Clustered Naive Bayes

by

Daniel Murphy Roy

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

Masters of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2006

Author .....
Department of Electrical Engineering and Computer Science
May 27, 2006

Certified by .....
Leslie Pack Kaelbling
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Clustered Naive Bayes

by

Daniel Murphy Roy

Submitted to the Department of Electrical Engineering and Computer Science
on May 27, 2006, in partial fulfillment of the
requirements for the degree of
Masters of Engineering in Electrical Engineering and Computer Science

Abstract

Humans effortlessly use experience from related tasks to improve their performance at novel tasks. In machine learning, we are often confronted with data from "related" tasks and asked to make predictions for a new task. How can we use the related data to make the best prediction possible? In this thesis, I present the Clustered Naive Bayes classifier, a hierarchical extension of the classic Naive Bayes classifier that ties several distinct Naive Bayes classifiers by placing a Dirichlet Process prior over their parameters. A priori, the model assumes that there exists a partitioning of the data sets such that, within each subset, the data sets are identically distributed. I evaluate the resulting model in a meeting domain, developing a system that automatically responds to meeting requests, partially taking on the responsibilities of a human office assistant. The system decides, based on a learned model of the user's behavior, whether to accept or reject the request on his or her behalf. The extended model outperforms the standard Naive Bayes model by using data from other users to influence its predictions.

Thesis Supervisor: Leslie Pack Kaelbling
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

All of my achievements to date were possible only with the support provided to me by a handful of individuals. Most immediately, this work would not have been possible without my thesis advisor, Leslie Kaelbling, who took me on as a student last year and has, since then, given me full reign to chase my interests. Leslie has an amazing ability to see through the clutter of my reasoning and extract the essence; the various drafts of this thesis improved by leaps and bounds each time Leslie gave me comments.

I would also like to thank Martin Rinard, who was the first professor at MIT to invite me to work with him. Martin encouraged me to apply to the PhD program and then championed my application. I learned many valuable lessons from Martin, most of them late at night, working on drafts of papers due the next day. Every so often, I hear Martin in the back of my head, exclaiming, "Be brilliant!" Of course!

I would like to thank Joann P. DiGennaro, Maite Ballestero and the Center for Excellence in Education for inviting me to participate in the Research Science Institute in 1998. They accepted my unorthodox application, taking a risk with a student with much less exposure to academic research than his peers. I flourished under their guidance and owe them an enormous debt of gratitude. I would also like to thank my teachers at Viewpoint School who pushed me to reach my full potential and more than prepared me for MIT.

I am incredibly grateful for the sacrifices my parents made in sending me to the best schools possible. I would like to thank my father for buying my first computer, my mother for buying my first cello, and my two brothers, Kenny and Matt, for being my closest friends growing up. Finally, I would like to thank Juliet Wagner, who five years ago changed my life with a single glance.

Contents

Chapter 1

Introduction

In machine learning, we are often confronted with multiple, related datasets and asked to make predictions. For example, in spam filtering, a typical dataset consists of thousands of labeled emails belonging to a collection of users. In this sense, we have multiple data sets—one for each user. Should we combine the datasets and ignore the prior knowledge that different users labeled each email? If we combine the data from a group of users who roughly agree on the definition of spam, we will have increased the available training data from which to make predictions. However, if the preferences within a population of users are heterogeneous, then we should expect that simply collapsing the data into an undifferentiated collection will make our predictions worse. Can we take advantage of all the data to improve prediction accuracy even if not all the data is relevant?

The process of using data from unrelated or partially related tasks is known as *transfer learning* or *multi-task learning* and has a growing literature (Thrun, 1996; Baxter, 2000; Guestrin et al., 2003; Xue et al., 2005; Teh et al., 2006). While humans effortlessly use experience from related tasks to improve their performance at novel tasks, machines must be given precise instructions on how to make such connections. In this thesis, I introduce such a set of instructions, based on the statistical assumption that there exists some partitioning of the data sets into groups such that each group is identically distributed. Because I have made no representational commitments, the assumptions are general but weak. Ultimately, any such model of sharing must be evaluated on real data to test its worth, and, to that end, I evaluate the resulting model in a meeting domain, developing a system that automatically

responds to meeting requests, partially taking on the responsibilities of a human office assistant. The system decides, based on a learned model of the user's behavior, whether to accept or reject the request on his or her behalf. The model with sharing outperforms its non-sharing counterpart by using data from other users to influence its predictions.

In the machine learning community, prediction is typically characterized as a *classification* problem. In binary classification, our goal is to learn to predict the outputs of a (possibly non-deterministic) function f that maps inputs X to output labels $Y \in \{0, 1\}$. Given n example input/output pairs $(X_i, Y_i)_{i=1}^n$ and an unlabeled input X_{n+1}, we predict the missing label Y_{n+1}. However, before a prediction is decided upon, we must first formally specify our preferences with respect to prediction error. This is accomplished by defining a loss function, $L(y, y')$, which specifies the penalty associated with predicting the label y' when the true label is y. For example,

$$L(y, y') = \begin{cases} 0 & y = y' \\ 1 & y \neq y', \end{cases} \tag{1.1}$$

is referred to as 0-1 loss. Inspecting Equation (1.1), we see that the classifier is penalized if it predicts the wrong label and is not penalized if it predicts the correct label. It follows from this definition that the optimal decision is to choose the most likely label. In a spam filtering setting, however, the 0-1 loss is inappropriate; we would prefer to allow the odd spam email through the filter if it lowered the chance that an authentic email would be discarded. Such loss functions are called *asymmetric* because they assign different loss depending on whether the true label is zero or one.

In the standard classification setting, the example input/output pairs are assumed to be independent and identically distributed (i.i.d.) samples from some unknown probability distribution, $P(X, Y)$. In fact, if we know $P(X, Y)$, then, given any unlabeled input $X_{n+1} = x$ and any loss function $L(y, y')$, we can determine the prediction that minimizes the expected loss by choosing the label

$$y_{\text{opt}} = \arg\min_{y'} E[L(Y, y')|X = x]. \tag{1.2}$$

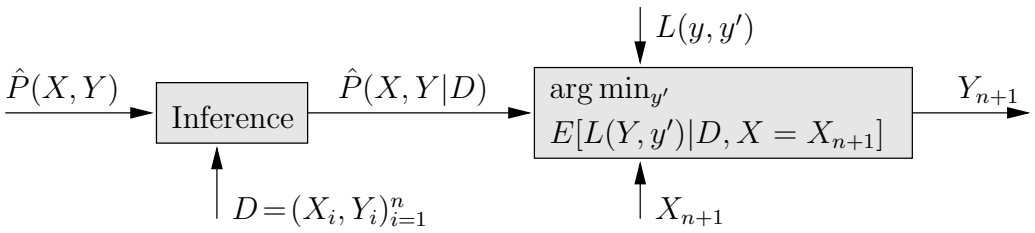

Figure 1-1: Given a probabilistic model, $\hat{P}(X, Y)$, we can incorporate training data $D = (X_i, Y_i)_{i=1}^n$ using probability theory. Given this updated probability model, $\hat{P}(X, Y|D)$, any loss function $L(y, y')$ and an unlabeled input X_{n+1}, we can immediately choose the label that minimizes the expected loss.

From this perspective, if we can build a probabilistic model $\hat{P}(X, Y)$ that is, in some sense, close to the true distribution $P(X, Y)$, then we can make predictions for a range of loss functions (see Figure 1-1). For any particular loss function, empirical assessments of loss can be used to discriminate between probabilistic models (Vapnik, 1995). However, in order to handle a range of loss functions, we should not optimize with respect to any one particular loss function. While we could handle over-specialization heuristically by using several representative loss functions, we will instead directly evaluate the probability assignments that each model makes.

Recall that the monotonicity of the log function implies that maximizing the likelihood is identical to maximizing the log-likelihood. Using Gibbs' inequality, we can interpret the effect of seeking models that maximize the *expected* log-likelihood. Given two distribution $p(\cdot)$ and $q(\cdot)$, Gibbs' inequality states that,

$$E_p[\log p(x)] \geq E_p[\log q(x)], \tag{1.3}$$

with equality only if $p = q$ almost everywhere. Consider the task of choosing a distribution q to model data generated by an unknown probability distribution p. By Gibbs inequality, the distribution q that maximizes the expected log-likelihood of the data,

$$E_p[\log q(x)], \tag{1.4}$$

is precisely $q = p$ (MacKay, 2003, pg. 34). Remarkably, the log-loss, $L(q(\cdot), a) = -\log q(a)$,

is the only loss function on distributions such that minimizing the expected loss leads invariably to true beliefs (Merhav, 1998).[1]

The difference between the expected log-likelihood under the true model p and any approximate model q is

$$E_p[\log p(x)] - E_p[\log q(x)] = E_p[\log \frac{p(x)}{q(x)}] \triangleq D(p||q), \tag{1.5}$$

where $D(p||q)$ is known as the *relative entropy of p with respect to q*.[2] Therefore, choosing the model that maximizes the expected log-likelihood is equivalent to choosing the model that is closest in relative entropy to the true distribution. Given any finite dataset, we can produce empirical estimates of these expectations and use them to choose between models and make predictions.[3]

Assuming that we have decided to model our data probabilistically, how do we take advantage of related data sets? Consider the case where the datasets are each associated with a user performing a task that we intend to learn to predict. In order to leverage other users' data, we must formally define how users are related. However, while we may know several types of relationships that could exist, unless we know the intended users personally, we cannot know which relationships will actually exist between users in an arbitrary population. Therefore, the set of encoded relationships should include those relationships that we expect to exist in real data (so they can be found) and those that can be used to improve prediction performance (so they are useful). With real data from a group of users, we first identify which of these relationships actually hold and then take advantage of them. For example, consider the prior belief that people who receive similar email are more likely to agree on the definition of spam. Given unlabeled emails from a user, we might benefit by grouping that user with other users whose mail is similar. On the other hand, while we may be able to discover that two users share the same birthday, this information is unlikely to aid in learning their spam preferences.

[1] Technically the log-loss is the only *local proper* loss function, where local implies that the loss function $L(q, a)$ only relies on the value of the distribution q at the value a and proper implies that it leads to Bayesian beliefs. The log-loss is known as the self-information loss in the information theory literature (Barron, 1998).

[2] This quantity is also known as the Kullback-Leibler (or, KL) divergence. Note that, while relative entropy is non-negative and zero if and only if $q = p$, it is not symmetric and therefore not a distance metric.

[3] I discuss the implementation of these ideas in Chapter 3.

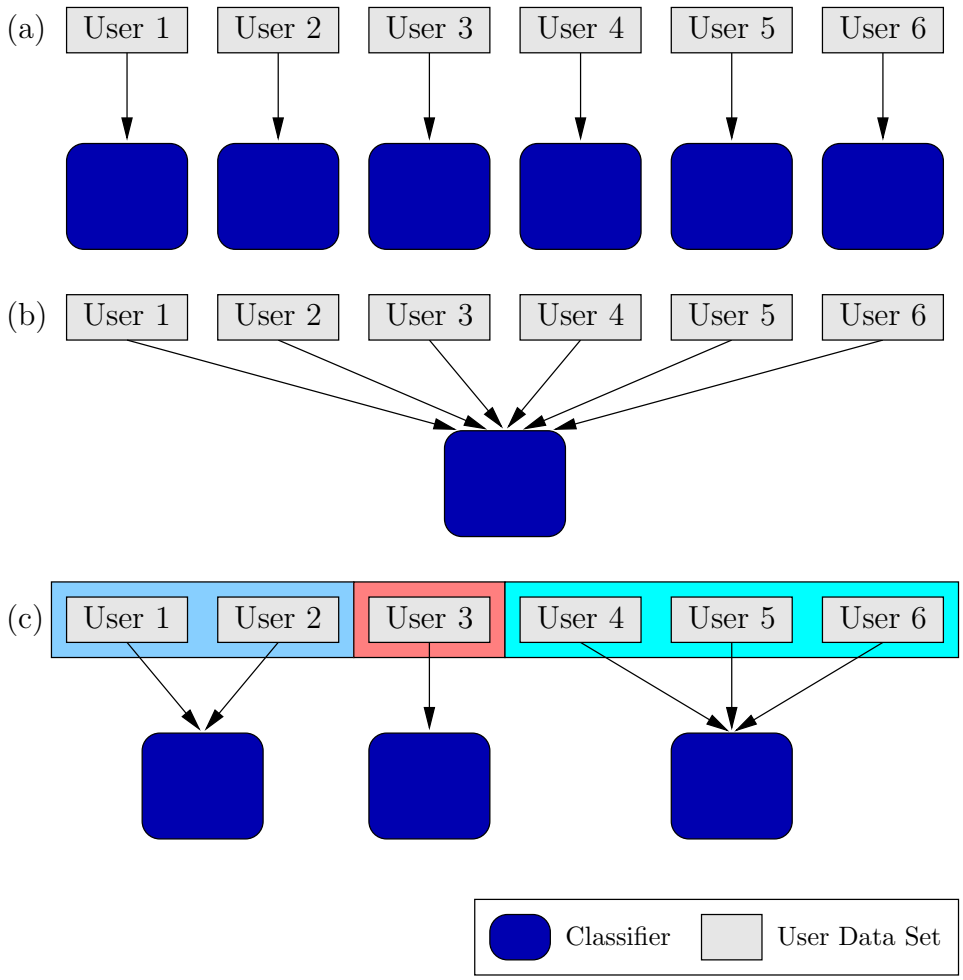

Figure 1-2: Transfer Learning Strategies: When faced with multiple datasets (six here, but in general n) over the same input space, we can extend a model that describes a single data set to one that describes the collection of datasets by constraining the parameters of each model. (a) If a model is applied to each dataset and no constraints are placed on the parameterizations of these models, then no transfer will be achieved. (b) Alternatively, if each of the models is constrained to have the same parameterization, it is as if we trained a single model on the entire dataset. Implicitly, we have assumed that all of the data sets are identically distributed. When this assumption is violated, we can expect prediction to suffer. (c) As a compromise, assume there exists a partitioning (coloring) of the datasets into groups such that the groups are identically distributed.

When faced with a classification task on a single data set, well-studied techniques abound (Boser et al., 1992; Herbrich et al., 2001; Lafferty et al., 2001). A popular classifier that works well in practice, despite its simplicity, is the Naive Bayes classifier (Maron, 1961) (which was casually introduced over 40 years ago). However, the Naive Bayes classifier cannot directly integrate multiple related datasets into its predictions. If we train a separate classifier on each data set then, of course, it follows that the other data sets will have no effect on predictive performance (see Figure 1-2(a)). One possible way of integrating related datasets is to combine them into a single, large data set (see Figure 1-2(b)). However, just as in the spam example, grouping the data and training a single Naive Bayes classifier will work only if users behave similarly.

Consider the following compromise between these two extreme forms of sharing. Instead of assuming that every dataset is identically distributed, we will assume that some grouping of the users exists such that the datasets within each group are identically distributed (see Figure 1-2(c)). Given a partitioning, we can then train a classifier on each group. To handle uncertainty in the number of groups and their membership, I define a generative process for datasets, such that they the datasets tend to cluster into identically distributed groups. At the heart of this process is a non-parametric prior known as the Dirichlet Process. This prior ties the parameters of separate Naive Bayes models that model each dataset. Because samples from the Dirichlet Process are discrete with probability one, and therefore, exhibit clustering, the model parameterizations also cluster, resulting in identically distributed groups of data sets. This model extends the applicability of the Naive Bayes classifier to the domain of multiple, (possibly) related data sets defined over the same input space. I call the resulting classifier the Clustered Naive Bayes classifier. Bayesian inference under this model simultaneously considers all possible groupings of the data, weighing the predictive contributions of each grouping by its posterior probability.

There have been a wide variety of approaches to the problem of transfer learning. Some of the earliest work focused on sequential transfer in neural networks, using weights from networks trained on related data to bias the learning of networks on novel tasks (Pratt, 1993; Caruana, 1997). Other work has focused on learning multiple tasks simultaneously. For example, Caruana (1997) describes a way of linking multiple neural networks together

to achieve transfer and then extends this approach to decision-tree induction and k-nearest neighbors. More recently, these ideas have been applied to modern supervised learning algorithms, like support vector machines (Wu and Dietterich, 2004). Unfortunately, it is very difficult to interpret these ad-hoc techniques; it is unclear whether the type of sharing they embody is justified or relevant. In contrast, the probabilistic model at the heart of the Clustered Naive Bayes classifier explicitly defines the type of sharing it models.

I have taken a Bayesian approach to modelling, and therefore probabilities should be interpreted as representing degrees of belief (Cox, 1946; Halpern, 1999; Arnborg and Sjdin, 2000). This thesis is related to a large body of transfer learning research conducted in the hierarchical Bayesian framework, in which common prior distributions are used to tie together model components across multiple datasets. My work is a direct continuation of work started by Zvika Marx and Michael Rosenstein on the CALO DARPA project. Marx and Rosenstein formulated the meeting acceptance task, chose relevant features, collected real-world data sets and built two distinct models that both achieved transfer. In the first model, Marx et al. (2005) biased the learning of a logistic regression model by using a prior over weights whose mean and variance matched the empirical mean and variance of the weights learned on related tasks. Rosenstein et al. (2005) constructed a hierarchical Bayesian model where each tasks was associated with a separate naive Bayes classifier. Transfer was achieved by first assuming that the parameters were drawn from a common distribution and then performing approximate Bayesian inference. The most serious problem with this model is that the chosen prior distribution is not flexible enough to handle more than one group of related tasks. In particular, if there exists two sufficiently different groups of tasks, then no transfer will occur. I describe their prior distribution in Section 2.4. This thesis improves their model by replacing the common distribution with a Dirichlet Process prior. It is possible to loosely interpret the resulting Clustered Naive Bayes model as grouping tasks based on a marginal likelihood metric. From this viewpoint, this work is related to transfer-learning research which aims to first determine which tasks are relevant before attempting transfer (Thrun and O'Sullivan, 1996).

