{\text{all-sources}}, L = +1) \\
&\leq \mathbb{P}\Bigg(\bigcup_{\substack{\widehat{u} \in \{1,\ldots,n\} \text{ s.t. } L_{\widehat{u}} = -1, \\ \widehat{\Delta} \in \mathcal{D}}} \{ 2\langle Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^*, W \rangle_T \geq \| Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^* \|_T^2 \} \Bigg) \\
&\overset{(1)}{\leq} \sum_{\substack{\widehat{u} \in \{1,\ldots,n\} \text{ s.t. } L_{\widehat{u}} = -1, \\ \widehat{\Delta} \in \mathcal{D}}} \mathbb{P}\big(2\langle Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^*, W \rangle_T \geq \| Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^* \|_T^2 \big) \\
&= \sum_{\substack{\widehat{u} \in \{1,\ldots,n\} \text{ s.t. } L_{\widehat{u}} = -1, \\ \widehat{\Delta} \in \mathcal{D}}} \mathbb{P}\big(\exp(2\phi \langle Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^*, W \rangle_T) \geq \exp(\phi \| Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^* \|_T^2) \big) \\
&\overset{(2)}{\leq} \sum_{\substack{\widehat{u} \in \{1,\ldots,n\} \text{ s.t. } L_{\widehat{u}} = -1, \\ \widehat{\Delta} \in \mathcal{D}}} \exp(-\phi \| Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^* \|_T^2) \mathbb{E}\big[\exp(2\phi \langle Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^*, W \rangle_T) \big] \\
&\overset{(3)}{\leq} \sum_{\substack{\widehat{u} \in \{1,\ldots,n\} \text{ s.t. } L_{\widehat{u}} = -1, \\ \widehat{\Delta} \in \mathcal{D}}} \exp(-\phi \| Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^* \|_T^2) \exp(4\phi^2 \sigma^2 \| Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^* \|_T^2) \\
&= \sum_{\substack{\widehat{u} \in \{1,\ldots,n\} \text{ s.t. } L_{\widehat{u}} = -1, \\ \widehat{\Delta} \in \mathcal{D}}} \exp\big(-(\phi - 4\phi^2\sigma^2) \| Y_{\widehat{u}} \circledast \widehat{\Delta} - Y_{u^*} \circledast \Delta^* \|_T^2 \big) \\
&\leq \sum_{\substack{\widehat{u} \in \{1,\ldots,n\} \text{ s.t. } L_{\widehat{u}} = -1, \\ \widehat{\Delta} \in \mathcal{D}}} \exp\big(-(\phi - 4\phi^2\sigma^2) \mathcal{G}^{(T)} \big) \\
&\leq n|\mathcal{D}| \exp\big(-(\phi - 4\phi^2\sigma^2) \mathcal{G}^{(T)} \big) \\
&= n(2\Delta_{\max} + 1) \exp\big(-(\phi - 4\phi^2\sigma^2) \mathcal{G}^{(T)} \big) \\
&\overset{(4)}{=} n(2\Delta_{\max} + 1) \exp\big(-\frac{1}{16\sigma^2} \mathcal{G}^{(T)} \big),
\end{aligned} \tag{3.10}$$

where step (1) is by a union bound, step (2) is by Markov's inequality for $\phi > 0$, step (3) is by sub-Gaussianity, and step (4) is by choosing $\phi = \frac{1}{8\sigma^2}$.

A similar calculation shows that

$$\mathbb{P}(\widehat{L}_{\mathrm{NN}}^{(T)}(Y) = -1 | L = +1) \leq n(2\Delta_{\max} + 1) \exp\big(-\frac{1}{16\sigma^2} \mathcal{G}^{(T)} \big), \tag{3.11}$$

Plugging in bounds (3.10) and (3.11) into equation (3.8) yields

$$\mathbb{P}(\widehat{L}_{\mathrm{NN}}^{(T)}(Y) \neq L \mid \mathcal{E}_{\text{all-sources}}) \leq (2\Delta_{\max}+1)n \exp\Big(-\frac{1}{16\sigma^2}\mathcal{G}^{(T)}\Big).$$

Finally,

$$\begin{aligned}
\mathbb{P}(\widehat{L}_{\mathrm{NN}}^{(T)}(Y) \neq L) &= \mathbb{P}(\mathcal{E}_{\text{all-sources}})\mathbb{P}(\widehat{L}_{\mathrm{NN}}^{(T)}(Y) \neq L \mid \mathcal{E}_{\text{all-sources}}) \\
&\quad + \mathbb{P}(\mathcal{E}_{\text{all-sources}}^c)\mathbb{P}(\widehat{L}_{\mathrm{NN}}^{(T)}(Y) \neq L \mid \mathcal{E}_{\text{all-sources}}^c) \\
&\leq \mathbb{P}(\widehat{L}_{\mathrm{NN}}^{(T)}(Y) \neq L \mid \mathcal{E}_{\text{all-sources}}) + \mathbb{P}(\mathcal{E}_{\text{all-sources}}^c) \\
&\leq (2\Delta_{\max}+1)n \exp\Big(-\frac{1}{16\sigma^2}\mathcal{G}^{(T)}\Big) + k\exp\Big(-\frac{n\pi_{\min}}{8}\Big).
\end{aligned}$$

Performance Guarantee for Generalized Weighted Majority Voting

Given event $\mathcal{E}_{\text{all-sources}}$, the probability of error for generalized weighted majority voting is

$$\begin{aligned}
&\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) \neq L \mid \mathcal{E}_{\text{all-sources}}) \\
&= \mathbb{P}(L=+1 \mid \mathcal{E}_{\text{all-sources}})\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) = -1 \mid \mathcal{E}_{\text{all-sources}}, L=+1) \\
&\quad + \mathbb{P}(L=-1 \mid \mathcal{E}_{\text{all-sources}})\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) = +1 \mid \mathcal{E}_{\text{all-sources}}, L=-1) \\
&= \mathbb{P}(L=+1)\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) = -1 \mid \mathcal{E}_{\text{all-sources}}, L=+1) \\
&\quad + \mathbb{P}(L=-1)\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) = +1 \mid \mathcal{E}_{\text{all-sources}}, L=-1),
\end{aligned} \tag{3.12}$$

where the second equality holds since the training data are independent of the label of a new time series generated. The rest of the proof primarily shows how to bound $\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) = -1 \mid \mathcal{E}_{\text{all-sources}}, L=+1)$. Bounding $\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) = +1 \mid \mathcal{E}_{\text{all-sources}}, L=-1)$ is almost identical. By Markov's inequality,

$$\begin{aligned}
&\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) = -1 \mid \mathcal{E}_{\text{all-sources}}, L=+1) \\
&= \mathbb{P}\left(\frac{1}{R^{(T)}(Y;\theta)} > \frac{1}{\tau} \,\middle|\, \mathcal{E}_{\text{all-sources}}, L=+1\right) \\
&\leq \tau\mathbb{E}\left[\frac{1}{R^{(T)}(Y;\theta)} \,\middle|\, \mathcal{E}_{\text{all-sources}}, L=+1\right]. \\
&= \tau\mathbb{E}\left[\frac{1}{R^{(T)}(Y_{u^*} \circledast \Delta^* + W;\theta)} \,\middle|\, \mathcal{E}_{\text{all-sources}}, L=+1\right],
\end{aligned} \tag{3.13}$$

where the last step uses equation (3.7), which, as a reminder, relied on event $\mathcal{E}_{\text{all-sources}}$ holding.

To proceed further, we begin by upper-bounding the term within the expectation, namely $1/R^{(T)}(Y_{u^*} \circledast \Delta^* + W;\theta)$. Note that for any $\widetilde{u} \in \{1,\dots,n\}$, shift $\widetilde{\Delta} \in \mathcal{D}$, and time series Q,

$$\frac{1}{R^{(T)}(Q;\theta)} \leq \frac{\sum_{u\in\{1,\dots,n\} \text{ s.t. } L_u=-1, \Delta\in\mathcal{D}} \exp(-\theta\|Y_u \circledast \Delta - Q\|_T^2)}{\exp(-\theta\|Y_{\widetilde{u}} \circledast \widetilde{\Delta} - Q\|_T^2)}.$$

In particular, we can choose $\widetilde{u} = u^*$ and $\widetilde{\Delta} = \Delta^*$, so

$$\frac{1}{R^{(T)}(Q;\theta)} \leq \frac{\sum_{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \Delta\in\mathcal{D}} \exp(-\theta\|Y_u \circledast \Delta - Q\|_T^2)}{\exp(-\theta\|Y_{u^*} \circledast \Delta^* - Q\|_T^2)}.$$

Recall that we're upper-bounding $1/R^{(T)}(Y_{u^*} \circledast \Delta^* + W;\theta)$, corresponding to setting $Q = Y_{u^*} \circledast \Delta^* + W$. Doing this substitution, we get

$$\begin{aligned}
&\frac{1}{R^{(T)}(Y_{u^*} \circledast \Delta^* + W;\theta)} \\
&\leq \frac{\sum_{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \Delta\in\mathcal{D}} \exp(-\theta\|Y_u \circledast \Delta - (Y_{u^*} \circledast \Delta^* + W)\|_T^2)}{\exp(-\theta\|Y_{u^*} \circledast \Delta^* - (Y_{u^*} \circledast \Delta^* + W)\|_T^2)} \\
&= \frac{\sum_{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \Delta\in\mathcal{D}} \exp(-\theta\|(Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*) - W\|_T^2)}{\exp(-\theta\|W\|_T^2)} \\
&= \frac{\sum_{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \Delta\in\mathcal{D}} e^{-\theta(\|Y_u\circledast\Delta - Y_{u^*}\circledast\Delta^*\|_T^2 - 2\langle Y_u\circledast\Delta - Y_{u^*}\circledast\Delta^*, W\rangle_T + \|W\|_T^2)}}{\exp(-\theta\|W\|_T^2)} \\
&= \sum_{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \Delta\in\mathcal{D}} \exp\Big(-\theta(\|Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*\|_T^2 - 2\langle Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*, W\rangle_T)\Big) \\
&= \sum_{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \Delta\in\mathcal{D}} \exp(-\theta\|Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*\|_T^2) \exp(2\theta\langle Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*, W\rangle_T).
\end{aligned}$$

What we actually want to bound is the expectation of the left-hand side, given event $\mathcal{E}_{\text{all-sources}}$ holding and $L = +1$. Taking this expectation for both sides above,

$$\begin{aligned}
&\mathbb{E}\left[\frac{1}{R^{(T)}(Y_{u^*} \circledast \Delta^* + W;\theta)} \,\middle|\, \mathcal{E}_{\text{all-sources}}, L = +1\right] \\
&\leq \mathbb{E}\left[\sum_{\substack{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \\ \Delta\in\mathcal{D}}} e^{-\theta\|Y_u\circledast\Delta - Y_{u^*}\circledast\Delta^*\|_T^2} e^{2\theta\langle Y_u\circledast\Delta - Y_{u^*}\circledast\Delta^*, W\rangle_T} \,\middle|\, \mathcal{E}_{\text{all-sources}}, L = +1\right] \\
&= \sum_{\substack{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \\ \Delta\in\mathcal{D}}} \exp(-\theta\|Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*\|_T^2)\mathbb{E}_W\big[\exp(2\theta\langle Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*, W\rangle_T)\big] \\
&\leq \sum_{\substack{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \\ \Delta\in\mathcal{D}}} \exp(-\theta\|Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*\|_T^2) \exp(4\theta^2\sigma^2\|Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*\|_T^2) \\
&= \sum_{\substack{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \\ \Delta\in\mathcal{D}}} \exp\Big(-(\theta - 4\theta^2\sigma^2)\|Y_u \circledast \Delta - Y_{u^*} \circledast \Delta^*\|_T^2\Big) \\
&\leq \sum_{\substack{u\in\{1,\ldots,n\} \text{ s.t. } L_u=-1, \\ \Delta\in\mathcal{D}}} \exp\Big(-(\theta - 4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big) \\
&\leq n|\mathcal{D}| \exp\Big(-(\theta - 4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big)
\end{aligned}$$

$$= n(2\Delta_{\max}+1)\exp\Big(-(\theta-4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big),$$

where the second inequality uses the fact that W consists of entries that are i.i.d. zero-mean sub-Gaussian with parameter $\sqrt{2}\sigma$. Plugging the above into inequality (3.13), we get

$$\mathbb{P}(\widehat{L}_\tau^{(T)}(Y;\theta)=-1 \mid \mathcal{E}_{\text{all-sources}}, L=+1) \leq \tau n(2\Delta_{\max}+1)\exp\Big(-(\theta-4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big). \quad (3.14)$$

A similar calculation yields

$$\mathbb{P}(\widehat{L}_\tau^{(T)}(Y;\theta)=+1 \mid \mathcal{E}_{\text{all-sources}}, L=-1) \leq \frac{1}{\tau} n(2\Delta_{\max}+1)\exp\Big(-(\theta-4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big). \quad (3.15)$$

Plugging inequalities (3.14) and (3.15) into equation (3.12) gives

$$\begin{aligned}
&\mathbb{P}(\widehat{L}_\tau^{(T)}(Y;\theta)\neq L \mid \mathcal{E}_{\text{all-sources}}) \\
&\leq \mathbb{P}(L=+1)\tau n(2\Delta_{\max}+1)\exp\Big(-(\theta-4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big) \\
&\quad + \mathbb{P}(L=-1)\frac{1}{\tau} n(2\Delta_{\max}+1)\exp\Big(-(\theta-4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big) \\
&= \left(\mathbb{P}(L=+1)\tau + \mathbb{P}(L=-1)\frac{1}{\tau}\right) n(2\Delta_{\max}+1)\exp\Big(-(\theta-4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big).
\end{aligned}$$

Finally,

$$\begin{aligned}
\mathbb{P}(\widehat{L}_\tau^{(T)}(Y;\theta)\neq L) &= \mathbb{P}(\mathcal{E}_{\text{all-sources}})\mathbb{P}(\widehat{L}_\tau^{(T)}(Y;\theta)\neq L \mid \mathcal{E}_{\text{all-sources}}) \\
&\quad + \mathbb{P}(\mathcal{E}^c_{\text{all-sources}})\mathbb{P}(\widehat{L}_\tau^{(T)}(Y;\theta)\neq L \mid \mathcal{E}^c_{\text{all-sources}}) \\
&\leq \mathbb{P}(\widehat{L}_\tau^{(T)}(Y;\theta)\neq L \mid \mathcal{E}_{\text{all-sources}}) + \mathbb{P}(\mathcal{E}^c_{\text{all-sources}}) \\
&\leq \left(\mathbb{P}(L=+1)\tau + \mathbb{P}(L=-1)\frac{1}{\tau}\right) n(2\Delta_{\max}+1)\exp\Big(-(\theta-4\theta^2\sigma^2)\mathcal{G}^{(T)}\Big) \\
&\quad + k\exp\left(-\frac{n\pi_{\min}}{8}\right).
\end{aligned}$$

■

■ 3.7.2 Proof of Theorem 3.5.1

We prove Theorem 3.5.1 using gap $\mathcal{G}^{(T)\dagger}$, which gives a stronger result than with gap $\mathcal{G}^{(T)*}$ that disregards the labels of the latent sources. The proof using gap $\mathcal{G}^{(T)*}$ is the same.

As with the proof of Theorem 3.4.1, we begin by assuming that there are more than $n\pi_{\min}/2$ training time series from every latent source, i.e, that event $\mathcal{E}_{\text{all-sources}}$ holds. By Lemma 2.2.2, this happens with probability at least $1-k\exp(-n\pi_{\min}/8)$.

Let Y_{+1} and Y_{-1} be two training time series with labels $+1$ and -1. These time series exist since event $\mathcal{E}_{\text{all-sources}}$ holds and, moreover, the latent source model assumes that the underlying latent sources don't all have the same label. Under the latent source model, $Y_{+1} = X^{(Y_{+1})} + W^{(Y_{+1})}$ and $Y_{-1} = X^{(Y_{-1})} + W^{(Y_{-1})}$, where $X^{(Y_{+1})}, X^{(Y_{-})} \in \{\mu_1,\ldots,\mu_k\}$ denote the true latent sources corresponding to Y_{+1} and Y_{-1}, and $W^{(Y_{+1})}$ and $W^{(Y_{-1})}$ are the noise signals corresponding to Y_{+1} and Y_{-1} that consist of i.i.d. $\mathcal{N}(0,\sigma^2)$ entries. Thus,

$$\|Y_{+1}-Y_{-1}\|_T^2$$

$$= \|(X^{(Y_{+1})} + W^{(Y_{+1})}) - (X^{(Y_{-1})} + W^{(Y_{-1})})\|_T^2$$
$$= \|(X^{(Y_{+1})} - X^{(Y_{-1})}) + (W^{(Y_{+1})} - W^{(Y_{-1})})\|_T^2$$
$$= \|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T^2 + 2\langle X^{(Y_{+1})} - X^{(Y_{-1})}, W^{(Y_{+1})} - W^{(Y_{-1})}\rangle_T + \|W^{(Y_{+1})} - W^{(Y_{-1})}\|_T^2.$$

We next want to analyze how far $\|Y_{+1} - Y_{-1}\|_T^2$ deviates from $\|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T^2$. With s_1 and s_2 as fixed positive constants, define $\mathcal{E}_{\text{all-distances-good}}$ to be the event where for every pair of training time series Y_{+1} and Y_{-1} with labels $+1$ and -1 respectively, we have both

$$\langle X^{(Y_{+1})} - X^{(Y_{-1})}, W^{(Y_{+1})} - W^{(Y_{-1})}\rangle_T > -\|W^{(Y_{+1})} - W^{(Y_{-1})}\|_T s_1, \tag{3.16}$$
$$\|W^{(Y_{+1})} - W^{(Y_{-1})}\|_T^2 > 2\sigma^2 T - 4\sigma^2\sqrt{T s_2}. \tag{3.17}$$

We show when event $\mathcal{E}_{\text{all-distances-good}}$ holds with sufficiently high probability:

- Bound (3.16): Note that $\langle X^{(Y_{+1})} - X^{(Y_{-1})}, W^{(Y_{+1})} - W^{(Y_{-1})}\rangle_T$ is zero-mean sub-Gaussian with parameter $\sqrt{2}\sigma\|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T$, so

  $$\mathbb{P}\big(\langle X^{(Y_{+1})} - X^{(Y_{-1})}, W^{(Y_{+1})} - W^{(Y_{-1})}\rangle_T \le -\|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T s_1\big) \le \exp\Big(-\frac{s_1^2}{4\sigma^2}\Big).$$

  A union bound over all pairs of training time series with opposite labels gives

  $$\mathbb{P}\Big(\bigcup_{Y_{+1},Y_{-1}} \big\{\langle X^{(Y_{+1})} - X^{(Y_{-1})}, W^{(Y_{+1})} - W^{(Y_{-1})}\rangle_T \le -\|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T s_1\big\}\Big)$$
  $$\le \sum_{Y_{+1},Y_{-1}} \mathbb{P}\big(\langle X^{(Y_{+1})} - X^{(Y_{-1})}, W^{(Y_{+1})} - W^{(Y_{-1})}\rangle_T \le -\|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T s_1\big)$$
  $$\le \sum_{Y_{+1},Y_{-1}} \exp\Big(-\frac{s_1^2}{4\sigma^2}\Big) = n_{+1}n_{-1}\exp\Big(-\frac{s_1^2}{4\sigma^2}\Big) \le \frac{n^2}{4}\exp\Big(-\frac{s_1^2}{4\sigma^2}\Big),$$

  where n_{+1} is the number of training time series with labels $+1$, and $n_{-1} = n - n_{+1}$ is the number of training time series with label -1.

- Bound (3.17): Due to a result by Laurent and Massart (2000, Lemma 1), we have

  $$\mathbb{P}\big(\|W^{(Y_{+1})} - W^{(Y_{-1})}\|_T^2 \le 2\sigma^2 T - 4\sigma^2\sqrt{Tb}\big) \le e^{-s_2}.$$

  A union bound gives

  $$\mathbb{P}\Big(\bigcup_{Y_{+1},Y_{-1}} \big\{\|W^{(Y_{+1})} - W^{(Y_{-1})}\|_T^2 \le 2\sigma^2 T - 4\sigma^2\sqrt{T s_2}\big\}\Big)$$
  $$\le \sum_{Y_{+1},Y_{-1}} \mathbb{P}\big(\|W^{(Y_{+1})} - W^{(Y_{-1})}\|_T^2 \le 2\sigma^2 T - 4\sigma^2\sqrt{T s_2}\big)$$
  $$\le \sum_{Y_{+1},Y_{-1}} e^{-s_2} = n_{+1}n_{-1}e^{-s_2} \le \frac{n^2}{4}e^{-s_2}.$$

By a union bound, the "good" event $\mathcal{E}_{\text{good}} \triangleq \mathcal{E}_{\text{all-sources}} \cap \mathcal{E}_{\text{all-distances-good}}$ happens with probability at least $1 - k\exp(-\frac{n\pi_{\min}}{8}) - \frac{n^2}{4}\exp(-\frac{s_1^2}{4\sigma^2}) - \frac{n^2}{4}e^{-s_2}$.

Assuming that event $\mathcal{E}_{\text{good}}$ holds and $\mathcal{G}^{(T)\dagger} \geq s_1^2$,

$$
\begin{aligned}
&\|Y_{+1} - Y_{-1}\|_T^2 \\
&= \|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T^2 + 2\langle X^{(Y_{+1})} - X^{(Y_{-1})}, W^{(Y_{+1})} - W^{(Y_{-1})}\rangle_T + \|W^{(Y_{+1})} - W^{(Y_{-1})}\|_T^2 \\
&> \|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T^2 - 2\|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T^2 s_1 + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2} \\
&= (\|X^{(Y_{+1})} - X^{(Y_{-1})}\|_T - s_1)^2 - s_1^2 + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2} \\
&\geq (\sqrt{\mathcal{G}^{(T)\dagger}} - s_1)^2 - s_1^2 + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2} \\
&= \mathcal{G}^{(T)\dagger} - 2s_1\sqrt{\mathcal{G}^{(T)\dagger}} + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2}.
\end{aligned}
$$

Minimizing both sides over training time series Y_{+1} and Y_{-1} with labels $+1$ and -1,

$$
\mathcal{G}^{(T)} \geq \mathcal{G}^{(T)\dagger} - 2s_1\sqrt{\mathcal{G}^{(T)\dagger}} + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2}. \tag{3.18}
$$

Using the same proof as for Theorem 3.4.1 except where now instead of only asking $\mathcal{E}_{\text{all-sources}}$ to hold, we ask that event $\mathcal{E}_{\text{good}}$ to hold, and we use the bound above relating $G^{(T)}$ to $G^{(T)\dagger}$, nearest-neighbor classification has probability of error

$$
\begin{aligned}
\mathbb{P}(\widehat{L}_{\text{NN}}^{(T)}(Y) \neq L) \leq\ & k\exp\Big(-\frac{n\pi_{\min}}{8}\Big) + \frac{n^2}{4}\exp\Big(-\frac{s_1^2}{4\sigma^2}\Big) + \frac{n^2}{4}\exp(-s_2) \\
& + n\exp\Big(-\frac{1}{16\sigma^2}\big(\mathcal{G}^{(T)\dagger} - 2s_1\sqrt{\mathcal{G}^{(T)\dagger}} + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2}\big)\Big),
\end{aligned}
$$

and generalized weighted majority voting has probability of error

$$
\begin{aligned}
&\mathbb{P}(\widehat{L}_{\tau}^{(T)}(Y;\theta) \neq L) \\
&\leq k\exp\Big(-\frac{n\pi_{\min}}{8}\Big) + \frac{n^2}{4}\exp\Big(-\frac{s_1^2}{4\sigma^2}\Big) + \frac{n^2}{4}\exp(-s_2) \\
&\quad + \Big(\tau\pi_{+1} + \frac{1}{\tau}\pi_{-1}\Big)n\exp\Big(-(\theta - 4\sigma^2\theta)^2\big(\mathcal{G}^{(T)\dagger} - 2s_1\sqrt{\mathcal{G}^{(T)\dagger}} + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2}\big)\Big).
\end{aligned}
$$

■

■ 3.7.3 Proof of Corollary 3.5.1

Again, here we can replace $\mathcal{G}^{(T)\dagger}$ with $\mathcal{G}^{(T)*}$, and the proof would still go through. Let $\delta \in (0,1)$. For the nearest-neighbor classification upper bound in Theorem 3.4.1, we ask that each of the four right-hand side terms be at most $\delta/4$. Specifically, we set $s_1 = 2\sigma\sqrt{\log\frac{n^2}{\delta}}$ and $s_2 = \log\frac{n^2}{\delta}$, which makes the second and third terms each exactly equal to $\delta/4$. For the first term, we ask that $n \geq \frac{8}{\pi_{\min}}\log\frac{4k}{\delta}$. Finally, for the fourth term, we do a worst-case analysis. Specifically, with our choices for s_1 and s_2, by completing the square, we have

$$
\mathcal{G}^{(T)\dagger} - 2s_1\sqrt{\mathcal{G}^{(T)\dagger}} + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2}
$$

$$
\begin{aligned}
&= \mathcal{G}^{(T)\dagger} - 4\sigma\sqrt{\mathcal{G}^{(T)\dagger}\log\frac{n^2}{\delta}} + 2\sigma^2 T - 4\sigma^2\sqrt{T\log\frac{n^2}{\delta}} \\
&= \left(\sqrt{\mathcal{G}^{(T)\dagger}} - 2\sigma\sqrt{\log\frac{n^2}{\delta}}\right)^2 - 4\sigma^2\log\frac{n^2}{\delta} + 2\sigma^2 T - 4\sigma^2\sqrt{T\log\frac{n^2}{\delta}} \\
&\geq -4\sigma^2\log\frac{n^2}{\delta} + 2\sigma^2 T - 4\sigma^2\sqrt{T\log\frac{n^2}{\delta}}.
\end{aligned}
$$

