9 *Generalized linear models and the exponential family*

9.1 Introduction

We have now encountered a wide variety of probability distributions: the Gaussian, the Bernoulli, the Student t, the uniform, the gamma, etc. It turns out that most of these are members of a broader class of distributions known as the **exponential family**.[1] In this chapter, we discuss various properties of this family. This allows us to derive theorems and algorithms with very broad applicability.

We will see how we can easily use any member of the exponential family as a class-conditional density in order to make a generative classifier. In addition, we will discuss how to build discriminative models, where the response variable has an exponential family distribution, whose mean is a linear function of the inputs; this is known as a generalized linear model, and generalizes the idea of logistic regression to other kinds of response variables.

9.2 The exponential family

Before defining the exponential family, we mention several reasons why it is important:

- It can be shown that, under certain regularity conditions, the exponential family is the only family of distributions with finite-sized sufficient statistics, meaning that we can compress the data into a fixed-sized summary without loss of information. This is particularly useful for online learning, as we will see later.
- The exponential family is the only family of distributions for which conjugate priors exist, which simplifies the computation of the posterior (see Section 9.2.5).
- The exponential family can be shown to be the family of distributions that makes the least set of assumptions subject to some user-chosen constraints (see Section 9.2.6).
- The exponential family is at the core of generalized linear models, as discussed in Section 9.3.
- The exponential family is at the core of variational inference, as discussed in Section 21.2.

[1] The exceptions are the Student t, which does not have the right form, and the uniform distribution, which does not have fixed support independent of the parameter values.

9.2.1 Definition

A pdf or pmf $p(\mathbf{x}|\boldsymbol{\theta})$, for $\mathbf{x} = (x_1, \ldots, x_m) \in \mathcal{X}^m$ and $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d$, is said to be in the **exponential family** if it is of the form

$$\begin{aligned} p(\mathbf{x}|\boldsymbol{\theta}) &= \frac{1}{Z(\boldsymbol{\theta})} h(\mathbf{x}) \exp[\boldsymbol{\theta}^T \boldsymbol{\phi}(\mathbf{x})] && (9.1) \\ &= h(\mathbf{x}) \exp[\boldsymbol{\theta}^T \boldsymbol{\phi}(\mathbf{x}) - A(\boldsymbol{\theta})] && (9.2) \end{aligned}$$

where

$$\begin{aligned} Z(\boldsymbol{\theta}) &= \int_{\mathcal{X}^m} h(\mathbf{x}) \exp[\boldsymbol{\theta}^T \boldsymbol{\phi}(\mathbf{x})] d\mathbf{x} && (9.3) \\ A(\boldsymbol{\theta}) &= \log Z(\boldsymbol{\theta}) && (9.4) \end{aligned}$$

Here $\boldsymbol{\theta}$ are called the **natural parameters** or **canonical parameters**, $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{R}^d$ is called a vector of **sufficient statistics**, $Z(\boldsymbol{\theta})$ is called the **partition function**, $A(\boldsymbol{\theta})$ is called the **log partition function** or **cumulant function**, and $h(\mathbf{x})$ is the a scaling constant, often 1. If $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$, we say it is a **natural exponential family**.

Equation 9.2 can be generalized by writing

$$p(\mathbf{x}|\boldsymbol{\theta}) = h(\mathbf{x}) \exp[\eta(\boldsymbol{\theta})^T \boldsymbol{\phi}(\mathbf{x}) - A(\eta(\boldsymbol{\theta}))] \tag{9.5}$$

where η is a function that maps the parameters $\boldsymbol{\theta}$ to the canonical parameters $\boldsymbol{\eta} = \eta(\boldsymbol{\theta})$. If $\dim(\boldsymbol{\theta}) < \dim(\eta(\boldsymbol{\theta}))$, it is called a **curved exponential family**, which means we have more sufficient statistics than parameters. If $\eta(\boldsymbol{\theta}) = \boldsymbol{\theta}$, the model is said to be in **canonical form**. We will assume models are in canonical form unless we state otherwise.

9.2.2 Examples

Let us consider some examples to make things clearer.

9.2.2.1 Bernoulli

The Bernoulli for $x \in \{0, 1\}$ can be written in exponential family form as follows:

$$\text{Ber}(x|\mu) = \mu^x (1-\mu)^{1-x} = \exp[x \log(\mu) + (1-x)\log(1-\mu)] = \exp[\boldsymbol{\phi}(x)^T \boldsymbol{\theta}] \tag{9.6}$$

where $\boldsymbol{\phi}(x) = [\mathbb{I}(x=0), \mathbb{I}(x=1)]$ and $\boldsymbol{\theta} = [\log(\mu), \log(1-\mu)]$. However, this representation is **over-complete** since there is a linear dependendence between the features:

$$\mathbf{1}^T \boldsymbol{\phi}(x) = \mathbb{I}(x=0) + \mathbb{I}(x=1) = 1 \tag{9.7}$$

Consequently $\boldsymbol{\theta}$ is not uniquely identifiable. It is common to require that the representation be **minimal**, which means there is a unique $\boldsymbol{\theta}$ associated with the distribution. In this case, we can just define

$$\text{Ber}(x|\mu) = (1-\mu) \exp\left[x \log\left(\frac{\mu}{1-\mu}\right)\right] \tag{9.8}$$