Ferguson (1973) was the first to study the Dirichlet Process and show that it can, simply speaking, model any other distribution arbitrarily closely. With the advent of faster com-

puters, large-scale inference under the Dirichlet Process has become possible, resulting in a rush of research applying this tool to a wide range of problems including document modelling (Blei et al., 2004), gene discovery (Dahl, 2003), and scene understanding (Sudderth et al., 2005). Teh et al. (2006) introduced the Hierarchical Dirichlet Process, achieving transfer learning in document modelling across multiple corpora. The work closest in spirit to this thesis was presented recently by Xue et al. (2005). They achieved transfer learning between multiple datasets by tying the parameters of logistic regression models together using the Dirichlet Process and deriving a variational method for performing inference. In the same way, the Clustered Naive Bayes model I introduce uses a Dirichlet Process prior to tie the parameters of several Naive Bayes models together to achieve transfer. There are several important differences: First, the logistic regression model is discriminative, meaning that it does not model the distribution of the inputs. Instead, it only models the conditional distribution of the outputs conditioned on the inputs. As a result, it cannot take advantage of unlabeled data. In the Clustered Naive Bayes model, the datasets are clustered with respect to a generative model which defines a probability distribution over both the inputs and outputs. Therefore, the Clustered Naive Bayes model is semi-supervised: it can take advantage of unlabeled data points. Implicit in this choice is the assumption that similar feature distributions are associated with similar predictive distributions. Whether this assumption is valid must be judged for each task; for the meeting acceptance task, the generative model of sharing seems appropriate and leads to improved results.

This thesis is in five parts. In Chapter 2, I formally define the Clustered Naive Bayes model as well as two baseline models against which the clustered variant will be compared. In Chapter 3, I define the meeting acceptance task and justify the metrics I will use to evaluate the Clustered Naive Bayes model's performance. In Chapter 4, I present the results for the meeting acceptance task. In Chapter 5, I conclude with a discussion of the results and ideas for future work. Finally, for my most ardent readers, I present the technical aspects of performing inference in each of the models in Appendix A.

Chapter 2

Models

In this thesis we are concerned with classification settings where the inputs and output labels are drawn from finite sets. In particular, the inputs are composed of F features, each taking values from finite sets. Over such input spaces, the standard Naive Bayes classifier normally takes the form of a product of multinomial distributions. In this chapter I introduce the Clustered Naive Bayes model in detail, as well as two additional models which will function as baselines for comparison in the meeting acceptance task.

Let the tuple $(X_1, X_2, \ldots, X_F)$ represent F features comprising an input X, where each X_i takes values from a finite set $\mathcal{V}_i$. We write (X, Y) to denote a labelled set of features, where $Y \in \mathcal{L}$ is a label. We will make an assumption of exchangeability; any permutation of the data has the same probability. This is a common assumption in the classification setting, that states that it is the value of the inputs and outputs that determine the probability assignment and not their order. Consider U data sets, each composed of N_u feature/label pairs. Simply speaking, each data set is defined over the same input space. It should be stressed, however, that, while each dataset is assumed to be identically distributed, we do not assume that the collection of datasets are identically distributed. We will associate each data set with a particular user. Therefore, we will write $D_{u,i} = (X, Y)_i$ and $D_u = (D_{u,1}, D_{u,2}, \ldots, D_{u,N_u})$ to indicate the N_u labelled features associated with the u-th user. Then D is the entire collection of data across all users, and $D_{u,j} = (X_{u,j}, Y_{u,j})$ is the j-th data point for the u-th user.

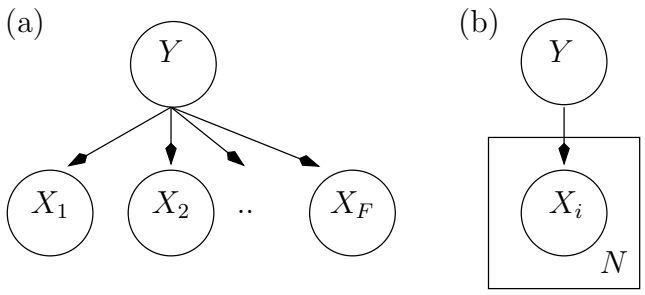

Figure 2-1: (a) A graphical model representing that the features X are conditionally independent given the label, Y. (b) We can represent the replication of conditional independence across many variables by using plates.

2.1 Naive Bayes models

Under the Naive Bayes assumption, the marginal distribution for j-th labelled input for the u-th user is

$$P(D_{u,j}) = P(X_{u,j}, Y_{u,j}) = P(Y_{u,j}) \prod_{f=1}^{F} P(X_{u,j,f}|Y_{u,j}). \tag{2.1}$$

This marginal distribution is depicted graphically in Figure 2-1. Recall that each label and feature is drawn from a finite set. Therefore, the conditional distribution, $P(X_{u,j,f}|Y_{u,j})$, associated with the feature/label pair y/f is completely determined by the collection of probabilities $\theta_{u,y,f} \triangleq \{\theta_{u,y,f,x} : x \in \mathcal{V}_f\}$, where

$$\theta_{u,y,f,x} \triangleq \Pr\{X_{u,f} = x|Y = y\}. \tag{2.2}$$

To define a distribution, the probabilities for each value of the feature X_f must sum to one. Specifically, $\sum_{x\in\mathcal{V}_f} \theta_{u,y,f,x} = 1$. Therefore, every feature-label pair is associated with a distribution. We write θ to denote the set of all conditional probability distributions that parameterize the model. Similarly, the marginal $P(Y_u)$ is completely determined by the probabilities $\phi_u \triangleq \{\phi_{u,y} : y \in \mathcal{L}\}$, where

$$\phi_{u,y} \triangleq \Pr\{Y_u = y\}. \tag{2.3}$$

Again, $\sum_{y\in\mathcal{L}} \phi_{u,y} = 1$. We will let ϕ denote the collection of marginal probability distributions that parameterize the model. Note that each dataset is parameterized separately and, therefore, we have so far not constrained the distributions in any way. We can now write the general form of the Naive Bayes model of the data:

$$P(D|\theta,\phi) = \prod_{u=1}^{U} P(D_u|\theta_u,\phi_u) \tag{2.4}$$

$$= \prod_{u=1}^{U}\prod_{n=1}^{N_u} P(D_{u,n}|\theta_u,\phi_u) \tag{2.5}$$

$$= \prod_{u=1}^{U}\prod_{n=1}^{N_u} P(Y_{u,n}|\phi_{u,y}) \prod_{f=1}^{F} P(X_{u,n,f}|Y_{u,y},\theta_{u,y,f}) \tag{2.6}$$

$$= \prod_{u=1}^{U}\prod_{y\in\mathcal{L}} \phi_{u,y}^{m_{u,y}} \prod_{f=1}^{F}\prod_{x\in\mathcal{V}_f} \theta_{u,y,f,x}^{n_{u,y,f,x}}, \tag{2.7}$$

where $m_{u,y}$ is the number of data points labeled $y \in \mathcal{L}$ in the data set associated with the u-th user and $n_{u,y,f,x}$ is the number of instances in the u-th dataset where feature f takes the value $x \in \mathcal{V}_f$ when its parent label takes value $y \in \mathcal{L}$. In (2.4), $P(D|\theta,\phi)$ is expanded into a product of terms, one for each dataset, reflecting that the datasets are independent conditioned on the parameterization. Equation (2.5) makes use of the exchangeability assumption; specifically, the label/feature pairs are independent of one another given the parameterization. In Equation (2.6), we have made use of the Naive Bayes assumption. Finally, in (2.7), we have used the fact that the distributions are multinomials.

To maximize $P(D|\theta,\phi)$ with respect to θ,ϕ, we choose

$$\phi_{u,y} = \frac{m_{u,y}}{\sum_{y\in\mathcal{L}} m_{u,y}} \tag{2.8}$$

and

$$\theta_{u,y,f,x} = \frac{n_{u,y,f,x}}{\sum_{x\in\mathcal{V}_f} n_{u,y,f,x}}. \tag{2.9}$$

Because each dataset is parameterized separately, it is no surprise that the maximum likelihood parameterization for each data set depends only on the data in that data set. In

order to induce sharing, it is clear that we must somehow constrain the parameterization across users. In the Bayesian setting, the prior distribution $P(\theta, \phi)$ can be used to enforce such constraints. Given a prior, the resulting joint distribution is

$$P(D) = \int_{\Theta \times \Phi} P(D|\theta, \phi)\, dP(\theta, \phi). \tag{2.10}$$

Each of the models introduced in this chapter are completely specified by particular prior distributions over the parameterization of the Naive Bayes model. As we will see, different priors result in different types of sharing.

2.2 No-Sharing Baseline Model

We have already seen that a ML parameterization of the Naive Bayes model ignores related data sets. In the Bayesian setting, any prior over the entire set of parameters that factors into distributions for each user's parameters will result in no sharing. In particular,

$$P(\theta, \phi) = \prod_{u=1}^{U} P(\theta_u, \phi_u), \tag{2.11}$$

is equivalent to the statement that the parameters for each user are independent of the parameters for other users. Under this assumption of independence, training the entire collection of models is identical to training each model separately on its own dataset. We therefore call this model the *no-sharing* model.

Having specified that the prior factors into parameter distributions for each user, we must specify the actual parameter distributions for each user. A reasonable (and tractable) class of distributions over multinomial parameters are the Dirichlet distributions which are conjugate to the multinomials. Therefore, the distribution over ϕ_u, which takes values in the $|\mathcal{L}|$-simplex, is

$$P(\phi_u) = \frac{\Gamma(\sum_{y \in \mathcal{L}} \alpha_{u,y})}{\prod_{y \in \mathcal{L}} \Gamma(\alpha_{u,y})} \prod_{y \in \mathcal{L}} \phi_{u,y}^{\alpha_{u,y}-1}. \tag{2.12}$$

Similarly, the distribution over $\theta_{u,y,f}$, which takes values in $|\mathcal{V}_f|$-simplex, is

$$P(\theta_{u_y,f}) = \frac{\Gamma(\sum_{x\in\mathcal{V}_f}\beta_{u,y,f,x})}{\prod_{x\in\mathcal{V}_f}\Gamma(\beta_{u,y,f,x})}\prod_{x\in\mathcal{V}_f}\theta_{u,y,f,x}^{\beta_{u,y,f,x}-1}. \tag{2.13}$$

We can write the resulting model compactly as a generative process:[1]

$$\phi_u \sim \text{Dirichlet}(\{\alpha_{u,y} : y \in \mathcal{L}\}) \tag{2.14}$$

$$Y_{u,n} \mid \phi_u \sim \text{Discrete}(\phi_u) \tag{2.15}$$

$$\theta_{u,y,f} \sim \text{Dirichlet}(\{\beta_{u,y,f,x} : x \in \mathcal{V}_f\}) \tag{2.16}$$

$$X_{u,n,f} \mid Y_{u,n}, \{\theta_{u,y,f} : \forall y \in Y\} \sim \text{Discrete}(\theta_{u,(Y_{u,n}),f}) \tag{2.17}$$

The No-Sharing model will function as a baseline against which we can compare alternative models that induce sharing.

2.3 Complete-Sharing Model

The Complete-Sharing model groups all the users' data together, learning a single model. We can express this constraint via a prior $P(\theta, \phi)$ by assigning zero probability if $\theta_{u,y,f} \neq \theta_{u',y,f}$ for any u, u' (and similarly for $\phi_{u,y}$). This forces the users to share the same parameterization. It is instructive to compare the graphical models for the baseline No-Sharing model (Figure 2-2) and the Complete-Sharing model (Figures 2-3). As in the No-Sharing model, each parameter will be drawn independently from a Dirichlet distribution. However, while each user had its own set of parameters in the No-Sharing model, all users share the same parameters in the Complete-Sharing model. Therefore, the prior probability of the complete parameterization is

$$P(\theta, \phi) = P(\phi)\prod_{f=1}^{F}\prod_{y\in\mathcal{L}}P(\theta_{y,f}), \tag{2.18}$$

[1]The $\sim$ symbol denotes that the variable to the left is distributed according to the distribution specified on the right. It should be noted that the Dirichlet distribution requires an ordered set of parameters while the definitions we specify provide an unordered set. To solve this, we define an arbitrary ordering of the elements of $\mathcal{L}$ and $\mathcal{V}_f$. We can then consider the elements of these sets as index variables when necessary.

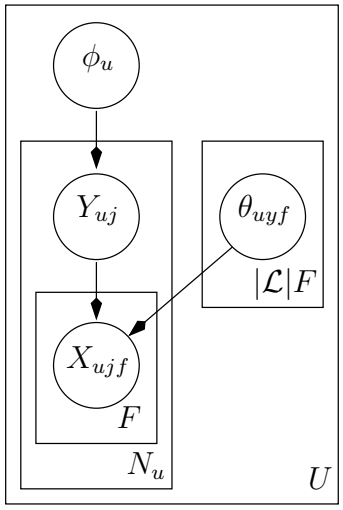

Figure 2-2: No-Sharing Graphical Model: Each model for each user has its own parameterization.

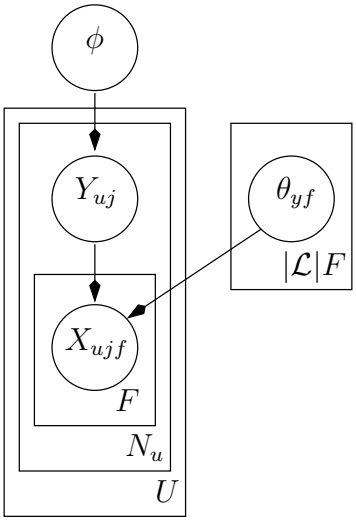

Figure 2-3: Graphical Model for Complete-Sharing model - The prior distribution over parameters constrains all users to have the same parameters.

where

$$P(\phi) = \frac{\Gamma(\sum_{y \in \mathcal{L}} \alpha_y)}{\prod_{y \in \mathcal{L}} \Gamma(\alpha_y)} \prod_{y \in \mathcal{L}} \phi_y^{\alpha_y - 1}, \tag{2.19}$$

and

$$P(\theta_{y,f}) = \frac{\Gamma(\sum_{x \in \mathcal{V}_f} \beta_{y,f,x})}{\prod_{x \in \mathcal{V}_f} \Gamma(\beta_{y,f,x})} \prod_{x \in \mathcal{V}_f} \theta_{y,f,x}^{\beta_{y,f,x} - 1}. \tag{2.20}$$

Again, we can represent the model compactly by specifying the generative process (c.f. (2.14)):

$$\phi \sim \text{Dirichlet}(\{\alpha_y : y \in \mathcal{L}\}) \tag{2.21}$$

$$Y_{u,n} \mid \phi \sim \text{Discrete}(\phi) \tag{2.22}$$

$$\theta_{y,f} \sim \text{Dirichlet}(\{\beta_{y,f,x} : x \in \mathcal{V}_f\}) \tag{2.23}$$

$$X_{u,n,f} \mid Y_{u,n}, \{\theta_{y,f} : \forall y \in Y\} \sim \text{Discrete}(\theta_{(Y_{u,n}),f}) \tag{2.24}$$

While the baseline model totally ignores other datasets when fitting its parameters, this model tries to find a single parameterization that fits all the data well. If the data are unrelated, then the resulting parameterization may describe none of individual data sets accurately. In some settings, this model of sharing is appropriate. For example, for a group of individuals who agree what constitutes spam, pooling their data would likely improve the performance of the resulting classifier. However, in situations where we do not know *a priori* whether the data sets are related, we need a more sophisticated model of sharing.

2.4 Prototype Model

Instead of assuming that every dataset is distributed identically, we can relax this assumption by assuming that there exists a prototypical distribution and that each data set is generated by a distribution that is a noisy copy of this prototype distribution. To model this idea, we might assume that every user's parameterization is drawn from a common unimodal parameter distribution shared between all users. Consider the following noise

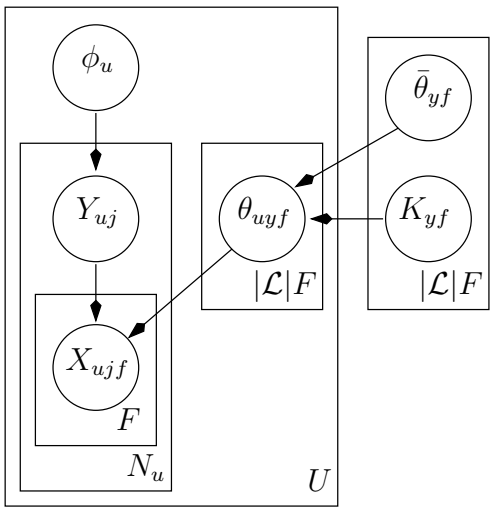

Figure 2-4: Prototype Model

model: For all features f and labels y, we define a prototype parameter

$$\bar{\theta}_{y,f} \triangleq \{\bar{\theta}_{y,f,x} : x \in \mathcal{V}_f\}, \tag{2.25}$$

and a strength parameter $K_{y,f} > 0$. Then, each parameter $\theta_{u,y,f}$ is drawn according to a Dirichlet distribution with parameters $\bar{\theta}_{y,f,x} * K_{y,f}$. I will refer to this distribution over the simplex as the noisy Dirichlet. As $K_{y,f}$ grows, the noisy Dirichlet density assigns most of its mass to a ball around the prototype $\bar{\theta}_{y,f}$. Figure 2-5 depicts some samples from this noise model.[2] One question is whether to share the parameterization for the marginal distribution, ϕ. Because we are most interested in transferring knowledge about the relationship between the features and the label, I have decided not to share the marginal.

[2] An alternative distribution over the simplex that has a larger literature is the logistic normal distribution (Aitchison, 1982). The noisy Dirichlet has reasonable properties when K is large. However, for $K < 1$, the distribution becomes multimodal, concentrated on the extreme points of the simplex.