Hence,

$$
\begin{aligned}
& n\exp\left(-\frac{1}{16\sigma^2}\left(\mathcal{G}^{(T)\dagger} - 2s_1\sqrt{\mathcal{G}^{(T)\dagger}} + 2\sigma^2 T - 4\sigma^2\sqrt{Ts_2}\right)\right) \\
& \leq n\exp\left(-\frac{1}{16\sigma^2}\left(-4\sigma^2\log\frac{n^2}{\delta} + 2\sigma^2 T - 4\sigma^2\sqrt{T\log\frac{n^2}{\delta}}\right)\right) \\
& \leq \frac{\delta}{4}
\end{aligned}
$$

where the second inequality holds when

$$
T \geq 2\log\frac{n^2}{\delta} + 2\sqrt{T\log\frac{n^2}{\delta}} + 8\log\frac{4n}{\delta}.
$$

This happens when

$$
T \geq 4\log\frac{n^2}{\delta} + 8\log\frac{4n}{\delta} + 2\sqrt{\left(3\log\frac{n^2}{\delta} + 8\log\frac{4n}{\delta}\right)\log\frac{n^2}{\delta}}. \qquad \blacksquare
$$

■ 3.7.4 Proof of Theorem 3.6.1

Let $\widehat{L}$ denote the predicted label for L using any classifier. Due to a result by Santhi and Vardy (2006, equation (4)), for binary classification,

$$
\mathbb{P}(\widehat{L} \neq L) \geq \int \frac{\mathbb{P}(L=+1)p(Y|L=+1)\mathbb{P}(L=-1)p(Y|L=-1)}{p(Y)}\,dY, \tag{3.19}
$$

where the right-hand side is referred to as the harmonic lower bound. We lower-bound this harmonic lower bound. For any two latent sources $X_{+1}, X_{-1} \in \{\mu_1, \ldots, \mu_k\}$ with labels $+1$ and -1 respectively,

$$
\begin{aligned}
&\int \frac{\mathbb{P}(L=+1)p(Y|L=+1)\mathbb{P}(L=-1)p(Y|L=-1)}{p(Y)}\,dY \\
&= \int \frac{\pi_{+1}\pi_{-1}\left(\sum_{\substack{g\in\{1,\ldots,k\}\\ \text{s.t. } \lambda_g=+1}} \pi_g\mathcal{N}(Y;\mu_g,\sigma^2\mathbf{I}_{T\times T})\right)\cdot\left(\sum_{\substack{g\in\{1,\ldots,k\}\\ \text{s.t. } \lambda_g=-1}} \pi_g\mathcal{N}(Y;\mu_g,\sigma^2\mathbf{I}_{T\times T})\right)}{p(Y)}\,dY \\
&\overset{(1)}{\geq} \int (2\pi)^{T/2}\sigma^T\pi_{+1}\pi_{-1}\left(\sum_{\substack{g\in\{1,\ldots,k\}\\ \text{s.t. } \lambda_g=+1}} \pi_g\mathcal{N}(Y;X_{+1},\sigma^2\mathbf{I}_{T\times T})\right)\left(\sum_{\substack{g\in\{1,\ldots,k\}\\ \text{s.t. } \lambda_g=-1}} \pi_g\mathcal{N}(Y;X_{-1},\sigma^2\mathbf{I}_{T\times T})\right)dY
\end{aligned}
$$

$$
\begin{aligned}
&\geq \int (2\pi)^{T/2}\sigma^T \pi_{+1}\pi_{-1}\pi_{\min}^2 \mathcal{N}(Y; X_{+1}, \sigma^2 \mathbf{I}_{T\times T})\mathcal{N}(Y; X_{-1}, \sigma^2 \mathbf{I}_{T\times T}) dY \\
&= (2\pi)^{T/2}\sigma^T \pi_{+1}\pi_{-1}\pi_{\min}^2 \int \mathcal{N}(Y; X_{-1}, \sigma^2 \mathbf{I}_{T\times T})(\mathcal{N}(Y; X_{+1}, \sigma^2 \mathbf{I}_{T\times T}) dY) \\
&= (2\pi)^{T/2}\sigma^T \pi_{+1}\pi_{-1}\pi_{\min}^2 \mathbb{E}_{Y\sim\mathcal{N}(X_{+1},\sigma^2\mathbf{I}_{T\times T})}[\mathcal{N}(Y; X_{-1}, \sigma^2 \mathbf{I}_{T\times T})] \\
&= \pi_{+1}\pi_{-1}\pi_{\min}^2 \mathbb{E}_{Y\sim\mathcal{N}(X_{+1},\sigma^2\mathbf{I}_{T\times T})}\Big[\exp\Big(-\frac{1}{2\sigma^2}\|Y - X_{-1}\|_T^2\Big)\Big] \\
&\overset{(2)}{\geq} \pi_{+1}\pi_{-1}\pi_{\min}^2 \mathbb{E}_{Y\sim\mathcal{N}(X_{+1},\sigma^2\mathbf{I}_{T\times T})}\Big[\exp\Big(-\frac{1}{2\sigma^2}(\|Y - X_{+1}\|_T + \|X_{+1} - X_{-1}\|_T)^2\Big)\Big] \\
&= \pi_{+1}\pi_{-1}\pi_{\min}^2 e^{-\frac{1}{2\sigma^2}\|X_{+1}-X_{-1}\|_T^2} \mathbb{E}_{Y\sim\mathcal{N}(X_{+1},\sigma^2\mathbf{I}_{T\times T})}\Big[e^{-\frac{1}{2\sigma^2}(\|Y-X_{+1}\|_T^2 + 2\|Y-X_{+1}\|_T\|X_{+1}-X_{-1}\|_T)}\Big],
\end{aligned}
\tag{3.20}
$$

where step (1) uses the fact that

$$
p(Y) = \sum_{g=1}^k \pi_g \mathcal{N}(Y; \mu_g, \sigma^2\mathbf{I}_{T\times T}) \leq \sum_{g=1}^k \pi_g \mathcal{N}(\mu_g; \mu_g, \sigma^2\mathbf{I}_{T\times T}) = \sum_{g=1}^k \frac{\pi_g}{(2\pi)^{T/2}\sigma^T} = \frac{1}{(2\pi)^{T/2}\sigma^T},
$$

and step (2) uses the triangle inequality

$$
\|Y - X_{-1}\|_T = \|(Y - X_{+1}) + (X_{+1} - X_{-1})\|_T \leq \|Y - X_{+1}\|_T + \|X_{+1} - X_{-1}\|_T.
$$

Let's bound the expectation in the last line of inequality (3.20). With Y treated as a random sample from $\mathcal{N}(X_{+1}, \sigma^2\mathbf{I}_{T\times T})$, then $Y = X_{+1} + W$, where $W \sim \mathcal{N}(0, \sigma^2\mathbf{I}_{T\times T})$. Therefore

$$
\begin{aligned}
&\mathbb{E}_{Y\sim\mathcal{N}(X_{+1},\sigma^2\mathbf{I}_{T\times T})}\Big[\exp\Big(-\frac{1}{2\sigma^2}(\|Y - X_{+1}\|_T^2 + 2\|Y - X_{+1}\|_T\|X_{+1} - X_{-1}\|_T)\Big)\Big] \\
&= \mathbb{E}_{W\sim\mathcal{N}(0,\sigma^2\mathbf{I}_{T\times T})}\Big[\exp\Big(-\frac{1}{2\sigma^2}(\|W\|_T^2 + 2\|W\|_T\|X_{+1} - X_{-1}\|_T)\Big)\Big] \\
&\overset{(1)}{\geq} \exp\Big(-\frac{1}{2\sigma^2}\mathbb{E}_{W\sim\mathcal{N}(0,\sigma^2\mathbf{I}_{T\times T})}\big[\|W\|_T^2 + 2\|W\|_T\|X_{+1} - X_{-1}\|_T\big]\Big) \\
&= \exp\Big(-\frac{1}{2\sigma^2}\big(\mathbb{E}_{W\sim\mathcal{N}(0,\sigma^2\mathbf{I}_{T\times T})}[\|W\|_T^2] + 2\|X_{+1} - X_{-1}\|_T\mathbb{E}_{W\sim\mathcal{N}(0,\sigma^2\mathbf{I}_{T\times T})}[\|W\|_T]\big)\Big) \\
&\overset{(2)}{=} \exp\Big(-\frac{1}{2\sigma^2}\Big(\sigma^2 T + 2\|X_{+1} - X_{-1}\|_T\sigma\sqrt{2}\frac{\Gamma((T+1)/2)}{\Gamma(T/2)}\Big)\Big) \\
&= \exp\Big(-\frac{T}{2} - \|X_{+1} - X_{-1}\|_T\frac{\sqrt{2}\Gamma((T+1)/2)}{\sigma\Gamma(T/2)}\Big),
\end{aligned}
\tag{3.21}
$$

where step (1) is by Jensen's inequality, and step (2) uses the fact that $\frac{1}{\sigma^2}\|W\|_T^2$ is distributed as a χ^2 random variable with T degrees of freedom, so $\mathbb{E}[\|W\|_T^2] = \sigma^2 T$, while $\frac{1}{\sigma}\|W\|_T$ is a χ random variable with T degrees of freedom, so $\mathbb{E}[\|W\|_T] = \sigma\sqrt{2}\frac{\Gamma((T+1)/2)}{\Gamma(T/2)}$.

Stringing together inequalities (3.19), (3.20), and (3.21), we get

$$
\mathbb{P}(\widehat{L} \neq L) \geq \pi_{+1}\pi_{-1}\pi_{\min}^2 \exp\Big(-\frac{1}{2\sigma^2}\|X_{+1} - X_{-1}\|_T^2 - \frac{T}{2} - \|X_{+1} - X_{-1}\|_T\frac{\sqrt{2}\Gamma((T+1)/2)}{\sigma\Gamma(T/2)}\Big).
$$

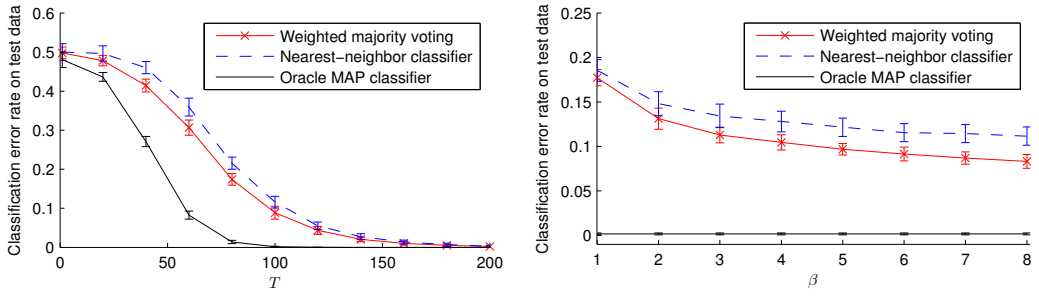
Since the above holds for all latent sources X_{+1} and X_{-1} with labels $+1$ and -1, we have

$$\mathbb{P}(\widehat{L} \neq L) \geq \pi_{+1}\pi_{-1}\pi_{\min}^2 \exp\Big(-\frac{1}{2\sigma^2}\mathcal{G}^{(T)}t - \frac{T}{2} - \sqrt{\mathcal{G}^{(T)}t}\frac{\sqrt{2}\Gamma((T+1)/2)}{\sigma\Gamma(T/2)}\Big). \qquad \blacksquare$$

3.8 Experimental Results

Synthetic data. We generate $k = 200$ latent sources that occur with equal probability, where each latent source is constructed by first sampling i.i.d. $\mathcal{N}(0, 100)$ entries per time step and then applying a 1D Gaussian smoothing filter with scale parameter 30. Half of the latent sources are labeled $+1$ and the other half -1. Then $n = \beta k \log k$ training time series are sampled, for various values of β, as per the latent source model where the noise added is i.i.d. $\mathcal{N}(0, 1)$, and the maximum time shift is $\Delta_{\max} = 100$. We similarly generate 1000 time series to use as test data. We set $\theta = 1/8$ for weighted majority voting. For $\beta = 8$, we compare the classification error rates on test data for weighted majority voting, nearest-neighbor classification, and the MAP classifier with oracle access to the true latent sources as shown in Figure 3.1(a). We see that weighted majority voting outperforms nearest-neighbor classification but as T grows large, the two methods' performances converge to that of the MAP classifier. Fixing $T = 100$, we then compare the classification error rates of the three methods using varying amounts of training data, as shown in Figure 3.1(b); the oracle MAP classifier is also shown but does not actually depend on training data. We see that as β increases, both weighted majority voting and nearest-neighbor classification steadily improve in performance.

Forecasting trending topics on twitter. We now summarize experimental results reported in (Nikolov, 2012) and subsequently in (Chen et al., 2013). From June 2012 user posts on Twitter, 500 examples of trends were sampled at random from a list of news trends, and



(a) (b)

Figure 3.1: Results on synthetic data. (a) Classification error rate vs. number of initial time steps T used; training set size: $n = \beta k \log k$ where $\beta = 8$. (b) Classification error rate at $T = 100$ vs. β. All experiments were repeated 20 times with newly generated latent sources, training data, and test data each time. Error bars denote one standard deviation above and below the mean value.

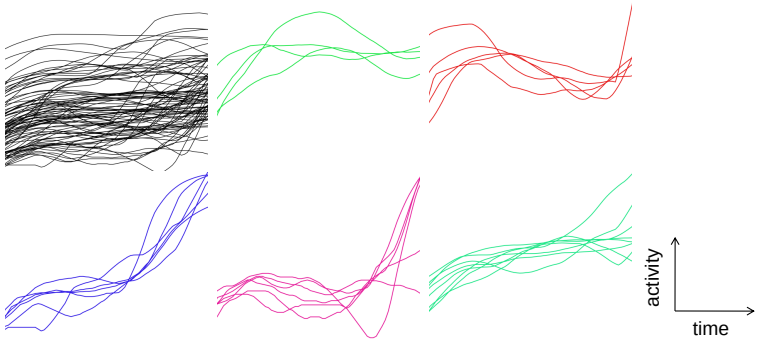


Figure 3.2: How news topics become trends on Twitter. The top left shows some time series of activity leading up to a news topic becoming trending. These time series superimposed look like clutter, but we can separate them into different clusters, as shown in the next five plots. Each cluster represents a "way" that a news topic becomes trending.

500 examples of non-trends were sampled based on phrases appearing in posts. As it's unknown to the public how Twitter chooses what phrases are considered as candidate phrases for trending topics, it's unclear what the size of the non-trend category is in comparison to the size of the trend category. Thus, for simplicity, the label class sizes were controlled to be equal, and standard weighted majority voting (3.2) was used to classify time series, where $\Delta_{\max}$ is set to the maximum possible (all shifts are considered). In practice, one could assemble the training data to have uneven label class sizes and then tune τ for generalized weighted majority voting (3.3).

Per topic, Nikolov (2012) created a time series based on a pre-processed version of the rate of how often the topic was shared by users on Twitter. Chen et al. (2013) empirically found that the time series for news topics that become trends tend to follow a finite number of patterns; a few examples of these patterns are shown in Figure 3.2. These few patterns could be thought of as latent sources. The trends and non-trends were randomly divided into two halves, one to use as training data and one to use as test data. Nikolov (2012) applied weighted majority voting, sweeping over θ, T, and data pre-processing parameters. As shown in Figure 3.3(a), with one choice of parameters, weighted majority voting detected trending topics in advance of Twitter 79% of the time, and on average 1.43 hours earlier, with a true positive rate (TPR) of 95% and a false positive rate (FPR) of 4%. Naturally, there are tradeoffs between TPR, FPR, and how early one wants to make a prediction (i.e., how small time horizon T is). As shown in Figure 3.3(c), an "aggressive" parameter setting yields early detection and high TPR but high FPR, and a "conservative" parameter setting yields low FPR but late detection and low TPR. An "in-between" setting can strike the right balance.

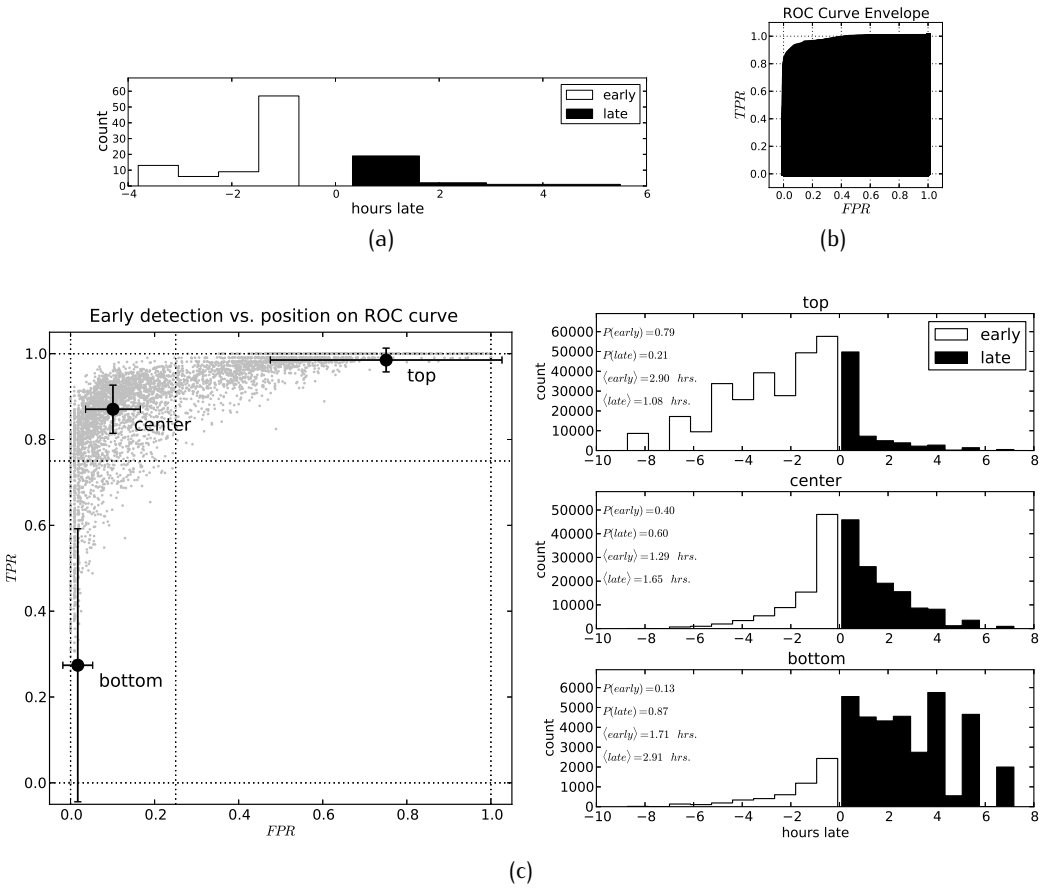


Figure 3.3: Results from Nikolov (2012) on Twitter data. (a) Weighted majority voting achieves a low error rate (FPR of 4%, TPR of 95%) and detects trending topics in advance of Twitter 79% of the time, with a mean of 1.43 hours when it does for a particular choice of parameters. (b) Envelope of all ROC curves shows the tradeoff between TPR and FPR. (c) Distribution of detection times for "aggressive" (top), "conservative" (bottom) and "in-between" (center) parameter settings.

■ 3.9 Discussion

Having to subsample the training data to keep the misclassification rate upper bounds from scaling with n seems strange. Shouldn't more data only help us? Or is it that by seeing more data, due to noise, as we get more and more samples, we're bound to get unlucky and encounter a training time series with the wrong label that is close to the time series we want to classify, causing nearest-neighbor classification to get confused and buckle? In fact, our later results in this thesis will also involve some training data subsampling, yet it's unclear whether this is really necessary for the nearest-neighbor-like inference methods we consider.

When the number of training data n is large, a more clever strategy that still involves subsampling but now uses all the training data is to randomly partition the training data into groups of size $\Theta(\frac{1}{\pi_{\min}} \log \frac{k}{\delta})$ each. Then we can apply weighted majority voting within each group. A final prediction can be made by a "meta" majority vote: choose the most popular label across the different groups' label predictions. This meta-voting strategy readily lends itself to analysis. In particular, since the training data in different groups are independent, weighted majority voting's predictions across the different groups are also independent and we know when we can ensure each of these predictions to be correct with probability at least $1 - \delta$. Then among the $\Theta(\frac{n\pi_{\min}}{\log(k/\delta)})$ groups, the number of correct label predictions stochastically dominates a Binomial$\big(\Theta(\frac{n\pi_{\min}}{\log(k/\delta)}), 1-\delta\big)$ random variable. We can then apply a binomial concentration inequality to lower-bound this meta-voting strategy's probability of success.

We end this chapter by relating separation gap $\mathcal{G}^{(T)\dagger}$ in Sections 3.5 and 3.6 to the margin condition of Mammen and Tsybakov (1999) and Tsybakov (2004) (recall that separation gap $\mathcal{G}^{(T)\dagger}$ between the true latent sources relates to the separation gap $\mathcal{G}^{(T)}$ in the training data via inequality (3.18)). The basic idea is that classification should be challenging if for observed time series Y, the posterior probability $\mathbb{P}(L = +1 \mid Y)$ is close to $1/2$. When this happens, it means that Y is close to the decision boundary and could plausibly be explained by both labels. Thus, if the probability that Y lands close to the decision boundary is sufficiently low, then an inference algorithm that, either explicitly or implicitly, estimates the decision boundary well should achieve a low misclassification rate. Mammen and Tsybakov (1999) and Tsybakov (2004) formalize this margin condition as follows:

$$\mathbb{P}\left(\left|\mathbb{P}(L = +1 \mid Y) - \frac{1}{2}\right| \leq s\right) \leq Cs^{\phi}, \tag{3.22}$$

for some finite $C > 0$, $\phi > 0$, and all $0 < s \leq s^*$ for some $s^* \leq 1/2$. Note that the randomness is over Y. With additional assumptions on the behavior of the decision boundary, Tsybakov (2004) and Audibert and Tsybakov (2007) showed that nonparametric classifiers can have misclassification rates that exceed the optimal Bayes error rate by as low as $\mathcal{O}(n^{-1})$ or even lower under a far more restrictive assumption on how label L relates to observation Y.