Now we have $\phi(x) = x$, $\theta = \log\left(\frac{\mu}{1-\mu}\right)$, which is the **log-odds ratio**, and $Z = 1/(1-\mu)$. We can recover the mean parameter μ from the canonical parameter using

$$\mu = \text{sigm}(\theta) = \frac{1}{1+e^{-\theta}} \tag{9.9}$$

9.2.2.2 Multinoulli

We can represent the multinoulli as a minimal exponential family as follows (where $x_k = \mathbb{I}(x = k)$):

$$\begin{aligned}
\text{Cat}(x|\boldsymbol{\mu}) &= \prod_{k=1}^{K} \mu_k^{x_k} = \exp\left[\sum_{k=1}^{K} x_k \log \mu_k\right] && (9.10)\\
&= \exp\left[\sum_{k=1}^{K-1} x_k \log \mu_k + \left(1 - \sum_{k=1}^{K-1} x_k\right) \log(1 - \sum_{k=1}^{K-1} \mu_k)\right] && (9.11)\\
&= \exp\left[\sum_{k=1}^{K-1} x_k \log\left(\frac{\mu_k}{1 - \sum_{j=1}^{K-1} \mu_j}\right) + \log(1 - \sum_{k=1}^{K-1} \mu_k)\right] && (9.12)\\
&= \exp\left[\sum_{k=1}^{K-1} x_k \log\left(\frac{\mu_k}{\mu_K}\right) + \log \mu_K\right] && (9.13)
\end{aligned}$$

where $\mu_K = 1 - \sum_{k=1}^{K-1} \mu_k$. We can write this in exponential family form as follows:

$$\begin{aligned}
\text{Cat}(x|\boldsymbol{\theta}) &= \exp(\boldsymbol{\theta}^T \boldsymbol{\phi}(\mathbf{x}) - A(\boldsymbol{\theta})) && (9.14)\\
\boldsymbol{\theta} &= [\log \frac{\mu_1}{\mu_K}, \ldots, \log \frac{\mu_{K-1}}{\mu_K}] && (9.15)\\
\boldsymbol{\phi}(x) &= [\mathbb{I}(x = 1), \ldots, \mathbb{I}(x = K-1)] && (9.16)
\end{aligned}$$

We can recover the mean parameters from the canonical parameters using

$$\mu_k = \frac{e^{\theta_k}}{1 + \sum_{j=1}^{K-1} e^{\theta_j}} \tag{9.17}$$

From this, we find

$$\mu_K = 1 - \frac{\sum_{j=1}^{K-1} e^{\theta_j}}{1 + \sum_{j=1}^{K-1} e^{\theta_j}} = \frac{1}{\sum_{j=1}^{K-1} e^{\theta_j}} \tag{9.18}$$

and hence

$$A(\boldsymbol{\theta}) = \log\left(1 + \sum_{k=1}^{K-1} e^{\theta_k}\right) \tag{9.19}$$

If we define $\theta_K = 0$, we can write $\boldsymbol{\mu} = \mathcal{S}(\boldsymbol{\theta})$ and $A(\boldsymbol{\theta}) = \log \sum_{k=1}^{K} e^{\theta_k}$, where $\mathcal{S}$ is the softmax function in Equation 4.39.

9.2.2.3 Univariate Gaussian

The univariate Gaussian can be written in exponential family form as follows:

$$\begin{aligned}
\mathcal{N}(x|\mu,\sigma^2) &= \frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\exp[-\frac{1}{2\sigma^2}(x-\mu)^2] && (9.20)\\
&= \frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\exp[-\frac{1}{2\sigma^2}x^2+\frac{\mu}{\sigma^2}x-\frac{1}{2\sigma^2}\mu^2] && (9.21)\\
&= \frac{1}{Z(\boldsymbol{\theta})}\exp(\boldsymbol{\theta}^T\boldsymbol{\phi}(x)) && (9.22)
\end{aligned}$$

where

$$\begin{aligned}
\boldsymbol{\theta} &= \begin{pmatrix}\mu/\sigma^2\\ \frac{-1}{2\sigma^2}\end{pmatrix} && (9.23)\\
\boldsymbol{\phi}(x) &= \begin{pmatrix}x\\ x^2\end{pmatrix} && (9.24)\\
Z(\mu,\sigma^2) &= \sqrt{2\pi}\sigma\exp[\frac{\mu^2}{2\sigma^2}] && (9.25)\\
A(\boldsymbol{\theta}) &= \frac{-\theta_1^2}{4\theta_2}-\frac{1}{2}\log(-2\theta_2)-\frac{1}{2}\log(2\pi) && (9.26)
\end{aligned}$$

9.2.2.4 Non-examples

Not all distributions of interest belong to the exponential family. For example, the uniform distribution, $X \sim \mathrm{Unif}(a,b)$, does not, since the support of the distribution depends on the parameters. Also, the Student T distribution (Section 11.4.5) does not belong, since it does not have the required form.