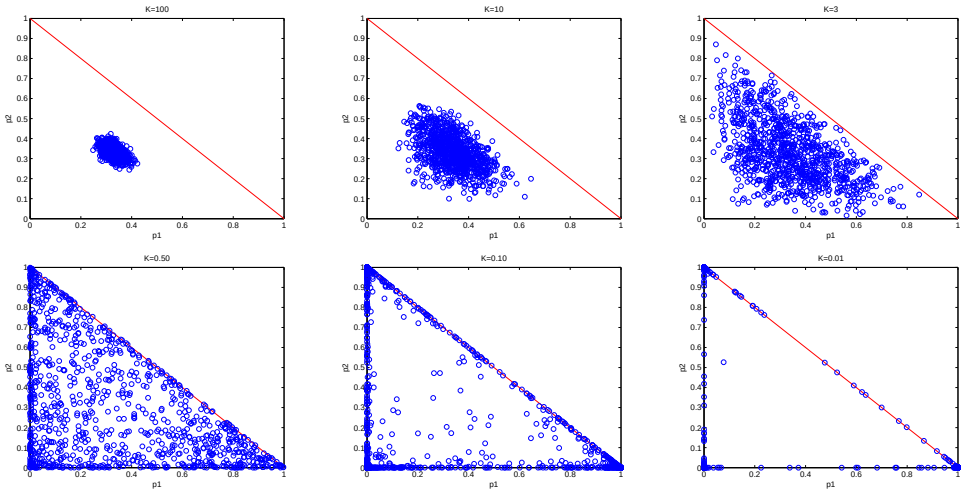

Figure 2-5: Noise Model Samples: The following diagrams each contain 1000 samples from the noisy Dirichlet distribution over the 3-simplex with prototype parameter $\bar{\theta} = \{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\}$ and strength parameter K=$\{100, 10, 3, 0.5, 0.1, 0.01\}$. When the value of K drops below the point at which $K\bar{\theta}$ has components less than 1 (in this diagram, $K = 3$), the distribution becomes multimodal, with most of its mass around the extreme points.

We can write the corresponding model as a generative process

$$\bar{\theta}_{y,f} \sim \text{Dirichlet}(\{\beta_{y,f,x} : x \in \mathcal{V}_f\}) \tag{2.26}$$

$$K_{y,f} \sim \Gamma(\gamma_1, \gamma_2) \tag{2.27}$$

$$\phi_u \sim \text{Dirichlet}(\{\alpha_{u,y} : y \in \mathcal{L}\}) \tag{2.28}$$

$$Y_{u,n} \mid \phi_u \sim \text{Discrete}(\phi_u) \tag{2.29}$$

$$\theta_{u,y,f} \sim \text{Dirichlet}(K_{y,f}\, \bar{\theta}_{y,f}) \tag{2.30}$$

$$X_{u,n,f} \mid Y_{u,n}, \{\theta_{u,y,f} : \forall y \in Y\} \sim \text{Discrete}(\theta_{u,(Y_{u,n}),f}) \tag{2.31}$$

If the parameterizations of each of the data sets are nearly identical, then a fitted unimodal noise model will be able to predict parameter values for users when there is insufficient data. However, this model of sharing is brittle: Consider two groups of data sets where each group is identically distributed, but the parameterizations of the two groups are very different. Presented with this dataset, the strength parameter K will be forced to zero in order to model the two different parameterizations. As a result, no sharing

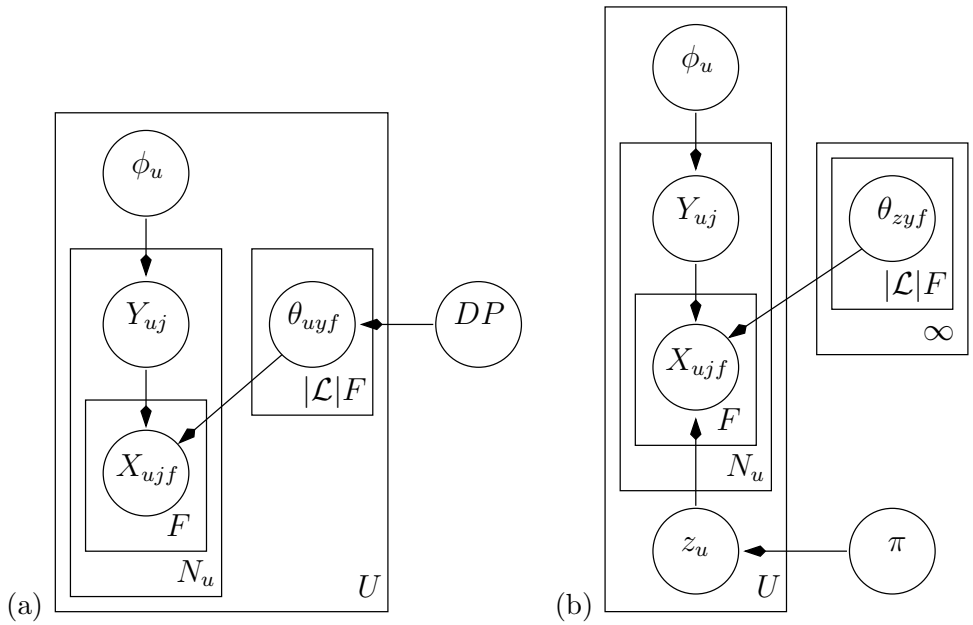

Figure 2-6: (a) The parameters of the Clustered Naive Bayes model are drawn from a Dirichlet Process. (b) In Chinese Restaurant Process representation, each user u is associated with a "table" z_u, which indexes an infinite vector of parameters drawn i.i.d. from the base distribution.

will occur. Because the unimodal nature of the prior seems to be the root cause of this brittleness, a natural step is to consider multimodal prior distributions.

2.5 The Clustered Naive Bayes Model

A more plausible assumption about a group of related data sets is that some partition exists where each subset is identically distributed. We can model this idea by making the prior distribution over the parameters a mixture model. Consider a partitioning of a collection of datasets into groups. Each group will be associated with a particular parameterization of a Naive Bayes model. Immediately, several modelling questions arise. First and foremost, how many groups are there? It seems reasonable that, as we consider additional data sets, we should expect the number of groups to grow. Therefore, it would be inappropriate to choose a prior distribution over the number of groups that was independent of the number of datasets. A stochastic process that has been used successfully in many recent papers to model exactly this type of intuition is the Dirichlet Process. In order to describe this

process, I will first describe a related stochastic process, the Chinese Restaurant Process.

The Chinese Restaurant Process (or CRP) is a stochastic process that induces distributions over partitions of objects (Aldous, 1985). The following metaphor was used to describe the process: Imagine a restaurant with an infinite number of indistinguishable tables. The first customer sits at an arbitrary empty table. Subsequent customers sit at an occupied table with probability proportional to the number of customers already seated at that table and sit at an arbitrary, new table with probability proportional to a parameter $\alpha > 0$ (see Figure 2.5). The resulting seating chart partitions the customers. It can be shown that, in expectation, the number of occupied tables after n customers is $\Theta(\log n)$ (Navarro et al., 2006; Antoniak, 1974).

To connect the CRP with the Dirichlet Process, consider this simple extension. Imagine that when a new user enters the restaurant and sits at a new table, they draw a complete parameterization of their Naive Bayes model from some base distribution. This parameterization is then associated with their table. If a user sits at an occupied table, they adopt the parameterization already associated with the table. Therefore, everyone at the same table uses the same rules for predicting.

This generative process is known as the Dirichlet Process Mixture Model and has been used very successfully to model latent groups. The Dirichlet process has two parameters, a mixing parameter α, which corresponds to the same parameter of the CRP, and a base distribution, from which the parameters are drawn at each new table. It is important to specify that we draw a complete parameterization of all the feature distributions, $\theta_{y,f}$, at each table. As in the prototype model, I have decided not to share the marginal distributions, ϕ, because we are most interested in knowledge relating features and labels.

It is important to note that we could have easily defined a separate generative process for each conditional distribution $\theta_{y,f}$. Instead, we have opted to draw a complete parameterization. If we were to share each conditional distribution separately, we would be sharing data in hopes of better predicting a distribution over the $|\mathcal{V}_f|$-simplex. However, the cardinality of each set $\mathcal{V}_f$ is usually small. In order to benefit from the sharing, we must gain a significant amount of predictive capability when we discover the underlying structure. I have found through empirical experimentation that, if we share knowledge only between single

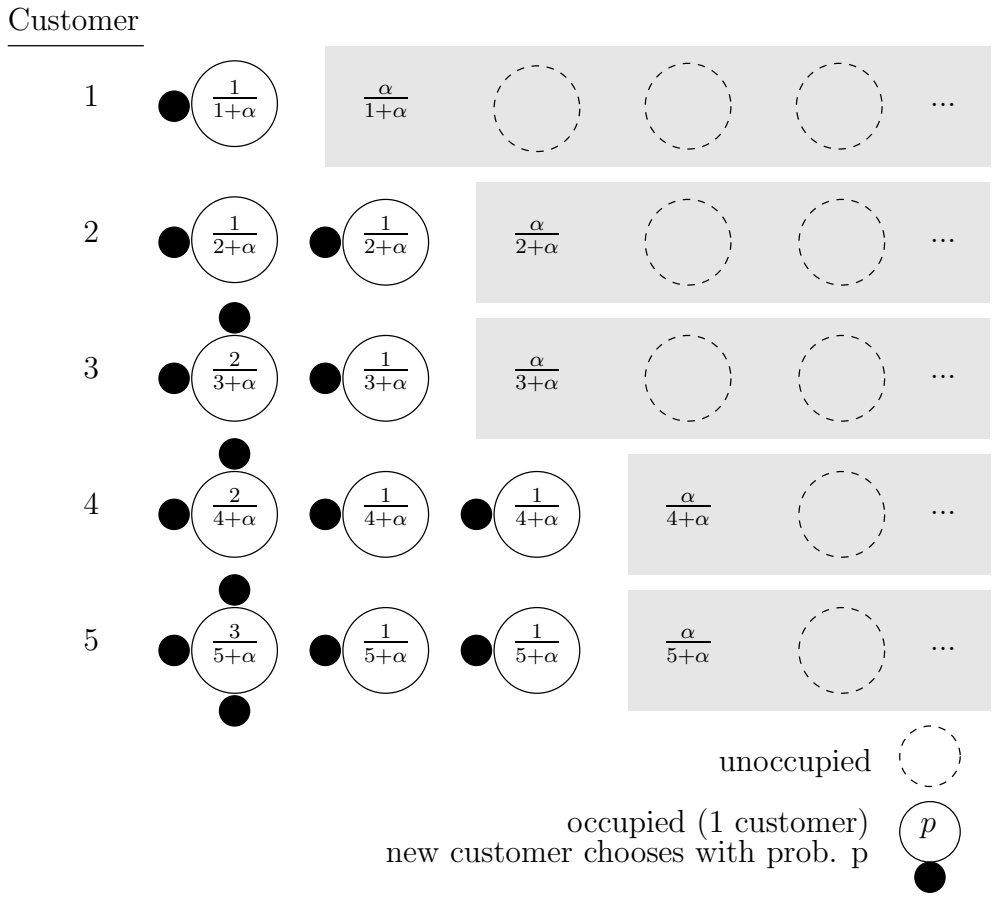

Figure 2-7: This figure illustrates the Chinese Restaurant Process. After the first customer sits at an arbitrary table, the next customer sits at the same table with probability $\frac{1}{1+\alpha}$ and a new table with probability $\frac{\alpha}{1+\alpha}$. In general, a customer sits a table with probability proportional to the number of customers already seated at the table. A customer sits at an arbitrary new table with probability proportional to α. In the diagram above, solid circles represent occupied tables while dashed circles represent unoccupied circles. Note that there are an infinite number of unoccupied tables, which is represented by the gray regions. The fractions in a table represents the probability that the next customer sits at that table.

features, it is almost always the case that, by the time we have enough data to discover a relationships, we have more than enough data to get good performance without sharing. As a result, we make a strong assumption about sharing in order to benefit from discovering the structure; when people agree about one feature distribution, they are likely to agree about all feature distributions. Therefore, at each table, a complete parameterization of the feature distributions is drawn. The base distribution is the same joint distribution used in the other two models. Specifically, each user's parameterization is drawn from a product of independent Dirichlet distributions for each feature-label pair.

Again, we can represent the model compactly by specifying the generative process:

$$\phi_u \sim \text{Dirichlet}(\{\alpha_{u,y} : y \in \mathcal{L}\}) \tag{2.32}$$

$$Y_{u,n} \mid \phi \sim \text{Discrete}(\phi_u) \tag{2.33}$$

$$\theta_u = (\theta_{u,y,f})_{f=1,2,\ldots,F}^{y\in\mathcal{L}} \sim \text{DP}(\alpha, \prod_{f=1}^{F} \prod_{y\in\mathcal{L}} \text{Dirichlet}(\{\beta_{y,f,x} : x \in \mathcal{V}_f\}) \tag{2.34}$$

$$X_{u,n,f} \mid Y_{u,n}, \{\theta_{u,y,f} : \forall y \in Y\} \sim \text{Discrete}(\theta_{(u,Y_{u,n}),f}) \tag{2.35}$$

Because the parameters are being clustered, I have named this model the Clustered Naive Bayes model. Bayesian inference under this generative model requires that we marginalize over the parameters and clusters. Because we have chosen conjugate priors, the base distribution can be analytically marginalized. However, as the number of partitions is large even for our small dataset, we cannot perform exact inference. Instead, we will build a Gibbs sampler to produce samples from the posterior distribution. See Appendix A for details. In the following chapter I define the meeting acceptance task and justify two metrics that I will use to evaluate the CNB model.

Chapter 3

The Meeting Task and Metrics

In the meeting acceptance task, we aim to predict whether a user would accept or reject an unseen meeting request based on a learned model of the user's behavior. Specifically, we are given unlabeled meeting requests and asked to predict the missing labels, having observed the true labels of several meeting requests. While we need training data to learn the preferences of a user, we cannot expect the user to provide enough data to get the accuracy we desire; if we were to require extensive training, we would undermine the usefulness of the system. In this way, the meeting task can benefit from transfer learning techniques.

The success of collaborative spam filters used by email services like Yahoo Mail and Google's GMail suggests that a similar "collaborative" approach might solve this problem. All users enjoy accurate spam filtering even if most users never label their own mail, because the system uses everyone's email to make decisions for each user. This is, however, not as simple as building a single spam filter by collapsing everyone's mail into a undifferentiated collection; the latter approach only works when all users roughly agree on the definition of spam. When the preferences of the users are heterogeneous, we must selectively use other users' data to improve classification performance.

Using the metaphor of collaborative spam filtering, meeting requests are email messages to be filtered (i.e. accepted or rejected). Ideally, the data (i.e. past behavior) of "related" users can be used to improve the accuracy of predictions.

A central question of this thesis is whether the type of sharing that the CNB model supports is useful in solving real world problems. The meeting acceptance task is a natural

candidate for testing the CNB model. In this chapter I describe the meeting acceptance task in detail and justify two metrics that I will use to evaluate the models on this task.

3.1 Meeting Task Definition

In order to make the meeting task a well-defined classification task, we must specify our preferences with respect to classification errors by specifying a loss function.[1] However, it is easy to imagine that different users would assign different loss values to prediction errors. For example, some users may never want the system to reject a meeting they would have accepted while others may be more willing to let the system make its best guess. As I argued in the introduction, when there is no single loss function that encompasses the intended use of a classifier, we should instead model the data probabilistically.

Therefore, our goal is to build a probabilistic model of the relationship between meeting requests and accept/reject decisions that integrates related data from other users. Given an arbitrary loss function and a candidate, probabilistic model, we will build a classifier in the obvious way:

1. Calculate the posterior probability of each missing label conditioned on both labelled and unlabeled meeting requests.

2. For each unlabeled meeting request, choose the label that minimizes the expected loss with respect to the posterior distribution over the label.

There are two things to note about this definition. First, our goal is to produce marginal distributions over each missing label, not joint distributions over all missing labels. This follows from the additivity of our loss function. Second, we are conditioning on all labelled meeting requests as well as on unlabeled meeting requests; it is possible that knowledge about pending requests could influence our decisions and, therefore, we will not presume otherwise. It is important to note that each request is accepted or rejected based on the state of the calendar at the time of the request (as if it were the only meeting request). As a result, the system will not correctly handle two simultaneous requests for the same

[1] We restrict our attention to additive loss; i.e. if we make k predictions, the total loss is the sum of the individual losses.

time slot. However, we can avoid this problem by handling such requests in the order they arrive.

3.1.1 Meeting Representation

In this thesis, I use a pre-existing formulation of the meeting acceptance task developed by Zvika Marx and Michael Rosenstein for the CALO Darpa project. To simplify the modelling process, they chose to represent each meeting request by a small set of features that describe aspects of the meeting, its attendees, their inter-relationships and the state of the user's calendar. The representation they have chosen implicitly encodes certain prior knowledge they have about meeting requests. For instance, they have chosen to encode the state of the users' calendar by the free time bordering the meeting request and a binary feature indicating whether the requested time is unallocated.[2] The use of relative, as opposed to absolute, descriptions of the state of a meeting request imposes assumptions that they, as modelers, have made about how different meeting requests are, in fact, similar. See Appendix B for a list of the features used to represent each meeting request and Figure 3-1 for a description of the input space.