To sketch how separation gap $\mathcal{G}^{(T)\dagger}$ relates to the above margin condition, we consider the one-dimensional Gaussian case with no time shifts where we have two latent sources: if Y has label $L = +1$ then it is generated from $\mathcal{N}(\mu, \sigma^2)$ and if Y has label $L = -1$ then it is generated from $\mathcal{N}(-\mu, \sigma^2)$ for constants $\mu > 0$, and $\sigma > 0$, and where $\mathbb{P}(L = +1) = \mathbb{P}(L = -1) = 1/2$. For this example, an optimal MAP decision rule classifies Y to have label $+1$ if $Y \geq 0$, and to have label -1 otherwise. Thus, the decision boundary is at $Y = 0$. Meanwhile, the separation gap is given by $\mathcal{G}^{(T)\dagger} = \mu - (-\mu) = 2\mu$. To relate to margin condition (3.22), note that for $s \in [0, 1/2)$,

$$\left|\mathbb{P}(L = +1 \mid Y) - \frac{1}{2}\right| \leq s \quad \Leftrightarrow \quad Y \in \Big[-\frac{\sigma^2}{2\mu} \log\Big(\frac{1+2s}{1-2s}\Big), \frac{\sigma^2}{2\mu} \log\Big(\frac{1+2s}{1-2s}\Big)\Big],$$

where $\log(\frac{1+2s}{1-2s}) = 0$ when $s = 0$, and $\log(\frac{1+2s}{1-2s}) \to \infty$ when $s \to 1/2$. The right-hand side

interval corresponds to the decision boundary $Y = 0$ up to some closeness parameter s. For this interval to be far away enough from the two latent source means $-\mu$ and μ, henceforth, we assume that s is small enough so that

$$\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right) < \mu.$$

In other words, the interval doesn't contain the latent source means $-\mu$ and μ. Then

$$\begin{aligned}
&\mathbb{P}\left(\left|\mathbb{P}(L=+1 \mid Y) - \frac{1}{2}\right| \le s\right) \\
&= \mathbb{P}\left(Y \in \left[-\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right), \frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right)\right]\right) \\
&= \mathbb{P}(L=+1)\mathbb{P}\left(Y \in \left[-\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right), \frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right)\right] \,\middle|\, L=+1\right) \\
&\quad + \mathbb{P}(L=-1)\mathbb{P}\left(Y \in \left[-\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right), \frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right)\right] \,\middle|\, L=-1\right) \\
\text{(by symmetry)} &= \mathbb{P}\left(Y \in \left[-\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right), \frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right)\right] \,\middle|\, L=+1\right) \\
&= \mathbb{P}\left(\mathcal{N}(\mu,\sigma^2) \in \left[-\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right), \frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right)\right]\right) \\
&\le \left[\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right) - \left(-\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right)\right)\right]\mathcal{N}\left(\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right); \mu, \sigma^2\right) \\
&= \frac{\sigma^2}{\mu}\log\left(\frac{1+2s}{1-2s}\right)\mathcal{N}\left(\frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right); \mu, \sigma^2\right) \\
&= \frac{\sigma}{\sqrt{2\pi}\mu}\log\left(\frac{1+2s}{1-2s}\right)\exp\left(-\frac{1}{\sigma^2}\left(\mu - \frac{\sigma^2}{2\mu}\log\left(\frac{1+2s}{1-2s}\right)\right)^2\right), \qquad (3.23)
\end{aligned}$$

where the inequality uses the fact that since $\frac{\sigma^2}{2\mu}\log(\frac{1+2s}{1-2s}) < \mu$, the largest value of the density of $\mathcal{N}(\mu,\sigma^2)$ within interval $[-\frac{\sigma^2}{2\mu}\log(\frac{1+2s}{1-2s}), \frac{\sigma^2}{2\mu}\log(\frac{1+2s}{1-2s})]$ is $\mathcal{N}(\frac{\sigma^2}{2\mu}\log(\frac{1+2s}{1-2s}); \mu, \sigma^2)$. Examining the Taylor expansion of (3.23), one can then derive an upper bound of the form $\mathbb{P}\big(|\mathbb{P}(L=+1 \mid Y) - \frac{1}{2}| \le s\big) \le Cs^{\phi}$ by choosing ϕ to be the (positive integer) order of the Taylor expansion, and then choosing constant C large enough. For example, if we want a linear bound ($\phi = 1$), note that the right-hand side of inequality (3.23) has Taylor expansion $\sqrt{\frac{8}{\pi}}\frac{\sigma^2 e^{-\mu^2/\sigma^2}}{\mu}s + \mathcal{O}(s^2)$. Choosing C to be sufficiently larger than $\sqrt{\frac{8}{\pi}}\frac{\sigma^2 e^{-\mu^2/\sigma^2}}{\mu}$ results in a bound for small enough s, thus satisfying margin condition (3.22).

We note that bound (3.23) decays exponentially with the separation gap $\mathcal{G}^{(T)\dagger} = 2\mu$. One way to intuit this result is that since our noise is sub-Gaussian, the probability that Y deviates significantly from its generating latent source decays exponentially as a function of how far Y is from this latent source. When the separation gap between latent sources of opposite labels is large, then it means that to land close to the decision boundary, Y would have to be quite far from even the two closest latent sources with opposite labels. This event's probability goes to 0 as the separation gap grows large.

Chapter 4

Online Collaborative Filtering

Recommendation systems have become ubiquitous in our lives, helping us filter the vast expanse of information we encounter into small selections tailored to our personal tastes. Prominent examples include Amazon recommending items to buy, Netflix recommending movies, and LinkedIn recommending jobs. In practice, recommendations are often made via *collaborative filtering*, which boils down to recommending an item to a user by considering items that other similar or "nearby" users liked. Collaborative filtering has been used extensively for decades now including in the GroupLens news recommendation system (Resnick et al., 1994), Amazon's item recommendation system (Linden et al., 2003), the Netflix $1 million grand prize winning algorithm by BellKor's Pragmatic Chaos (Koren, 2009; Töscher and Jahrer, 2009; Piotte and Chabbert, 2009), and a recent song recommendation system (Aiolli, 2013) that won the Million Song Dataset Challenge (Bertin-Mahieux et al., 2011).

Most such systems operate in the "online" setting, where items are constantly recommended to users over time. In many scenarios, it does not make sense to recommend an item that is already consumed. For example, once Alice watches a movie, there's little point to recommending the same movie to her again, at least not immediately, and one could argue that recommending unwatched movies and already watched movies could be handled as separate cases. Finally, what matters is whether a *likable* item is recommended to a user rather than an *unlikable* one. In short, a good online recommendation system should recommend different likable items continually over time.

Despite the success of collaborative filtering, there has been little theoretical development to justify its effectiveness in the online setting. Instead, most work (e.g., Cai et al. (2010); Candès and Recht (2009); Keshavan et al. (2010a,b); Recht (2011)) has been in the "offline" setting, where we freeze time, have access to all "revealed" ratings users have provided so far, and predict all "missing" ratings for items users haven't yet rated. This offline problem setup perhaps gained enormous popularity amongst both academics and practitioners after Netflix offered a $1 million dollar grand prize for a solution to the problem that outperformed theirs by a pre-specified performance metric. The setup itself is a matrix completion problem, where we consider a matrix of ratings where rows index users and columns index items (e.g., movies), and the (u, i)-th entry is either the rating that user u gave item i, or marked as missing. The goal then is to fill in these missing ratings, typically by imposing a low-rank constraint on the ratings matrix. The theoretical guarantees for such methods usually assume that the items that users view are uniform

at random, which is not the case in reality and also doesn't account for the fact that real recommendation systems should and are biasing users into rating certain items, hopefully items that they like. Only recently has this assumption been lifted for theoretical analysis of the offline setting (Lee and Shraibman, 2013). Of course, none of these model the true online nature of recommendation systems with time marching forward and the system continuously providing recommendations and receiving user feedback.

Moving to this online setting, most work has been phrased in the context of the classical so-called *multi-armed bandit* problem, first introduced by Thompson (1933). The name of the problem originates from the "one-armed bandit" slot machine found in casinos in which a gambler pulls the arm of the machine and receives a random reward. Naturally, the m-armed bandit problem refers to when the gambler has m such machines to play and seeks to maximize her or his cumulative reward over time, playing one machine at each time step. Translated to the context of online recommendation systems, the m machines are items, and playing a machine refers to recommending an item. We remark that in this standard setup of the multi-armed bandit problem, there is only a single user and hence no concept of collaboration between users, and only recently has there been work on incorporating a pool of users (Bui et al., 2012; Gentile et al., 2014).

Even so, there hasn't been a justification for why existing commonly used *collaborative filtering* methods work in this online setting, with the closest related work analyzing the asymptotic consistency of cosine-similarity collaborative filtering in predicting the rating of the next unseen item (Biau et al., 2010), or examining the ability of a collaborative filtering method to predict ratings in an asymptotic information-theoretic setting (Barman and Dabeer, 2012). Dabeer (2013) uses a similar model as ours but analyzes an algorithm the knows the number of user and item types, which we won't assume we have access to.

In this chapter, we justify when, why, and how well a variant of cosine-similarity collaborative filtering works. This main result relies on an underlying model for an online recommendation system as well as a clear objective of what a recommendation method should be optimizing, both of which are presented in Section 4.1. Our setup imposes the constraint that once an item is consumed by a user, the system can't recommend the item to the same user again. In the context of the multi-armed bandit problem, once a gambler pulls the arm of a machine, the machine breaks and is no longer available!

For analysis, similar to the time series classification setting, we begin with an oracle MAP inference algorithm that knows the underlying model parameters in Section 4.2. We show how to approximate this oracle recommendation algorithm with cosine-similarity collaborative filtering, where it becomes apparent when we should expect cosine-similarity filtering to work well. Before the approximation works well though, there should be sufficient exploration upfront. We include two types of exploration, one to probe the space of items and the other to probe the space of users. This leads to a simple cosine-similarity collaborative filtering algorithm described in Section 4.3, which we modify slightly to obtain a more readily analyzable algorithm called Collaborative-Greedy. This sets up the stage for our main result of this chapter in Section 4.4, which provides a theoretical performance guarantee for Collaborative-Greedy. Its proof is in Section 4.5.

We present experimental results in Section 4.6. First, we provide evidence that

real movie recommendation data do exhibit clustering behavior across users. In fact, items cluster as well, which we do not account for in our model. Then, to test our algorithm Collaborative-Greedy, we simulate an online recommendation system that uses ratings from real movie recommendation datasets and show that Collaborative-Greedy outperforms several existing collaborative filtering methods.

■ 4.1 A Latent Source Model and a Learning Problem

We consider a system with n users and m items. At each time step, each user is recommended an item that she or he hasn't consumed yet, upon which, for simplicity, we assume that the user immediately consumes the item and rates it $+1$ (like) or -1 (dislike).[1] The reward earned by the recommendation system up to any time step is the total number of liked items that have been recommended so far across all users. Formally, index time by $t \in \{1, 2, \dots\}$, and users by $u \in \{1, \dots, n\}$. Let $\psi_{ut} \in \{1, \dots, m\}$ be the item recommended to user u at time t. Let $Y_{ui}^{(t)} \in \{-1, 0, +1\}$ be the rating provided by user u for item i up to and including time t, where 0 indicates that no rating has been given yet. A reasonable objective is to maximize the expected reward $r^{(T)}$ up to time T:

$$r^{(T)} \triangleq \sum_{t=1}^{T} \sum_{u=1}^{n} \mathbb{E}[Y_{u\psi_{ut}}^{(T)}] = \sum_{i=1}^{m} \sum_{u=1}^{n} \mathbb{E}[Y_{ui}^{(T)}].$$

The ratings are noisy: the latent item preferences for user u are represented by a length-m vector $p_u \in [0, 1]^m$, where user u likes item i with probability p_{ui}, independently across items. For a user u, we say that item i is *likable* if $p_{ui} > 1/2$ and *unlikable* otherwise. To maximize the expected reward $r^{(T)}$, clearly likable items for the user should be recommended before unlikable ones.

In this chapter, we focus on recommending likable items. Thus, instead of maximizing the expected reward $r^{(T)}$, we aim to maximize the expected number of *likable* items recommended up to time T:

$$r_+^{(T)} \triangleq \sum_{t=1}^{T} \sum_{u=1}^{n} \mathbb{E}[X_{ut}], \tag{4.1}$$

where X_{ut} is the indicator random variable for whether the item recommended to user u at time t is likable, i.e., $X_{ut} = +1$ if $p_{u\psi_{ut}} > 1/2$ and $X_{ut} = 0$ otherwise. Maximizing $r^{(T)}$ and $r_+^{(T)}$ differ since the former asks that we prioritize items according to their probability of being liked.

Recommending likable items for a user in an arbitrary order is sufficient for many real recommendation systems such as for movies and music. For example, we suspect that users wouldn't actually prefer to listen to music starting from the songs that their

[1] In practice, a user could ignore the recommendation. To keep our exposition simple, however, we stick to this setting that resembles song recommendation systems like Pandora that per user continually recommends a single item at a time. For example, if a user rates a song as "thumbs down" then we assign a rating of -1 (dislike), and any other action corresponds to $+1$ (like).

user type would like with highest probability to the ones their user type would like with lowest probability; instead, each user would listen to songs that she or he finds likable, ordered such that there is sufficient diversity in the playlist to keep the user experience interesting. We target the modest goal of merely recommending likable items, in any order. Of course, if all likable items have the same probability of being liked and similarly for all unlikable items, then maximizing $r^{(T)}$ and $r_+^{(T)}$ are equivalent.

The fundamental challenge is that to learn about a user's preference for an item, we need the user to rate (and thus consume) the item. But then we cannot recommend that item to the user again! Thus, the only way to learn about a user's preferences is through collaboration, or inferring from other users' ratings. Broadly, such inference is possible if the users' preferences are somehow related.

In this chapter, we assume a simple structure for shared user preferences. We posit that there are $k \ll n$ different types of users, where users of the same type have identical item preference vectors. Specifically, we denote the k underlying item preference vectors as $\mu_1, \ldots, \mu_k \in [0,1]^m$. If users u and v are of the same type $g \in \{1, \ldots, k\}$, then $p_u = p_v = \mu_g$. The number of user types k represents the heterogeneity in the population. For ease of exposition, in this chapter we assume that a user belongs to each user type with probability $1/k$. We refer to the overall model as a *latent source model for online collaborative filtering*, where each user type corresponds to a latent source of users. We remark that there is evidence suggesting real movie recommendation data to be well modeled by clustering of both users and items (Sutskever et al., 2009). Our model only assumes clustering over users.

Our problem setup relates to some versions of the multi-armed bandit problem. A fundamental difference between our setup and that of the standard stochastic multi-armed bandit problem (Thompson, 1933; Bubeck and Cesa-Bianchi, 2012) is that the latter allows each item to be recommended an infinite number of times. Thus, the solution concept for the stochastic multi-armed bandit problem is to determine the best item (arm) and keep choosing it (Auer et al., 2002). This observation applies also to "clustered bandits" (Bui et al., 2012), which like our work seeks to capture collaboration between users. On the other hand, "sleeping bandits" (Kleinberg et al., 2010) allow for the available items at each time step to vary, but the analysis is worst-case in terms of which items are available over time. In our setup, the sequence of items that are available is not adversarial. Our model combines the collaborative aspect of clustered bandits with dynamic item availability from sleeping bandits, where we impose a strict structure on how items become unavailable.

■ 4.2 Approximating an Oracle Algorithm

We begin with an algorithm that has oracle access to the true item preference vectors $\mu_1, \ldots, \mu_k \in [0,1]^m$ of the k user types, but does not know which type each user belongs to. One could show that at time $t+1$, the MAP recommendation given the past ratings

for user u is to choose item i that user u has not consumed that maximizes

$$\sum_{g=1}^{k} \mu_{gi} \exp\left\{ - \sum_{j \in \mathrm{supp}(Y_u^{(t)})} D\big(\mathrm{Ber}\big((Y_{uj}^{(t)}+1)/2\big) \big\| \mathrm{Ber}(\mu_{gj})\big)\right\}, \tag{4.2}$$

where $\mathrm{supp}(\cdot)$ denotes the support of a vector (its set of nonzero entries), $\mathrm{Ber}(\cdot)$ denotes the Bernoulli distribution, and $D(\cdot\|\cdot)$ denotes KL divergence; as a reminder, $Y_u^{(t)}$ is the revealed ratings vector for user u up to and including time t. Notice that the MAP recommendation is a weighted plurality vote. Each user type g casts a vote with weight $\mu_{gi} \exp\big\{ - \sum_{j \in \mathrm{supp}(Y_u^{(t)})} D\big(\mathrm{Ber}\big((Y_{uj}^{(t)}+1)/2\big) \big\| \mathrm{Ber}(\mu_{gj})\big)\big\}$ for item i. The weight is the probability that user type g likes item i downweighted by how misaligned user u's revealed ratings so far $Y_u^{(t)}$ are with μ_g. To obtain a recommendation, we sum all the votes for each item that user u has not consumed yet, and choose the item with the highest vote.

In reality, we know neither the number of user types k nor the item preference vectors $\mu_1, \ldots, \mu_k$. If these vectors are in some sense sufficiently well-separated, then after enough revealed ratings for user u, the exponentially decaying weight in equation (4.2) should eliminate all the user types' contributions except for that of user u's type, and furthermore, if the revealed ratings aren't too noisy, then the exponentially decaying weight for user u's type should be going to 1. In other words, if user u belongs to type h, then for sufficiently large t,

$$\sum_{g=1}^{k} \mu_{gi} \exp\left\{ - \sum_{j \in \mathrm{supp}(Y_u^{(t)})} D\big(\mathrm{Ber}\big((Y_{uj}^{(t)}+1)/2\big) \big\| \mathrm{Ber}(\mu_{gj})\big)\right\} \to \mu_{hi} \quad \text{as} \quad t \to \infty. \tag{4.3}$$

Of course, we do not know μ_{hi}. We could estimate it using neighbors for user u, where we use cosine similarity to define the neighborhoods as it is widely used in collaborative filtering. Specifically, we estimate μ_{hi} with:

$$\widehat{p}_{ui}^{(t)} \triangleq \begin{cases} \dfrac{\sum_{v \in N_u^{(t)}} \mathbb{1}\{Y_{vi}^{(t)} = +1\}}{\sum_{v \in N_u^{(t)}} \mathbb{1}\{Y_{vi}^{(t)} \neq 0\}} & \text{if } \sum_{v \in N_u^{(t)}} \mathbb{1}\{Y_{vi}^{(t)} \neq 0\} > 0, \\ 1/2 & \text{otherwise,} \end{cases} \tag{4.4}$$

where the user's neighborhood $N_u^{(t)}$ is defined as

$$N_u^{(t)} \triangleq \{v \in \{1, \ldots, n\} : \langle Y_u^{(t)}, Y_v^{(t)} \rangle \geq \theta |\mathrm{supp}(Y_u^{(t)}) \cap \mathrm{supp}(Y_v^{(t)})|\}\,.$$

Here θ is a pre-specified parameter. In particular, user v is a neighbor of user u precisely if the cosine similarity between their revealed ratings over items they have both rated is at least θ. To see this, let $\Omega_{uv} \triangleq \mathrm{supp}(Y_u^{(t)}) \cap \mathrm{supp}(Y_v^{(t)})$ be the support overlap of $Y_u^{(t)}$ and $Y_v^{(t)}$, and let $\langle \cdot, \cdot \rangle_{\Omega_{uv}}$ be the dot product restricted to entries in Ω_{uv}. Then

$$\frac{\langle Y_u^{(t)}, Y_v^{(t)} \rangle}{|\Omega_{uv}|} = \frac{\langle Y_u^{(t)}, Y_v^{(t)} \rangle_{\Omega_{uv}}}{\sqrt{\langle Y_u^{(t)}, Y_u^{(t)} \rangle_{\Omega_{uv}}} \sqrt{\langle Y_v^{(t)}, Y_v^{(t)} \rangle_{\Omega_{uv}}}}\,,$$

is the cosine similarity of revealed rating vectors $Y_u^{(t)}$ and $Y_v^{(t)}$ restricted to the overlap of their supports.

The above approximation assumed two conditions: low noise and user type separation. Both of these conditions turn up in our theoretical performance guarantee in Section 4.4, where the separation will be in terms of cosine similarity. With both conditions satisfied, after enough items are explored and revealed, approximation (4.3) should work well and could be thought of as *exploiting* what we've learned thus far about the users' preferences. However, in the initial transition period in which we have not seen enough ratings for user u, we shouldn't just be exploiting. Whereas the oracle MAP recommender neatly handles this transition period, we'll opt for the simpler action of randomized exploration to develop two collaborative filtering algorithms in the next section. We remark that if instead of cosine similarity, we used another similarity measure between users, we suspect that our main theoretical result to follow could still work with a different user type separation condition.

■ 4.3 Collaborative Filtering with Two Exploration Types

For clarity of presentation, we begin by describing a simpler recommendation algorithm Simple-Collaborative-Greedy. To make Simple-Collaborative-Greedy more amenable to analysis, we modify it slightly to obtain Collaborative-Greedy. Both algorithms are syntactically similar to an algorithm called ε-Greedy for the standard multi-armed bandit setting, which explores items with probability ε and otherwise greedily chooses the best item seen so far based on a plurality vote (Sutton and Barto, 1998). The exploration probability ε is allowed to decay with time: as we learn more about the different bandit machines, or items in our setting, we should be able to explore less and exploit more.

The standard multi-armed bandit setting does not have user collaboration, and one could interpret asking each user to randomly explore an item as probing the space of items. To explicitly encourage user collaboration, we could ask users to all explore the same item, which probes the space of users. Accounting for the constraint in our setting that an item can't be recommended to the same user more than once, we thus have the two following exploration types:

- *Random exploration.* For every user, recommend an item that she or he hasn't consumed yet uniformly at random.
- *Joint exploration.* Ask every user to provide a rating for the next unseen item in a shared, randomly chosen sequence of the m items.

Our first algorithm Simple-Collaborative-Greedy thus does one of three actions at each time step t: With probability ε_R, we do the above random exploration step. With probability ε_J, we do the above joint exploration step. Finally, if we do do neither of these exploration steps, then we do a greedy exploitation step for every user: recommend whichever item i user u has not consumed yet that maximizes the plurality vote $\widehat{p}_{ui}^{(t)}$ given by equation (4.4), which relied on cosine similarity to find nearby users.

We choose the exploration probabilities ε_R and ε_J as follows. For a pre-specified rate $\alpha \in (0, 4/7]$, we set the probability of random exploration to be $\varepsilon_R(n) = 1/n^\alpha$ (decaying with the number of users), and the probability of joint exploration to be $\varepsilon_J(t) = 1/t^\alpha$ (decaying with time). For ease of presentation, we set the two explorations to have the same decay rate α, but our proof easily extends to encompass different decay rates for the two exploration types. Furthermore, the constant $4/7 \geq \alpha$ is not special. It could be different and only affects another constant in our proof. The resulting algorithm is given in Algorithm 1.

Algorithm 1: Simple-Collaborative-Greedy (and Collaborative-Greedy)

Input: Parameters $\theta \in [0, 1]$, $\alpha \in (0, 4/7]$.
Select a random ordering ω of the items $\{1, \ldots, m\}$. Define

$$\varepsilon_R(n) = \frac{1}{n^\alpha}, \qquad \text{and} \qquad \varepsilon_J(t) = \frac{1}{t^\alpha}.$$

for *time step* $t = 1, 2, \ldots, T$ **do**
With prob. $\varepsilon_R(n)$: (**random exploration**) for each user, recommend a random item that the user has not rated.
With prob. $\varepsilon_J(t)$: (**joint exploration**) for each user, recommend the first item in ω that the user has not rated.
With prob. $1 - \varepsilon_J(t) - \varepsilon_R(n)$: (**exploitation**) for each user u, recommend an item i that the user has not rated and that maximizes score $\widehat{p}_{ui}^{(t)}$ given by equation (4.4), which depends on threshold θ. (For Collaborative-Greedy, use score $\widetilde{p}_{ui}^{(t)}$ given by equation (4.5) instead.)

The main technical hurdle in analyzing Simple-Collaborative-Greedy is that it's not trivial reasoning about the items that two users have both rated, especially the items recommended by cosine similarity exploitation. In other words, which items have revealed ratings follows a nontrivial probability distribution. We can easily circumvent this issue by changing the definition of the neighborhood of a user u to only consider items that have been jointly explored. Specifically, if we denote t_J to be the number of joint exploration steps up to time t, then we're guaranteed that there's a subset of t_J items chosen uniformly at random that all users have consumed and rated (this is the first t_J items in random item sequence ω in Algorithm 1). The algorithm Collaborative-Greedy results from this slight change. Formally, we replace a user's item score $\widehat{p}_{ui}^{(t)}$ given in equation (4.4) with

$$\widetilde{p}_{ui}^{(t)} \triangleq \begin{cases} \dfrac{\sum_{v \in \widetilde{N}_u^{(t)}} \mathbb{1}\{Y_{vi}^{(t)} = +1\}}{\sum_{v \in \widetilde{N}_u^{(t)}} \mathbb{1}\{Y_{vi}^{(t)} \neq 0\}} & \text{if } \sum_{v \in \widetilde{N}_u^{(t)}} \mathbb{1}\{Y_{vi}^{(t)} \neq 0\} > 0, \\ 1/2 & \text{otherwise,} \end{cases} \tag{4.5}$$

where the neighborhood of user u is given by

$$\widetilde{N}_u^{(t)} \triangleq \{v \in \{1, \ldots, n\} : \langle \widetilde{Y}_u^{(t)}, \widetilde{Y}_v^{(t)} \rangle \geq \theta |\mathrm{supp}(\widetilde{Y}_u^{(t)}) \cap \mathrm{supp}(\widetilde{Y}_v^{(t)})|\},$$

and $\widetilde{Y}_u^{(t)}$ consists of the revealed ratings of user u restricted to the first t_J jointly explored items up to time t. In other words,

$$\widetilde{Y}_{ui}^{(t)} = \begin{cases} Y_{ui}^{(t)} & \text{if item } i \text{ is among the first } t_J \text{ items in random sequence } \omega \text{ of Algorithm 1,} \\ 0 & \text{otherwise.} \end{cases}$$

We also give Collaborative-Greedy in Algorithm 1. Our experimental results in Section 4.6 suggest that the two algorithms have similar performance.

■ 4.4 A Theoretical Performance Guarantee

We now present the main result of this chapter that characterizes the performance of Collaborative-Greedy. This result depends on two conditions, hinted at earlier:

- *Low noise.* There exists a constant $\sigma \in [0, 1/2)$ such that

  $$\min\{1 - \mu_{gi}, \mu_{gi}\} \leq \sigma$$

  for all user types $g \in \{1, \ldots, k\}$ and items $i \in \{1, \ldots, m\}$. In particular, σ measures how far the probabilities of liking items are from 0 or 1. If $\sigma = 0$, then the probabilities are all 0 or 1, so user ratings are deterministic and there is no noise. If $\sigma = 1/2$ (which is actually disallowed by the condition), then there is an item with probability 1/2 of being liked. We can't hope to predict whether a user will like this item better than chance.