9.2.3 Log partition function

An important property of the exponential family is that derivatives of the log partition function can be used to generate **cumulants** of the sufficient statistics.[2] For this reason, $A(\boldsymbol{\theta})$ is sometimes called a **cumulant function**. We will prove this for a 1-parameter distribution; this can be generalized to a K-parameter distribution in a straightforward way. For the first

[2] The first and second cumulants of a distribution are its mean $\mathbb{E}[X]$ and variance $\mathrm{var}[X]$, whereas the first and second moments are its mean $\mathbb{E}[X]$ and $\mathbb{E}[X^2]$.

derivative we have

$$\frac{dA}{d\theta} = \frac{d}{d\theta}\left(\log\int \exp(\theta\phi(x))h(x)dx\right) \tag{9.27}$$

$$= \frac{\frac{d}{d\theta}\int \exp(\theta\phi(x))h(x)dx}{\int \exp(\theta\phi(x))h(x)dx} \tag{9.28}$$

$$= \frac{\int \phi(x)\exp(\theta\phi(x))h(x)dx}{\exp(A(\theta))} \tag{9.29}$$

$$= \int \phi(x)\exp(\theta\phi(x) - A(\theta))h(x)dx \tag{9.30}$$

$$= \int \phi(x)p(x)dx = \mathbb{E}\left[\phi(x)\right] \tag{9.31}$$

For the second derivative we have

$$\frac{d^2A}{d\theta^2} = \int \phi(x)\exp\left(\theta\phi(x) - A(\theta)\right)h(x)(\phi(x) - A'(\theta))dx \tag{9.32}$$

$$= \int \phi(x)p(x)(\phi(x) - A'(\theta))dx \tag{9.33}$$

$$= \int \phi^2(x)p(x)dx - A'(\theta)\int \phi(x)p(x)dx \tag{9.34}$$

$$= \mathbb{E}\left[\phi^2(X)\right] - \mathbb{E}\left[\phi(x)\right]^2 = \text{var}\left[\phi(x)\right] \tag{9.35}$$

where we used the fact that $A'(\theta) = \frac{dA}{d\theta} = \mathbb{E}\left[\phi(x)\right]$.

In the multivariate case, we have that

$$\frac{\partial^2 A}{\partial\theta_i\partial\theta_j} = \mathbb{E}\left[\phi_i(x)\phi_j(x)\right] - \mathbb{E}\left[\phi_i(x)\right]\mathbb{E}\left[\phi_j(x)\right] \tag{9.36}$$

and hence

$$\nabla^2 A(\boldsymbol{\theta}) = \text{cov}\left[\boldsymbol{\phi}(\mathbf{x})\right] \tag{9.37}$$

Since the covariance is positive definite, we see that $A(\boldsymbol{\theta})$ is a convex function (see Section 7.3.3).

9.2.3.1 Example: the Bernoulli distribution

For example, consider the Bernoulli distribution. We have $A(\theta) = \log(1 + e^\theta)$, so the mean is given by

$$\frac{dA}{d\theta} = \frac{e^\theta}{1+e^\theta} = \frac{1}{1+e^{-\theta}} = \text{sigm}(\theta) = \mu \tag{9.38}$$

The variance is given by

$$\frac{d^2A}{d\theta^2} = \frac{d}{d\theta}(1+e^{-\theta})^{-1} = (1+e^{-\theta})^{-2}.e^{-\theta} \tag{9.39}$$

$$= \frac{e^{-\theta}}{1+e^{-\theta}}\frac{1}{1+e^{-\theta}} = \frac{1}{e^\theta+1}\frac{1}{1+e^{-\theta}} = (1-\mu)\mu \tag{9.40}$$

9.2.4 MLE for the exponential family

The likelihood of an exponential family model has the form

$$p(\mathcal{D}|\boldsymbol{\theta}) = \left[\prod_{i=1}^N h(\mathbf{x}_i)\right] g(\boldsymbol{\theta})^N \exp\left(\boldsymbol{\eta}(\boldsymbol{\theta})^T [\sum_{i=1}^N \boldsymbol{\phi}(\mathbf{x}_i)]\right) \tag{9.41}$$

We see that the sufficient statistics are N and

$$\boldsymbol{\phi}(\mathcal{D}) = [\sum_{i=1}^N \phi_1(\mathbf{x}_i), \ldots, \sum_{i=1}^N \phi_K(\mathbf{x}_i)] \tag{9.42}$$

For example, for the Bernoulli model we have $\boldsymbol{\phi} = [\sum_i \mathbb{I}(x_i = 1)]$, and for the univariate Gaussian, we have $\boldsymbol{\phi} = [\sum_i x_i, \sum_i x_i^2]$. (We also need to know the sample size, N.)

The **Pitman-Koopman-Darmois theorem** states that, under certain regularity conditions, the exponential family is the only family of distributions with finite sufficient statistics. (Here, finite means of a size independent of the size of the data set.)

One of the conditions required in this theorem is that the support of the distribution not be dependent on the parameter. For a simple example of such a distribution, consider the uniform distribution

$$p(x|\theta) = U(x|\theta) = \frac{1}{\theta}\mathbb{I}(0 \leq x \leq \theta) \tag{9.43}$$

The likelihood is given by

$$p(\mathcal{D}|\boldsymbol{\theta}) = \theta^{-N}\mathbb{I}(0 \leq \max\{x_i\} \leq \theta) \tag{9.44}$$

So the sufficient statistics are N and $s(\mathcal{D}) = \max_i x_i$. This is finite in size, but the uniform distribution is not in the exponential family because its support set, $\mathcal{X}$, depends on the parameters.