3.1.2 Dataset specification

A meeting request is *labelled* if the user has chosen whether to accept or reject the meeting. In the meeting acceptance task, there are U users, $U \in \{1, 2, \dots\}$. For each user there is a collection of meeting requests, some of which are labelled. Consider the u-th user, $u \in \{1, 2, \dots, U\}$. Let $N_u \in \{0, 1, \dots\}$ denote the total number of meeting requests for that user and let $M_u \in \{0, 1, \dots, N_u\}$ denote the number of these requests that are labelled. Then $X_u^+ \triangleq (X_{u,1}, X_{u,2}, \dots, X_{u,M_j})$ denotes this user's labelled meeting requests and $X_u^- \triangleq (X_{u,M_j+1}, \dots, X_{u,N_j})$ denotes the unlabeled meeting requests, where

[2] A fair question to ask is whether the features I have chosen to use can be automatically extracted from raw data. If not, then the system is not truly autonomous. Most of the features I inherited from the CALO formulation can be extracted automatically from meeting requests served by programs like Microsoft Outlook. Others, like the feature that indicates whether the requester is the user's supervisor, rely upon the knowledge of the social hierarchy in which the user is operating (i.e. the organizational chart in a company, chain of command in the military, etc). Fortunately, this type of knowledge remains relatively fixed. Alternatively, it could be learned in an unsupervised manner. Other features, e.g. the feature that describes the importance of the meeting topic from the user's perspective, would need to be provided by the user or learned by another system.

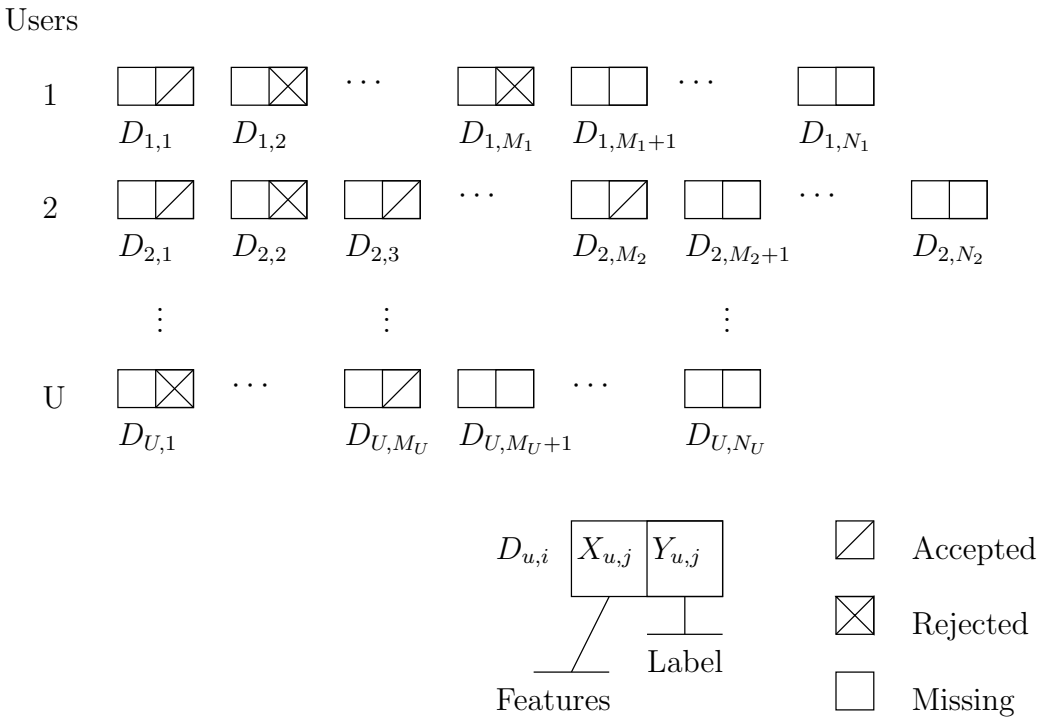

Figure 3-1: A representation of a meeting task data set.

each $X_{u,j} = (x_{u,j,1}, x_{u,j,2}, \ldots, x_{u,j,F})$ is a vector composed of F features, the i-th feature, $i \in \{0, 1, \ldots, F\}$, taking values from the finite set $\mathcal{V}_i$. Together, the concatenation of X_u^+ and X_u^- is written X_u, denoting the entire collection of meeting requests. Similarly, let $Y_u^+ \triangleq (Y_{u,1}, Y_{u,2}, \ldots, Y_{u,M_u})$ denote the labels for the M_u labelled meeting requests. We will write Y_u^- to denote the $N_u - M_u$ unknown labels labels corresponding to X_u^-. Each $Y_{u,i} \in \{\texttt{accept}, \texttt{reject}\} \equiv \{1, 0\}$. For $j \in \{1, 2, \ldots, M_u\}$, $D_{u,j} = (X_{u,j}, Y_{u,j})$ denotes the features and label pair of the j-th labelled meeting request. Then, $D_u^+ \triangleq (X_u^+, Y_u^+)$ the set of all labelled meeting requests for the u-th user. By omitting the subscript indicating the user, D^+ represents the entire collection labelled meeting requests across all users. The same holds for X^- and Y^-.

Having defined the meeting acceptance task, we now turn to specifying how we will evaluate candidate models.

3.2 Evaluating probabilistic models on data

Each of the models I evaluate are defined by a prior distribution $P(\Theta)$ over a common, parameterized family of distributions $P(D|\Theta)$. From a Bayesian machine learning perspective, the prior defines the hypothesis space by its support (i.e. regions of positive probability) and our inductive bias as to which models to prefer *a priori*.[3] After making some observations, D, new predictions are formed by averaging the predictions of each model according to its posterior probability, $P(\theta|D)$.

In the classification setting, the goal is to produce optimal label assignments. However, without a particular loss function in mind, we must justify another metric by which to evaluate candidate models. Since the true distribution has the property that it can perform optimally with respect to any loss function, in some sense we want to choose metrics that result in distributions that are "close" to the true distribution.

In this thesis, I use two approaches to evaluate proposed models, both of which can be

[3]In the non-Bayesian setting, priors are used to mix predictions from the various models in a family. After data is observed, the prior is updated by Bayes rule, just as in the Bayesian case. This "mixture approach" was first developed in the information theory community and shown to be optimal in various ways with respect to the self-information loss, or log-loss, as its known in the machine learning community. Consequently, the log-loss plays a special role in Bayesian analysis for model averaging and selection.

		actual label	
		yes	no
predicted	yes	true positive (tp)	false positive (fp)
label	no	false negative (fn)	true negative (tn)

Table 3.1: Confusion Matrix: A confusion matrix details the losses associated each of the four possibilities when performing binary classification. The optimal decision is only a function of $\Delta_1 = fp - tn$ and $\Delta_0 = fn - tp$.

understood as empirical evaluations of two distinct classes of loss functions. The first class of loss functions evaluates label assignments, while the second class evaluates probability assignments. From the first class, I have chosen a range of asymmetric loss functions as well as the the symmetric 0-1 loss, which measures the frequency with which the most probable label under the model is the true label. From the second class, I have chosen the marginal likelihood (or log-loss, or self-information loss), which can be shown to favor models that are close to the "true" distribution (where distance is measured in relative entropy). Furthermore, I will show that the log-loss plays a central role in combining predictions from several models in a principled fashion.

3.2.1 Loss Functions on Label Assignments

When a probabilistic model is built for a classification setting, it is common for the model to be evaluated using a loss function representative of the intended domain. The symmetric 0-1 loss is a common proxy, and optimal decisions under this loss correspond to choosing the maximum a posteriori (MAP) label assignment. For this reason, the resulting classifier is sometimes referred to as the minimum-probability-of-error estimator. Intuitively, if the most probable label according to the model is often the true label, then the model is deemed a good fit. However, this intuition can be misleading.

Consider the following simple example involving a Bernoulli random variable X taking values in $\{0, 1\}$. Let us consider a set of models P, where each member $p \in P$ is a Bernoulli random variable with probability p. We will evaluate these models by computing their expected loss with respect to the true distribution, $Pr[X = 1] = q$. As we will see, the resulting model choices can be counter-intuitive.

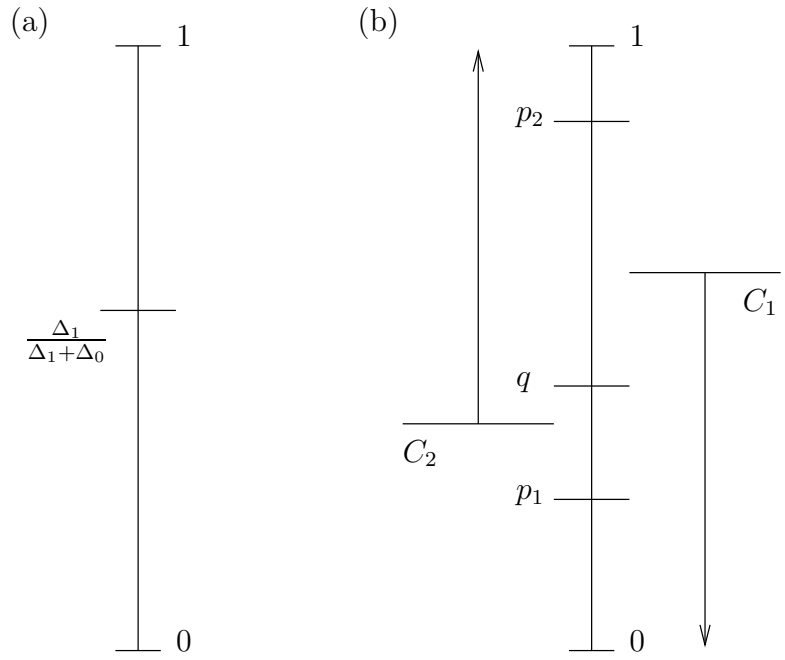

Figure 3-2: (a) Every loss function, (Δ_0, Δ_1), induces a probability threshold, below which models predict $X = 0$ and above which models predict $X = 1$. (b) Two thresholds, C_1 and C_2, divide the space. The side of each threshold on which q lies defines the optimal decision for that threshold. Therefore, loss functions corresponding with the C_1 threshold will prefer the p_1 model while those corresponding to C_2 will prefer the p_2 model.

Let $\Delta_1 > 0$ be the difference in loss between a false positive and true negative and $\Delta_0 > 0$ be the difference in loss between a false negative and true positive (see Table 3.1). It then follows that the optimal decision for a model $p \in P$ is to predict $X = 1$ if

$$p > \frac{\Delta_1}{\Delta_0 + \Delta_1}, \tag{3.1}$$

and $X = 0$ otherwise. Therefore, every loss function defines a probability threshold between 0 and 1, below which all models always predict $X = 0$ and above which all models predict $X = 1$. If the true probability q lies below the threshold, then the optimal decision is $X = 0$, and vice versa (see Figure 3-2a). However, all models that lie on the same side of the threshold will exhibit identical behavior and therefore, will be indistinguishable. In addition, it is obvious that, given any two models p_1, p_2 satisfying $p_1 < q < p_2$, there exists a pair of loss functions (specifically, thresholds) C_1 and C_2 such that, upon evaluation, C_1 prefers p_1 and C_2 prefers p_2, regardless of how close either p_1, p_2 is to the true probability q (see Figure 3-2b). For example, consider the concrete setting where $q = 0.51$, $p_1 = 0.49$ and $p_2 = 1.0$. Then our intuition tells us to prefer p_1 but a symmetric loss function ($\Delta_1 = \Delta_0$) will actually prefer the second model. If we shift the threshold beyond 0.51, then the first model is preferred.

The above example, while contrived, demonstrates an important point: optimizing for one type of loss function can result in model choices that conflict with common-sense intuitions about what constitutes a "good" fit. The above example, however, does suggest that it might be possible to use multiple loss functions to separate out the best model. Unfortunately, as we consider more complex modelling situations, it will become increasingly difficult to design loss functions that will reveal which model should be preferred. Despite its drawback, this approach does result in easy-to-interpret values (e.g. percentage correct on a classification dataset), possibly explaining its widespread use. Besides its intuitiveness, I have chosen to use this metric because it is a common metric in the classification literature. In addition, I will evaluate not only the symmetric loss function but a range of asymmetric loss functions in the hope of highlighting any differences between the models.

3.2.2 Loss Functions on Probability Assignments

The second approach I use in this thesis prefers models that assign higher likelihood to the data. By Gibbs inequality, the model q that maximizes the expected log-likelihood of data generated by some unknown distribution p,

$$E_p[\log q(x)], \tag{3.2}$$

is precisely $q = p$. This suggests that we evaluate the empirical log-loss of each of our candidate models. Choosing the best one corresponds to choosing the distribution that minimizes the relative entropy to the true distribution.

Despite the apparent simplicity of this approach, there are many implementation issues in practice. First and foremost, we cannot evaluate the expectation (3.2) because we do not know the true distribution p. Therefore, we instead calculate the empirical log-loss on the available dataset. The specifics of how we calculate the log-loss depends, at the very least, on how we intend to use the model class on future data. Each of the model classes we are evaluating is a parameterized family of distributions, $P(D|\Theta)$, with a prior distribution $P(\Theta)$, where predictions are made by performing full Bayesian inference. If we imagine that after every prediction we will receive feedback as to whether the correct decision was made, then the appropriate log-loss metric is the marginal likelihood, $P(D) = \int P(D;\Theta)P(\Theta)\,d\Theta$. To see this, note that if $D = (D_1, D_2, \ldots, D_N)$, then we can write

$$P(D) = P(D_1)\,P(D_2|D_1)\,P(D_3|D_1, D_2)\cdots . \tag{3.3}$$

Often, this quantity is difficult to calculate as it involves a high-dimensional integral. However, we will employ stochastic Monte-Carlo techniques that allow us to estimate it efficiently (Neal, 1993).

In the Bayesian setting, we can think about the question of future performance in a fundamentally different way. Surprisingly, we will find that the marginal likelihood is once again the metric of interest. Instead of choosing a model with which to make predictions, we can combine the predictions of several models together. Intuitively, the more we trust a

model (based upon its performance heretofore), the more weight we assign to its predictions. Under consideration are a set of models $\mathcal{H}$, for which we have prior knowledge in the form of a prior $P(H)$, $H \in \mathcal{H}$. After observing data D, the optimal Bayesian prediction averages each classes' prediction according to its posterior probability, $P(H|D)$. By Bayes rule, the posterior probability of a model is proportional to the model's prior probability and the marginal likelihood of the data. Therefore,

$$P(H|D) \propto P(D|H)\, P(H). \tag{3.4}$$

The marginal likelihood, $P(D|H)$, is also known as the *evidence*.

In many classification tasks, we are required only to predict the label, Y, given full knowledge of the features, X. Therefore, it is unnecessary to model the distribution of features and we can instead focus on modelling the conditional distribution $P(Y|X)$. If we use the marginal likelihood to compare models, we will penalize models that poorly predict the features. If we are concerned only with the matching the conditional distribution, the hypotheses (models) we are considering should be independent of the features, i.e. $P(H|X) = P(H)$. This is known as the discriminative setting. The predictive distribution of interest is $P(Y_n|X_n, D)$, where $D = (X_i, Y_i)_{i=1}^{n-1}$ are observed (labelled) meeting requests, X_n is an unlabeled meeting request, and Y_n is the corresponding, but unknown, label. If we are considering several models $H_i \in \mathcal{H}$, we will average according to the posterior probability of the model given the value of the features.

$$\begin{aligned}
P(Y_n|X_n, D) &= \sum_i P(H_i, Y_n|X_n, D) && (3.5)\\
&\propto \sum_i P(Y_n|X_n, D, H_i)\, P(H_i|X_n, D) && (3.6)\\
&\propto \sum_i P(Y_n|X_n, D, H_i)\, P(Y_{i=1}^{n-1}|X_{i=1}^{n-1}, X_n, H_i)\, P(H_i|X_n, X_{i=1}^{n-1}) && (3.7)\\
&\propto \sum_i P(Y_n|X_n, D, H_i)\, P(Y_{i=1}^{n-1}|X_{i=1}^{n-1}, H_i)\, P(H_i) && (3.8)
\end{aligned}$$

Typically $P(Y_n|X_n, D, H) = P(Y_n|X_n, H)$, i.e., the label Y_n is independent of past labels and features conditioned on X_n. When making new predictions, (3.8) illustrates that our

a priori preference for a model and its conditional evidence, $P(Y|X, H)$ determines how much we trust its predictions. For the models presented in this thesis, I will compute both the marginal and conditional evidence. Having defined the meeting task and several metrics, we now analyze actual results on the dataset.

Chapter 4

Results

The usefulness of the type of sharing that the Clustered Naive Bayes supports can only be assessed on real data sets. To that end, we collected a datasets of roughly 180 meeting requests from 21 users across multiple universities, one company and a military training exercise, for a total of about 4000 data points. The clustered model was evaluated on this dataset using the metrics I justified in Chapter 3.2. As a baseline, we compared the performance of the clustered model to the no-sharing and complete-sharing models.

What we find is very interesting. As suggested by theoretical results that provide bounds on the generalization performance of Naive Bayes (Domingos and Pazzani, 1997), the Clustered variant does not perform significantly better than the standard Naive Bayes classifier when evaluated by 0-1 loss. However, as the loss function is made asymmetric, CNB significantly outperforms the standard Naive Bayes model. After comparing the models empirically with respect to the log loss metric (marginal likelihood ratio, Bayes factor), we see why the Clustered Naive Bayes model performing better over the range of loss functions; it apparently does a much better job modelling the data.