  Note that this low noise condition holding with parameter σ implies that it also holds with parameter $\sigma' \in (\sigma, 1/2)$.

- *Cosine separation.* There exists a constant $\mathcal{G}^* \in (0, 1]$ such that for two different user types g and h,

  $$\frac{1}{m}\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1}\rangle \leq (1 - \mathcal{G}^*)(1 - 2\sigma)^2.$$

  where $\mathbf{1}$ is the all ones vector. The left-hand side is an expected cosine similarity. To see this, let Y_u^* and Y_v^* be fully-revealed rating vectors of users u and v from types g and h respectively. Then $\mathbb{E}[\frac{1}{m}\langle Y_u^*, Y_v^*\rangle] = \frac{1}{m}\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1}\rangle$ is the expected cosine similarity between fully-revealed rating vectors Y_u^* and Y_v^*, where the expectation is over the random ratings of items. Constant $\mathcal{G}^*$ can be thought of as a true gap: the larger it is, the smaller the cosine similarities between different user types can be. To combat noise, when σ increases, the condition asks that the cosine similarity between different user types decrease, i.e., that user types become more separated.

  Note that this cosine separation condition holding with parameter $\mathcal{G}^*$ implies that it also holds with parameter $\mathcal{G}^{*\prime} \in (0, \mathcal{G}^*)$.

The low noise condition ensures that with a finite number of samples, we can correctly classify each item as either likable or unlikable. The cosine separation condition ensures that using cosine similarity can tease apart users of different types over time. A worry may be that the cosine separation condition is too stringent and might only hold in with expected cosine similarity $\mathbb{E}[\frac{1}{m}\langle Y_u^*, Y_v^*\rangle]$ scaling as $o(1)$. We provide some examples after the statement of this chapter's main result for which the cosine separation condition holds with $\mathbb{E}[\frac{1}{m}\langle Y_u^*, Y_v^*\rangle]$ scaling as $\Theta(1)$.

We assume that the number of users satisfies $n = \mathcal{O}(m^C)$ for some constant $C > 1$. This is without loss of generality since otherwise, we can randomly divide the n users into separate population pools, each of size $\mathcal{O}(m^C)$ and run the recommendation algorithm independently for each pool to achieve the same overall performance guarantee.

Finally, we define ζ, the minimum proportion of likable items for any user (and thus any user type):

$$\zeta \triangleq \min_{g\in\{1,\dots,k\}} \frac{\sum_{i=1}^{m} \mathbb{1}\{\mu_{ui} > 1/2\}}{m}.$$

We're now ready to state this chapter's main theorem.

Theorem 4.4.1. *Let $\delta \in (0,1)$ be a pre-specified tolerance. Suppose that the low noise condition holds with parameter σ, and the cosine separation condition holds with parameter $\mathcal{G}^*$. Take as input to* Collaborative-Greedy *$\theta = \frac{1}{2}(1-2\sigma)^2\mathcal{G}^*$ where $\mathcal{G}^* \in (0,1]$, and $\alpha \in (0, 4/7]$. Under the latent source model for online collaborative filtering, if the number of users $n = \mathcal{O}(m^C)$ satisfies*

$$n = \Omega\left(km\log\frac{1}{\delta} + \left(\frac{4}{\delta}\right)^{1/\alpha}\right),$$

then for any $T_{learn} \le T \le \zeta m$, the expected proportion of likable items recommended by Collaborative-Greedy *up until time T satisfies*

$$\frac{r_+^{(T)}}{Tn} \ge \left(1 - \frac{T_{learn}}{T}\right)(1-\delta),$$

where

$$T_{learn} = \Theta\left(\left(\frac{\log\frac{km}{(1-2\sigma)\delta}}{(1-2\sigma)^4(\mathcal{G}^*)^2}\right)^{1/(1-\alpha)} + \left(\frac{4}{\delta}\right)^{1/\alpha}\right).$$

The precise conditions (without using big O notation) on the number of users n and learning duration T_{learn} are provided in Lemma 4.5.3 for which no attempt has been made to optimize the many large constants.

The above theorem says that there are T_{learn} initial time steps for which Collaborative-Greedy may be giving poor recommendations. Afterward, for $T_{\text{learn}} < T < \zeta m$, the algorithm becomes near-optimal, recommending a fraction of likable items $1-\delta$ close to what an optimal oracle algorithm (that recommends all likable items first) would achieve. (Note that this oracle algorithm is different from the MAP oracle algorithm presented

earlier, which does not get to know which user is assigned to which latent source.) Then for time horizon $T > \zeta m$, we can no longer guarantee that there are likable items left to recommend. Indeed, if the user types each have the same fraction of likable items, then even an oracle recommender would use up the ζm likable items by this time. To give a sense of how long the learning period T_{learn} is, note that when $\alpha = 1/2$, we have T_{learn} scaling as $\log^2(km)$, and if we choose α close to 0, then T_{learn} becomes nearly $\log(km)$. In summary, after T_{learn} initial time steps, which could be made nearly $\log(km)$, and with number of users scaling as km, Collaborative-Greedy is essentially optimal. This recovers the informal statement of Theorem 1.2.2.

To provide intuition for the cosine separation condition, we calculate parameter $\mathcal{G}^*$ for three examples that build on top of each other.

Example 4.4.1. *Consider when there is no noise, i.e., $\sigma = 0$. Then users' ratings are deterministic given their user type. We construct the true underlying item preference vectors $\mu_1, \dots, \mu_k \in [0,1]^m$ by sampling every entry μ_{gi} ($g \in \{1, \dots, k\}$ and $i \in \{1, \dots, m\}$) to be i.i.d. Bernoulli(1/2). In this case, the cosine separation condition, with true gap $\mathcal{G}^* = 1 - \sqrt{\frac{\log m}{m}}$, holds with probability at least $1 - \frac{k^2}{m}$.*

To show this, note that for any item i and pair of distinct user types g and h, the product $(2\mu_{gi} - 1)(2\mu_{hi} - 1)$ is a Rademacher random variable ($+1$ or -1 each with probability $\frac{1}{2}$), and thus the dot product $\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1} \rangle$ is equal to the sum of m i.i.d. Rademacher random variables, each of which is sub-Gaussian with parameter 1. Hence, the sum is zero-mean sub-Gaussian with parameter $\sqrt{m}$, implying that

$$\mathbb{P}\big(\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1} \rangle \geq s\big) \leq \exp\Big(-\frac{s^2}{2m}\Big).$$

Plugging in $s = m\sqrt{\frac{\log m}{m}}$, we see that

$$\mathbb{P}\Big(\frac{1}{m}\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1} \rangle \geq \sqrt{\frac{\log m}{m}}\Big) \leq \frac{1}{m}.$$

Union-bounding over all distinct pairs of user types,

$$\mathbb{P}\left(\bigcup_{g,h \in \{1,\dots,k\} \text{ s.t. } g \neq h} \left\{\frac{1}{m}\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1} \rangle \geq \sqrt{\frac{\log m}{m}}\right\}\right) \leq \binom{k}{2}\frac{1}{m} \leq \frac{k^2}{m}.$$

Hence, with probability at least $1 - \frac{k^2}{m}$, we have

$$\frac{1}{m}\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1} \rangle < \sqrt{\frac{\log m}{m}}$$

for every distinct pair of user types g and h. Noting that $\sigma = 0$, the cosine separation condition holds with parameter $\mathcal{G}^*$ to be $1 - \sqrt{\frac{\log m}{m}}$.

Example 4.4.2. *We expand on the previous example by introducing noise with parameter $\sigma \in (0, 1/2)$. Now let the item preference vectors $\mu_1, \dots, \mu_k \in [0,1]^m$ have i.i.d. entries that are $1-\sigma$ (likable) or σ (unlikable) with probability $\frac{1}{2}$ each. Then for a distinct pair of user types g and h, if $\mu_{gi} = \mu_{hi}$ (which happens with probability 1/2), then $\mathbb{E}[(2\mu_g - 1)\cdot(2\mu_h - 1)] = (1-\sigma)^2 + \sigma^2 - 2\sigma(1-\sigma) = (1-2\sigma)^2$, and if $\mu_{gi} \neq \mu_{hi}$ (so $\mu_{gi} = 1-\mu_{hi}$ in this example, also occurring with probability 1/2), then $\mathbb{E}[(2\mu_g - 1)\cdot(2\mu_h - 1)] = 2\sigma(1-\sigma) - (1-\sigma)^2 - \sigma^2 = -(1-2\sigma)^2$. This means that $\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1}\rangle$ is again a sum of Rademacher random variables, except now scaled by $(1-2\sigma)^2$. This sum is sub-Gaussian with parameter $\sqrt{m}(1-2\sigma)^2$. By a similar calculation as the previous example, with probability at least $1 - k^2/m$,*

$$\frac{1}{m}\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1}\rangle < (1-2\sigma)^2\sqrt{\frac{\log m}{m}}$$

for every distinct pair of user types g and h. Thus, the cosine separation condition holds with parameter $\mathcal{G}^* = 1 - \sqrt{\frac{\log m}{m}}$.

Example 4.4.3. *Building off our second example, we now suppose that entries in the item preference vectors $\mu_1, \dots, \mu_k \in [0,1]^m$ have entries that are $1-\sigma$ (likable) with probability $\zeta \in (0, 1/2)$, and σ (unlikable) with probability $1-\zeta$. Then for item i and different user types g and h, $\mu_{gi} = \mu_{hi}$ with probability $\zeta + (1-\zeta)^2$. This implies that $\mathbb{E}[\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1}\rangle] = m(1-2\sigma)^2(1-2\zeta)^2$, and one can verify that the dot product $\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1}\rangle$ is still sub-Gaussian with parameter $\sqrt{m}(1-2\sigma)^2$. Using a similar calculation as before but now accounting for the mean of the dot product no longer being 0, with probability at least $1 - k^2/m$,*

$$\frac{1}{m}\langle 2\mu_g - \mathbf{1}, 2\mu_h - \mathbf{1}\rangle < (1-2\sigma)^2\left((1-2\zeta)^2 + \sqrt{\frac{\log m}{m}}\right)$$

for every distinct pair of user types g and h. Then the cosine separation condition holds with parameter $\mathcal{G}^* = 1 - (1-2\zeta)^2 - \sqrt{\frac{\log m}{m}}$.

■ 4.5 Proof of Theorem 4.4.1

Recall that X_{ut} is the indicator random variable for whether the item ψ_{ut} recommended to user u at time t is likable, i.e., $p_{u\psi_{ut}} > 1/2$. This is equivalent to the event that $p_{u\psi_{ut}} \geq 1-\sigma$, under the low noise condition (with parameter σ) holding. Then the expected proportion of likable items is

$$\frac{r_+^{(T)}}{Tn} = \frac{1}{Tn}\sum_{t=1}^{T}\sum_{u=1}^{n}\mathbb{E}[X_{ut}] = \frac{1}{Tn}\sum_{t=1}^{T}\sum_{u=1}^{n}\mathbb{P}(X_{ut}=1).$$

Our proof focuses on lower-bounding $\mathbb{P}(X_{ut}=1)$. The key idea is to condition on what we call the "good neighborhood" event $\mathcal{E}_{\text{good}}(u,t)$:

$\mathcal{E}_{\text{good}}(u,t)$

$$= \Big\{ \text{ at time } t, \text{ user } u \text{ has } \geq \frac{n}{5k} \text{ neighbors from the same user type ("good neighbors"),}$$
$$\text{and } \leq \frac{(1-2\sigma)tn^{1-\alpha}}{20km} \text{ neighbors from other user types ("bad neighbors")} \Big\}.$$

This good neighborhood event will enable us to argue that after an initial learning time, with high probability there are at most $(\frac{1}{2} - \sigma)$ as many ratings from bad neighbors as there are from good neighbors.

The proof of Theorem 4.4.1 consists of two parts. The first part uses joint exploration to show that after a sufficient amount of time, the good neighborhood event $\mathcal{E}_{\text{good}}(u, t)$ holds with high probability.

Lemma 4.5.1. *For user u, after*

$$t \geq \left(\frac{32 \log(20kmn^{\alpha}/(1-2\sigma))}{(1-2\sigma)^4 (\mathcal{G}^*)^2} \right)^{1/(1-\alpha)}$$

time steps,

$$\mathbb{P}(\mathcal{E}_{good}(u, t)) \geq 1 - \exp\left(-\frac{n}{8k} \right) - 12 \exp\left(-\frac{(1-2\sigma)^4 (\mathcal{G}^*)^2 t^{1-\alpha}}{320} \right).$$

In the above lower bound, the first exponentially decaying term could be thought of as the penalty for not having enough users from the k user types, and the second decaying term could be thought of as the penalty for not yet clustering the users correctly.

The second part of our proof to Theorem 4.4.1 shows that, with high probability, the good neighborhoods have, through random exploration, accurately estimated the probability of liking each item. Thus, we correctly classify each item as likable or not with high probability, which leads to a lower bound on $\mathbb{P}(X_{ut} = 1)$.

Lemma 4.5.2. *For user u at time t, if the good neighborhood event $\mathcal{E}_{good}(u, t)$ holds and $t \leq \zeta m$, then*

$$\mathbb{P}(X_{ut} = 1) \geq 1 - 2m \exp\left(-\frac{(1-2\sigma)^2 tn^{1-\alpha}}{160km} \right) - \frac{1}{t^{\alpha}} - \frac{1}{n^{\alpha}}.$$

Here, the first exponentially decaying term could be thought of as the cost of not classifying items correctly as likable or unlikable, and the last two decaying terms together could be thought of as the cost of exploration (we explore with probability $\varepsilon_J(t) + \varepsilon_R(n) = 1/t^{\alpha} + 1/n^{\alpha}$).

We defer the proofs of Lemmas 4.5.1 and 4.5.2 to Sections 4.5.1 and 4.5.2. Combining these lemmas and choosing appropriate constraints on the numbers of users and items, we produce the following lemma.

Lemma 4.5.3. *Let $\delta \in (0, 1)$ be a pre-specified tolerance. If the number of users n and items m satisfy*

$$n \geq \max\left\{ 8k \log\frac{4}{\delta}, \left(\frac{4}{\delta}\right)^{1/\alpha} \right\},$$

$$\zeta m \geq t \geq \max\left\{\left(\frac{32\log(20kmn^{\alpha}/(1-2\sigma))}{(1-2\sigma)^4(\mathcal{G}^*)^2}\right)^{1/(1-\alpha)}, \left(\frac{320\log(96/\delta)}{(1-2\sigma)^4(\mathcal{G}^*)^2}\right)^{1/(1-\alpha)}, \left(\frac{4}{\delta}\right)^{1/\alpha}\right\},$$
$$nt^{1-\alpha} \geq \frac{160km}{(1-2\sigma)^2}\log\left(\frac{16m}{\delta}\right),$$

then $\mathbb{P}(X_{ut}=1) \geq 1-\delta$.

Proof. With the above conditions on n and t satisfied, we combine Lemmas 4.5.1 and 4.5.2 to obtain

$$\begin{aligned}\mathbb{P}(X_{ut}=1) &\geq 1-\exp\left(-\frac{n}{8k}\right)-12\exp\left(-\frac{(1-2\sigma)^4(\mathcal{G}^*)^2t^{1-\alpha}}{320}\right)\\ &\quad -2m\exp\left(-\frac{(1-2\sigma)^2tn^{1-\alpha}}{160km}\right)-\frac{1}{t^{\alpha}}-\frac{1}{n^{\alpha}}\\ &\geq 1-\frac{\delta}{4}-\frac{\delta}{8}-\frac{\delta}{8}-\frac{\delta}{4}-\frac{\delta}{4}=1-\delta.\end{aligned}$$

■

Theorem 4.4.1 follows as a corollary to Lemma 4.5.3. As previously mentioned, we take the number of users to satisfy $n=\mathcal{O}(m^C)$, separating users into separate pools as necessary. Then with number of users n satisfying

$$\mathcal{O}(m^C)=n=\Omega\left(km\log\frac{1}{\delta}+\left(\frac{4}{\delta}\right)^{1/\alpha}\right),$$

and for any time step t satisfying

$$\zeta m \geq t \geq \Theta\left(\left(\frac{\log\frac{km}{(1-2\sigma)\delta}}{(1-2\sigma)^4(\mathcal{G}^*)^2}\right)^{1/(1-\alpha)}+\left(\frac{4}{\delta}\right)^{1/\alpha}\right) \triangleq T_{\text{learn}},$$

we simultaneously meet all of the conditions of Lemma 4.5.3. Note that the upper bound on number of users n appears since without it, T_{learn} would depend on n (observe that in Lemma 4.5.3, we ask that t be greater than a quantity that depends on n). Provided that the time horizon satisfies $T \leq \zeta m$, then

$$\frac{r_+^{(T)}}{Tn} \geq \frac{1}{Tn}\sum_{t=T_{\text{learn}}}^{T}\sum_{u=1}^{n}\mathbb{P}(X_{ut}=1) \geq \frac{1}{Tn}\sum_{t=T_{\text{learn}}}^{T}\sum_{u=1}^{n}(1-\delta)=\frac{(T-T_{\text{learn}})(1-\delta)}{T},$$

yielding the theorem statement. ■

We present the proofs of Lemmas 4.5.1 and 4.5.2 next. In our derivations, if it is clear from context, we omit argument (t) indexing time, for example writing Y_u instead of $Y_u(t)$.

■ 4.5.1 Proof of Lemma 4.5.1

We reproduce Lemma 4.5.1 below for ease of presentation.

Lemma 4.5.1. *For user u, after*

$$t \geq \left(\frac{32 \log(20kmn^{\alpha}/(1-2\sigma))}{(1-2\sigma)^4 (\mathcal{G}^*)^2} \right)^{1/(1-\alpha)}$$

time steps,

$$\mathbb{P}(\mathcal{E}_{good}(u,t)) \geq 1 - \exp\left(-\frac{n}{8k}\right) - 12\exp\left(-\frac{(1-2\sigma)^4(1-(1-\mathcal{G}^*))^2 t^{1-\alpha}}{320}\right).$$

To derive this lower bound on the probability that the good neighborhood event $\mathcal{E}_{\text{good}}(u,t)$ occurs, we prove four lemmas (Lemmas 4.5.4, 4.5.5, 4.5.6, and 4.5.7). We begin by ensuring that enough users from each of the k user types are in the system.

Lemma 4.5.4. *For a user u,*

$$\mathbb{P}\Big(\textit{user } u\textit{'s type has} \leq \frac{n}{2k} \textit{ users}\Big) \leq \exp\left(-\frac{n}{8k}\right).$$

Proof. This is simply Lemma 2.2.1 with every cluster occurring with probability $1/k$. ■

Next, we ensure in Lemma 4.5.5 that sufficiently many items have been jointly explored across all users. This will subsequently be used for bounding both the number of good neighbors and the number of bad neighbors. To prove Lemma 4.5.5, we will use the following fact.

Fact 4.5.1 (Bernstein's inequality). *Suppose $X_1, \ldots, X_n$ are independent with finite expectations, and that there exists a constant C such that $|X_u - \mathbb{E}[X_u]| \leq C$ for all $u \in \{1, \ldots, n\}$. Then for any $s > 0$,*

$$\mathbb{P}\Big(\sum_{u=1}^{n} X_u \geq \mathbb{E}\Big[\sum_{u=1}^{n} X_u\Big] + s\Big) \leq \exp\left(-\frac{\frac{1}{2}s^2}{\sum_{u=1}^{n} var(X_u) + \frac{1}{3}Cs}\right).$$

Lemma 4.5.5. *After t time steps,*

$$\mathbb{P}(\textit{fewer than } t^{1-\alpha}/2 \textit{ jointly explored items}) \leq \exp(-t^{1-\alpha}/20).$$

Proof. Let Z_s be the indicator random variable for the event that the algorithm jointly explores at time s. Thus, the number of jointly explored items up to time t is $\sum_{s=1}^{t} Z_s$. By our choice for the time-varying joint exploration probability ε_J, we have $\mathbb{P}(Z_s = 1) = \varepsilon_J(s) = \frac{1}{s^{\alpha}}$ and $\mathbb{P}(Z_s = 0) = 1 - \frac{1}{s^{\alpha}}$. Note that the centered random variable $\overline{Z}_s = \mathbb{E}[Z_s] - Z_s = \frac{1}{s^{\alpha}} - Z_s$ has zero mean, and $|\overline{Z}_s| \leq 1$ with probability 1. Then,

$$\begin{aligned}
\mathbb{P}\left(\sum_{s=1}^{t} Z_s < \frac{1}{2}t^{1-\alpha}\right) &= \mathbb{P}\left(\sum_{s=1}^{t} \overline{Z}_s > \sum_{s=1}^{t} \mathbb{E}[Z_s] - \frac{1}{2}t^{1-\alpha}\right) \overset{(1)}{\leq} \mathbb{P}\left(\sum_{s=1}^{t} \overline{Z}_s > \frac{1}{2}t^{1-\alpha}\right) \\
&\overset{(2)}{\leq} \exp\left(-\frac{\frac{1}{8}t^{2(1-\alpha)}}{\sum_{s=1}^{t} \mathbb{E}[\overline{Z}_s^2] + \frac{1}{6}t^{1-\alpha}}\right) \overset{(3)}{\leq} \exp\left(-\frac{\frac{1}{8}t^{2(1-\alpha)}}{\frac{t^{1-\alpha}}{1-\alpha} + \frac{1}{6}t^{1-\alpha}}\right)
\end{aligned}$$

$$= \exp\left(-\frac{3(1-\alpha)t^{1-\alpha}}{4(7-\alpha)}\right) \overset{(4)}{\leq} \exp(-t^{1-\alpha}/20),$$

where step (1) uses the fact that $\sum_{s=1}^{t}\mathbb{E}[Z_s] = \sum_{s=1}^{t} 1/s^{\alpha} \geq t/t^{\alpha} = t^{1-\alpha}$, step (2) is Bernstein's inequality, step (3) uses the fact that $\sum_{s=1}^{t}\mathbb{E}[\overline{Z}_s^2] \leq \sum_{s=1}^{t}\mathbb{E}[Z_s^2] = \sum_{s=1}^{t} 1/s^{\alpha} \leq t^{1-\alpha}/(1-\alpha)$, and step (4) uses the fact that $\alpha \leq 4/7$. (We remark that the choice of constant 4/7 isn't special; changing it would simply modify the constant in the decaying exponentially to potentially no longer be 1/20). ■

Assuming that the bad events for the previous two lemmas do not occur, we now provide a lower bound on the number of good neighbors that holds with high probability. In what follows, the following function that decays with time appears numerous times, so we give it a name:

$$\xi \triangleq \exp\left(-\frac{1}{16}(1-2\sigma)^4(\mathcal{G}^*)^2 t^{1-\alpha}\right). \tag{4.6}$$

Lemma 4.5.6. *Suppose that the low noise condition holds with parameter σ, that there are more than $\frac{n}{2k}$ users of user u's type, and that all users have rated at least $t^{1-\alpha}/2$ items as part of joint exploration. For user u, let n_{good} be the number of "good" neighbors of user u. If $\xi \leq \frac{1}{10}$, then*

$$\mathbb{P}\left(n_{good} \leq (1-\xi)\frac{n}{4k}\right) \leq 10\xi.$$

We defer the proof of Lemma 4.5.6 to Section 4.5.3.

Finally, we verify that the number of bad neighbors for any user is not too large, again conditioned on there being enough jointly explored items.

Lemma 4.5.7. *Suppose that the minimum number of rated items in common between any pair of users is $t^{1-\alpha}/2$ and suppose that the cosine separation condition holds with parameter $\mathcal{G}^*$. For user u, let n_{bad} be the number of "bad" neighbors of user u. Then*

$$\mathbb{P}(n_{bad} \geq n\sqrt{\xi}) \leq \sqrt{\xi}.$$

We defer the proof of Lemma 4.5.7 to Section 4.5.4.

We now prove Lemma 4.5.1, which union bounds over the four bad events of Lemmas 4.5.4, 4.5.5, 4.5.6, and 4.5.7. Recall that the good neighborhood event $\mathcal{E}_{\text{good}}(u,t)$ holds if at time t, user u has more than $\frac{n}{5k}$ good neighbors and fewer than $\frac{(1-2\sigma)tn^{1-\alpha}}{20km}$ bad neighbors. By assuming that the four bad events don't happen, then Lemma 4.5.6 tells us that there are more than $(1-\xi)\frac{n}{4k}$ good neighbors provided that $\xi \leq \frac{1}{10}$. Thus, to ensure that there are more than $\frac{n}{5k}$ good neighbors, it suffices to have $(1-\xi)\frac{n}{4k} \geq \frac{n}{5k}$, which happens when $\xi \leq \frac{1}{5}$, but we already require that $\xi \leq \frac{1}{10}$. Similarly, Lemma 4.5.7 tells us that there are fewer than $n\sqrt{\xi}$ bad neighbors, so to ensure that there are fewer than $\frac{(1-2\sigma)tn^{1-\alpha}}{20km}$ bad neighbors it suffices to have $n\sqrt{\xi} \leq \frac{(1-2\sigma)tn^{1-\alpha}}{20km}$, which happens when

$\xi \le (\frac{(1-2\sigma)t}{20kmn^\alpha})^2$. We can satisfy all constraints on ξ by asking that $\xi \le (\frac{1-2\sigma}{20kmn^\alpha})^2$, which is tantamount to asking that

$$t \ge \left(\frac{32 \log(20kmn^\alpha/(1-2\sigma))}{(1-2\sigma)^4(\mathcal{G}^*)^2} \right)^{1/(1-\alpha)}$$

since $\xi = \exp(-\frac{1}{16}(1-2\sigma)^4(\mathcal{G}^*)^2 t^{1-\alpha})$.