We now descibe how to compute the MLE for a canonical exponential family model. Given N iid data points $\mathcal{D} = (x_1, \ldots, x_N)$, the log-likelihood is

$$\log p(\mathcal{D}|\boldsymbol{\theta}) = \boldsymbol{\theta}^T\boldsymbol{\phi}(\mathcal{D}) - NA(\boldsymbol{\theta}) \tag{9.45}$$

Since $-A(\boldsymbol{\theta})$ is concave in $\boldsymbol{\theta}$, and $\boldsymbol{\theta}^T\boldsymbol{\phi}(\mathcal{D})$ is linear in $\boldsymbol{\theta}$, we see that the log likelihood is concave, and hence has a unique global maximum. To derive this maximum, we use the fact that the derivative of the log partition function yields the expected value of the sufficient statistic vector (Section 9.2.3):

$$\nabla_{\boldsymbol{\theta}} \log p(\mathcal{D}|\boldsymbol{\theta}) = \boldsymbol{\phi}(\mathcal{D}) - N\mathbb{E}\left[\boldsymbol{\phi}(\mathbf{X})\right] \tag{9.46}$$

Setting this gradient to zero, we see that at the MLE, the empirical average of the sufficient statistics must equal the model's theoretical expected sufficient statistics, i.e., $\hat{\boldsymbol{\theta}}$ must satisfy

$$\mathbb{E}\left[\boldsymbol{\phi}(\mathbf{X})\right] = \frac{1}{N}\sum_{i=1}^N \boldsymbol{\phi}(\mathbf{x}_i) \tag{9.47}$$

This is called **moment matching**. For example, in the Bernoulli distribution, we have $\phi(X) = \mathbb{I}(X = 1)$, so the MLE satisfies

$$\mathbb{E}\left[\phi(X)\right] = p(X = 1) = \hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(x_i = 1) \tag{9.48}$$

9.2.5 Bayes for the exponential family *

We have seen that exact Bayesian analysis is considerably simplified if the prior is conjugate to the likelihood. Informally this means that the prior $p(\boldsymbol{\theta}|\boldsymbol{\tau})$ has the same form as the likelihood $p(\mathcal{D}|\boldsymbol{\theta})$. For this to make sense, we require that the likelihood have finite sufficient statistics, so that we can write $p(\mathcal{D}|\boldsymbol{\theta}) = p(\mathbf{s}(\mathcal{D})|\boldsymbol{\theta})$. This suggests that the only family of distributions for which conjugate priors exist is the exponential family. We will derive the form of the prior and posterior below.

9.2.5.1 Likelihood

The likelihood of the exponential family is given by

$$p(\mathcal{D}|\boldsymbol{\theta}) \propto g(\boldsymbol{\theta})^N \exp\left(\boldsymbol{\eta}(\boldsymbol{\theta})^T \mathbf{s}_N\right) \tag{9.49}$$

where $\mathbf{s}_N = \sum_{i=1}^{N} \mathbf{s}(\mathbf{x}_i)$. In terms of the canonical parameters this becomes

$$p(\mathcal{D}|\boldsymbol{\eta}) \propto \exp(N\boldsymbol{\eta}^T \overline{\mathbf{s}} - NA(\boldsymbol{\eta})) \tag{9.50}$$

where $\overline{\mathbf{s}} = \frac{1}{N}\mathbf{s}_N$.

9.2.5.2 Prior

The natural conjugate prior has the form

$$p(\boldsymbol{\theta}|\nu_0, \boldsymbol{\tau}_0) \propto g(\boldsymbol{\theta})^{\nu_0} \exp\left(\boldsymbol{\eta}(\boldsymbol{\theta})^T \boldsymbol{\tau}_0\right) \tag{9.51}$$

Let us write $\boldsymbol{\tau}_0 = \nu_0 \overline{\boldsymbol{\tau}}_0$, to separate out the size of the prior pseudo-data, ν_0, from the mean of the sufficient statistics on this pseudo-data, $\overline{\boldsymbol{\tau}}_0$. In canonical form, the prior becomes

$$p(\boldsymbol{\eta}|\nu_0, \overline{\boldsymbol{\tau}}_0) \propto \exp(\nu_0 \boldsymbol{\eta}^T \overline{\boldsymbol{\tau}}_0 - \nu_0 A(\boldsymbol{\eta})) \tag{9.52}$$

9.2.5.3 Posterior

The posterior is given by

$$p(\boldsymbol{\theta}|\mathcal{D}) = p(\boldsymbol{\theta}|\nu_N, \boldsymbol{\tau}_N) = p(\boldsymbol{\theta}|\nu_0 + N, \boldsymbol{\tau}_0 + \mathbf{s}_N) \tag{9.53}$$

So we see that we just update the hyper-parameters by adding. In canonical form, this becomes

$$\begin{aligned} p(\boldsymbol{\eta}|\mathcal{D}) &\propto \exp\left(\boldsymbol{\eta}^T(\nu_0 \overline{\boldsymbol{\tau}}_0 + N\overline{\mathbf{s}}) - (\nu_0 + N)A(\boldsymbol{\eta})\right)\right) && (9.54) \\ &= p(\boldsymbol{\eta}|\nu_0 + N, \frac{\nu_0 \overline{\boldsymbol{\tau}}_0 + N\overline{\mathbf{s}}}{\nu_0 + N}) && (9.55) \end{aligned}$$

So we see that the posterior hyper-parameters are a convex combination of the prior mean hyper-parameters and the average of the sufficient statistics.