4.1 No-Sharing Model

If we train a separate Naive Bayes model on each data set, then we have already shown that no transfer learning can possibly occur. Therefore, the standard Naive Bayes classifier provides a useful baseline against which we can compare the performance of our model of

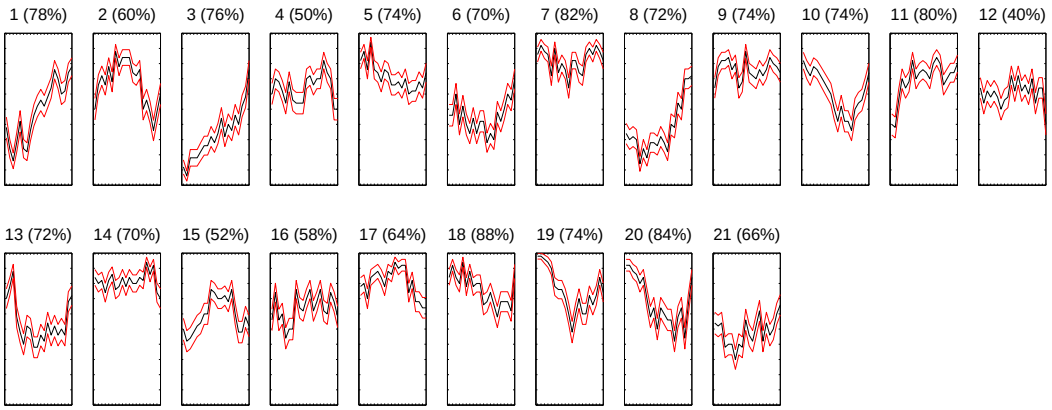

Figure 4-1: Learning curves for all users under the Naive Bayes classifier with 0-1 loss. The y-axis of each graph represents the size of the data set on which the classifier is trained before it is asked to make a small set of predictions. As we move to the right in each graph, more of the data for that user is provided as training. At the far left only 10 data points comprise the training set. At the far right of each graph all but a single point comprise the training set. The vertical axis plots the predictive accuracy from 0% to 100% (each tick marks 10%). Each graph is labelled with an ID number representing the user as well as the performance on the final, leave-one-out trial. Error bars represent the variance of the maximum-likelihood estimator for the 0-1 loss (i.e. we treat each trial as learning a probability).

sharing.

The baseline model was evaluated by performing leave-n-out cross-validation trials, for n takes values between 1 and 98% of the size of each dataset, in 50 equal steps. For each trial, we trained on a random set of the data and calculated the 0-1 loss associated with predictions on held out data. Results were averaged over ten repetitions. The result of these trials for all 21 users appears as Figure 4-1.

Inspecting Figure 4-1 we see that, for several users, the model correctly predicts 80% of the decisions on held-out data. However, the model performs very poorly on other users providing performance near and below chance (50%). That some models perform worse as they get more data suggests that the Naive Bayes model assumption is violated, as we expected it would be.

In order to visualize what the Naive Bayes classifier has learned, we can inspect the MAP estimates of the parameters for all the features. Recall that, under the discriminative

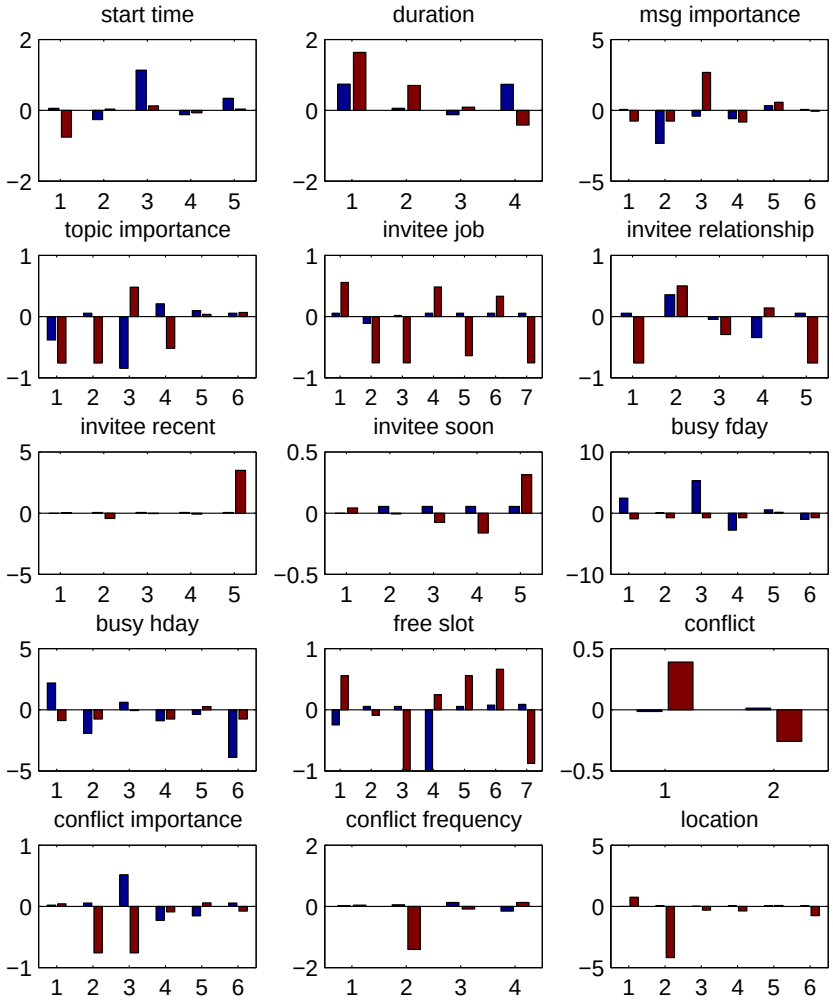

Figure 4-2: Under the discriminative hypothesis space of the Naive Bayes model, the ratio of probability assigned to each feature under both labels is a sufficient statistic for predicting the label. The following 15 graphs correspond to the 15 features in the Meeting Task. Each graph is composed of pairs of log probability ratios, one pair for each value that the feature can take. The left bar of each pair is the probability ratio of the model for User #1 (which performs poorly). The right bar of each pair is associated with the model for User #12 (which performs well). Probabilities correspond to a MAP parameterization. Positive ratios favor acceptance while negative ratios favor rejection. Note that the scale of the vertical axis differs between the graphs (apologies to Tufte).

hypothesis space implicit in the Naive Bayes model, knowing the ratio of the probability of each feature under both labels is a sufficient for predicting the label. In Figure 4-2, we have plotted the log ratios for each feature using the MAP parameterization for the 1st and 12th user. Users 1 and 12 represent the two extremes of performance (78% and 40%).[1] Log ratios close to 0 correspond to features that do not provide discriminatory power under this model. Large negative and positive log ratios, however, dramatically favor one of the hypothesis. The model learned for the first user ignores the features describing (i) how recently he and the requester met, (ii) how soon the meeting is, (iii) whether there is a conflict, (iv) the frequency of a conflict if there is one and (v) the location of the meeting. In contrast (i) the importance of the topic, and (ii) how busy the day have a large effect on the resulting label.

The model learned for the 12th user, which performed below chance, places more weight on (i) the rank of the user requesting the meeting, (ii) whether there is a conflict and (iii) the location of the meeting. The model differs dramatically from the model learned for the 1st user. Interestingly, if we apply the model learned for the 1st user to the 12th user, performance increases to 76%, out-predicting the model learned on all but one of the 12th user's data. Again, this suggests that either the Naive Bayes model is a mismatch or that the data for the 12th user is, in some way, unrepresentative. As we will see, performance for the 12th user improves under the Clustered Naive Bayes, where the 12th user is grouped with two other users.

4.2 Complete-Sharing Model

In some individual cases, predictive performance is not improved by more data, suggesting that the model is flawed. However, on average, as the number of training data grows the accuracy improves. In particular, predicting for the 12th user using training data from the first user improved performance. Perhaps everyone is making identical decisions?

As a first cut, we can assess the hypothesis that all the users are identically distributed

[1] Obviously the decisions made by the 12th model could be inverted to achieve a 60% rate which is higher than several other models. Regardless, we will analyze what assumptions this model is making as it represents the worst performance.

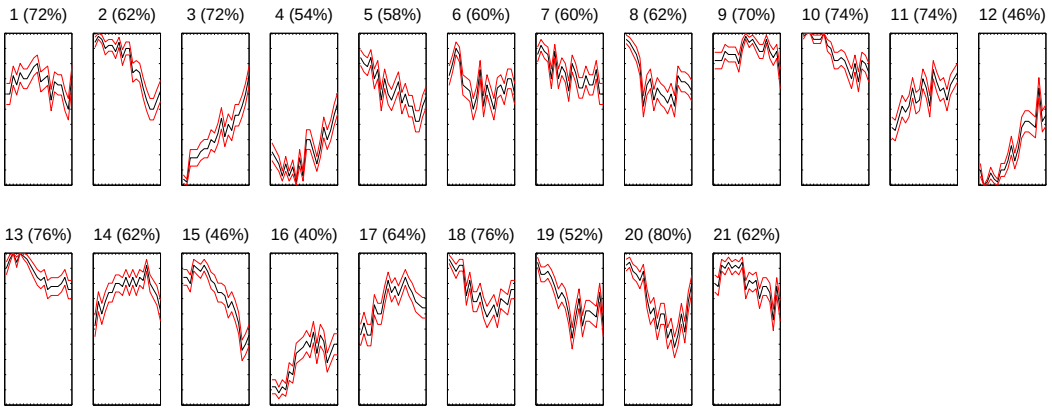

Figure 4-3: Learning curves for all users under the Naive Bayes classifier with 0-1 loss when trained on data from all users. The y-axis of each graph represents the size of the data set on which the classifier is trained before it is asked to make a small set of predictions. As we move to the right in each graph, more of data from all users is provided as training. At the far left 10 data points from each user comprise the training set. At the far right of each graph all but a single data point from each user comprise the training set. The vertical axis plots the predictive accuracy from 0% to 100% (each tick marks 10%). Each graph is labelled with an ID number representing the user as well as the performance on the final, leave-one-out trial. Error bars represent the variance of the maximum-likelihood estimator for the 0-1 loss (i.e. we treat each trial as learning a probability).

according to some Naive Bayes model. The Complete-Sharing model encapsulates this idea (see Section 2.3). Figure 4-3 contains results from a cross-validation experiment where a single classifier is trained on progressively larger subsets of the entire data set (c.f. Figure 4-1). Again, the effect of more data has detrimental effects on the model prediction accuracy for some users. Figure 4-4 directly compares the two models by assessing leave-one-out cross-validation error under 0-1 loss. In this experiment, no-sharing significantly outperformed complete sharing.

Given the no-sharing and complete-sharing models, we can ask which is favored by the data. A quick glance at both models reveals that the no-sharing model has U times as many parameters as the complete-sharing model, where U is the number of users. It is clear that the maximum-likelihood parameterization of the no-sharing model will assign higher likelihood to the data than the complete-sharing model can. However, in the Bayesian framework, models are compared by marginalizing over their parameters to compute the

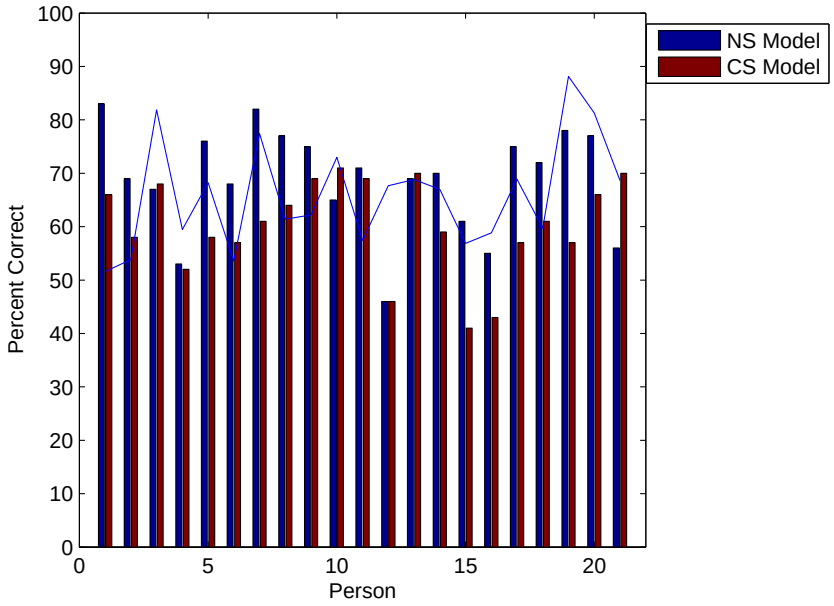

Figure 4-4: Comparison of leave-one-out prediction error for the no-sharing and complete-sharing models under 0-1 loss. No-sharing outperforms complete-sharing on almost all users. The line represents the performance of the classifier that predicts solely on the basis of the marginal probability of the label, ignoring the features entirely. That both models often perform worse than this simple strategy suggests that the Naive Bayes assumption is violated and affecting classification accuracy.

marginal likelihood of the data. Combined with prior distributions over the models, we can compute the posterior probability for both model. In this setting, large model classes are penalized for their complexity.

To determine which model's predictions we should prefer, we will compute the ratio of posterior probability of each model conditioned on our data $D \triangleq \{Y_{d,n}, \mathbf{X}_{d,n}\}$. Let H_{NS} represent the model assumptions of the no-sharing model and H_{CS} represent the model assumptions of the complete-sharing model. Then,

$$\frac{P(H_{NS}|D)}{P(H_{CS}|D)} = \frac{P(H_{NS})P(D|H_{NS})}{P(H_{CS})P(D|H_{CS})} \tag{4.1}$$

For the moment let us concentrate on the ratio of marginal likelihoods.[2] Once we calculate this ratio exactly, it will be clear that our prior knowledge regarding the two hypotheses

[2]We will also refer to the marginal likelihood as the *evidence*.

would be unlikely to affect our posterior beliefs qualitatively (see Appendix A.2 for marginal likelihood equations).

The marginal likelihood under the complete-sharing model, H_{CS}, can be easily calculated by treating all data as if it belonged to a single user. Using the meeting acceptance task data, we find that the log ratio of the marginal likelihoods is:

$$\log \frac{P(D|H_{NS})}{P(D|H_{CS})} = -6.2126 \times 10^4 + 7.3044 \times 10^4 = 1.0918 \times 10^4 \tag{4.2}$$

Therefore, the data alone favor the no-sharing model to the complete-sharing model by a factor of more than $e^{10000} : 1$. Therefore, in light of this likelihood ratio, our prior knowledge concerning the two hypotheses is largely irrelevant; no-sharing is massively preferred by the data. As we might expect from a large group of users, their distributions over meeting requests and decisions are not identical.

In the meeting acceptance task, we are always provided the entire set of feature values and asked to produce a decision (the label). In Section 3.2.2, this is described as a situation where the discriminative approach to classification can be better suited than the generative approach. In a discriminative setting, our hypotheses do not model the features. Therefore models are compared by the likelihood they assign to the labels conditioned on the features (i.e. the *conditional evidence*). In most discriminative settings, the model is a (parameterized) family of conditional distributions $P(Y|X, \Theta, H)$ of labels given features. In order to compute the conditional evidence, it is necessary to marginalize over the parameters. Because, the features are independent of the parameters, the conditional can be written

$$P(Y|X, H) = \sum_{\theta} P(Y|X, \theta, H)P(\theta|H). \tag{4.3}$$

In order to compute the conditional evidence we must rewrite (4.3) in terms of distributions that define the Naive Bayes model. Applying the product rule to $P(Y|X, H)$,

$$P(Y|X, H) = \frac{P(Y, X|H)}{P(X|H)} \tag{4.4}$$

$$= \frac{P(Y, X|H)}{\sum_Y P(Y, X|H)}, \tag{4.5}$$

we see that, because our model is generative in nature, $P(Y|X,H)$ is a function of $P(X|H)$. While we have an efficient way to calculate the numerator (see Appendix (A.2)), the denominator requires that we marginalize over Y. The meeting acceptance task has roughly 4000 data points. Therefore, a brute force approach to calculating the denominator term would require 2^{4000} computations.[3] Instead of computing this quantity exactly, we will approximate it using Monte Carlo methods (Neal, 1993).

Note that $P(X|H)$ is the normalizing constant for the posterior distribution of Y and θ conditioned on X:

$$P(Y,\theta|X,H) = \frac{P(X,Y|\theta,H)\,P(\theta|H)}{P(X|H)} \tag{4.6}$$

We can use importance sampling techniques to estimate $P(X)$ by building a sequence of Markov chains whose invariant distributions converge to (4.6). We can then compute $P(Y|X)$ using Equation (4.4). See Section A.2 for the details of calculating normalizing constants using importance sampling.

For our dataset, the ratio of conditional evidence for the no-sharing, H_{NS}, and complete-sharing, H_{CS}, models is:

$$\log \frac{P(Y|X,H_{NS})}{P(Y|X,H_{CS})} = 1.4587 \times 10^3 \tag{4.7}$$

Therefore, if we compare models only on their ability to discriminate, the no-sharing model is still preferred, but by a smaller factor. Again our prior knowledge is largely irrelevant since no-sharing is preferred more than $e^{1400} : 1$.

With less data we might expect the preference for the no-sharing models to be smaller. Indeed this intuition is correct. Figure 4-5 contains a plot of the empirical average evidence and conditional evidence ratios versus the size of a random subset of the data. On average, as we get more data, the data prefer the no-sharing model.

In light of these results, it appears that there is strong evidence against complete sharing between individuals. On the other hand, we have seen that in some cases, using data from

[3] The formula actually relies upon a set of sufficient statistics (in this case, counts). Therefore, fewer than 2^{4000} computations are needed, but the number is still exponential in the number of data points.