Finally, with t satisfying the inequality above, the union bound over the four bad events can be further bounded to complete the proof:

$$\begin{aligned}\mathbb{P}(\mathcal{E}_{\text{good}}(u,t)) &\ge 1 - \exp\left(-\frac{n}{8k}\right) - \exp(-t^{1-\alpha}/20) - 10\xi - \sqrt{\xi} \\ &\ge 1 - \exp\left(-\frac{n}{8k}\right) - 12\exp\left(-\frac{(1-2\sigma)^4(\mathcal{G}^*)^2 t^{1-\alpha}}{320}\right).\end{aligned}$$

■

■ 4.5.2 Proof of Lemma 4.5.2

We reproduce Lemma 4.5.2 below.

Lemma 4.5.2. *For user u at time t, if the good neighborhood event $\mathcal{E}_{good}(u,t)$ holds and $t \le \zeta m$, then*

$$\mathbb{P}(X_{ut} = 1) \ge 1 - 2m\exp\left(-\frac{(1-2\sigma)^2 t n^{1-\alpha}}{160km}\right) - \frac{1}{t^\alpha} - \frac{1}{n^\alpha}.$$

To prove this, we will use the following fact multiple times.

Fact 4.5.2 (Hoeffding's inequality). *Suppose $X_1, \dots, X_n$ are independent random variables, and X_u takes on a value in the interval $[a_u, b_u]$. Then for any $s > 0$,*

$$\mathbb{P}\left(\frac{1}{n}\sum_{u=1}^n X_u \ge \mathbb{E}\left[\frac{1}{n}\sum_{u=1}^n X_u\right] + s\right) \le \exp\left(-\frac{2n^2 s^2}{\sum_{u=1}^n (b_u - a_u)^2}\right).$$

We begin by checking that when the good neighborhood event $\mathcal{E}_{\text{good}}(u,t)$ holds for user u, the items have been rated enough times by the good neighbors.

Lemma 4.5.8. *For user u at time t, suppose that the good neighborhood event $\mathcal{E}_{good}(u,t)$ holds. Then for a given item i,*

$$\mathbb{P}\left(\textit{item } i \textit{ has} \le \frac{tn^{1-\alpha}}{10km} \textit{ ratings from good neighbors of } u\right) \le \exp\left(-\frac{tn^{1-\alpha}}{40km}\right).$$

Proof. The number of user u's good neighbors who have rated item i stochastically dominates a Binomial$(\frac{n}{5k}, \frac{\varepsilon_R(n)t}{m})$ random variable, where $\frac{\varepsilon_R(n)t}{m} = \frac{t}{mn^\alpha}$ (here, we have critically used the lower bound on the number of good neighbors user u has when the good neighborhood event $\mathcal{E}_{\text{good}}(u,t)$ holds). By a Chernoff bound for the binomial distribution,

$$\mathbb{P}\left(\text{Binomial}\left(\frac{n}{5k}, \frac{t}{mn^\alpha}\right) \le \frac{tn^{1-\alpha}}{10km}\right) \le \exp\left(-\frac{1}{2}\frac{(\frac{tn^{1-\alpha}}{5km} - \frac{tn^{1-\alpha}}{10km})^2}{\frac{tn^{1-\alpha}}{5km}}\right) \le \exp\left(-\frac{tn^{1-\alpha}}{40km}\right).$$

■

Next, we show a sufficient condition for which the algorithm correctly classifies every item as likable or unlikable for user u.

Lemma 4.5.9. *Suppose that the low noise condition holds with parameter σ. For user u at time t, suppose that the good neighborhood event $\mathcal{E}_{good}(u,t)$ holds. Provided that every item $i \in \{1,\dots,m\}$ has more than $\frac{tn^{1-\alpha}}{10km}$ ratings from good neighbors of user u, then with probability at least $1 - m\exp(-\frac{(1-2\sigma)^2 tn^{1-\alpha}}{80km})$, we have that for every item $i \in \{1,\dots,m\}$,*

$$\widetilde{p}_{ui} > \frac{1}{2} \quad \text{if item } i \text{ is likable by user } u,$$
$$\widetilde{p}_{ui} < \frac{1}{2} \quad \text{if item } i \text{ is unlikable by user } u.$$

Proof. Let A be the number of ratings that good neighbors of user u have provided. Suppose item i is likable by user u. Then when we condition on $A = a_0 \triangleq \lceil \frac{tn^{1-\alpha}}{10km} \rceil$, $\widetilde{p}_{ui}$ stochastically dominates

$$q_{ui} \triangleq \frac{\text{Binomial}(a_0, p_{ui})}{a_0 + (\frac{1}{2} - \sigma)a_0} = \frac{\text{Binomial}(a_0, p_{ui})}{(1 + (\frac{1}{2} - \sigma))a_0},$$

which is the worst-case variant of $\widetilde{p}_{ui}$ that insists that all $(\frac{1}{2} - \sigma)a_0$ bad neighbors provided rating "-1" for likable item i (here, we have critically used the upper bound on the number of bad neighbors user u has when the good neighborhood event $\mathcal{E}_{\text{good}}(u,t)$ holds). Then

$$\begin{aligned}
\mathbb{P}(q_{ui} \le \frac{1}{2} \mid A = a_0) &= \mathbb{P}\left(\text{Binomial}(a_0, p_{ui}) \le \frac{(1 + (\frac{1}{2} - \sigma))a_0}{2} \;\middle|\; A = a_0 \right) \\
&= \mathbb{P}\left(a_0 p_{ui} - \text{Binomial}(a_0, p_{ui}) \ge a_0 \left(p_{ui} - \frac{1}{2} - \frac{(\frac{1}{2} - \sigma)}{2} \right) \;\middle|\; A = a_0 \right) \\
&\overset{(1)}{\le} \exp\left(-2a_0 \left(p_{ui} - \frac{1}{2} - \frac{(\frac{1}{2} - \sigma)}{2} \right)^2 \right) \\
&\overset{(2)}{\le} \exp\left(-\frac{1}{2} a_0 \left(\frac{1}{2} - \sigma \right)^2 \right) \\
&\overset{(3)}{\le} \exp\left(-\frac{(\frac{1}{2} - \sigma)^2 tn^{1-\alpha}}{20km} \right),
\end{aligned}$$

where step (1) is Hoeffding's inequality, step (2) follows from item i being likable by user u (i.e., $p_{ui} \ge 1 - \sigma$), and step (3) is by our choice of a_0. Conclude then that

$$\mathbb{P}(\widetilde{p}_{ui} \le \frac{1}{2} \mid A = a_0) \le \exp\left(-\frac{(\frac{1}{2} - \sigma)^2 tn^{1-\alpha}}{20km} \right) = \exp\left(-\frac{(1-2\sigma)^2 tn^{1-\alpha}}{80km} \right).$$

Finally,

$$\mathbb{P}\left(\widetilde{p}_{ui} \le \frac{1}{2} \;\middle|\; A \ge \frac{tn^{1-\alpha}}{10km} \right) = \frac{\sum_{a=a_0}^{\infty} \mathbb{P}(A = a)\mathbb{P}(\widetilde{p}_{ui} \le \frac{1}{2} \mid A = a)}{\mathbb{P}(A \ge \frac{tn^{1-\alpha}}{10km})}$$

$$\leq \frac{\sum_{a=a_0}^{\infty} \mathbb{P}(A = a) \exp(-\frac{(1-2\sigma)^2 t n^{1-\alpha}}{80km})}{\mathbb{P}(A \geq \frac{tn^{1-\alpha}}{10km})}$$
$$= \exp\Big(-\frac{(1-2\sigma)^2 t n^{1-\alpha}}{80km}\Big).$$

A similar argument holds for when item i is unlikable. Union-bounding over all m items yields the claim. ■

We now prove Lemma 4.5.2. First off, provided that $t \leq \zeta m$, we know that there must still exist an item likable by user u that user u has yet to consume. For user u at time t, supposing that event $\mathcal{E}_{\text{good}}(u,t)$ holds, then every item has been rated more than $\frac{tn^{1-\alpha}}{10km}$ times by the good neighbors of user u with probability at least $1 - m\exp(-\frac{tn^{1-\alpha}}{40km})$. This follows from union-bounding over the m items with Lemma 4.5.8. Applying Lemma 4.5.9, and noting that we only exploit with probability $1 - \varepsilon_I(t) - \varepsilon_R(n) = 1 - 1/t^\alpha - 1/n^\alpha$, we finish the proof:

$$\mathbb{P}(X_{ut} = 1) \geq 1 - m\exp\Big(-\frac{tn^{1-\alpha}}{40km}\Big) - m\exp\Big(-\frac{(1-2\sigma)^2 t n^{1-\alpha}}{80km}\Big) - \frac{1}{t^\alpha} - \frac{1}{n^\alpha}$$
$$\geq 1 - 2m\exp\Big(-\frac{(1-2\sigma)^2 t n^{1-\alpha}}{160km}\Big) - \frac{1}{t^\alpha} - \frac{1}{n^\alpha}.$$ ■

■ 4.5.3 Proof of Lemma 4.5.6

We begin with a preliminary lemma that upper-bounds the probability of two users of the same type not being declared as neighbors.

Lemma 4.5.10. *Suppose that the low noise condition holds with parameter σ. Let users u and v be of the same type, and suppose that they have rated at least Γ_0 items in common (explored jointly). Then for $\theta \in (0, (1-2\sigma)^2)$,*

$$\mathbb{P}(\textit{users } u \textit{ and } v \textit{ are not declared as neighbors}) \leq \exp\Big(-\frac{((1-2\sigma)^2 - \theta)^2}{2}\Gamma_0\Big).$$

Proof. Let us first suppose that users u and v have rated exactly Γ_0 items in common. The two users are not declared to be neighbors if $\langle \widetilde{Y}_u, \widetilde{Y}_v \rangle < \theta\Gamma_0$. Let $\Omega \subseteq \{1, \ldots, m\}$ such that $|\Omega| = \Gamma_0$. We have

$$\mathbb{E}\big[\langle \widetilde{Y}_u, \widetilde{Y}_v \rangle \big| \text{supp}(\widetilde{Y}_u) \cap \text{supp}(\widetilde{Y}_v) = \Omega\big] = \sum_{i \in \Omega} \mathbb{E}[\widetilde{Y}_{ui}\widetilde{Y}_{vi} \mid \widetilde{Y}_{ui} \neq 0, \widetilde{Y}_{vi} \neq 0]$$
$$= \sum_{i \in \Omega} (p_{ui}^2 + (1-p_{ui})^2 - 2p_{ui}(1-p_{ui}))$$
$$= 4\sum_{i \in \Omega} \Big(p_{ui} - \frac{1}{2}\Big)^2. \tag{4.7}$$

Since $\langle \widetilde{Y}_u, \widetilde{Y}_v \rangle = \sum_{i\in\Omega} \widetilde{Y}_{ui}\widetilde{Y}_{vi}$ is the sum of terms $\{\widetilde{Y}_{ui}\widetilde{Y}_{vi}\}_{i\in\Omega}$ that are each bounded within $[-1,1]$, Hoeffding's inequality yields

$$\mathbb{P}\big(\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \le \theta\Gamma_0 \;\big|\; \mathrm{supp}(\widetilde{Y}_u)\cap\mathrm{supp}(\widetilde{Y}_v) = \Omega\big) \le \exp\left(-\frac{\big[\overbrace{4\textstyle\sum_{i\in\Omega}\big(p_{gi}-\frac{1}{2}\big)^2}^{\text{equation (4.7)}} - \theta\Gamma_0\big]^2}{2\Gamma_0}\right). \tag{4.8}$$

Under the low noise condition with parameter σ, we have $(\frac{1}{2}-\sigma) \le |p_{ui}-1/2|$ for all users u and items i. Thus, our choice of θ guarantees that

$$4\sum_{i\in\Omega}\left(p_{ui}-\frac{1}{2}\right)^2 - \theta\Gamma_0 \ge 4\Gamma_0\left(\frac{1}{2}-\sigma\right)^2 - \theta\Gamma_0 = ((1-2\sigma)^2-\theta)\Gamma_0 \ge 0. \tag{4.9}$$

Combining inequalities (4.8) and (4.9), and observing that the above holds for all subsets Ω of cardinality Γ_0, we obtain the desired bound on the probability that users u and v are not declared as neighbors:

$$\mathbb{P}(\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \le \theta\Gamma_0 \mid |\mathrm{supp}(\widetilde{Y}_u)\cap\mathrm{supp}(\widetilde{Y}_v)| = \Gamma_0) \le \exp\left(-\frac{((1-2\sigma)^2-\theta)^2}{2}\Gamma_0\right). \tag{4.10}$$

Now to handle the case that users u and v have jointly rated more than Γ_0 items, observe that, with shorthand $\Gamma_{uv} \triangleq |\mathrm{supp}(\widetilde{Y}_u)\cap\mathrm{supp}(\widetilde{Y}_v)|$,

$$\begin{aligned}
&\mathbb{P}(u \text{ and } v \text{ not declared neighbors} \,|\, p_u = p_v, \Gamma_{uv}\ge\Gamma_0)\\
&= \mathbb{P}(\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle < \theta\Gamma_{uv} \mid p_u = p_v,\ \Gamma_{uv}\ge\Gamma_0)\\
&= \frac{\mathbb{P}(\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \le \theta\Gamma_{uv}, \Gamma_{uv}\ge\Gamma_0 \mid p_u = p_v)}{\mathbb{P}(\Gamma_{uv}\ge\Gamma_0 \mid p_u = p_v)}\\
&= \frac{\sum_{\ell=\Gamma_0}^m \mathbb{P}(\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \le \theta\ell, \Gamma_{uv}=\ell \mid p_u = p_v)}{\mathbb{P}(\Gamma_{uv}\ge\Gamma_0 \mid p_u = p_v)}\\
&= \frac{\sum_{\ell=\Gamma_0}^m \begin{matrix}\big[\mathbb{P}(\Gamma_{uv}=\ell \mid p_u=p_v)\\ \cdot\mathbb{P}(\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \le \theta\ell \mid p_u=p_v, \Gamma_{uv}=\ell)\big]\end{matrix}}{\mathbb{P}(\Gamma_{uv}\ge\Gamma_0 \mid p_u = p_v)}\\
&\le \frac{\sum_{\ell=\Gamma_0}^m \mathbb{P}(\Gamma_{uv}=\ell \mid p_u=p_v)\exp\big(-\frac{((1-2\sigma)^2-\theta)^2}{2}\Gamma_0\big)}{\mathbb{P}(\Gamma_{uv}\ge\Gamma_0 \mid p_u = p_v)}\\
&\quad \text{by inequality (4.10)}\\
&= \exp\left(-\frac{((1-2\sigma)^2-\theta)^2}{2}\Gamma_0\right).
\end{aligned}$$

■

We now prove Lemma 4.5.6. Suppose that the event in Lemma 4.5.4 holds. Let $\mathcal{G}$ be $\frac{n}{2k}$ users from the same user type as user u; there could be more than $\frac{n}{2k}$ such users but it suffices to consider $\frac{n}{2k}$ of them. We define an indicator random variable

$$G_v \triangleq \mathbb{1}\{\text{users } u \text{ and } v \text{ are neighbors}\} = \mathbb{1}\{\langle \widetilde{Y}_u^{(t)}, \widetilde{Y}_v^{(t)}\rangle \ge \theta t^{1-\alpha}/2\}.$$

Thus, the number of good neighbors of user u is lower-bounded by $W = \sum_{v \in \mathcal{G}} G_v$. Note that the G_v's are not independent. To arrive at a lower bound for W that holds with high probability, we use Chebyshev's inequality:

$$\mathbb{P}(W - \mathbb{E}[W] \leq -\mathbb{E}[W]/2) \leq \frac{4\mathrm{Var}(W)}{(\mathbb{E}[W])^2}. \tag{4.11}$$

Let $\xi = \exp(-((1-2\sigma)^2 - \theta)^2 \Gamma_0/2)$ be the probability bound from Lemma 4.5.10. By our choice of $\theta = \frac{1}{2}(1-2\sigma)^2\mathcal{G}^*$ and with $\Gamma_0 = \frac{t^{1-\alpha}}{2}$, we have $\xi = \exp(-\frac{1}{16}(1-2\sigma)^4(\mathcal{G}^*)^2 t^{1-\alpha})$, which turns out to be equal to what ξ was defined as in Section 4.5.1.

Applying Lemma 4.5.10, we have $\mathbb{E}[W] \geq (1-\xi)\frac{n}{2k}$, and hence

$$(\mathbb{E}[W])^2 \geq (1-2\xi)\frac{n^2}{4k^2}. \tag{4.12}$$

We now upper-bound

$$\mathrm{Var}(W) = \sum_{v \in \mathcal{G}} \mathrm{Var}(G_v) + \sum_{v \neq w} \mathrm{Cov}(G_v, G_w).$$

Since $G_v = G_v^2$,

$$\mathrm{Var}(G_v) = \mathbb{E}[G_v] - \mathbb{E}[G_v]^2 = \underbrace{\mathbb{E}[G_v]}_{\leq 1}(1 - \mathbb{E}[G_v]) \leq \xi,$$

where the last step uses Lemma 4.5.10.

Meanwhile,

$$\mathrm{Cov}(G_v, G_w) = \mathbb{E}[G_v G_w] - \mathbb{E}[G_v]\mathbb{E}[G_w] \leq 1 - (1-\xi)^2 \leq 2\xi.$$

Putting together the pieces,

$$\mathrm{Var}(W) \leq \frac{n}{2k} \cdot \xi + \frac{n}{2k} \cdot \left(\frac{n}{2k} - 1\right) \cdot 2\xi \leq \frac{n^2}{2k^2} \cdot \xi. \tag{4.13}$$

Plugging (4.12) and (4.13) into (4.11) gives

$$\mathbb{P}(W - \mathbb{E}[W] \leq -\mathbb{E}[W]/2) \leq \frac{8\xi}{1-2\xi} \leq 10\xi,$$

provided that $\xi \leq \frac{1}{10}$. Thus, $n_{\text{good}} \geq W \geq \mathbb{E}[W]/2 \geq (1-\xi)\frac{n}{4k}$ with probability at least $1 - 10\xi$. ■

■ 4.5.4 Proof of Lemma 4.5.7

We begin with a preliminary lemma that upper-bounds the probability of two users of different types being declared as neighbors.

Lemma 4.5.11. *Let users u and v be of different types, and suppose that they have rated at least Γ_0 items in common via joint exploration. Further suppose that the cosine separation condition holds with parameter $\mathcal{G}^*$. If $\theta \geq (1-\mathcal{G}^*)(1-2\sigma)^2$, then*

$$\mathbb{P}(\textit{users } u \textit{ and } v \textit{ are declared to be neighbors}) \leq \exp\Big(-\frac{(\theta-(1-\mathcal{G}^*)(1-2\sigma)^2)^2}{2}\Gamma_0\Big).$$

Proof. As with the proof of Lemma 4.5.10, we first analyze the case where users u and v have rated exactly Γ_0 items in common. Users u and v are declared to be neighbors if $\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \geq \theta\Gamma_0$. We now crucially use the fact that joint exploration chooses these Γ_0 items as a random subset of the m items. For our random permutation ω of m items, we have $\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle = \sum_{i=1}^{\Gamma_0} \widetilde{Y}_{u,\omega(i)}\widetilde{Y}_{v,\omega(i)} = \sum_{i=1}^{\Gamma_0} Y_{u,\omega(i)}Y_{v,\omega(i)}$, which is the sum of terms $\{Y_{u,\omega(i)}Y_{v,\omega(i)}\}_{i=1}^{\Gamma_0}$ that are each bounded within $[-1,1]$ and drawn without replacement from a population of all possible items. Hoeffding's inequality (which also applies to the current scenario of sampling without replacement (Hoeffding, 1963)) yields

$$\mathbb{P}\big(\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \geq \theta\Gamma_0 \mid p_u \neq p_v\big) \leq \exp\left(-\frac{\big(\theta\Gamma_0 - \mathbb{E}[\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \mid p_u \neq p_v]\big)^2}{2\Gamma_0}\right). \tag{4.14}$$

By the cosine separation condition (with parameter $\mathcal{G}^*$) and our choice of θ,

$$\theta\Gamma_0 - \mathbb{E}\big[\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \mid p_u \neq p_v\big] \geq \theta\Gamma_0 - (1-\mathcal{G}^*)(1-2\sigma)^2\Gamma_0 = (\theta-(1-\mathcal{G}^*)(1-2\sigma)^2)\Gamma_0 \geq 0. \tag{4.15}$$

Above, we used the fact that Γ_0 randomly explored items are a random subset of m items, and hence

$$\mathbb{E}\big[\tfrac{1}{\Gamma_0}\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle\big] = \mathbb{E}\big[\tfrac{1}{m}\langle Y_u^*, Y_v^*\rangle\big],$$

with Y_u^*, Y_v^* representing the fully-revealed (random) vector of ratings of users u and v respectively.

Combining inequalities (4.14) and (4.15) yields

$$\mathbb{P}\big(\langle \widetilde{Y}_u, \widetilde{Y}_v\rangle \geq \theta\Gamma_0 \mid p_u \neq p_v\big) \leq \exp\Big(-\frac{(\theta-(1-\mathcal{G}^*)(1-2\sigma)^2)^2}{2}\Gamma_0\Big).$$

A similar argument as the ending of Lemma 4.5.10's proof establishes that the bound holds even if users u and v have jointly explored more than Γ_0 items. ■

We now prove Lemma 4.5.7. Let $\xi = \exp(-(\theta-(1-\mathcal{G}^*)(1-2\sigma)^2)^2\Gamma_0/2)$ be the probability bound from Lemma 4.5.11. By our choice of $\theta = \frac{1}{2}(1-2\sigma)^2\mathcal{G}^*$ and with $\Gamma_0 = \frac{t^{1-\alpha}}{2}$, we have $\xi = \exp(-\frac{1}{16}(1-2\sigma)^4(\mathcal{G}^*)^2 t^{1-\alpha})$, which turns out to be equal to what ξ was defined as in Section 4.5.1.

By Lemma 4.5.11, for a pair of users u and v with at least $t^{1-\alpha}/2$ items jointly explored, the probability that they are erroneously declared neighbors is upper-bounded by ξ.

Denote the set of users of type different from u by $\mathcal{B}$, and write

$$n_{\text{bad}} = \sum_{v \in \mathcal{B}} \mathbb{1}\{u \text{ and } v \text{ are declared to be neighbors}\},$$

whence $\mathbb{E}[n_{\text{bad}}] \leq n\xi$. Markov's inequality gives

$$\mathbb{P}(n_{\text{bad}} \geq n\sqrt{\xi}) \leq \frac{\mathbb{E}[n_{\text{bad}}]}{n\sqrt{\xi}} \leq \frac{n\xi}{n\sqrt{\xi}} = \sqrt{\xi}\,. \qquad \blacksquare$$

■ 4.6 Experimental Results

We demonstrate our algorithms Simple-Collaborative-Greedy and Collaborative-Greedy on two datasets, showing that they have comparable performance and that they both outperform two existing recommendation algorithms Popularity Amongst Friends (PAF) (Barman and Dabeer, 2012) and Deshpande and Montanari's method (DM) (Deshpande and Montanari, 2013). At each time step, PAF finds nearest neighbors ("friends") for every user and recommends to a user the "most popular" item, i.e., the one with the most number of +1 ratings, among the user's friends. DM doesn't do any collaboration beyond a preprocessing step that computes item feature vectors via matrix completion. Then during online recommendation, DM learns user feature vectors over time with the help of item feature vectors and recommends an item to each user based on whether it aligns well with the user's feature vector.

We simulate an online recommendation system based on movie ratings from the Movielens10m and Netflix datasets, each of which provides a sparsely filled user-by-movie rating matrix with ratings out of 5 stars. Unfortunately, existing collaborative filtering datasets such as the two we consider don't offer the interactivity of a real online recommendation system, nor do they allow us to reveal the rating for an item that a user didn't actually rate. For simulating an online system, the former issue can be dealt with by simply revealing entries in the user-by-item rating matrix over time. We address the latter issue by only considering a dense "top users vs. top items" subset of each dataset. In particular, we consider only the "top" users who have rated the most number of items, and the "top" items that have received the most number of ratings. While this dense part of the dataset is unrepresentative of the rest of the dataset, it does allow us to use actual ratings provided by users without synthesizing any ratings. A rigorous validation would require an implementation of an actual interactive online recommendation system, which is beyond the scope of this thesis.