9.2.5.4 Posterior predictive density

Let us derive a generic expression for the predictive density for future observables $\mathcal{D}' = (\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_{N'})$ given past data $\mathcal{D} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ as follows. For notational brevity, we will combine the sufficient statistics with the size of the data, as follows: $\tilde{\boldsymbol{\tau}}_0 = (\nu_0, \boldsymbol{\tau}_0)$, $\tilde{\mathbf{s}}(\mathcal{D}) = (N, \mathbf{s}(\mathcal{D}))$, and $\tilde{\mathbf{s}}(\mathcal{D}') = (N', \mathbf{s}(\mathcal{D}'))$. So the prior becomes

$$p(\boldsymbol{\theta}|\tilde{\boldsymbol{\tau}}_0) = \frac{1}{Z(\tilde{\boldsymbol{\tau}}_0)} g(\boldsymbol{\theta})^{\nu_0} \exp(\boldsymbol{\eta}(\boldsymbol{\theta})^T \boldsymbol{\tau}_0) \tag{9.56}$$

The likelihood and posterior have a similar form. Hence

$$\begin{aligned} p(\mathcal{D}'|\mathcal{D}) &= \int p(\mathcal{D}'|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} && (9.57) \\ &= \left[\prod_{i=1}^{N'} h(\tilde{\mathbf{x}}_i)\right] Z(\tilde{\boldsymbol{\tau}}_0 + \tilde{\mathbf{s}}(\mathcal{D}))^{-1} \int g(\boldsymbol{\theta})^{\nu_0 + N + N'} d\boldsymbol{\theta} && (9.58) \\ &\quad \times \exp\left(\sum_k \eta_k(\boldsymbol{\theta})(\tau_k + \sum_{i=1}^{N} s_k(\mathbf{x}_i) + \sum_{i=1}^{N'} s_k(\tilde{\mathbf{x}}_i)\right) d\boldsymbol{\theta} && (9.59) \\ &= \left[\prod_{i=1}^{N'} h(\tilde{\mathbf{x}}_i)\right] \frac{Z(\tilde{\boldsymbol{\tau}}_0 + \tilde{\mathbf{s}}(\mathcal{D}) + \tilde{\mathbf{s}}(\mathcal{D}'))}{Z(\tilde{\boldsymbol{\tau}}_0 + \tilde{\mathbf{s}}(\mathcal{D}))} && (9.60) \end{aligned}$$

If $N = 0$, this becomes the marginal likelihood of $\mathcal{D}'$, which reduces to the familiar form of normalizer of the posterior divided by the normalizer of the prior, multiplied by a constant.

9.2.5.5 Example: Bernoulli distribution

As a simple example, let us revisit the Beta-Bernoulli model in our new notation.

The likelihood is given by

$$p(\mathcal{D}|\theta) = (1-\theta)^N \exp\left(\log(\frac{\theta}{1-\theta}) \sum_i x_i\right) \tag{9.61}$$

Hence the conjugate prior is given by

$$\begin{aligned} p(\theta|\nu_0, \tau_0) &\propto (1-\theta)^{\nu_0} \exp\left(\log(\frac{\theta}{1-\theta})\tau_0\right) && (9.62) \\ &= \theta^{\tau_0}(1-\theta)^{\nu_0 - \tau_0} && (9.63) \end{aligned}$$

If we define $\alpha = \tau_0 + 1$ and $\beta = \nu_0 - \tau_0 + 1$, we see that this is a beta distribution.

We can derive the posterior as follows, where $s = \sum_i \mathbb{I}(x_i = 1)$ is the sufficient statistic:

$$\begin{aligned} p(\theta|\mathcal{D}) &\propto \theta^{\tau_0 + s}(1-\theta)^{\nu_0 - \tau_0 + n - s} && (9.64) \\ &= \theta^{\tau_n}(1-\theta)^{\nu_n - \tau_n} && (9.65) \end{aligned}$$

We can derive the posterior predictive distribution as follows. Assume $p(\theta) = \text{Beta}(\theta|\alpha, \beta)$, and let $s = s(\mathcal{D})$ be the number of heads in the past data. We can predict the probability of a given sequence of future heads, $\mathcal{D}' = (\tilde{x}_1, \ldots, \tilde{x}_m)$, with sufficient statistic $s' = \sum_{i=1}^m \mathbb{I}(\tilde{x}_i = 1)$, as follows:

$$\begin{aligned} p(\mathcal{D}'|\mathcal{D}) &= \int_0^1 p(\mathcal{D}'|\theta|\text{Beta}(\theta|\alpha_n, \beta_n)d\theta & (9.66) \\ &= \frac{\Gamma(\alpha_n + \beta_n)}{\Gamma(\alpha_n)\Gamma(\beta_n)} \int_0^1 \theta^{\alpha_n + t' - 1}(1-\theta)^{\beta_n + m - t' - 1} d\theta & (9.67) \\ &= \frac{\Gamma(\alpha_n + \beta_n)}{\Gamma(\alpha_n)\Gamma(\beta_n)} \frac{\Gamma(\alpha_{n+m})\Gamma(\beta_{n+m})}{\Gamma(\alpha_{n+m} + \beta_{n+m})} & (9.68) \end{aligned}$$

where

$$\begin{aligned} \alpha_{n+m} &= \alpha_n + s' = \alpha + s + s' & (9.69) \\ \beta_{n+m} &= \beta_n + (m - s') = \beta + (n - s) + (m - s') & (9.70) \end{aligned}$$