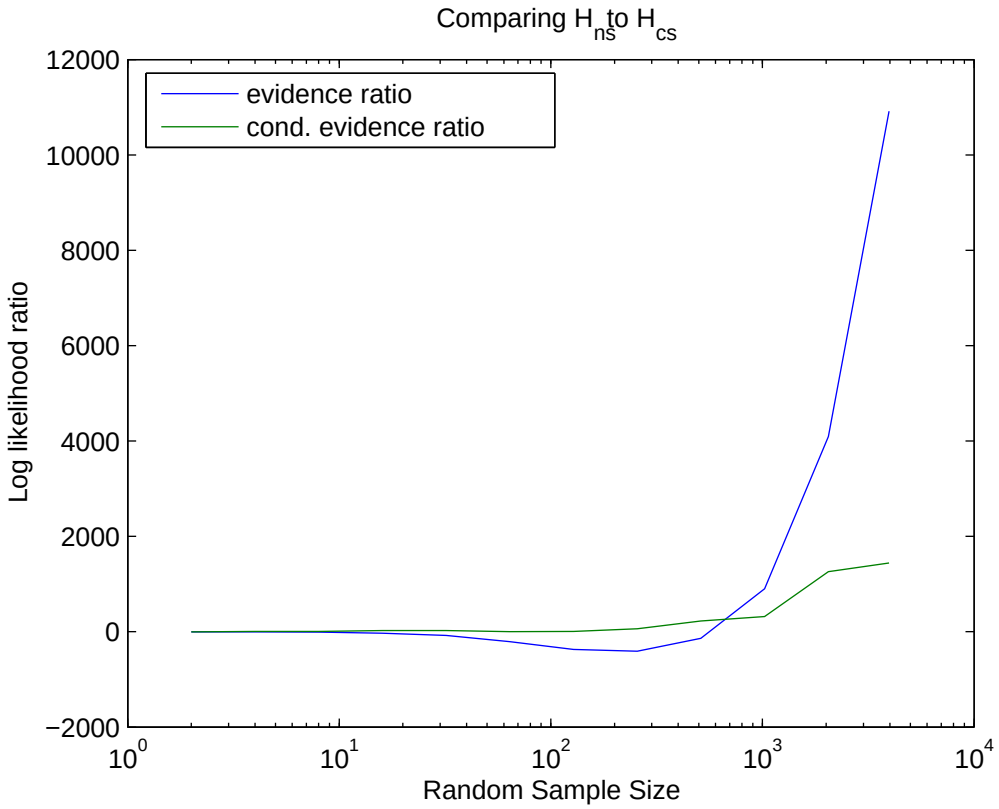

Figure 4-5: Average Evidence and Conditional Evidence Ratios for No-sharing and Complete-sharing as a function of data set size. From both an evidence and conditional evidence perspective, the no-sharing model is preferred when the entire data set is taken into consideration. However, not until more than 100 data points have been witnessed does the preference for the no-sharing model become overwhelmingly. With few data points, the complete-sharing model is actually preferred by the evidence ratio.

other users actually improves classification performance. As a compromise we can investigate partial sharing. In particular, I have already argued that a reasonable assumption about a collection of related data sets is that some grouping exists such that each group is identically distributed. The Clustered Naive Bayes model is exactly the application of this intuition to the standard Naive Bayes classifier.

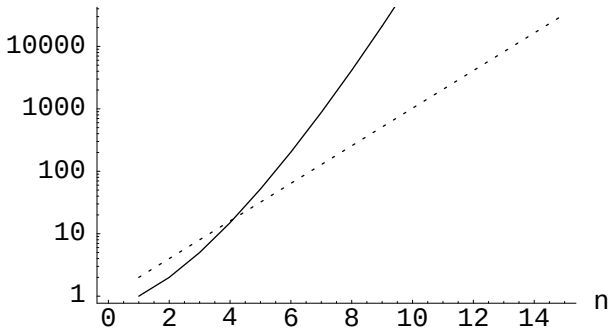

Figure 4-6: Log-plot of the Bell numbers (solid) and powers of 2 (dashed). The Bell numbers grow super-exponentially and the n^{th} Bell number is equivalent to the number of unrestricted partitions of n objects.

4.3 Clustered Naive Bayes Model

We begin our comparison of the clustered and no-sharing model by computing the evidence ratio as we did for the no-sharing and complete-sharing models. Unfortunately, computing the evidence term is much more complicated in the clustered model than in the no-sharing model. In particular, recall that the number of partitions of n objects follows the Bell numbers which grow super-exponentially (see Figure 4-6). In particular, for $n = 21$ objects, there are roughly 475 trillion distinct partitions. Therefore, we have no hope of calculating the exact marginal likelihood for this model by brute force. In contrast, we can easily obtain a lower bound on the evidence that will be sufficient to show that the clustered model assigns higher marginal likelihood to the data than the no-sharing model.

In the clustered model, we have introduced a latent partition z, whose prior distribution is governed by the CRP. By the total probability theorem, we can decompose the marginal likelihood into the contributions of each partition averaged by their prior probabilities. For $n = 21$ objects, the log probability of any partition generated by a CRP with parameter

$\alpha \leq 1$ is bounded below by

$$\log \Pr\{z\} \geq \log \frac{\alpha}{1+\alpha}\frac{\alpha}{2+\alpha}\cdots\frac{\alpha}{20+\alpha} \tag{4.8}$$

$$= \log \frac{\alpha^{20}\Gamma(\alpha)}{\Gamma(20+\alpha)} \tag{4.9}$$

$$= 20\log\alpha + \log\Gamma(\alpha) - \log\Gamma(20+\alpha). \tag{4.10}$$

For our choice of $\alpha = 1$, $\log \Pr\{z\} > -43$. (Note that a uniform prior over partitions of 21 objects would assign approximately -33.8 log probability to every partition.) Therefore, for any partition z^*,

$$\log P(D|H_C) = \log \sum_z P(D|z, H_C)\, P(z|H_C) \tag{4.11}$$

$$> \log P(D|z = z^*, H_C) + \log P(z = z^*|H_C) \tag{4.12}$$

$$> \log P(D|z = z^*, H_C) - 43. \tag{4.13}$$

Therefore, we can lower bound the marginal likelihood of the data assigned by the clustered model by finding a good partition. I performed a stochastic search over the space of partitions, keeping track of the partition that assigned the highest marginal likelihood to the data. Based on the best partition found, it follows that

$$\log P(D|H_C) > -6.1719 \times 10^4 - 43 > -6.1762 \times 10^4. \tag{4.14}$$

Therefore, the log likelihood ratio of the clustered and no-sharing models is

$$\log \frac{P(D|H_C)}{P(D|H_{NS})} > -6.1762 \times 10^4 + 6.2126 \times 10^4 \approx 364. \tag{4.15}$$

Therefore, the clustered model is favored by a factor of at least $e^{364}:1$ over the no-sharing model.

Having compared the models using their marginal likelihood, we return to classic metrics. Figure 4-7 compares the leave-one-out cross-validation performance of the clustered model against both the no-sharing and complete-sharing models. While the clustered model does

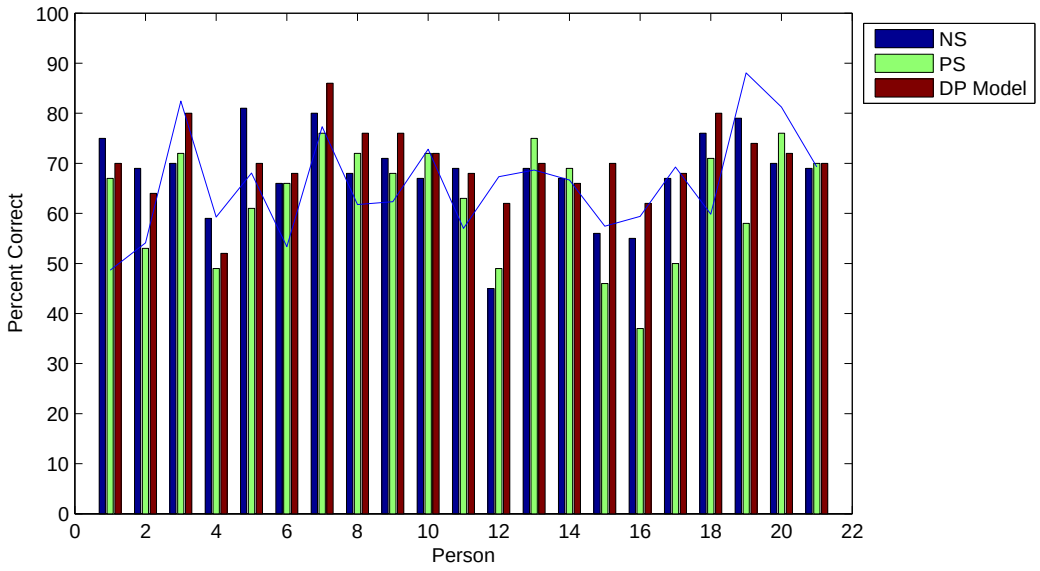

Figure 4-7: Leave-one-out cross validation results for clustered, no-sharing and complete-sharing models. The clustered model performs marginally better. Again, the solid line indicates the *expected* performance of the simple strategy of predicting the most common label for each user (i.e. it ignores the features entirely). In most cases the clustered model performs better than all three other strategies, but there are some cases when the simplest marginal strategy outperforms all methods, suggesting that the Naive Bayes models is a mismatch.

not outperform its competition in every domain, it performs marginally better on average.

Given the dramatic marginal likelihood ratio between the CNB and No-Sharing model, the lackluster results under 0-1 loss are perplexing. As I mentioned in Chapter 3, using a single cost function to discriminate between models can lead to violations of common sense. To tease apart the difference between the CNB and No-Sharing models I tested them both under a range of loss functions of the form

$$L_n(y, y') = \begin{cases} 0 & y = y' \\ 1 & y \neq y' = 1 \qquad \text{false positive} \\ 2^n & y \neq y' = 0, \qquad \text{false negative} \end{cases} \tag{4.16}$$

where n takes on positive and negative values. For $n = 0$, L_0 is the 0-1 loss function. For

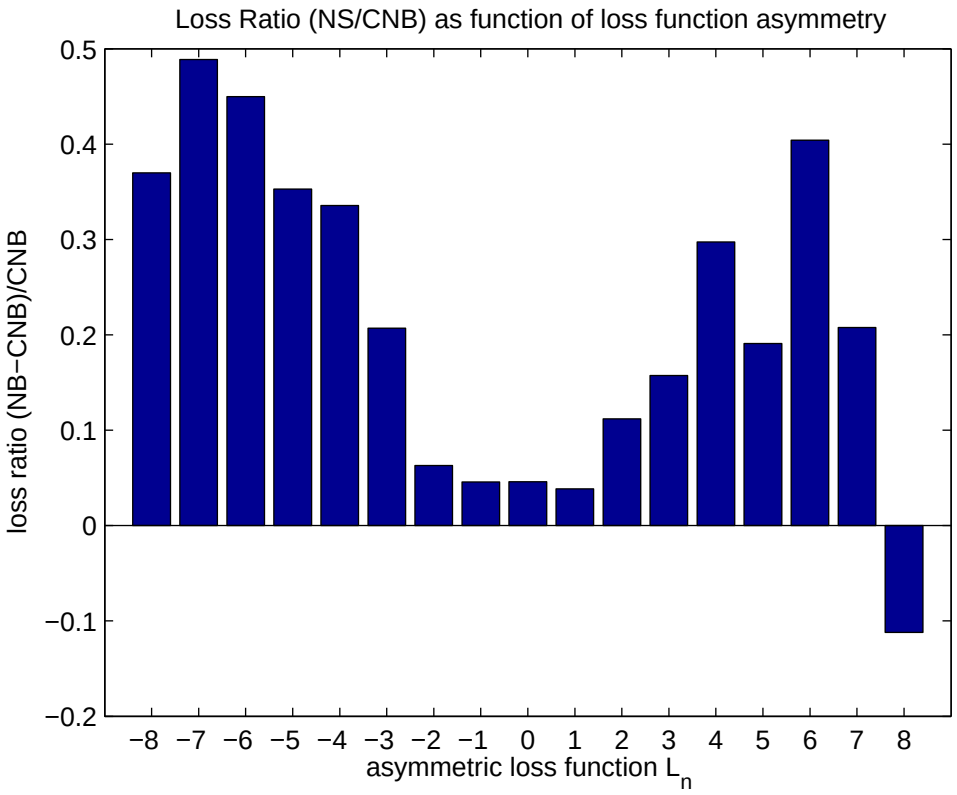

Figure 4-8: Asymmetric loss

Group	User ID	Institution
1	1,2,3,5,6,8	Military
2	4,7	Military
3	9,11	MIT, Michigan
4	10,14,15,21	SRI
5	12,17,20	MIT, MIT, Oregon State
6	13	SRI
7	16	Berkeley
8	18	SRI
9	19	CMU

Table 4.1: Most Common Partitioning of Users: Of the three main groups (military, SRI, professors), none intermixed in the most common partitioning of the data.

$n = -2$, L_{-2} specifies that a false positive is four times worse that a false negative. I conducted leave-5%-out cross validation experiments, under L_n for $n = 0, \pm 1, \pm 2, \ldots, \pm 8$, averaging the results over 90 trials. Figure 4-8 shows the results of this experiment. Under 0-1 loss, L_0, we see that the CNB model only marginally outperforms the no-sharing model. However, as the loss functions become more asymmetric we see that the CNB model begins to significantly outperform the no-sharing model. The trend breaks down at the extremes (especially $n = 8$) as we should expect because both models are approximations and under microscopic scrutiny, inconsistencies in both will be magnified. I have already argued that asymmetric cost functions are more appropriate in the meeting domain (as well as many real-world classification tasks). From this plot we see the benefit of the CNB model modelling the data more closely.

Understanding what has been "learned" by the clustered model is more difficult than under the no-sharing model where MAP parameter estimates are a good approximation to full Bayesian inference. Typically, we must inspect the posterior distribution over partitions using samples from the Markov Chain simulations. Luckily, the posterior distribution over partitions for the meeting task was peaked around a single mode with 9 groups, making it easy to visualize.[4] Table 4.3 shows which users belonged to each group.

Inspecting the partitions, they fall along understandable lines; only military personnel are grouped with military personnel, only researchers at SRI are grouped with other SRI

[4]Any permutation of the numbering within a partitioning is identical. To avoid this we can canonicalize the description of partitions with respect to a fixed number of the domains.

employees, and only professors are grouped with other professors. In addition to the larger groups, there are four singleton groups.

From the Bayesian model selection perspective, the data support the clustered model over its no-sharing counterpart. In addition, inspecting the posterior distribution over partitions, we see that the model finds reasonable groupings. This explains why the Clustered Naive Bayes classifier outperforms the baseline model most significantly when the cost function is asymmetric; it is under these more skewed conditions that the probability assignments become more relevant and the superior modelling capability of the Clustered Naive Bayes model is highlighted.

Chapter 5

Conclusion

The central goal of this thesis was to evaluate the Clustered Naive Bayes model in a transfer-learning setting. This model is characterized by a statistical assumption that, within a collection of datasets, there exists a partitioning such that each cluster of datasets is identically distributed. The Clustered Naive Bayes model is the first application of this idea to a generative model in order to achieve transfer learning. To evaluate the model, I measured its performance on a real-world meeting acceptance task and showed that it fits the data better than its no-sharing counterpart and performs better at prediction, especially under asymmetric loss. Unfortunately, inference under the Clustered Naive Bayes model is considerably more difficult than under its Naive Bayes sibling; simulations that take seconds using the Naive Bayes model take minutes using the Clustered Naive Bayes model. Deriving a more efficient inference algorithm, perhaps using variational techniques, would be a crucial next step before this model could be applied to larger datasets. In addition, the choice to use the Naive Bayes model was not motivated by goodness-of-fit criteria but instead by ease-of-analysis and implementation considerations. A logical next step is to investigate this model of sharing on more sophisticated base models.

In this thesis I have used a generative model to perform clustering; implicit in this choice is the additional assumption that similar feature distributions imply similar predictive distributions. On the meeting acceptance task, this assumption improves performance under asymmetric loss. This aspect of the Clustered Naive Bayes model sets it apart from recent work that extends logistic regression to the multi-task setting.

The Clustered Naive Bayes model uses a Dirichlet Process prior to cluster the parameters of models applied to separate tasks. This model of sharing is immediately applicable to any collection of tasks whose data are modelled by the same parameterized family of distributions, whether those models are generative or discriminative. This thesis suggests that clustering parameters with the Dirichlet Process is worthwhile and can improve prediction performance it situations where we are presented with multiple, related tasks. A theoretical question that deserves attention is whether we can get improved generalization bounds using this technique. Unfortunately, requiring that each of the datasets be modelled by the same parameterized family rules out interesting scenarios where transfer would be useful. More work is necessary to understand how transfer can be achieved between distinct types of tasks.

Appendix A

Implementing Inference

Let $m_{u,y}$ denote the number of data points labeled $y \in \mathcal{L}$ in the data set associated with the uth user and let $n_{u,y,f,x}$ denote the number of instances in the u-th dataset where the f feature takes the value $x \in \mathcal{V}_f$ when its parent label takes value $y \in \mathcal{L}$. Recall that the family of Naive Bayes models with a finite number of labels and F features taken values from finite sets can be written in general as

$$P(D|\theta,\phi) = \prod_{u=1}^{U}\prod_{n=1}^{N_u} P(Y_{u,n}|\phi_{u,y}) \prod_{f=1}^{F} P(X_{u,n,f}|Y_{u,y},\theta_{u,y,f}) \tag{A.1}$$

$$= \prod_{u=1}^{U}\prod_{y\in\mathcal{L}} \phi_{u,y}^{m_{u,y}} \prod_{f=1}^{F}\prod_{x\in\mathcal{V}_f} \theta_{u,y,f,x}^{n_{u,y,f,x}}, \tag{A.2}$$

where

$$\theta_{u,y,f,x} \triangleq \Pr\{X_{u,f} = x|Y = y\}, \tag{A.3}$$

and

$$\phi_{u,y} \triangleq \Pr\{Y_u = y\}. \tag{A.4}$$

We stated in Section 2.1 that the maximum likelihood parameterization is given by

$$\phi_{u,y} = \frac{m_{u,y}}{\sum_{y \in \mathcal{L}} m_{u,y}} \tag{A.5}$$

and

$$\theta_{u,y,f,x} = \frac{n_{u,y,f,x}}{\sum_{x \in \mathcal{V}_f} n_{u,y,f,x}}. \tag{A.6}$$

Each of the models considered in this thesis are defined by a particular prior distribution $P(\theta, \phi)$. Let

$$S^k = \{(\gamma_1, \ldots, \gamma_n) \in (0,1)^k : \sum_{i=1}^{n} \gamma_i = 1\} \tag{A.7}$$

be the k-simplex. Recall that ϕ_u parameterizes the $|\mathcal{L}|$ element discrete distribution $P(Y_{u,j})$. Therefore $\phi_u \in S^{|\mathcal{L}|}$. We let $\phi = \{\phi_u : u = 1, 2, \ldots, U\}$ denote the entire collection of distributions over all users. Similarly, $\theta_{u,y,f}$ parameterizes the $|\mathcal{V}_f|$ element discrete distribution $P(X_{u,n,f}|Y_{u,n})$. Therefore, $\theta_{u,y,f} \in S^{|\mathcal{V}_f|}$. We let $\theta = \{\theta_{u,y,f} : u = 1, 2, \ldots, U, y \in \mathcal{L}, f = 1, 2, \ldots, F\}$ denote the entire collection of distributions over all users, features and labels. Let $\Theta = (S^{\mathcal{L}})^U$ be the space of all ϕ_u. Let $\Phi = ((S^{\mathcal{V}_1} \times \cdots \times \mathcal{V}_F)^{|\mathcal{L}|})^U$ be the space of all $\theta_{u,y,f}$. From a Bayesian perspective, the quantity of interest is now the marginal likelihood

$$P(D) = \int_{\Theta \times \Phi} P(D|\theta, \phi)\, dP(\theta, \phi). \tag{A.8}$$

In the following section I describe the implementation details associated with calculating the posterior probability of labels conditioned on labelled data. In the second part of this chapter, I discuss the strategy for calculating the marginal likelihood via stochastic techniques.