An initial question to ask is whether the dense movie ratings matrices we consider could be reasonably explained by our latent source model. We automatically learn the structure of these matrices using the method by Grosse et al. (2012) and find Bayesian clustered tensor factorization (BCTF) to accurately model the data. This finding isn't surprising as BCTF has previously been used to model movie ratings data (Sutskever et al., 2009). BCTF effectively clusters both users and movies so that we get structure such as that shown in Figure 4.1 for the Movielens10m "top users vs. top items" matrix.

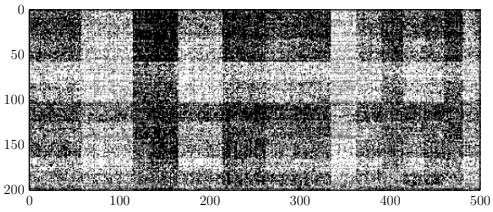


Figure 4.1: Top users by top movies ratings matrix with rows and columns reordered to show clustering of users and items for the Movielens10m dataset.

Our latent source model could reasonably model movie ratings data as it only assumes clustering of users.

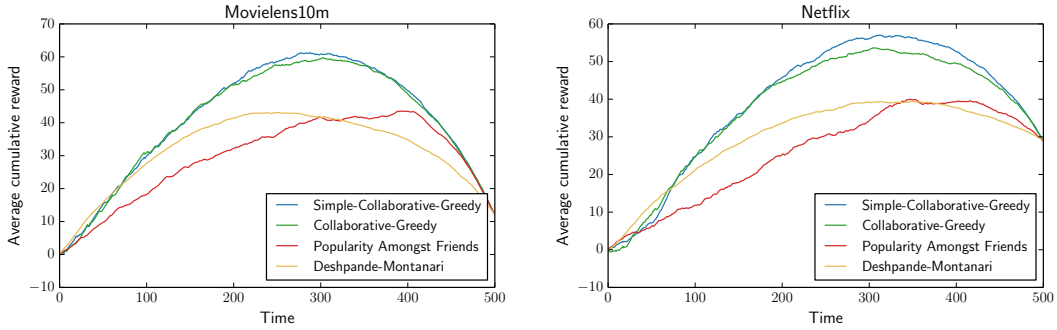
Following the experimental setup of Barman and Dabeer (2012), we quantize a rating of 4 stars or more to be $+1$ (likable), and a rating of 3 stars or less to be -1 (unlikable). While we look at a dense subset of each dataset, there are still missing entries. If a user u hasn't rated item j in the dataset, then we set the corresponding true rating to 0, meaning that in our simulation, upon recommending item j to user u, we receive 0 reward, but we still mark that user u has consumed item j; thus, item j can no longer be recommended to user u. For both Movielens10m and Netflix datasets, we consider the top $n = 200$ users and the top $m = 500$ movies. For Movielens10m, the resulting user-by-rating matrix has 80.7% nonzero entries. For Netflix, the resulting matrix has 86.0% nonzero entries. For an algorithm that recommends item ψ_{ut} to user u at time t, we measure the algorithm's average cumulative reward up to time T as

$$\frac{1}{n}\sum_{t=1}^{T}\sum_{u=1}^{n} Y_{u\psi_{ut}}^{(T)},$$

where we average over users.

For all methods, we recommend items until we reach time $T = 500$, i.e., we make movie recommendations until each user has seen all $m = 500$ movies. We disallow the matrix completion step for DM to see the users that we actually test on, but we allow it to see the the same items as what is in the simulated online recommendation system in order to compute these items' feature vectors (using the rest of the users in the dataset). Furthermore, when a rating is revealed, we provide DM both the thresholded and non-thresholded ratings, the latter of which DM uses to estimate user feature vectors.

Parameters θ and α for Simple-Collaborative-Greedy and Collaborative-Greedy are chosen using training data: We sweep over the two parameters on training data consisting of 200 users that are the "next top" 200 users, i.e., ranked 201 to 400 in number movie ratings they provided. For simplicity, we discretize our search space to $\theta \in \{0.0, 0.1, \ldots, 1.0\}$ and $\alpha \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. We choose the parameter setting achieving the highest area under the cumulative reward curve. For both Movielens10m and Netflix datasets, this corresponded to setting $\theta = 0.1$ and $\alpha = 0.5$ for Simple-Collaborative-Greedy, and $\theta = 0.0$ and $\alpha = 0.5$ for Collaborative-Greedy. In contrast,



(a) (b)

Figure 4.2: Average cumulative rewards over time: (a) Movielens10m, (b) Netflix.

the parameters for PAF and DM are chosen to be the best parameters for the test data among a wide range of parameters. The results are shown in Figure 4.2. We find that our algorithms Simple-Collaborative-Greedy and Collaborative-Greedy have similar performance, and both outperform PAF and DM. We remark that the curves are roughly concave, which is expected since once we've finished recommending likable items (roughly around time step 300), we end up recommending mostly unlikable items until we've exhausted all the items.

■ 4.7 Discussion

Our model for online recommendation systems clearly oversimplifies how any such real system would work but enables one to analyze the performance of recommendation algorithms. Moreover, it could provide a benchmark: a good recommendation algorithm should at least be able to perform well under our model. We highlight four aspects unaccounted for by our model that are promising future directions for exploration:

- *Item similarity.* First, for the actual revealed user ratings, we make very few assumptions, mainly just assuming clustering of ratings across users, but item ratings also cluster, which is incredibly important for making recommendations. For example, if Alice likes the movie *Star Wars: A New Hope*, then she probably also likes the other two movies in the original *Star Wars* trilogy. Incorporating item similarity leads to what is called *item-item* collaborative filtering (in contrast to the *user-user* collaborative filtering we have analyzed, which looks at user similarity). Theory for item-item collaborative filtering is discussed in (Voloch, 2015).

- *Item diversification.* Disallowing an item from being recommended a second time to the same user is a hard diversifying constraint. One could consider a soft constraint that penalizes recommending an item multiple times to the same user, at which point it becomes apparent that some items should be penalized more than others. For

example, if Alice buys a rice cooker, she probably does not want to buy another rice cooker, at least not in the near future. So if there are multiple rice cookers, all of them should be downweighted for being recommended to Alice again. What is a good way to model this item diversity?

- *User experience.* One could ask why we slowly decay the probabilities of exploration initially rather than just doing a phase of only random exploration and only joint exploration. Nearly all of our proof would remain identical. The issue is that such an algorithm fundamentally would give bad user experience, especially since the initial number of exploration steps would, from the theoretical analysis, be a bit conservative. In reality, if a user were forced to explore initially for many time steps, they would just leave the system! How do we incorporate this user experience as part of the recommendation objective?

- *Temporal dynamics.* Users come and go. Items come and go. User preferences change. These are the realities encountered by a real online recommendation system. How do we model all of these dynamics yet retain ease of analyzing recommendation algorithms under the model? What would an optimal algorithm be under this setting, and could a nonparametric approach be optimal?

We end by relating our problem setup to that of learning mixture distributions (c.f., Chaudhuri and Rao (2008); Belkin and Sinha (2010); Moitra and Valiant (2010); Anandkumar et al. (2012)), where one observes samples from a mixture distribution and the goal is to learn the mixture components and weights. Existing results assume that one has access to the entire high-dimensional sample or that the samples are produced in an exogenous manner (not chosen by the algorithm). Neither assumption holds in our setting, as we only see each user's revealed ratings thus far and not the user's entire preference vector, and the recommendation algorithm affects which samples are observed (by choosing which item ratings are revealed for each user). These two aspects make our setting more challenging than the standard setting for learning mixture distributions. However, our goal is more modest. Rather than learning the k item preference vectors, we settle for classifying them as likable or unlikable. Despite this, we suspect having two types of exploration to be useful in general for efficiently learning mixture distributions in the active learning setting.

Chapter 5

Patch-Based Image Segmentation

Delineating objects in an image is a fundamental inference task that feeds into a flurry of analyses downstream, from parsing what is happening in a scene to discovering indicators for disease in the case of medical image analysis. To learn what a particular object looks like, whether a cat or a tumor, we rely on seeing examples of the object. Luckily, we now have access to enormous, ever-growing repositories of images of virtually everything, for example, on photo sharing websites like Flickr, contained within videos on YouTube, or privately stored in hospitals in the case of medical images. Often we could easily procure in excess of thousands of training images for a particular object, perhaps with the help of crowdsourcing manual labeling via Amazon Mechanical Turk.

At the outset though, thousands or even millions of images might seem small since the space of possible images is enormous. Consider a two-dimensional image of 100-by-100 pixels, where at each pixel, we store one of two values. The number of such images is 2^{10000}, which happens to be larger than the estimated number of atoms in the observable universe ($\approx 10^{80}$). Nearly all of these possible images would be completely implausible to us as naturally occurring based on what we see with our eyes day-to-day. What structure is present in an image that makes it "plausible", and how do we exploit this structure for inference?

In this chapter, we focus on the simple setting of segmenting out where a foreground object of interest is from the background, a problem referred to as binary image segmentation. Our running example is the task of finding a human organ in a medical image. Specifically for medical image segmentation, nearest-neighbor and weighted majority voting methods have been widely used, originally at the pixel (or voxel, for 3D images) level (Depa et al., 2010; Sabuncu et al., 2010) and more recently for image patches (Bai et al., 2013; Coupé et al., 2011; Rousseau et al., 2011; Wachinger et al., 2014). Of course, in the extreme case, a patch can just be a single pixel, so patch-based approaches subsume pixel-based approaches.

We specifically study patch-based binary image segmentation, which sits at a middle ground between other patch-based inference tasks and our earlier nonparametric time series classification case study. To relate to patch-based inference tasks beyond binary image segmentation, note that rather than predicting a binary label per pixel, we could predict real-valued labels or patches of such labels leading to patch-based methods for image denoising, reconstruction, restoration, and synthesis (e.g., Buades et al. (2005); Rousseau and Studholme (2013); Konukoglu et al. (2013); Iglesias et al. (2013)). As

these patch-based methods are all syntactically similar, analysis of the binary image segmentation setting could possibly be extended to handle these more sophisticated tasks as well. To relate to the earlier time series classification work, consider a myopic approach to patch-based segmentation, where we predict the label of each pixel using only the observed image patch centered at that pixel. This patch could be viewed as a time series! Thus, our main result for times series classification could help explain the performance of myopic nearest-neighbor-like segmentation. In doing so, we want to account for image structure: pixels nearby, as well as their associated patches, are similar. One of our key contributions in this chapter is to introduce such a structural property.

For medical image segmentation in particular, perhaps the primary reason for the popularity of nonparametric patch-based methods is that the main competing approach to the problem, called label fusion, requires robust alignment of images, whereas patch-based methods do not. Specifically, standard label fusion techniques begin by first aligning all the images into a common coordinate frame and henceforth effectively assuming the alignment to be perfect. Because alignment quality substantially affects label fusion's segmentation performance, alignments are typically computed by so-called *nonrigid registration* that estimates a highly nonlinear transformation between two images. When nonrigid registration works well, then label fusion also works well. The main issue is that nonrigid registration often fails when the images exhibit large anatomical variablity as in whole body CT scans, or when the images have other quality problems commonly found in clinical scans taken at hospitals, such as low resolution and insufficient contrast (Sridharan, 2015). Unlike standard label fusion, patch-based methods do not require input images to be aligned perfectly, making them a promising alternative when the images we encounter present significant alignment challenges.

Of course, another reason for patch-based methods' growing popularity is their efficiency of computation, due to increasing availability of fast approximate nearest-neighbor search algorithms both for general high-dimensional spaces (as discussed back in Section 1.1) and tailored specifically for image patches (Barnes et al., 2009; Ta et al., 2014). Thus, if the end goal is segmentation or a decision based on segmentation, for many problems solving numerous nonrigid registration subproblems required for standard label fusion could be a computationally expensive detour that, even if successful, might not produce better solutions than a patch-based approach.

Many patch-based image segmentation methods are precisely variants of the nearest-neighbor-like inference methods we have seen throughout the rest of this thesis. In the simplest case, to determine whether a pixel in an input image should be labeled as the foreground object of interest or background, we consider the patch centered at that pixel. We compare this image patch to patches in a training database that are labeled labeled either foreground or background depending on the pixel at the center of the training patch. We transfer the label from the closest patch in the training database to the pixel of interest in the new image. A plethora of embellishments improve this algorithm, such as, but not limited to, using K nearest neighbors or weighted majority voting instead of only the nearest neighbor (Coupé et al., 2011; Rousseau et al., 2011; Wachinger et al.,

2014), incorporating hand-engineered or learned feature descriptors (Wachinger et al., 2014), cleverly choosing the shape of a patch (Wachinger et al., 2014), and enforcing consistency among adjacent pixels by assigning each training intensity image patch to a label patch rather than a single label (Rousseau et al., 2011; Wachinger et al., 2014) or by employing a Markov random field (Freeman and Liu, 2011).

Despite the broad popularity and success of nonparametric patch-based image segmentation and the smorgasbord of tricks to enhance its performance, the existing work has been empirical with no theoretical justification for when and why such methods should work and, if so, how well and with how much training data. In this chapter, we provide this theoretical justification for several patch-based nearest-neighbor-like segmentation algorithms. As with the other case studies, our theoretical guarantees depend on an underlying probabilistic model. We build up our model in two stages, first incorporating local structure and then modeling global structure.

We begin in Section 5.1 with a simple case of our model that corresponds to inferring each pixel's label separately from other pixels. For this special case of so-called pointwise segmentation, we provide theoretical performance guarantees for patch-based nearest-neighbor and weighted majority voting segmentation in terms of the available training data. Our analysis borrows heavily from the time series classification work in Chapter 3, and crucially relies on the newly introduced local structural property called the *jigsaw condition*, which holds when neighboring patches are sufficiently similar.

We present our full model in Section 5.2 and derive a new iterative patch-based image segmentation algorithm that combines ideas from patch-based image restoration (Zoran and Weiss, 2011) and distributed optimization (Boyd et al., 2011). This algorithm alternates between inferring label patches separately and merging these local estimates to form a globally consistent segmentation. We show how various existing patch-based algorithms can be viewed as special cases of this new algorithm, and present experimental results showing that it significantly outperforms the pointwise segmentation algorithms that we provide theoretical guarantees for.

■ 5.1 Pointwise Segmentation

For an image A, we use $A(i)$ to denote the value of image A at pixel i, and $A[i]$ to denote the patch of image A centered at pixel i based on a pre-specified patch shape; $A[i]$ can include feature descriptors in addition to raw intensity values. Each pixel i belongs to a finite, uniformly sampled lattice $\mathcal{I}$.

Given an intensity image Y, we aim to infer its label image L that delineates an object of interest in image Y. In particular, for each pixel $i \in \mathcal{I}$, we infer label $L(i) \in \{+1, -1\}$, where $+1$ corresponds to foreground (object of interest) and -1 to background. To make this inference, we use patches of image Y, each patch of dimensionality d. For example, for a 2D image, if we use 5x5 patches, then $d = 25$, and for a 3D image, if we use 5x5x5 patches, then $d = 125$.

We model the joint distribution $p(L(i), Y[i])$ of label $L(i)$ and image patch $Y[i]$ as a generalization of a Gaussian mixture model with diagonal covariances, where each mixture

component corresponds to either $L(i) = +1$ or $L(i) = -1$. We call this generalization a *diagonal sub-Gaussian mixture model*, to be described shortly. First, we provide a concrete example where label $L(i)$ and patch $Y[i]$ are related through a Gaussian mixture model with k_i mixture components. Mixture component $c \in \{1, \ldots, k_i\}$ occurs with probability $\pi_{ic} \in (0, 1]$ and has mean vector $\mu_{ic} \in \mathbb{R}^d$ and label $\lambda_{ic} \in \{+1, -1\}$. In this example, we assume that all covariance matrices are $\sigma^2 \mathbf{I}_{d\times d}$, and that there exists constant $\pi_{\min} > 0$ such that $\pi_{ic} \geq \pi_{\min}$ for all i, c. Thus, image patch $Y[i]$ belongs to mixture component c with probability π_{ic}, in which case $Y[i] = \mu_{ic} + W_i$, where vector $W_i \in \mathbb{R}^d$ consists of white Gaussian noise with variance σ^2, and $L(i) = \lambda_{ic}$. Formally,

$$p(L(i), Y[i]) = \sum_{c=1}^{k_i} \pi_{ic} \mathcal{N}(Y[i]; \mu_{ic}, \sigma^2 \mathbf{I}_{d\times d}) \mathbb{1}\{L(i) = \lambda_{ic}\},$$

where $\mathcal{N}(\cdot; \mu, \Sigma)$ is a Gaussian density with mean μ and covariance Σ, and $\mathbb{1}\{\cdot\}$ is the indicator function. Each pixel i has its own mixture model with parameters $(\pi_{ic}, \mu_{ic}, \lambda_{ic})$ for $c = 1, \ldots, k_i$. The *diagonal sub-Gaussian mixture model* refers to a simple generalization where noise vector W_i consists of entries that are i.i.d. zero-mean sub-Gaussian with parameter σ. This generalization precisely corresponds to our latent source model for time series classification where we disallow time shifts (so $\Delta_{\max} = 0$) and the time horizon T is replaced by the patch dimensionality d.

We assume that every pixel is associated with its own diagonal sub-Gaussian mixture model whose parameters are fixed but unknown. Similar to recent work on modeling natural imagery patches (Zoran and Weiss, 2011, 2012), we do not model how different overlapping patches behave jointly and instead only model how each individual patch, viewed alone, behaves. In natural imagery, image patches turn out to very accurately behave like samples from a Gaussian mixture model (Zoran and Weiss, 2012).

As with our other case studies, rather than learning the mixture model components, we instead take a nonparametric approach, using available training data in nearest-neighbor or weighted majority voting schemes to infer label $L(i)$ from image patch $Y[i]$. To this end, we assume we have access to n i.i.d. training intensity-label *image* pairs $(Y_1, L_1), \ldots, (Y_n, L_n)$ that obey our probabilistic model above.

■ 5.1.1 Inference

We translate the three nonparametric time series classifiers from Section 3.1 to the patch-based image segmentation setting. The resulting methods operate on each pixel i separately, inferring label $L(i)$ only based on image patch $Y[i]$:

- *Pointwise nearest-neighbor segmentation* first finds which training intensity image Y_u has a patch centered at pixel j that is closest to observed intensity patch $Y[i]$. This amounts to solving

$$(\widehat{u}, \widehat{j}) = \operatorname*{arg\,min}_{u\in\{1,2,\ldots,n\}, j\in N(i)} \|Y_u[j] - Y[i]\|^2,$$

where $\|\cdot\|$ denotes Euclidean norm, and $N(i)$ refers to a user-specified finite set of pixels that are neighbors of pixel i. Label $L(i)$ is estimated to be the same as the closest training patch's label:

$$\widehat{L}_{\mathrm{NN}}(i|Y[i]) = L_{\widehat{u}}(\widehat{j}).$$

- *Pointwise weighted majority voting segmentation* first computes the following weighted votes for labels $+1$ and -1:

$$V_{+1}(i|Y[i];\theta) = \sum_{u=1}^{n} \sum_{j\in N(i)} \exp\big(-\theta\|Y_u[j]-Y[i]\|^2\big)\mathbb{1}\{L_u(j)=+1\},$$
$$V_{-1}(i|Y[i];\theta) = \sum_{u=1}^{n} \sum_{j\in N(i)} \exp\big(-\theta\|Y_u[j]-Y[i]\|^2\big)\mathbb{1}\{L_u(j)=-1\},$$

  where θ is a scale parameter, and $N(i)$ again refers to user-specified neighboring pixels of pixel i. Label $L(i)$ is estimated to be the label with the higher vote:

$$\widehat{L}_{\mathrm{WMV}}(i|Y[i];\theta) = \begin{cases} +1 & \text{if } V_{+1}(i|Y[i];\theta) \geq V_{-1}(i|Y[i];\theta), \\ -1 & \text{otherwise.} \end{cases}$$

  We obtain pointwise nearest-neighbor segmentation with $\theta \to \infty$. For an identical reason as in the time series classification setting (c.f., Section 3.3), the prediction made at each pixel i by pointwise weighted majority voting segmentation approximates an oracle MAP classifier that knows the parameters for the diagonal sub-Gaussian mixture model at pixel i. This oracle MAP classifier is myopic, inferring label $L(i)$ only given patch $Y[i]$.

  Pointwise weighted majority voting has been used extensively for patch-based segmentation (Bai et al., 2013; Coupé et al., 2011; Rousseau et al., 2011; Wachinger et al., 2014), where we note that our formulation readily allows for one to choose which training image patches are considered neighbors, what the patch shape is, and whether feature descriptors are part of the intensity patch vector $Y[i]$. For example, a simple choice of feature at pixel i is the coordinate for pixel i itself. Thus, we can encode as part of the exponentially decaying weight $\exp(-\theta\|Y_u[j]-Y[i]\|^2)$ how far apart pixels i and $j \in N(i)$ are, yielding a weighted voting algorithm previously derived from a Bayesian model that explicitly models this displacement (Bai et al., 2013).

- *Pointwise generalized weighted majority voting segmentation* biases the votes for label -1 by scaling it by $\tau > 0$:

$$\widehat{L}_{\tau}(i|Y[i];\theta) = \begin{cases} +1 & \text{if } V_{+1}(i|Y[i];\theta) \geq \tau V_{-1}(i|Y[i];\theta), \\ -1 & \text{otherwise,} \end{cases}$$

  where choosing $\tau = 1$ yields pointwise weighted majority voting segmentation.

■ 5.1.2 Theoretical Guarantees

The model above allows nearby pixels to be associated with dramatically different mixture models. However, real images are smooth, with patches centered at two adjacent pixels likely similar. We incorporate this smoothness via a structural property on the sub-Gaussian mixture model parameters associated with nearby pixels.

To build some intuition, we consider two extremes. First, it could be that the $|\mathcal{I}|$ mixture models (one per pixel) are actually all identical. This means that every intensity patch comes from the same distribution. If we know this, then when we do pointwise nearest-neighbor or weighted majority voting segmentation, we could compare an observed intensity patch $Y[i]$ with a training image patch centered *anywhere* in the training image, since the patches all follow the same distribution. On the opposite extreme, the $|\mathcal{I}|$ mixture models could have no commonalities, and so when we do pointwise nearest-neighbor or weighted majority voting segmentation, it only makes sense to compare intensity patch $Y[i]$ with training image patches also centered at pixel i. We can interpolate between these two extremes by saying how far away two pixels have to be while still sharing mixture component parameters, which we formalize as follows:

> *Jigsaw condition.* For every mixture component $(\pi_{ic}, \mu_{ic}, \lambda_{ic})$ of the diagonal sub-Gaussian mixture model associated with pixel i, there exists a neighbor $j \in N^*(i)$ such that the diagonal sub-Gaussian mixture model associated with pixel j also has a mixture component with mean μ_{ic}, label λ_{ic}, and mixture weight at least $\pi_{\min}$; this weight need not be equal to π_{ic}.

The name of this structural property is inspired by a jigsaw puzzle, where the pieces of the puzzle somehow need to fit with nearby pieces to produce the final picture.

The shape and size of neighborhood $N^*(i)$, which is fixed and unknown like the mixture model parameters, control how similar the mixture models are across image pixels. For the two extremes discussed previously, the true neighborhood $N^*(i)$ corresponds to the full space of pixels in the first case, and $N^*(i) = \{i\}$ in the second case. As we already hinted at, what $N^*(i)$ is affects how far from pixel i we should look for training patches, i.e., how to choose neighborhood $N(i)$ in pointwise nearest-neighbor and weighted majority voting segmentation, where ideally $N(i) = N^*(i)$.

As with our main result for time series classification, our main result here depends on the separation gap between training intensity image patches that correspond to the two different labels:

$$\mathcal{G} \triangleq \min_{\substack{u,v \in \{1,\ldots,n\}, \\ i \in \mathcal{I}, j \in N(i) \text{ s.t. } L_u(i) \neq L_v(j)}} \|Y_u[i] - Y_v[j]\|^2.$$

Intuitively, a small separation gap corresponds to the case of two training intensity image patches that are very similar but one corresponds to foreground and the other to background. In this case, a nearest-neighbor approach may easily select a patch with the wrong label, resulting in an error.

We now state our main theoretical result of this chapter. The proof is deferred to Section 5.4.