9.2.6 Maximum entropy derivation of the exponential family *

Although the exponential family is convenient, is there any deeper justification for its use? It turns out that there is: it is the distribution that makes the least number of assumptions about the data, subject to a specific set of user-specified constraints, as we explain below. In particular, suppose all we know is the expected values of certain features or functions:

$$\sum_{\mathbf{x}} f_k(\mathbf{x})p(\mathbf{x}) = F_k \tag{9.71}$$

where F_k are known constants, and $f_k(\mathbf{x})$ is an arbitrary function. The principle of **maximum entropy** or **maxent** says we should pick the distribution with maximum entropy (closest to uniform), subject to the constraints that the moments of the distribution match the empirical moments of the specified functions.

To maximize entropy subject to the constraints in Equation 9.71, and the constraints that $p(\mathbf{x}) \geq 0$ and $\sum_{\mathbf{x}} p(\mathbf{x}) = 1$, we need to use Lagrange multipliers. The Lagrangian is given by

$$J(p, \boldsymbol{\lambda}) = -\sum_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x}) + \lambda_0(1 - \sum_{\mathbf{x}} p(\mathbf{x})) + \sum_k \lambda_k(F_k - \sum_{\mathbf{x}} p(\mathbf{x}) f_k(\mathbf{x})) \tag{9.72}$$

We can use the calculus of variations to take derivatives wrt the function p, but we will adopt a simpler approach and treat p as a fixed length vector (since we are assuming $\mathbf{x}$ is discrete). Then we have

$$\frac{\partial J}{\partial p(\mathbf{x})} = -1 - \log p(\mathbf{x}) - \lambda_0 - \sum_k \lambda_k f_k(\mathbf{x}) \tag{9.73}$$

Setting $\frac{\partial J}{\partial p(\mathbf{x})} = 0$ yields

$$p(\mathbf{x}) = \frac{1}{Z} \exp(-\sum_k \lambda_k f_k(\mathbf{x})) \tag{9.74}$$

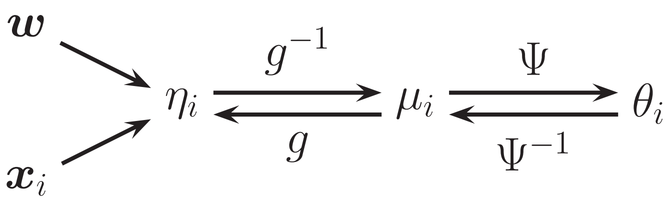


Figure 9.1 A visualization of the various features of a GLM. Based on Figure 8.3 of (Jordan 2007).

where $Z = e^{1+\lambda_0}$. Using the sum to one constraint, we have

$$1 \;=\; \sum_{\mathbf{x}} p(\mathbf{x}) = \frac{1}{Z} \sum_{\mathbf{x}} \exp(-\sum_k \lambda_k f_k(\mathbf{x})) \tag{9.75}$$

Hence the normalization constant is given by

$$Z \;=\; \sum_{\mathbf{x}} \exp(-\sum_k \lambda_k f_k(\mathbf{x})) \tag{9.76}$$

Thus the maxent distribution $p(\mathbf{x})$ has the form of the exponential family (Section 9.2), also known as the **Gibbs distribution**.

9.3 Generalized linear models (GLMs)

Linear and logistic regression are examples of **generalized linear model**s, or **GLM**s (McCullagh and Nelder 1989). These are models in which the output density is in the exponential family (Section 9.2), and in which the mean parameters are a linear combination of the inputs, passed through a possibly nonlinear function, such as the logistic function. We describe GLMs in more detail below. We focus on scalar outputs for notational simplicity. (This excludes multinomial logistic regression, but this is just to simplify the presentation.)

9.3.1 Basics

To understand GLMs, let us first consider the case of an unconditional dstribution for a scalar response variable:

$$p(y_i|\theta, \sigma^2) \;=\; \exp\left[\frac{y_i\theta - A(\theta)}{\sigma^2} + c(y_i, \sigma^2)\right] \tag{9.77}$$

where σ^2 is the **dispersion parameter** (often set to 1), θ is the natural parameter, A is the partition function, and c is a normalization constant. For example, in the case of logistic regression, θ is the log-odds ratio, $\theta = \log(\frac{\mu}{1-\mu})$, where $\mu = \mathbb{E}[y] = p(y = 1)$ is the mean parameter (see Section 9.2.2.1). To convert from the mean parameter to the natural parameter