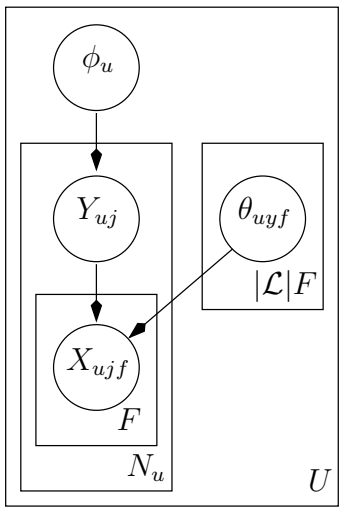

Figure A-1: Graphical Model for No-Sharing model

A.1 Posterior Distributions: Derivations and Samplers

A.1.1 No-Sharing Model

The No-Sharing model assumes that each user's parameters are independently and identically distributed. Specifically,

$$P(\theta, \phi) = \prod_{u=1}^{U} P(\phi_u) \prod_{y \in \mathcal{L}} \prod_{f=1}^{F} P(\theta_{u,y,f}). \tag{A.9}$$

Both $P(\phi_u)$ and $P(\theta_{u,y,f})$ are Dirichlet distributions over the $|\mathcal{L}|$-simplex and $|\mathcal{V}_f|$ simplex, respectively. Recall the generative definition of the No-Sharing (see also Figure A.1.2):

$$\phi_u \sim \text{Dirichlet}(\{\alpha_{u,y} : y \in \mathcal{L}\}) \tag{A.10}$$

$$Y_{u,n} \mid \phi_u \sim \text{Discrete}(\phi_u) \tag{A.11}$$

$$\theta_{u,y,f} \sim \text{Dirichlet}(\{\beta_{u,y,f,x} : x \in \mathcal{V}_f\}) \tag{A.12}$$

$$X_{u,n,f} \mid Y_{u,n}, \{\theta_{u,y,f} : \forall y \in Y\} \sim \text{Discrete}(\theta_{u,(Y_{u,n}),f}) \tag{A.13}$$

The quantity of interest when performing classification is $P(Y^-|D^+, X^-)$, the probability of missing labels, Y^-, given the labelled data, D^+, and the unlabelled meeting requests,

X^-. To calculate this quantity we must relate it to our likelihood functions $P(Y_{u,j}|\phi_u)$ and $P(X_{u,j,f}|Y_{u,j},\{\theta_{u,y,f}\})$. By the total probability theorem,

$$P(Y^-|D^+,X^-) = \int_{\Phi\times\Theta} P(Y^-,\theta,\phi|D^+,X^-)\, d(\theta\times\phi). \tag{A.14}$$

Applying Bayes rule,

$$P(Y^-,\theta,\phi|D^+,X^-) = \frac{P(Y^-,X^-,D^+|\theta,\phi)P(\theta,\phi)}{P(D^+,X^-)} \tag{A.15}$$

$$\propto P(Y^-,X^-,D^+|\theta,\phi)P(\theta,\phi), \tag{A.16}$$

where we have dropped multiplicative terms that are not functions of the parameters or Y^-. But $P(Y^-,X^-,D^+|\theta,\phi) = P(D|\theta,\phi)$ (see Equation (A.2)), where D is the entire dataset, labelled and unlabelled. Plugging in our priors and rearranging,

$$P(Y^-|D^+,X^-) \tag{A.17}$$

$$\propto \int_{\Phi\times\Theta} \left(\prod_{u=1}^{U}\prod_{y\in\mathcal{L}} \phi_{u,y}^{m_{u,y}} \prod_{f=1}^{F}\prod_{x\in\mathcal{V}_f} \theta_{u,y,f,x}^{n_{u,y,f,x}}\right)\left(\prod_{u=1}^{U} P(\phi_u) \prod_{y\in\mathcal{L}}\prod_{f=1}^{F} P(\theta_{u,y,f})\right) d(\theta\times\phi) \tag{A.18}$$

$$= \int_{\Phi\times\Theta} \left(\prod_{u=1}^{U} P(\phi_u) \prod_{y\in\mathcal{L}} \phi_{u,y}^{m_{u,y}}\right)\left(\prod_{u=1}^{U}\prod_{y\in\mathcal{L}}\prod_{f=1}^{F} P(\theta_{u,y,f}) \prod_{x\in\mathcal{V}_f} \theta_{u,y,f,x}^{n_{u,y,f,x}}\right) d(\theta\times\phi) \tag{A.19}$$

Recall that the individual parameter distributions are distributed by Dirichlet distributions. Specifically,

$$P(\phi_u) = \frac{1}{Z(\{\alpha_{u,y} : y\in\mathcal{L}\})} \prod_{y\in\mathcal{L}} \phi_{u,y}^{(\alpha_{u,y}-1)}, \tag{A.20}$$

and

$$P(\theta_{u,y,f}) = \frac{1}{Z(\{\beta_{u,y,f,x} : x\in\mathcal{V}_f\})} \prod_{x\in\mathcal{V}_f} \theta_{u,y,f,x}^{(\beta_{u,y,f,x}-1)}, \tag{A.21}$$

where the function

$$Z(\mathcal{X}) \triangleq \frac{\prod_{x\in\mathcal{X}} \Gamma(x)}{\Gamma(\sum_{x\in\mathcal{X}} x)}. \tag{A.22}$$

maps a finite *multiset* of positive real numbers to a positive real number that normalizes the distribution for the specific choice of hyperparameters. Substituting (A.20) and (A.21) into (A.19), and distributing the integrals, we get

$$P(Y^-|D^+, X^-) \tag{A.23}$$

$$\propto \int_{\Phi\times\Theta} \left(\prod_{u=1}^{U} \frac{1}{Z(\{\alpha_{u,y} : y \in \mathcal{L}\})} \prod_{y\in\mathcal{L}} \phi_{u,y}^{(m_{u,y}+\alpha_{u,y}-1)} \right) \tag{A.24}$$

$$\left(\prod_{u=1}^{U} \prod_{y\in\mathcal{L}} \prod_{f=1}^{F} \frac{1}{Z(\{\beta_{u,y,f,x} : x \in \mathcal{V}_f\})} \prod_{x\in\mathcal{V}_f} \theta_{u,y,f,x}^{(n_{u,y,f,x}+\beta_{u,y,f,x}-1)} \right) d(\theta\times\phi) \tag{A.25}$$

$$= \prod_{u=1}^{U} \frac{1}{Z(\{\alpha_{u,y} : y \in \mathcal{L}\})} \int_{S^{|\mathcal{L}|}} \left(\prod_{y\in\mathcal{L}} \phi_{u,y}^{(m_{u,y}+\alpha_{u,y}-1)} \right) d\phi_u \tag{A.26}$$

$$\times \prod_{u=1}^{U} \prod_{y\in\mathcal{L}} \prod_{f=1}^{F} \frac{1}{Z(\{\beta_{u,y,f,x} : x \in \mathcal{V}_f\})} \int_{S^{|\mathcal{V}_f|}} \left(\prod_{x\in\mathcal{V}_f} \theta_{u,y,f,x}^{(n_{u,y,f,x+\beta_{u,y,f,x}-1)}} \right) d\theta_{u,y,f} \tag{A.27}$$

Recall that, assuming $a, b > 0$,

$$\int_0^1 \theta^{(a-1)}(1-\theta)^{(b-1)}\, d\theta = \frac{\Gamma(a)\,\Gamma(b)}{\Gamma(a+b)} = Z(a,b). \tag{A.28}$$

This integral can be generalized to the k-simplex, S^k. Assuming $a_i > 0$,

$$\int_{S^k} \prod_{i=1}^{k} \gamma_i^{(a_i-1)}\, d\gamma = \frac{\prod_{i=1}^{k} \Gamma(a_i)}{\Gamma(\sum_{i=1}^{k} a_i)} = Z(\{a_1, \ldots, a_k\}). \tag{A.29}$$

Therefore, we can rewrite (A.27) as

$$P(Y^-|D^+, X^-) \propto \prod_{u=1}^{U} \frac{Z(\{m_{u,y} + \alpha_{u,y} : y \in \mathcal{L}\})}{Z(\{\alpha_{u,y} : y \in \mathcal{L}\})} \prod_{y \in \mathcal{L}} \prod_{f=1}^{F} \frac{Z(\{n_{u,y,f,x} + \beta_{u,y,f,x} : x \in \mathcal{V}_f\})}{Z(\{\beta_{u,y,f,x} : x \in \mathcal{V}_f\})} \tag{A.30}$$

In fact, we can drop the denominator of the first term as it is not a function of Y^-. Generally we are only interested in the marginal $P(Y_{u,j}|D^+, X^-)$, because our loss functions are additive. However, computing this marginal can be difficult as we must marginalize over the $N_u - M_u - 1$ other unknown labels. By brute force, $2^{(N_u - M_u - 1)}$ additions must be performed. However, by inspecting (A.30), it is clear that only the counts n, m affect the probability. It is possible that clever counting arguments could make the marginalization more tractable. Instead, we will use Markov Chain Monte Carlo (MCMC) techniques to simultaneously calculate the marginals over the entire set of unknown labels (Neal, 1993).

We can immediately define a Gibbs sampler for our posterior: starting with some arbitrary labelling, we resample each unknown label $Y_{u,j}$, fixing the values for $Y_{u',i}, (u,j) \neq (u',i)$. The conditional distribution is exactly Equation A.30. Note that the update rule is the same for each unknown label.

A.1.2 Complete-Sharing Model

The Complete-Sharing model assumes that $\phi_u = \phi_{u'}$ and $\theta_{u,y,f} = \theta_{u',y,f}$ for all users u and u'. Because corresponding parameters between users are constrained to be identical, we will write ϕ instead of ϕ_u and $\theta_{y,f,x}$ instead of $\theta_{u,y,f,x}$. Therefore the prior is

$$P(\theta, \phi) = P(\phi) \prod_{y \in \mathcal{L}} \prod_{f=1}^{F} P(\theta_{y,f}), \tag{A.31}$$

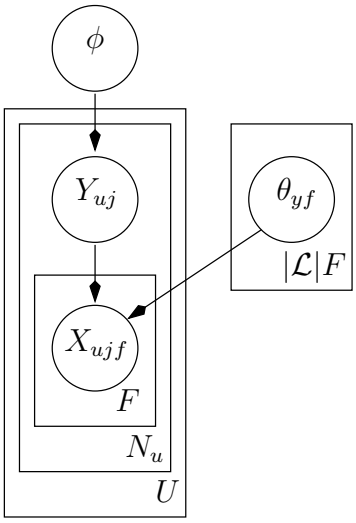

Figure A-2: Complete-Sharing Graphical Model

where the component distributions are Dirichlet distributions. Formally, the Complete-Sharing model is defined by the following generative process:

$$\phi \sim \text{Dirichlet}(\{\alpha_y : y \in \mathcal{L}\}) \tag{A.32}$$

$$Y_{d,n} \mid \phi \sim \text{Discrete}(\phi) \tag{A.33}$$

$$\theta_{y,f} \sim \text{Dirichlet}(\{\beta_{y,f,x} : x \in \mathcal{V}_f\}) \tag{A.34}$$

$$X_{d,n,f} \mid Y_{d,n}, \{\theta_{y,f} : \forall y \in Y\} \sim \text{Discrete}(\theta_{(Y_{d,n}),f}) \tag{A.35}$$

In order to derive the posterior distribution and sampler, we need only recognize that the complete-sharing model is equivalent to the no-sharing model where all the data belongs to a single user whose parameters are ϕ and θ.

Let m_y denote the number of data points labeled $y \in \mathcal{L}$ across the entire data set (i.e. across all users) and let $n_{y,f,x}$ denote the number of instances where the f feature takes the value $x \in \mathcal{V}_f$ when its parent label takes value $y \in \mathcal{L}$. In terms of the quantities $m_{u,y}$ and $n_{u,y,f,x}$,

$$m_y = \sum_{u=1}^{U} m_{u,y} \tag{A.36}$$

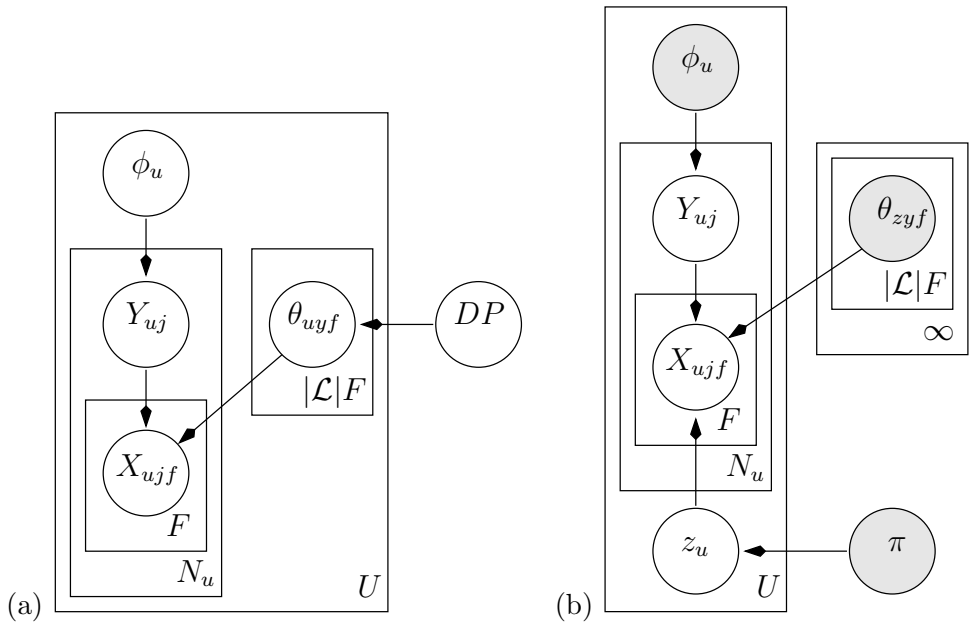

Figure A-3: Clustered Naive Bayes Graphical Model: (a) The parameters of the Clustered Naive Bayes model are drawn from a Dirichet Process. (b) (b) In Chinese Restaurant Process representation, each user u is associated with a "table" z_u, which indexes an infinite vector of parameters drawn i.i.d. from the base distribution. The darkened nodes are marginalized over analytically. The resulting model cannot be represented graphically using standard notation.

and

$$n_{y,f,x} = \sum_{u=1}^{U} n_{u,y,f,x}. \tag{A.37}$$

Therefore, the posterior over missing labels conditioned on labeled data and features is,

$$P(Y^- | D^+, X^-) \propto \frac{Z(\{m_y + \alpha_y : y \in \mathcal{L}\})}{Z(\{\alpha_y : y \in \mathcal{L}\})} \prod_{y \in \mathcal{L}} \prod_{f=1}^{F} \frac{Z(\{n_{y,f,x} + \beta_{y,f,x} : x \in \mathcal{V}_f\})}{Z(\{\beta_{y,f,x} : x \in \mathcal{V}_f\})} \tag{A.38}$$

Similarly, the Gibbs sampler involves resampling each unknown label while holding all others fixed. The conditional (sampling) distribution is proportional to (A.38).