Theorem 5.1.1. *Let $N(i)$ be the user-specified neighborhood of pixel i. Denote $k \triangleq \max_{i\in\mathcal{I}} k_i$, $|N| \triangleq \max_{i\in\mathcal{I}} |N(i)|$, $\pi_{+1}(i) \triangleq \mathbb{P}(L(i) = +1) = \sum_{c=1}^{k_i} \pi_{ic}\mathbb{1}\{\lambda_{ic} = +1\}$, and $\pi_{-1}(i) \triangleq \mathbb{P}(L(i) = -1) = \sum_{c=1}^{k_i} \pi_{ic}\mathbb{1}\{\lambda_{ic} = -1\}$. Under the model above with n training intensity-label image pairs and provided that the jigsaw condition holds with neighborhood N^* such that $N^*(i) \subseteq N(i)$ for every pixel i:*

(a) Pointwise nearest-neighbor segmentation has expected pixel labeling error rate

$$\mathbb{E}\bigg[\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbb{1}\{\widehat{L}_{NN}(i|Y[i]) \neq L(i)\}\bigg] \leq |\mathcal{I}|k\exp\Big(-\frac{n\pi_{\min}}{8}\Big) + |N|n\exp\Big(-\frac{\mathcal{G}}{16\sigma^2}\Big).$$

(b) Pointwise generalized weighted majority voting segmentation has expected pixel labeling error rate

$$\begin{aligned}&\mathbb{E}\bigg[\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbb{1}\{\widehat{L}_{\tau}(i|Y[i];\theta) \neq L(i)\}\bigg]\\ &\leq |\mathcal{I}|k\exp\Big(-\frac{n\pi_{\min}}{8}\Big) + \frac{|N|n}{|\mathcal{I}|}\exp\big(-(\theta - 4\sigma^2\theta^2)\mathcal{G}\big)\sum_{i\in\mathcal{I}}\Big(\tau\pi_{+1}(i) + \frac{1}{\tau}\pi_{-1}(i)\Big).\end{aligned}$$

As with our time series classification result, these two upper bounds match when $\tau = 1$ (corresponding to regular pointwise weighted majority voting segmentation) and $\theta = \frac{1}{8\sigma^2}$. Thus, to interpret this theorem, we just examine the pointwise nearest-neighbor segmentation bound, considering sufficient conditions for each term on the right-hand side to be at most $\delta/2$, for $\delta \in (0, 1)$. For the first term, the number of training intensity-label image pairs n should be sufficiently large so that we see enough of the different mixture model components in our training data: $n \geq \frac{8}{\pi_{\min}}\log\big(2|\mathcal{I}|k/\delta\big)$. For the second term, the gap $\mathcal{G}$ should be sufficiently large so that the nearest training intensity image patch found does not produce a segmentation error: $\mathcal{G} \geq 16\sigma^2\log\big(2|N|n/\delta\big)$.

There are different ways to change the gap, such as changing the shape of the patch and including hand-engineered or learned patch features. For example, if the mixture models are all Gaussian mixture models, then provided that no true mean vectors of opposite labels are the same, then as shown in inequality (3.18), gap $\mathcal{G}$ turns out to grow as $\Omega(\sigma^2 d)$ (time horizon T in inequality (3.18) corresponds to dimensionality d). Intuitively, using larger patches d should widen the gap. But using larger patches also means that the (maximum) number of mixture components k needed to represent a patch increases, possibly quite dramatically.

As with our main time series classification result, to prevent the second terms in the upper bounds in Theorem 5.1.1 from scaling linearly with n, we could subsample the training data so that n is large enough to capture the diversity of mixture model components yet not so large that it overcomes the gap. In other words, with estimates or bounds on k, σ^2, and $\pi_{\min}$, then collecting $n = \Theta(\frac{1}{\pi_{\min}}\log(|\mathcal{I}|k/\delta))$ training image pairs and with a gap $\mathcal{G} = \Omega\big(\sigma^2\log\big(\frac{|N|}{\pi_{\min}\delta}\log\big(\frac{|\mathcal{I}|k}{\delta}\big)\big)\big)$, both algorithms achieve an expected error rate of at most δ.

■ 5.2 Multipoint Segmentation

We generalize the basic model to infer label patches $L[i]$ rather than just a single pixel's label $L(i)$. Every label patch $L[i]$ is assumed to have dimensionality d', where d and d' need not be equal. For example, $Y[i]$ could be a 5-by-5 patch, whereas $L[i]$ could be a 3-by-3 patch. When $d' > 1$, estimates of label patches must be merged to arrive at a globally consistent estimate of label image L. This case is referred to as multipoint segmentation.

In this general case, we assume there to be K underlying latent label images $\Lambda_1, \ldots, \Lambda_K$ that occur with probabilities $\Pi_1, \ldots, \Pi_K$. To generate intensity image Y, we first sample label image $\Lambda \in \{\Lambda_1, \ldots, \Lambda_K\}$ according to probabilities $\Pi_1, \ldots, \Pi_K$. Then we sample label image L to be a perturbed version of Λ such that $p(L|\Lambda) \propto \exp(-\alpha \mathbf{d}(L, \Lambda))$ for some constant $\alpha \geq 0$ and differentiable "distance" function $\mathbf{d}(\cdot, \cdot)$. For example, $\mathbf{d}(L, \Lambda)$ could relate to volume overlap between the segmentations represented by label images L and Λ with perfect overlap yielding distance 0. Finally, intensity image Y is generated so that for each pixel $i \in \mathcal{I}$, patch $Y[i]$ is a sample from a mixture model patch prior $p(Y[i]|L[i])$. If $\alpha = 0$, $d' = 1$, and the mixture model is diagonal sub-Gaussian, we obtain our earlier model. We remark that this generative model describes, for every pixel i, the joint distribution between intensity image patch $Y[i]$ and the full label image L. As with our pointwise segmentation model, we do not specify how overlapping intensity image patches are jointly distributed.

We refer to this formulation as a *latent source model* since the intensity image patches could be thought of as generated from the latent "prototypical" label images $\Lambda_1, \ldots, \Lambda_K$ combined with the latent mixture model clusters linking $L[i]$ to $Y[i]$. This hierarchical structure enables local appearances around a given pixel to be shared across the prototypical label images.

■ 5.2.1 Inference

We derive an iterative algorithm based on the expected patch log-likelihood (EPLL) framework (Zoran and Weiss, 2011). First, note that our full latent source model for multipoint segmentation prescribes a joint distribution for label image L and image patch $Y[i]$. Thus, assuming that we know the model parameters, the MAP estimate for L given $Y[i]$ is

$$\widehat{L} = \underset{L \in \{+1,-1\}^{|\mathcal{I}|}}{\arg\max} \left\{ \log \left(\sum_{g=1}^{K} \Pi_g \exp(-\alpha \mathbf{d}(L, \Lambda_g)) \right) + \log p(Y[i]|L[i]) \right\}.$$

If we average the objective function above across all pixels i, then we obtain the EPLL objective function, which we approximately maximize to segment an image:

$$\widehat{L} = \underset{L \in \{+1,-1\}^{|\mathcal{I}|}}{\arg\max} \left\{ \log \left(\sum_{g=1}^{K} \Pi_g \exp(-\alpha \mathbf{d}(L, \Lambda_g)) \right) + \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \log p(Y[i]|L[i]) \right\}.$$

The first term in the objective function encourages label image L to be close to the true label images $\Lambda_1, \ldots, \Lambda_K$. The second term is the "expected patch log-likelihood", which favors solutions whose local label patches agree well on average with the local intensity patches according to the patch priors.

Since latent label images $\Lambda_1, \ldots, \Lambda_K$ are unknown, we use training label images $L_1, \ldots, L_n$ as proxies instead, replacing the first term in the objective function with $F(L;\alpha) \triangleq \log\left(\frac{1}{n}\sum_{u=1}^{n}\exp(-\alpha \mathbf{d}(L, L_u))\right)$. Next, we approximate the unknown patch prior $p(Y[i]|L[i])$ with a kernel density estimate

$$\widetilde{p}(Y[i]|L[i];\gamma) \propto \sum_{u=1}^{n}\sum_{j\in\mathcal{N}(i)} \mathcal{N}\left(Y[i]; Y_u[j], \frac{1}{2\gamma}\mathbf{I}_{d\times d}\right)\mathbb{1}\{L[i] = L_u[j]\},$$

where the user specifies a neighborhood $\mathcal{N}(i)$ of pixel i, and constant $\gamma > 0$ that controls the Gaussian kernel's bandwidth. We group the pixels so that nearby pixels within a small block all share the same kernel density estimate. This approximation essentially assumes a stronger version of the jigsaw condition from Section 5.1 since the algorithm operates as if nearby pixels have the same mixture model as a patch prior. Hence, we maximize objective $F(L;\alpha) + \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\log\widetilde{p}(Y[i]|L[i];\gamma)$ to determine label image L.

Similar to the original EPLL method (Zoran and Weiss, 2011), we introduce an auxiliary variable $\xi_i \in \mathbb{R}^{d'}$ for each patch $L[i]$, where ξ_i acts as a local estimate for $L[i]$. Whereas two patches $L[i]$ and $L[j]$ that overlap in label image L must be consistent across the overlapping pixels, there is no such requirement on their local estimates ξ_i and ξ_j. In summary, we solve

$$\widehat{L} = \underset{\substack{L\in\{+1,-1\}^{|\mathcal{I}|},\\ (\xi_i\in\mathbb{R}^{d'}, i\in\mathcal{I})\\ \text{s.t. } L[i]=\xi_i \text{ for } i\in\mathcal{I}}}{\arg\,min} \left\{-F(L;\alpha) - \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\log\widetilde{p}(Y[i]|\xi_i;\gamma) + \frac{\beta}{2}\sum_{i\in\mathcal{I}}\|L[i]-\xi_i\|^2\right\}, \quad (5.1)$$

where $\beta > 0$ is a user-specified constant.

The original EPLL method (Zoran and Weiss, 2011) progressively increases β and, for each choice of β, alternates between updating the label image L and the auxiliary variables ξ_i, ignoring the constraints $L[i] = \xi_i$ for $i \in \mathcal{I}$. The idea is that as β grows large, these constraints will eventually be satisfied. However, it is unclear how to increase β in a principled manner. While heuristics could be used, an alternative approach is to fix β and instead introduce a Lagrange multiplier η_i for each constraint $L[i] = \xi_i$ and iteratively update these Lagrange multipliers. This can be achieved by the Alternating Direction Method of Multipliers (ADMM) for distributed optimization (Boyd et al., 2011). Specifically, we form Lagrangian

$$\mathcal{L}_\beta(L,\xi,\eta) = -F(L;\alpha) - \sum_{i\in\mathcal{I}}\log\widetilde{p}(Y[i]|\xi_i;\gamma) + \frac{\beta}{2}\sum_{i\in\mathcal{I}}\|L[i]-\xi_i\|^2 + \sum_{i\in\mathcal{I}}\eta_i^T(L[i]-\xi_i),$$

where $\eta = (\eta_i \in \mathbb{R}^{d'}, i \in \mathcal{I})$ is the collection of Lagrange multipliers, and $\xi = (\xi_i, i \in \mathcal{I})$ is the collection of auxiliary variables. Indexing iterations with superscripts, the ADMM

update equations are given by:

$$\xi^{t+1} \leftarrow \arg\min_{\xi} \mathcal{L}_\beta(L^t, \xi, \eta^t) \qquad \text{(minimize Lagrangian along direction } \xi\text{)},$$
$$L^{t+1} \leftarrow \arg\min_{L} \mathcal{L}_\beta(L, \xi^{t+1}, \eta^t) \qquad \text{(minimize Lagrangian along direction } L\text{)},$$
$$\eta^{t+1} \leftarrow \eta^t + \beta(\xi^{t+1} - L^{t+1}) \qquad \text{(update Lagrange multipliers } \eta\text{)}.$$

By looking at what terms matter for each update equation, we can rewrite the above three steps as follows:

1. *Update label patch estimates.* We update estimate ξ_i for label patch $L[i]$ given observed image patch $Y[i]$ in parallel across $i \in \mathcal{I}$:

$$\xi_i^{t+1} \leftarrow \arg\min_{\xi_i} \Big\{ -\frac{1}{|\mathcal{I}|} \log \widetilde{p}(Y[i] \| \xi_i; \gamma) + \frac{\beta}{2} \|L^t[i] - \xi_i\|^2 + (\eta_i^t)^T (L^t[i] - \xi_i) \Big\}.$$

   This minimization only considers ξ_i among training label patches for which $\widetilde{p}$ is defined. Thus, this minimization effectively scores different nearest-neighbor training label patches found and chooses the one with the best score.

2. *Merge label patch estimates.* Fixing ξ_i, we update label image L:

$$L^{t+1} \leftarrow \arg\min_{L} \Big\{ -F(L; \alpha) + \frac{\beta}{2} \sum_{i \in \mathcal{I}} \|L[i] - \xi_i^{t+1}\|^2 + \sum_{i \in \mathcal{I}} (\eta_i^t)^T (L[i] - \xi_i^{t+1}) \Big\}.$$

   With the assumption that F is differentiable, gradient methods could be used to numerically solve this subproblem.

3. *Update Lagrange multipliers.* Set $\eta_i^{t+1} \leftarrow \eta_i^t + \beta(\xi_i^{t+1} - L^{t+1}[i])$. This penalizes large discrepancies between ξ_i and $L[i]$.

Parameters α, β, and γ are chosen using held-out data or cross-validation.

Step 2 above corresponds to merging local patch estimates to form a globally consistent segmentation. This is the only step that involves expression $F(L; \alpha)$. With $\alpha = 0$ and forcing the Lagrange multipliers to always be zero, the merging becomes a simple averaging of overlapping label patch estimates ξ_i. This algorithm corresponds to existing multipoint patch-based segmentation algorithms (Coupé et al., 2011; Rousseau et al., 2011; Wachinger et al., 2014) and the in-painting technique achieved by the original EPLL method. Setting $\alpha = \beta = 0$ and $d' = 1$ yields pointwise weighted majority voting with parameter $\theta = \gamma$. When $\alpha > 0$, a global correction is applied, shifting the label image estimate closer to the training label images. This should produce better estimates when the full training label images can, with small perturbations as measured by $\mathbf{d}(\cdot, \cdot)$, explain new intensity images.

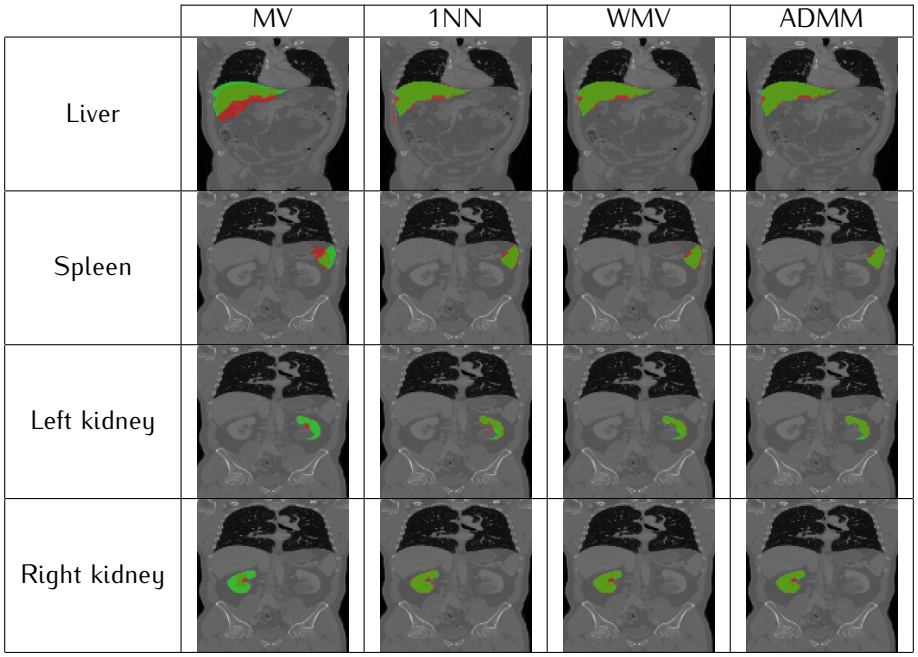
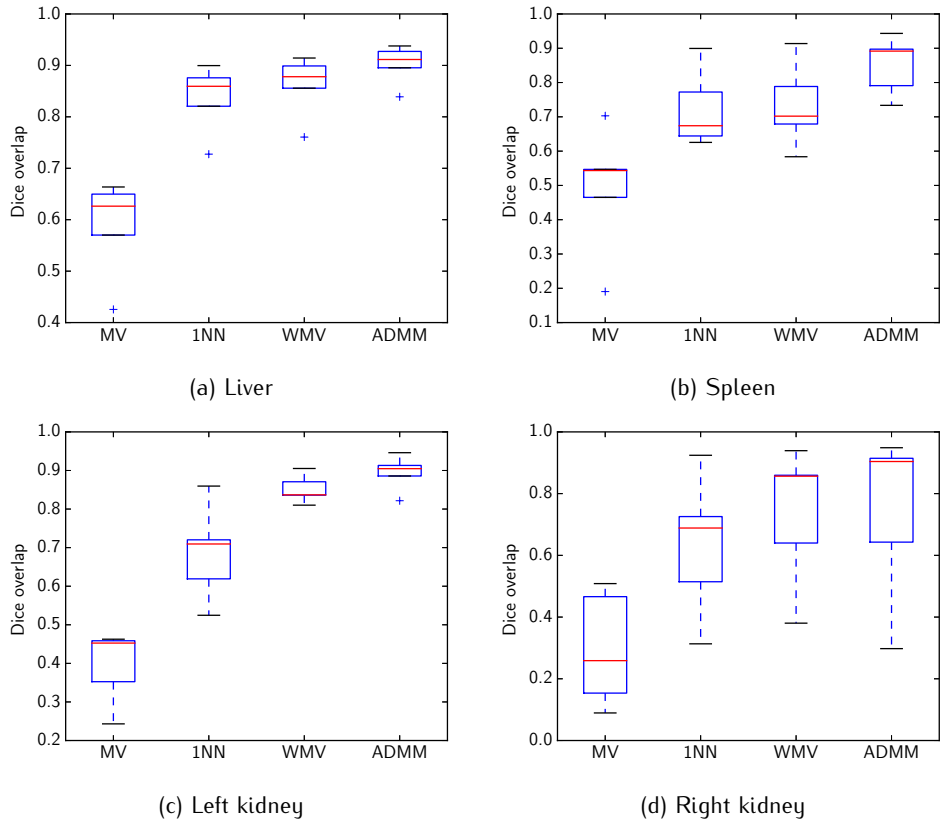


Figure 5.1: Example image segmentation results. Green denotes the ground truth label and red denotes the estimated label, where a good segmentation result has the green and red overlaid on top of each other.

■ 5.3 Experimental Results

We empirically explore the new iterative algorithm on 20 labeled thoracic-abdominal contrast-enhanced CT scans from the Visceral ANATOMY3 dataset (Hanbury et al., 2012). We train the model on 15 scans and test on the remaining 5 scans. The training procedure amounted to using 10 of the 15 training scans to estimate the algorithm parameters in an exhaustive sweep, using the rest of the training scans to evaluate parameter settings. Finally, the entire training dataset of 15 scans is used to segment the test dataset of 5 scans using the best parameters found during training. For each test scan, we first use a fast affine registration to roughly align each training scan to the test scan. Then we apply four different algorithms: a baseline majority voting algorithm (denoted "MV") that simply averages the training label images that are now roughly aligned to the test scan, pointwise nearest neighbor (denoted "1NN") and weighted majority voting (denoted "WMV") segmentation that both use approximate nearest patches, and finally our proposed iterative algorithm (denoted "ADMM"), setting distance $\mathbf{d}$ to one minus Dice overlap. Dice overlap measures volume overlap between the true and estimated pixels of an object, where 1 is perfect overlap and 0 is no overlap. It can be written as a differentiable function by relaxing our optimization to allow each label to take on a value in $[-1, 1]$, in which case the Dice overlap of label images L and Λ is given by $2\langle\widetilde{L}, \widetilde{\Lambda}\rangle / (\langle\widetilde{L}, \widetilde{L}\rangle + \langle\widetilde{\Lambda}, \widetilde{\Lambda}\rangle)$, where $\widetilde{L} = (L+1)/2$ and $\widetilde{\Lambda} = (\Lambda+1)/2$.

We segmented the liver, spleen, left kidney, and right kidney. We show example segmentations in Fig. 5.1 and report the Dice overlap scores for in Fig. 5.2 using the four algorithms. In all cases, the proposed algorithm outperforms pointwise weighted majority voting, which outperforms both pointwise nearest-neighbor segmentation and the baseline majority voting. For the organs segmented, there was little benefit to having $\alpha > 0$, suggesting the local patch estimates to already be quite consistent and require no global correction.



(a) Liver (b) Spleen

(c) Left kidney (d) Right kidney

Figure 5.2: Dice volume overlap scores.

■ 5.4 Proof of Theorem 5.1.1

The proof is nearly identical to that of our main time series classifcation result, Theorem 3.4.1, except that it crucially relies on the jigsaw condition and has extra bookkeeping since there are numerous pixels with associated patches that we are predicting labels for. Thus, we primarily detail the main changes, over sketching the parts of the proof that are the same as that of Theorem 3.4.1.

In what follows, we abbreviate "diagonal sub-Gaussian mixture model" by "MM",

letting MM(i) denote the diagonal sub-Gaussian mixture model associated with pixel i. For predicting the label for a single pixel, we interchangeably use the terms "segment" and "classify".

We begin with a lemma that establishes how many training intensity-label image pairs are sufficient so that for every mixture model (associated with every pixel in $\mathcal{I}$), we see every mixture component enough times.

Lemma 5.4.1. *For the pointwise segmentation model with n training image pairs,*

$$\mathbb{P}\Big(\exists \text{ a pixel } i \text{ for which a mixture component of } MM(i) \text{ occurs} \leq \frac{n\pi_{\min}}{2} \text{ times}\Big) \leq |\mathcal{I}| k \exp\Big(-\frac{n\pi_{\min}}{8}\Big).$$

Proof. Fix pixel $i \in \mathcal{I}$. For training intensity images $Y_1, \ldots, Y_n$, note that patches $Y_1[i], \ldots, Y_n[i]$ behave as if they were drawn from MM(i), which has at most k mixture components each with weight at least $\pi_{\min}$. Thus, by Lemma 2.2.2, we have

$$\mathbb{P}\Big(\exists \text{ mixture component of MM}(i) \text{ occurring} \leq \frac{n\pi_{\min}}{2} \text{ times}\Big) \leq k \exp\Big(-\frac{n\pi_{\min}}{8}\Big).$$

Union-bounding over all $|\mathcal{I}|$ pixels yields the claim. ■

Now let $\mathcal{E}_{\text{all-sources}}$ be the event that for every pixel i and for every mixture component $(\pi_{ic}, \mu_{ic}, \lambda_{ic})$ in MM(i), there exists some training intensity image whose patch at pixel i is generated by mixture component $(\pi_{ic}, \mu_{ic}, \lambda_{ic})$. By the result of Lemma 5.4.1, $\mathbb{P}(\mathcal{E}_{\text{all-sources}}) \geq 1 - |\mathcal{I}| k \exp\big(-\frac{n\pi_{\min}}{8}\big)$. For the remainder of the proof, we assume that event $\mathcal{E}_{\text{all-sources}}$ holds.

An important piece in the proof of Theorem 3.4.1 is writing the time series to be classified in terms of one of the training data. We do the same here, writing patch $Y[i]$ in terms of one of the training patches, for every pixel $i \in \mathcal{I}$.

Fix pixel $i \in \mathcal{I}$. Under the pointwise segmentation model, there exists some mixture component $c \in \{1, \ldots, k_i\}$ of MM(i) that patch $Y[i]$ is generated from. In particular,

$$Y[i] = \mu_{ic} + W_1, \tag{5.2}$$

where noise $W_1 \in \mathbb{R}^d$ consists of i.i.d. zero-mean sub-Gaussian entries with parameter σ. Moreover, the label of patch $Y[i]$ is $\lambda_{ic} \in \{+1, -1\}$.

Using the jigsaw condition, there exists a neighbor $j^* \in N^*(i)$ such that MM(j^*) has a mixture component that also has mean μ_{ic} and label λ_{ic}. Furthermore, since event $\mathcal{E}_{\text{all-sources}}$ holds, we know that there exists some training intensity image Y_{u^*} for which $Y_{u^*}[j^*]$ is generated from a mixture component with mean μ_{ic} and label λ_{ic}. Hence,

$$Y_{u^*}[j^*] = \mu_{ic} + W_2, \tag{5.3}$$

where noise W_2 consists of i.i.d. zero-mean sub-Gaussian entries with parameter σ. Putting together equations (5.2) and (5.3),

$$Y[i] = \mu_{ic} + W_1 = (Y_{u^*}[j^*] - W_2) + W_1 = Y_{u^*}[j^*] + \underbrace{(W_1 - W_2)}_{\triangleq W}, \tag{5.4}$$

where W consists of i.i.d. zero-mean sub-Gaussian entries now with parameter $\sqrt{2}\sigma$.