Distrib.	Link $g(\mu)$	$\theta = \psi(\mu)$	$\mu = \psi^{-1}(\theta) = \mathbb{E}[y]$
$\mathcal{N}(\mu, \sigma^2)$	identity	$\theta = \mu$	$\mu = \theta$
$\text{Bin}(N, \mu)$	logit	$\theta = \log(\frac{\mu}{1-\mu})$	$\mu = \text{sigm}(\theta)$
$\text{Poi}(\mu)$	log	$\theta = \log(\mu)$	$\mu = e^\theta$

Table 9.1 Canonical link functions ψ and their inverses for some common GLMs.

we can use a function ψ, so $\theta = \Psi(\mu)$. This function is uniquely determined by the form of the exponential family distribution. In fact, this is an invertible mapping, so we have $\mu = \Psi^{-1}(\theta)$. Furthermore, we know from Section 9.2.3 that the mean is given by the derivative of the partition function, so we have $\mu = \Psi^{-1}(\theta) = A'(\theta)$.

Now let us add inputs/ covariates. We first define a linear function of the inputs:

$$\eta_i = \mathbf{w}^T \mathbf{x}_i \tag{9.78}$$

We now make the mean of the distribution be some invertible monotonic function of this linear combination. By convention, this function, known as the **mean function**, is denoted by g^{-1}, so

$$\mu_i = g^{-1}(\eta_i) = g^{-1}(\mathbf{w}^T \mathbf{x}_i) \tag{9.79}$$

See Figure 9.1 for a summary of the basic model.

The inverse of the mean function, namely $g()$, is called the **link function**. We are free to choose almost any function we like for g, so long as it is invertible, and so long as g^{-1} has the appropriate range. For example, in logistic regression, we set $\mu_i = g^{-1}(\eta_i) = \text{sigm}(\eta_i)$.

One particularly simple form of link function is to use $g = \psi$; this is called the **canonical link function**. In this case, $\theta_i = \eta_i = \mathbf{w}^T \mathbf{x}_i$, so the model becomes

$$p(y_i|\mathbf{x}_i, \mathbf{w}, \sigma^2) = \exp\left[\frac{y_i \mathbf{w}^T \mathbf{x}_i - A(\mathbf{w}^T \mathbf{x}_i)}{\sigma^2} + c(y_i, \sigma^2)\right] \tag{9.80}$$

In Table 9.1, we list some distributions and their canonical link functions. We see that for the Bernoulli/ binomial distribution, the canonical link is the logit function, $g(\mu) = \log(\eta/(1-\eta))$, whose inverse is the logistic function, $\mu = \text{sigm}(\eta)$.

Based on the results in Section 9.2.3, we can show that the mean and variance of the response variable are as follows:

$$\mathbb{E}\left[y|\mathbf{x}_i, \mathbf{w}, \sigma^2\right] = \mu_i = A'(\theta_i) \tag{9.81}$$

$$\text{var}\left[y|\mathbf{x}_i, \mathbf{w}, \sigma^2\right] = \sigma_i^2 = A''(\theta_i)\sigma^2 \tag{9.82}$$

To make the notation clearer, let us consider some simple examples.

- For linear regression, we have

$$\log p(y_i|\mathbf{x}_i, \mathbf{w}, \sigma^2) = \frac{y_i \mu_i - \frac{\mu_i^2}{2}}{\sigma^2} - \frac{1}{2}\left(\frac{y_i^2}{\sigma^2} + \log(2\pi\sigma^2)\right) \tag{9.83}$$

  where $y_i \in \mathbb{R}$, and $\theta_i = \mu_i = \mathbf{w}^T \mathbf{x}_i$ Here $A(\theta) = \theta^2/2$, so $\mathbb{E}[y_i] = \mu_i$ and $\text{var}[y_i] = \sigma^2$.

- For binomial regression, we have

$$\log p(y_i|\mathbf{x}_i, \mathbf{w}) = y_i \log(\frac{\pi_i}{1-\pi_i}) + N_i \log(1-\pi_i) + \log \binom{N_i}{y_i} \tag{9.84}$$

  where $y_i \in \{0, 1, \ldots, N_i\}$, $\pi_i = \text{sigm}(\mathbf{w}^T\mathbf{x}_i)$, $\theta_i = \log(\pi_i/(1-\pi_i)) = \mathbf{w}^T\mathbf{x}_i$, and $\sigma^2 = 1$. Here $A(\theta) = N_i \log(1+e^\theta)$, so $\mathbb{E}[y_i] = N_i\pi_i = \mu_i$, $\text{var}[y_i] = N_i\pi_i(1-\pi_i)$.

- For **poisson regression**, we have

$$\log p(y_i|\mathbf{x}_i, \mathbf{w}) = y_i \log \mu_i - \mu_i - \log(y_i!) \tag{9.85}$$

  where $y_i \in \{0, 1, 2, \ldots\}$, $\mu_i = \exp(\mathbf{w}^T\mathbf{x}_i)$, $\theta_i = \log(\mu_i) = \mathbf{w}^T\mathbf{x}_i$, and $\sigma^2 = 1$. Here $A(\theta) = e^\theta$, so $\mathbb{E}[y_i] = \text{var}[y_i] = \mu_i$. Poisson regression is widely used in bio-statistical applications, where y_i might represent the number of diseases of a given person or place, or the number of reads at a genomic location in a high-throughput sequencing context (see e.g., (Kuan et al. 2009)).