A.1.3 Clustered Naive Bayes

The Clustered Naive Bayes model assumes that the prior distribution over the feature parameters is a Dirichlet Process. Specifically,

$$\theta_u \sim \text{DirichletProcess}(\gamma, P(\theta)), \tag{A.39}$$

where γ is the stick-breaking constant and $P(\theta)$ is the base distribution. In our case, the base distribution is a product of Dirichlet distributions,

$$P(\theta) = \prod_{y \in \mathcal{L}} \prod_{f=1}^{F} P(\theta_{y,f}), \tag{A.40}$$

where the component distributions, $P(\theta_{y,f})$, are Dirichlet distributions over the $|\mathcal{V}_f|$-simplex, respectively. The marginal parameters ϕ are drawn independently from Dirichlet distributions over the $|\mathcal{L}|$-simplex. The distribution of each data point for user u conditioned on the parameter Θ_u is:

$$Y_{u,n} \mid \phi_u \sim \text{Discrete}(\phi_u) \tag{A.41}$$

$$X_{u,n,f} \mid Y_{u,n}, \{\theta_{u,y,f}\} \sim \text{Discrete}(\theta_{u,Y_{u,n},f}) \tag{A.42}$$

We will use the CRP representation of the Dirichlet Process to build a Gibbs sampler. In the CRP representation, the DP is represented as an infinite vector of parameters indexed by mixture component indices z_u which follow the Chinese Restaurant Process. Because the base distribution is conjugate with our sampling distributions, we can analytically integrate out the infinite vector. We will build a Gibbs sampler. The distributions we need to specify are the conditional distribution of a mixture index given all other indices and the data, $P(z_u|z_{-u}, X, Y)$, and the conditional distribution of a missing label conditioned on the mixture indices, and all other features and labels, $P(Y_{u,i}|Y, X, z)$. Using Bayes rule,

$$P(z_u = z^*|z_{-u}, \gamma, Y, X) \propto P(X|Y, z)\ P(z_u = z^*|z_{-u}), \tag{A.43}$$

where $P(z_u|z_{-u})$ is governed by the CRP and

$$P(X|Y,z) \propto \prod_y \prod_f \frac{\Gamma(\sum_{z\neq z^*} m_{z,y} + \alpha_y)}{\Gamma(\sum_z m_{z,y} + \alpha_y)} \prod_{x\in\mathcal{V}_f} \frac{\Gamma(\sum_z n_{z,y,f,x} + \beta_{y,f,x})}{\Gamma(\sum_{z\neq z^*} n_{z,y,f,x} + \beta_{y,f,x})}, \tag{A.44}$$

where $n_{z,y,f,x}$ are the number of features f taken the value x with parent label y in all data grouped in the z-th mixture (similarly for $m_{z,y}$ for labels). The conditional distribution needed to resample a missing label Y is:

$$P(Y_{u,i} = y|Y,X,z) \propto P(X|Y,Y_{u,i},z) \underbrace{\int_\phi P(Y_{u,i} = y, Y|\phi)P(\phi)\,d\phi}_{\mathcal{Y}}, \tag{A.45}$$

where

$$P(X|Y,Y_{u,i},z) \propto \prod_f \frac{n_{z,y,f,x} + \beta_{y,f,x}}{m_{z,y} + \sum_x \beta y,f,x} \tag{A.46}$$

and

$$\mathcal{Y} \propto m_{z,y} + \alpha_y. \tag{A.47}$$

A.2 Calculating Evidence and Conditional Evidence

A.2.1 Evidence

To determine which model's predictions we should prefer, we can compute the ratio of posterior probability of each model conditioned on our data $D \triangleq \{Y_{d,n}, \mathbf{X}_{d,n}\}$.

$$\frac{P(H_{NS}|D)}{P(H_{PS}|D)} = \frac{P(H_{NS})P(D|H_{NS})}{P(H_{PS})P(D|H_{PS})} \tag{A.48}$$

In this section I discuss the work necessary to calculate the marginal likelihood ratios (are known as "evidence" ratios). For the complete-sharing, we can derive the joint distribution analytically because we have chosen conjugate prior distributions. We will specialize to the

case where all the Dirichlet distributions are uniform. Specifically,

$$P(D|H_{CS}) \tag{A.49}$$

$$= \int_{\phi} \int_{\{\theta_{y,f}\}} P(D|\{\phi\}, \{\theta_{y,f}\}, H_{CS})\, p(\phi)\, p(\{\theta_{y,f}\})\, d\{\theta_{y,f}\}\, d\phi \tag{A.50}$$

$$= \Gamma(|Y|) \frac{\prod_y \Gamma(1 + |\{n : Y_n = y\}|)}{\Gamma(|Y| + N)} \prod_{y,f} \Gamma(|X_f|) \frac{\prod_x \Gamma(1 + |\{n : Y_n = y, X_{n,f} = x\}|)}{\Gamma(|X_f| + |\{n : Y_n = y\}|)} \tag{A.51}$$

The marginal likelihood for the no-sharing model can be easily calculated by treating every data set as if it were an isolated complete-sharing model.

A.2.2 Conditional Evidence

We can estimate the expectation of a function $\phi(x)$ under a distribution $P(x)$ given N independent samples $(x_1, \ldots, x_n)$ from $P(x)$ using

$$\langle \hat{\phi} \rangle = \frac{1}{N} \sum \phi(x_n) \tag{A.52}$$

This estimator is asymptotically unbiased. However, in practice we cannot generate independent samples from the distribution of interest. Instead, we can only evaluate $P(x)$ to a multiplicative constant, i.e. we can evaluate $P^*(x) \propto P(x)$.

Importance sampling is a means by which one can evaluate the expectation of a function $\phi(x)$ under a distribution $P(x) = \frac{1}{Z_P} P^*(x)$ (MacKay, 2003, pg. 361). Importance sampling requires a sampling distribution $Q(x) = \frac{1}{Z_Q} Q^*(x)$ from which we can draw indepedent samples and that we can evaluate to a multiplicative constant. The sampling density must also assign positive density to all points with positive density under $P(x)$, i.e. $P(x) > 0 \implies Q(x) > 0$. The idea is to sample from $Q(x)$ instead of $P(X)$ and treat those samples as if they were from $P(X)$. However, for x such that $Q(x) > P(x)$, we expect $\phi(x)$ to be overrepresented at that point, and for x such that $P(x) > Q(x)$ we expect $\phi(x)$ to be underrepresented. The solution is to weigh the sample x_n according to the ratio $P^*(x_n)/Q^*(x_n)$. Given such a sampling distribution, an asymptotically unbiased estimator

of $\langle \phi(x) \rangle$ is:

$$\langle \phi \rangle_{IS} = \frac{1}{N} \frac{\sum w_n \phi(x_n)}{\sum w_n} \tag{A.53}$$

where $w_n = P^*(x_n)/Q^*(x_n)$. The expectation of $\sum w_n$ is

$$\langle \sum w_n \rangle = \int Q(x) \frac{P*(x)}{Q^*(x)} \mathrm{d}x = \frac{Z_P}{Z_Q} \tag{A.54}$$

Therefore, we can use importance sampling as a means of generating a estimate of the ratio of normalizing constants. If we know Z_Q we can then get estimates of Z_P directly. Alternatively, if we use the same sampling distribution with two densities, we can compute the ratio of $\frac{Z_{P_1}}{Z_{P_2}}$ using estimates of $\frac{Z_{P_1}}{Z_Q}$ and $\frac{Z_{P_2}}{Z_Q}$.

Importance sampling is not without its troubles. If $Q(x)$ and $P(x)$ assign significantly different densities to certain regions, then we can expect that a few samples with enormous weight will dominate, resulting in high variance estimates. There are pedagogical examples that exist in one dimesion where the variance of the estimate is infinite (MacKay, 2003, pg. 386). In practice, even small mismatches between high-dimensional $P(x)$ and $Q(x)$ lead to estimates with high variance, rendering importance sampling ineffective.

Neal (2001) introduced annealed importance sampling (AIS), a method that combines simulated annealing, Markov chain Monte Carlo methods and importance sampling. Unlike Markov chain Monte Carlo (MCMC) techniques, AIS does not require convergence to work. In addition, AIS does not suffer from the problems of high variance associated with importance sampling because it performs importance sampling in an extended state space induced by a sequence of distributions p_τ that smoothly transition from a distribution, $Q(X)$, we can sample from, to the distribution of interest, $P(x)$. The approach resembles the simulated annealing heuristic employed in optimization and MCMC because it uses a sequence of distributions parameterized by a temperature τ. As $\tau \to 1$, $p_\tau \to P(x)$, the distribution of interest. The temperature schedule $\tau_0, \tau_1, \ldots, \tau_M$ is manipulated to make the difference between successive distributions $p_{\tau_j}(x)$ and $p_{\tau_{j+1}}(x)$ in the sequence arbitrarily small, thereby controlling the variance of individual importance weights. AIS produces independent, weighted samples from $P(x)$ and unbiased estimates of $Z_1/Z_0 = Z_P/Z_0$ with

variance that can be decreased by increasing the resolution of the temperature schedule at the expense of processing time.

A.2.3 Implementing AIS to compute the conditional evidence

According to equation (4.6), $P(Y, \Theta|X)$ is a distribution whose normalizing constant, $P(X)$, is the one we seek. The sequence of distributions we use to compute $P(X)$ is

$$p_\tau(Y, \Theta) = P(X, Y|\Theta)^\tau P(\Theta) \tag{A.55}$$

for $\tau \in [0, 1]$. By definition, $p_0(y) = P(Y)$ and $p_1(y) \propto P(Y, \Theta|X)$. Therefore, the AIS process will produce unbiased estimates of Z_1/Z_0, where $Z_1 = P(X)$ and Z_0 is the normalizing constant for our prior. We can compute Z_0 exactly by noting that Θ and Y are independent under p_0. $P(\Theta)$ already accounts for contribution of Θ to the normalizing constant. Under p_0, the labels are independent and uniformly distributed binary random variables. Therefore $1/Z_0$ is simply $2^{|\mathcal{L}|}$.

For each distribution p_τ, AIS requires a corresponding Markov chains that leaves p_τ invariant. We can readily obtain independent samples from the prior distribution. What remains is to define Markov chains for every $\tau \in (0, 1]$. For arbitrary τ, our parameterized distribution can be rewritten as

$$p_\tau(\Theta, y) = P(\Theta)P(D|\Theta)^\tau = (\prod_n P(Y_n|\phi) \prod_f P(X_n f|Y_n, \Theta_f, y))^\tau P(\Theta) \tag{A.56}$$

$$= \prod_n (P(Y_n|\phi)^\tau) \prod_f (P(X_n f|Y_n, \Theta_f, y))^\tau) P(\Theta) \tag{A.57}$$

Close inspection suggests that we may be able to treat the distribution p_τ as if it were p_1 with τ copies of each sample of Y and X. Because our data model is an exponential family distribution with conjugate priors, it is closed under IID sampling; τ copies (even fractional ones) of data points are easily handled by manipulating sufficient statistics. After we formally derive Gibbs updates, we will see that this intuition is correct.

For intermediate distributions, we define a parameterized Gibbs sampler that induces a Markov chain M_τ that leaves p_τ invariant. A Gibbs sampler is a type of Metropolis-Hasting

sampler, whose proposal distributions are conditional distributions for the parameters (Θ and the labels Y in our case). The conditional distribution for a label Y_n is

$$p_\tau(Y_n|\Theta, X, Y/Y_n) = \frac{p_\tau(Y, \Theta|X)}{p_\tau(Y/Y_n, \Theta|X)} \tag{A.58}$$

$$\propto p_\tau(Y, \Theta|X) \tag{A.59}$$

$$= P(X, Y|\Theta)^\tau P(\Theta) \tag{A.60}$$

$$\propto \prod_F (\phi_{(F),(Y_n),(X_{n,f})})^\tau \tag{A.61}$$

As we suspected, each sample of Y has τ times the effect. Clearly, for $\tau = 0$, the Y's are sampled uniformly. We will need to take this into account when calculating the normalizing constant of p_0, Z_0. The updates for ϕ and each $\theta_{y,f}$ are equally straightforward.

$$p_\tau(\phi|X, Y, \theta) = \frac{p_\tau(Y, \Theta|X)}{p_\tau(Y, \theta|X)} \tag{A.62}$$

$$\propto p_\tau(Y, \Theta|X) \tag{A.63}$$

$$= P(X, Y|\Theta)^\tau P(\Theta) \tag{A.64}$$

$$\propto \prod_{y=1}^{Y} \phi_y^{\alpha_y - 1 + \tau * |\{n:Y_n = y\}|} \tag{A.65}$$

Letting $\alpha'_y = a_y + \tau * |\{n : Y_n = y\}|$, we can see that resampling ϕ is equivalent to sampling from a Dirichlet distribution with parameters $\{\alpha'_y\}$. The conditional distribution for each $\theta_{y,f}$ is similar.

$$p_\tau(\theta_{y,f}|X, Y, \Theta/\theta_{y,f}) = \frac{p_\tau(Y, \Theta|X)}{p_\tau(Y, \Theta/\theta_{y,f}|X)} \tag{A.66}$$

$$\propto p_\tau(Y, \Theta|X) \tag{A.67}$$

$$= P(X, Y|\Theta)^\tau P(\Theta) \tag{A.68}$$

$$\propto \prod_{x=1}^{|X_f|} \theta_{y,f,x}^{\alpha_{y,f,x} - 1 + \tau * |\{n:X_{n,f} = x \wedge Y_n = y\}|} \tag{A.69}$$

Define $\alpha'_{y,f,x} = \alpha_{y,f,x} + \tau * |\{n : X_{n,f} = x \wedge Y_n = y\}|$. Then it is clear that a sample from the conditional distribution of $\theta_{y,f}$ is simply a sample from a Dirichlet distribution with

parameters $\{\alpha'_{y,f,x}\}$.

Let $\tau_0 = 0 < \tau_1 < \tau_2 < \cdots < \tau_M = 1$ be a sequence of temperatures that define our actual sequence of distributions. To compute a single sample of Z_1/Z_0 we first produce an independent sample $w_0 = (y_0, \Theta_0)$ from p_{τ_0}. We then update w_0 using the Markov chain M_{τ_1}, creating a new sample w_1. w_1 is then updated by M_{τ_2} and so on until we create w_{M-1} by updating w_{M-2} with M_{M-1}. With these M samples in hand, our sample of Z_1/Z_0 is:

$$\frac{p_{\tau_1}(w_0)}{p_{\tau_0}(w_0)} \frac{p_{\tau_2}(w_1)}{p_{\tau_1}(w_1)} \cdots \frac{p_{\tau_M}(w_{M-1})}{p_{\tau_{M-1}}(w_{M-1})} = \prod_{j=0}^{M-1} \frac{p_{\tau_{j+1}}(w_j)}{p_{\tau_j}(w_j)} \tag{A.70}$$

The average of N such samples is an unbiased estimate of Z_1/Z_0. For more details see (Neal, 2001).

Appendix B

Features

For each meeting request we extract the following features:

1. start time: morning, afternoon, early, mid, late
2. duration: 15, 30, 60, XXX minutes
3. message importance: other, unimportant, possibly important, important, very important, critical
4. topic importance: other, unimportant, possibly important, important, very important, critical
5. invitee job: other, officer, planner, student, faculty, administrator, funding
6. invitee relation: other, subordinate, peer, supervisor, family
7. invitee recent: 0, 1, 3, 7, XXX
8. invitee soon: 0,1,3,7,XXX
9. busy fday: light-free, light-busy, busy-free, busy-busy, booked-free, booked-busy
10. busy hday: light-free, light-busy, busy-free, busy-busy, booked-free, booked-busy
11. free slot time: 0, 15, 30, 60, 90, 135, XXX
12. conflict?: no, yes
13. conflict importance: other,unimportant,possibly important,important,very important,critical
14. conflict frequency: other,one time,occasionally,regularly
15. location (new-old): local-none,away-none,local-local,away-local,local-away,away-away

Bibliography

J. Aitchison. The statistical analysis of compositional data. *Journal of the Royal Statistical Society, Series B*, 44:139–177, 1982.

D. Aldous. Exchangeability and related topics. *Springer Lecture Notes in Math., No. 1117*, pages 1–198, 1985.

C. E. Antoniak. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *Annals of Statistics*, 2:1152–1174, 1974.

S. Arnborg and G. Sjdin. Bayes rules in finite models. In *European Conference on Artificial Intelligence*, 2000.

A. Barron. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44, 1998.

J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

D. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems 16*, 2004.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *5th Annual ACM Workshop on COLT*, 1992.

R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

R. T. Cox. Probability, frequency, and reasonable expectation. *American Journal of Physics*, 14:1–13, 1946.

D. B. Dahl. Modeling differential gene expression using a dirichlet process mixture model. In *Proceedings of the American Statistical Association*, 2003.

P. Domingos and M. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Journal of Machine Learning*, 29, 1997.

T. Ferguson. A bayesian analysis of non-parametric problems. *Annals of Statistics*, 1, 1973.

C. Guestrin, D. Koller, C. Gearhart, and N. Kanodia. Generalizing plans to new environments in relational mdps. In *International Joint Conference on Artificial Intelligence*, 2003.

J. Y. Halpern. A counterexample to theorems of cox and fine. 10:67–85, 1999.

R. Herbrich, T. Graepel, and C. Campbell. Bayes point machines. 1:245–279, 2001.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of the Eighteenth International Conference on Machine Learning (ICML)*, 2001.

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

M. E. Maron. Automatic indexing: An experimental inquiry. *Journal of the ACM*, 8(3): 404–417, 1961. ISSN 0004-5411.

Z. Marx, M. T. Rosenstein, L. P. Kaelbling, and T. G. Dietterich. Transfer learning with an ensemble of background tasks. In *NIPS Workshop on Inductive Transfer: 10 Years Later*, 2005.

N. Merhav. Universal prediction. *IEEE Transactions on Information Theory*, 44, 1998.

D. Navarro, T. Griffiths, M. Steyvers, and M. Lee. Modeling individual differences using dirichlet processes. *Journal of Mathematical Psychology*, 50:101–122, 2006.

R. M. Neal. Probabilistic inference using markov chain monte carlo methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, 1993.

R. M. Neal. Annealed importance sampling. In *Statistics and Computing*, volume 11, pages 125–139, 2001.

L. Y. Pratt. Non-literal transfer among neural network learners. In R. J. Mammone, editor, *Artificial Neural Networks for Speech and Vision*, pages 143–169. Chapman and Hall, 1993.

M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich. Transfer learning with an ensemble of background tasks. In *NIPS Workshop on Inductive Transfer: 10 Years Later*, 2005.

E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Describing visual scenes using transformed dirichlet processes. In *Advances in Neural Information Processing Systems 18*, 2005.

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 2006.

S. Thrun. Is learning the n-th thing any easier than learning the first? In *Advances in Neural Information Processing Systems 8*, 1996.

S. Thrun and J. O'Sullivan. Discovering structure in multiple learning tasks: The TC algorithm. In L. Saitta, editor, *Proceedings of the 13th International Conference on Machine Learning ICML-96*, San Mateo, CA, 1996. Morgen Kaufmann.

V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, 1995.

P. Wu and T. G. Dietterich. Improving svm accuracy by training on auxiliary data sources. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 110, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-828-5.

Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Learning multiple classifiers with dirichlet process mixture priors. In *Advances in Neural Information Processing Systems 18*, 2005.