Pointwise Nearest-Neighbor Segmentation Guarantee

The probability that pointwise nearest-neighbor segmentation misclassifies pixel i is

$$\begin{aligned}
&\mathbb{P}(\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i) \mid \mathcal{E}_{\text{all-sources}}) \\
&= \mathbb{P}(L(i) = +1 \mid \mathcal{E}_{\text{all-sources}})\mathbb{P}(\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i) \mid \mathcal{E}_{\text{all-sources}}, L(i) = +1) \\
&\quad + \mathbb{P}(L(i) = -1 \mid \mathcal{E}_{\text{all-sources}})\mathbb{P}(\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i) \mid \mathcal{E}_{\text{all-sources}}, L(i) = -1).
\end{aligned} \tag{5.5}$$

We upper-bound $\mathbb{P}(\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i) \mid \mathcal{E}_{\text{all-sources}}, L(i) = +1)$. Suppose $L(i) = +1$. Then nearest-neighbor segmentation makes an error and declares $\widehat{L}_{\mathrm{NN}}(i|Y[i]) = -1$ when the closest training intensity patch $Y_{\widehat{u}}[\widehat{j}\,]$ to patch $Y[i]$ has label $L_{\widehat{u}}(\widehat{j}\,) = -1$, where

$$(\widehat{u}, \widehat{j}) = \operatorname*{arg\,min}_{u \in \{1,2,\ldots,n\}, j \in \mathcal{N}(i)} \|Y_u[j] - Y[i]\|^2 = \operatorname*{arg\,min}_{u \in \{1,2,\ldots,n\}, j \in \mathcal{N}(i)} \|Y_u[j] - (Y_{u^*}[j^*] + W)\|^2,$$

and the last step above uses equation (5.4). By optimality of $(\widehat{u}, \widehat{j})$ for the above optimization problem, we have

$$\|Y_u[j] - (Y_{u^*}[j^*] + W)\|^2 \geq \|Y_{\widehat{u}}[\widehat{j}\,] - (Y_{u^*}[j^*] + W)\|^2$$

for all $u \in \{1, 2, \ldots, n\}$ and $j \in \mathcal{N}(i)$. Setting $u = u^*$ and $j = j^*$ (the latter of which we can do because $\mathcal{N}(i) \supseteq \mathcal{N}^*(i)$) and rearranging terms, we obtain

$$2\langle Y_{\widehat{u}}[\widehat{j}\,] - Y_{u^*}[j^*], W\rangle \geq \|Y_{\widehat{u}}[\widehat{j}\,] - Y_{u^*}[j^*]\|^2. \tag{5.6}$$

Pointwise nearest-neighbor segmentation makes an error and declares $\widehat{L}_{\mathrm{NN}}(i|Y[i]) = -1$ when there exists $\widehat{u} \in \{1, \ldots, n\}$ and $\widehat{j} \in \mathcal{N}(i)$ with $L_{\widehat{u}}(\widehat{j}\,) = -1$ such that optimality condition (5.6) holds. Thus, using the same steps as in deriving inequality (3.10),

$$\begin{aligned}
&\mathbb{P}(\widehat{L}_{\mathrm{NN}}(i|Y[i]) = -1 \mid \mathcal{E}_{\text{all-sources}}, L(i) = +1) \\
&\leq \mathbb{P}\left(\bigcup_{\substack{\widehat{u} \in \{1,\ldots,n\}, \\ \widehat{j} \in \mathcal{N}(i) \\ \text{s.t. } L_{\widehat{u}}(\widehat{j}\,) = -1}} \left\{ 2\langle Y_{\widehat{u}}[\widehat{j}\,] - Y_{u^*}[j^*], W\rangle \geq \|Y_{\widehat{u}}[\widehat{j}\,] - Y_{u^*}[j^*]\|^2 \right\} \right) \leq n|\mathcal{N}| \exp\left(-\frac{1}{16\sigma^2}\mathcal{G} \right),
\end{aligned}$$

where instead of the number of possible time shifts $2\Delta_{\max} + 1$ showing up, we have maximum number of possible neighboring pixels $|\mathcal{N}|$. A similar calculation shows that the same bound holds for $\mathbb{P}(\widehat{L}_{\mathrm{NN}}(i|Y[i]) = +1 \mid \mathcal{E}_{\text{all-sources}}, L(i) = -1)$, and so

$$\mathbb{P}(\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i) \mid \mathcal{E}_{\text{all-sources}}) \leq n|\mathcal{N}| \exp\left(-\frac{1}{16\sigma^2}\mathcal{G} \right).$$

Then

$$\mathbb{P}\big(\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i)\big) \leq \mathbb{P}(\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i) \mid \mathcal{E}_{\text{all-sources}}) + \mathbb{P}(\mathcal{E}^c_{\text{all-sources}})$$
$$\leq n|\mathcal{N}| \exp\Big(-\frac{1}{16\sigma^2}\mathcal{G}\Big) + |\mathcal{I}|k \exp\Big(-\frac{n\pi_{\min}}{8}\Big).$$

Finally, we establish the pointwise nearest-neighbor segmentation performance guarantee by noting that the expected pixel mislabeling rate satisfies

$$\mathbb{E}\Big[\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbb{1}\{\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i)\}\Big] = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbb{P}\big(\widehat{L}_{\mathrm{NN}}(i|Y[i]) \neq L(i)\big)$$
$$\leq n|\mathcal{N}| \exp\Big(-\frac{1}{16\sigma^2}\mathcal{G}\Big) + |\mathcal{I}|k \exp\Big(-\frac{n\pi_{\min}}{8}\Big).$$

Pointwise Generalized Weighted Majority Voting Segmentation Guarantee

The proof here is basically identical to that of generalized weighted majority voting for time series classification except with our new definition of event $\mathcal{E}_{\text{all-sources}}$ and where we relate each patch $Y[i]$ to training data via equation (5.4). As a reminder, equation (5.4) relied on good event $\mathcal{E}_{\text{all-sources}}$ holding as well as the jigsaw condition.

As our proof of pointwise nearest-neighbor segmentation suggests, instead of searching over $2\Delta_{\max}+1$ time shifts, we search over at most $|\mathcal{N}|$ neighboring pixels. Thus, the analogue of the generalized weighted majority voting time series classification result in this pointwise segmentation setting, for a single pixel i, is

$$\mathbb{P}\big(\widehat{L}_\tau(i|Y[i]\theta) \neq L(i)\big)$$
$$\leq \mathbb{P}(\widehat{L}_\tau(i|Y[i];\theta) \neq L(i) \mid \mathcal{E}_{\text{all-sources}}) + \mathbb{P}(\mathcal{E}^c_{\text{all-sources}})$$
$$\leq |\mathcal{I}|k \exp\Big(-\frac{n\pi_{\min}}{8}\Big) + \Big(\tau\pi_{+1}(i) + \frac{1}{\tau}\pi_{-1}(i)\Big) n|\mathcal{N}| \exp\big(-(\theta - 4\sigma^2\theta^2)\mathcal{G}\big).$$

Therefore the expected pixel labeling error rate satisfies

$$\mathbb{E}\Big[\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbb{1}\{\widehat{L}_\tau(i|Y[i];\theta) \neq L(i)\}\Big] = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbb{P}\big(\widehat{L}_\tau(i|Y[i];\theta) \neq L(i)\big)$$
$$\leq |\mathcal{I}|k \exp\Big(-\frac{n\pi_{\min}}{8}\Big) + \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \Big(\tau\pi_{+1}(i) + \frac{1}{\tau}\pi_{-1}(i)\Big) n|\mathcal{N}| \exp\big(-(\theta - 4\sigma^2\theta^2)\mathcal{G}\big).$$

■

■ 5.5 Discussion

By introducing the jigsaw condition, we were able to import our theoretical development on nonparametric time series classification into nonparametric patch-based image segmentation. Our main result indicates that if nearby patches behave as mixture models with sufficient similarity, then a myopic segmentation works well, where its quality is stated in terms of the available training data.

From a modeling standpoint, understanding the joint behavior of patches could yield substantial new insights into exploiting macroscopic structure in images rather than relying only on local properties that enable myopic inference. In a related direction, while we have modeled the individual behavior of patches, an interesting theoretical question is to understand what joint distributions on image pixels could possibly lead to such marginal distributions on patches. Do such joint distributions exist? If not, is there a joint distribution whose marginals for patches approximate the mixture models we use? These questions outline rich areas for future research.

Chapter 6

Conclusions and Future Directions

We set out to justify when, why, and how well nearest-neighbor-like inference works, grounding our analysis in three seemingly disparate case studies. For each case study, we imposed structure by modeling data to be generated by unknown latent sources, and the structure turns out to be enough to lead to theoretical performance guarantees for nearest-neighbor-like inference. Across all three case studies, our results asked for the number of training data n to scale linearly with the number of latent sources k modulo log factors, and for there to be a separation gap that combats noise.

6.1 More Training Data, More Problems?

A limitation across the three main theoretical results is that they become weak when the number of training data n grows too large. Our workaround each time was to subsample n to be smaller but in a way that depended on k. Is this really necessary? Is our worst-case analysis too pessimistic?

One way to see why larger n might cause nearest-neighbor inference to struggle is to consider a simple setup where data are generated i.i.d. with equal probability from one of two Gaussians, either $\mathcal{N}(0, \sigma^2)$ or $\mathcal{N}(\mu, \sigma^2)$ for constants $\mu > 0, \sigma > 0$. The goal is to predict which of these two Gaussians a new data point is from with the help of training data $Y_1, \ldots, Y_n$ drawn i.i.d. from the same model with known labels $L_1, \ldots, L_n \in \{+1, -1\}$, where $+1$ corresponds to $\mathcal{N}(0, \sigma^2)$, and -1 corresponds to $\mathcal{N}(\mu, \sigma^2)$. This is the time series classification setting with $T = 1$, no time shifts, and Gaussian noise. When n grows large, with high probability, we'll encounter training data generated from $\mathcal{N}(0, \sigma^2)$ that exceed μ and thus plausibly appear to come from the second Gaussian, and vice versa! This is disastrous as it means that with large amounts of training data, the separation gap $\mathcal{G}^{(T)}$ could become vanishingly small.

To sketch why this mishap happens, first note that Lemma 2.2.2 implies that collecting $n \geq 16 \log(2/\delta)$ training data, then with probability at least $1 - \delta$, we have more than $n/4$ samples from each of the two Gaussians. Assuming this event holds, we next rely on the fact that for random variables $X_1, \ldots, X_d$ drawn i.i.d. from $\mathcal{N}(0, \sigma^2)$ (Kamath, 2015),

$$\sigma \sqrt{\frac{\log d}{\pi \log 2}} \leq \mathbb{E}\big[\max_{u=1,\ldots,d} X_u\big] \leq \sigma \sqrt{2 \log d}.$$

Hence, we know that the maximum of the training data generated from the first Gaussian $\mathcal{N}(0, \sigma^2)$ (for which there are more than $n/4$ such training data) has an expected value of at least $\sigma\sqrt{\frac{\log(n/4)}{\pi \log 2}}$, which for large enough n exceeds μ. One could then apply a concentration inequality to argue that as n grows large, with high probability, there will be at least one training data point generated from $\mathcal{N}(0, \sigma^2)$ that is larger than μ, and thus is more plausibly explained as being generated from the second Gaussian. A similar calculation could be used with the two Gaussians swapped. Increasing n makes it more likely that more of these "bad" training data appear.

■ 6.2 Outlier Screening

The key observation that provides a promising solution to the disaster scenario above is that the "bad" training data that stray far from their corresponding (Gaussian) latent sources are *outliers*. We could thus screen for and remove outliers. For example, one simple nonparametric way to do this is for a training data point to look within a ball around it to find its nearest neighbors, and then to ask whether its label agrees with the most popular label amongst its nearest neighbors; if not, we classify the point as an outlier and discard it. A variant of this method was devised by Wilson (1972), where the label of a point is compared to the most popular label amongst its K nearest neighbors to determine whether to retain a point or not. Outlier screening is an active area of research with numerous other methods devised over the years to empirically improve classification performance (e.g., Sánchez et al. (2003); Smith and Martinez (2011)). While it's unclear how outlier screening integrates with the theory we developed, we suspect that adding it as a preprocessing step could yield nonparametric inference methods with stronger theoretical performance guarantees than those presented in this thesis.

■ 6.3 Fast, Scalable Nonparametric Inference

A separate question is how to actually build fast nonparametric inference methods that can handle massive, ever-growing amounts of training data for applications that might even demand real-time interactive inference for streaming test data. As with the online collaborative filtering setup, test data we see now could become part of our training data in the future! Developing these inference methods requires a significant amount of systems engineering and leads to many theoretical questions of interest. How does using *approximate* nearest-neighbor search (e.g., with locality-sensitive hashing (Gionis et al., 1999)) affect misclassification rate? What other approximations could be made to expedite running time, and could we provide theoretical guarantees for the overall system with easy-to-tune parameters that trade off inference quality, computation time, and the amount of training data needed? Building such a system and answering these theoretical questions could lead to a general-purpose nonparametric inference engine that is fast and broadly applicable, with a solid theoretical foundation characterizing its performance.

Bibliography

Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *Symposium on Theory of Computing*, 2006.

Fabio Aiolli. A preliminary study on a recommender system for the million songs dataset challenge. In *Italian Information Retrieval Workshop*, 2013.

Anima Anandkumar, Rong Ge, Daniel Hsu, Sham M. Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models, 2012. arXiv:1210.7559.

Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Foundations of Computer Science*, 2006.

Jean-Yves Audibert and Alexandre B. Tsybakov. Fast learning rates for plug-in classifiers. *Annals of Statistics*, 2007.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multi-armed bandit problem. *Machine Learning*, 2002.

Anthony Bagnall, Luke Davis, Jon Hills, and Jason Lines. Transformation based ensembles for time series classification. In *International Conference on Data Mining*, 2012.

Wenjia Bai, Wenzhe Shi, Declan P. O'Regan, Tong Tong, Haiyan Wang, Shahnaz Jamil-Copley, Nicholas S. Peters, and Daniel Rueckert. A probabilistic patch-based label fusion model for multi-atlas segmentation with registration refinement: application to cardiac MR images. *IEEE Transactions in Medical Imaging*, 2013.

Kishor Barman and Onkar Dabeer. Analysis of a collaborative filter based on popularity amongst neighbors. *IEEE Transactions on Information Theory*, 2012.

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics*, 2009.

Gustavo E. A. P. A. Batista, Xiaoyue Wang, and Eamonn J. Keogh. A complexity-invariant distance measure for time series. In *International Conference on Data Mining*, 2011.

Mayank Bawa, Tyson Condie, and Prasanna Ganesan. Lsh forest: self-tuning indexes for similarity search. In *International World Wide Web Conference*, 2005.

Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on Twitter. In *International Conference on Weblogs and Social Media*, 2011.

Mikhail Belkin and Kaushik Sinha. Polynomial learning of distribution families. In *Foundations of Computer Science*, 2010.

Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *International Conference on Music Information Retrieval*, 2011.

Gérard Biau, Benoît Cadre, and Laurent Rouvière. Statistical analysis of k-nearest neighbor collaborative recommendation. *Annals of Statistics*, 38(3):1568–1592, 2010.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 2011.

Guy Bresler, George H. Chen, and Devavrat Shah. A latent source model for online collaborative filtering. In *Neural Information Processing Systems*, 2014.

Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A non-local algorithm for image denoising. In *Computer Vision and Pattern Recognition*, 2005.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 2012.

Loc Bui, Ramesh Johari, and Shie Mannor. Clustered bandits, 2012. arXiv:1206.4169.

Valerii V. Buldygin and Yuriy V. Kozachenko. Sub-gaussian random variables. *Ukrainian Mathematical Journal*, 1980.

Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 2010.

Emmanuel J. Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 2009.

Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *International Workshop on Multimedia Data Mining*, 2010.

Wanpracha Art Chaovalitwongse, Ya-Ju Fan, and Rajesh C. Sachdeo. On the time series k-nearest neighbor classification of abnormal brain activity. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, Nov 2007.

Kamalika Chaudhuri and Satish Rao. Learning mixtures of product distributions using correlations and independence. In *Conference on Learning Theory*, 2008.

George H. Chen, Stanislav Nikolov, and Devavrat Shah. A latent source model for nonparametric time series classification. In *Neural Information Processing Systems*, 2013.

George H. Chen, Devavrat Shah, and Polina Golland. A latent source model for patch-based image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, 2015.

Pierrick Coupé, José V. Manjón, Vladimir Fonov, Jens Pruessner, Montserrat Robles, and D. Louis Collins. Patch-based segmentation using expert priors: Application to hippocampus and ventricle segmentation. *NeuroImage*, 2011.

Thomas M. Cover. Rates of convergence for nearest neighbor procedures. In *Hawaii International Conference on Systems Sciences*, 1968.

Thomas M. Cover and Peter E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 1967.

Antonio Criminisi, Jamie Shotton, and Ender Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 2011.

Onkar Dabeer. Adaptive collaborating filtering: The low noise regime. In *International Symposium on Information Theory*, 2013.

Sanjoy Dasgupta and Samory Kpotufe. Optimal rates for k-nn density and mode estimation. In *Neural Information Processing Systems*, 2014.

Sanjoy Dasgupta and Leonard Schulman. A probabilistic analysis of EM for mixtures of separated, spherical gaussians. *Journal of Machine Learning Research*, 2007.

Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Symposium on Computational Geometry*, 2004.

Michal Depa, Mert R. Sabuncu, Godtfred Holmvang, Reza Nezafat, Ehud J. Schmidt, and Polina Golland. Robust atlas-based segmentation of highly variable anatomy: Left atrium segmentation. In *Statistical Atlases and Computational Models of the Heart*, 2010.

Yash Deshpande and Andrea Montanari. Linear bandits in high dimension and recommendation systems, 2013. arXiv:1301.1722.

Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 2008.

Dan Feldman, Matthew Faulkner, and Andreas Krause. Scalable training of mixture models via coresets. In *Neural Information Processing Systems*, 2011.

William T. Freeman and Ce Liu. Markov random fields for super-resolution and texture synthesis. In *Advances in Markov Random Fields for Vision and Image Processing*. MIT Press, 2011.

Keinosuke Fukunaga. *Introduction to statistical pattern recognition (2nd ed.)*. Academic Press Professional, Inc., 1990.

Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits, 2014. arXiv:1401.8257.

Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *International Conference on Very Large Data Bases*, 1999.

Roger B. Grosse, Ruslan Salakhutdinov, William T. Freeman, and Joshua B. Tenenbaum. Exploiting compositionality to explore a large space of model structures. In *Uncertainty in Artificial Intelligence*, 2012.

Peter Hall, Byeong U. Park, and Richard J. Samworth. Choice of neighbor order in nearest-neighbor classification. *Annals of Statistics*, 2008.

Allan Hanbury, Henning Müller, Georg Langs, Marc André Weber, Bjoern H. Menze, and Tomas Salas Fernandez. Bringing the algorithms to the data: cloud–based benchmarking for medical image analysis. In *Conference and Labs of the Evaluation Forum*, 2012.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 1963.

Daniel Hsu and Sham M. Kakade. Learning mixtures of spherical gaussians: Moment methods and spectral decompositions, 2013. arXiv:1206.5766.

Juan Eugenio Iglesias, Ender Konukoglu, Darko Zikic, Ben Glocker, Koen Van Leemput, and Bruce Fischl. Is synthesizing MRI contrast useful for inter-modality analysis? In *Medical Image Computing and Computer-Assisted Intervention*, 2013.

László Kaján, Attila Kertész-Farkas, Dino Franklin, Neli Ivanova, András Kocsor, and Sándor Pongor. Application of a simple likelihood ratio approximant to protein sequence classification. *Bioinformatics*, 2006.

Gautam C. Kamath. Bounds on the expectation of the maximum of samples from a gaussian, 2015. URL http://www.gautamkamath.com/writings/gaussian_max.pdf.

Shiva Prasad Kasiviswanathan, Prem Melville, Arindam Banerjee, and Vikas Sindhwani. Emerging topic detection using dictionary learning. In *Conference on Information and Knowledge Management*, 2011.

Shiva Prasad Kasiviswanathan, Huahua Wang, Arindam Banerjee, and Prem Melville. Online l1-dictionary learning with application to novel document detection. In *Neural Information Processing Systems*, 2012.

Raghunandan H. Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 2010a.

Raghunandan H. Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from noisy entries. *Journal of Machine Learning Research*, 2010b.

Robert Kleinberg, Alexandru Niculescu-Mizil, and Yogeshwer Sharma. Regret bounds for sleeping experts and bandits. *Machine Learning*, 2010.

Ender Konukoglu, André J. W. van der Kouwe, Mert Rory Sabuncu, and Bruce Fischl. Example-based restoration of high-resolution magnetic resonance image acquisitions. In *Medical Image Computing and Computer-Assisted Intervention*, 2013.

Yehuda Koren. The BellKor solution to the Netflix grand prize, 2009. URL http://www.netflixprize.com/assets/GrandPrize2009_BPC_BellKor.pdf.

Samory Kpotufe. k-nn regression adapts to local intrinsic dimension. In *Neural Information Processing Systems*, 2011.

Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, 2000.

Troy Lee and Adi Shraibman. Matrix completion from any given set of observations. In *Neural Information Processing Systems*, 2013.

Yen-Hsien Lee, Chih-Ping Wei, Tsang-Hsiang Cheng, and Ching-Ting Yang. Nearest-neighbor-based approach to time-series classification. *Decision Support Systems*, 2012.

Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 2003.

Enno Mammen and Alexandre B. Tsybakov. Smooth discrimination analysis. *Annals of Statistics*, 1999.

Michael Mathioudakis and Nick Koudas. Twittermonitor: trend detection over the Twitter stream. In *International Conference on Management of Data*, 2010.

Charles Mathy, Nate Derbinsky, José Bento, and Jonathan Yedidia. The boundary forest algorithm for online supervised and unsupervised learning. In *AAAI Conference on Artificial Intelligence*, 2015.

Ankur Moitra and Gregory Valiant. Settling the polynomial learnability of mixtures of gaussians. In *Foundations of Computer Science*, 2010.

Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application*, 2009.

Alex Nanopoulos, Rob Alcock, and Yannis Manolopoulos. Feature-based classification of time-series data. *International Journal of Computer Research*, 2001.

Stanislav Nikolov. Trend or no trend: A novel nonparametric method for classifying timeseries. Master's thesis, Massachusetts Institute of Technology, 2012.

Martin Piotte and Martin Chabbert. The pragmatic theory solution to the netflix grand prize, 2009. URL http://www.netflixprize.com/assets/GrandPrize2009_BPC_PragmaticTheory.pdf.

Benjamin Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 2011.

Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Conference on Computer Supported Cooperative Work*, 1994.

Juan J. Rodríguez and Carlos J. Alonso. Interval and dynamic time warping-based decision trees. In *Symposium on Applied Computing*, 2004.

François Rousseau and Colin Studholme. A supervised patch-based image reconstruction technique: Application to brain MRI super-resolution. In *International Symposium on Biomedical Imaging*, 2013.

François Rousseau, Piotr A. Habas, and Colin Studholme. A supervised patch-based approach for human brain labeling. *IEEE Transactions on Medical Imaging*, 2011.

Mert R. Sabuncu, B. T. Thomas Yeo, Koenraad Van Leemput, Bruce Fischl, and Polina Golland. A generative model for image segmentation based on label fusion. *IEEE Transactions on Medical Imaging*, 2010.

José Salvador Sánchez, Ricardo Barandela, Al Marqués, Roberto Alejo, and Jorge Badenas. Analysis of new techniques to obtain quality training sets. *Pattern Recognition Letters*, 2003.

Nandakishore Santhi and Alexander Vardy. On an improvement over Rényi's equivocation bound. In *Allerton Conference on Communication, Control, and Computing*, 2006.

Michael R. Smith and Tony Martinez. Improving classification accuracy by identifying and removing instances that should be misclassified. In *International Joint Conference on Neural Networks*, 2011.

Ramesh Sridharan. *Visualization and Analysis of Large Medical Image Collections Using Pipelines*. PhD thesis, Massachusetts Institute of Technology, 2015.

Ilya Sutskever, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Modelling relational data using bayesian clustered tensor factorization. In *Neural Information Processing Systems*, 2009.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Vinh-Thong Ta, Rémi Giraud, D. Louis Collins, and Pierrick Coupé. Optimized patch-match for near real time and accurate label fusion. In *Medical Image Computing and Computer-Assisted Intervention*, 2014.

William R. Thompson. On the Likelihood that one Unknown Probability Exceeds Another in View of the Evidence of Two Samples. *Biometrika*, 1933.

Andreas Töscher and Michael Jahrer. The BigChaos solution to the Netflix Grand Prize, 2009. URL http://www.netflixprize.com/assets/GrandPrize2009_BPC_BigChaos.pdf.

Alexandre B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 2004.

Santosh Vempala and Grant Wang. A spectral algorithm for learning mixture models. *Journal of Computer and System Sciences*, 2004.

Luis F. Voloch. Collaborative filtering with low regret. Master's thesis, Massachusetts Institute of Technology, 2015.

Christian Wachinger, Matthew Brennan, Greg C. Sharp, and Polina Golland. On the importance of location and features for the patch-based segmentation of parotid glands. In *MICCAI Workshop on Image-Guided Adaptive Radiation Therapy*, 2014.

Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 2009.

Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man and Cybernetics*, 1972.

Yi Wu and Edward Y. Chang. Distance-function design and fusion for sequence data. In *Conference on Information and Knowledge Management*, 2004.

Xiaopeng Xi, Eamonn J. Keogh, Christian R. Shelton, Li Wei, and Chotirat Ann Ratanamahatana. Fast time series classification using numerosity reduction. In *International Conference on Machine Learning*, 2006.

Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *International Conference on Computer Vision*, 2011.

Daniel Zoran and Yair Weiss. Natural images, gaussian mixtures and dead leaves. In *Neural Information Processing Systems*, 2012.