9.3.2 ML and MAP estimation

One of the appealing properties of GLMs is that they can be fit using exactly the same methods that we used to fit logistic regression. In particular, the log-likelihood has the following form:

$$\ell(\mathbf{w}) = \log p(\mathcal{D}|\mathbf{w}) = \frac{1}{\sigma^2}\sum_{i=1}^N \ell_i \tag{9.86}$$

$$\ell_i \triangleq \theta_i y_i - A(\theta_i) \tag{9.87}$$

We can compute the gradient vector using the chain rule as follows:

$$\frac{d\ell_i}{dw_j} = \frac{d\ell_i}{d\theta_i}\frac{d\theta_i}{d\mu_i}\frac{d\mu_i}{d\eta_i}\frac{d\eta_i}{dw_j} \tag{9.88}$$

$$= (y_i - A'(\theta_i))\frac{d\theta_i}{d\mu_i}\frac{d\mu_i}{d\eta_i}x_{ij} \tag{9.89}$$

$$= (y_i - \mu_i)\frac{d\theta_i}{d\mu_i}\frac{d\mu_i}{d\eta_i}x_{ij} \tag{9.90}$$

If we use a canonical link, $\theta_i = \eta_i$, this simplifies to

$$\nabla_{\mathbf{w}}\ell(\mathbf{w}) = \frac{1}{\sigma^2}\left[\sum_{i=1}^N (y_i - \mu_i)\mathbf{x}_i\right] \tag{9.91}$$

which is a sum of the input vectors, weighted by the errors. This can be used inside a (stochastic) gradient descent procedure, discussed in Section 8.5.2. However, for improved efficiency, we should use a second-order method. If we use a canonical link, the Hessian is given by

$$\mathbf{H} = -\frac{1}{\sigma^2}\sum_{i=1}^N \frac{d\mu_i}{d\theta_i}\mathbf{x}_i\mathbf{x}_i^T = -\frac{1}{\sigma^2}\mathbf{X}^T\mathbf{S}\mathbf{X} \tag{9.92}$$

Name	Formula
Logistic	$g^{-1}(\eta) = \mathrm{sigm}(\eta) = \frac{e^\eta}{1+e^\eta}$
Probit	$g^{-1}(\eta) = \Phi(\eta)$
Log-log	$g^{-1}(\eta) = \exp(-\exp(-\eta))$
Complementary log-log	$g^{-1}(\eta) = 1 - \exp(-\exp(\eta))$

Table 9.2 Summary of some possible mean functions for binary regression.

where $\mathbf{S} = \mathrm{diag}(\frac{d\mu_1}{d\theta_1}, \ldots, \frac{d\mu_N}{d\theta_N})$ is a diagonal weighting matrix. This can be used inside the IRLS algorithm (Section 8.3.4). Specifically, we have the following Newton update:

$$\mathbf{w}_{t+1} = (\mathbf{X}^T\mathbf{S}_t\mathbf{X})^{-1}\mathbf{X}^T\mathbf{S}_t\mathbf{z}_t \quad (9.93)$$

$$\mathbf{z}_t = \boldsymbol{\theta}_t + \mathbf{S}_t^{-1}(\mathbf{y} - \boldsymbol{\mu}_t) \quad (9.94)$$

where $\boldsymbol{\theta}_t = \mathbf{X}\mathbf{w}_t$ and $\boldsymbol{\mu}_t = g^{-1}(\boldsymbol{\eta}_t)$.

If we extend the derivation to handle non-canonical links, we find that the Hessian has another term. However, it turns out that the expected Hessian is the same as in Equation 9.92; using the expected Hessian (known as the Fisher information matrix) instead of the actual Hessian is known as the **Fisher scoring method**.

It is straightforward to modify the above procedure to perform MAP estimation with a Gaussian prior: we just modify the objective, gradient and Hessian, just as we added ℓ_2 regularization to logistic regression in Section 8.3.6.

9.3.3 Bayesian inference

Bayesian inference for GLMs is usually conducted using MCMC (Chapter 24). Possible methods include Metropolis Hastings with an IRLS-based proposal (Gamerman 1997), Gibbs sampling using adaptive rejection sampling (ARS) for each full-conditional (Dellaportas and Smith 1993), etc. See e.g., (Dey et al. 2000) for futher information. It is also possible to use the Gaussian approximation (Section 8.4.1) or variational inference (Section 21.8.1.1).

9.4 Probit regression

In (binary) logistic regression, we use a model of the form $p(y = 1|\mathbf{x}_i, \mathbf{w}) = \mathrm{sigm}(\mathbf{w}^T\mathbf{x}_i)$. In general, we can write $p(y = 1|\mathbf{x}_i, \mathbf{w}) = g^{-1}(\mathbf{w}^T\mathbf{x}_i)$, for any function g^{-1} that maps $[-\infty, \infty]$ to $[0, 1]$. Several possible mean functions are listed in Table 9.2.

In this section, we focus on the case where $g^{-1}(\eta) = \Phi(\eta)$, where $\Phi(\eta)$ is the cdf of the standard normal. This is known as **probit regression**. The probit function is very similar to the logistic function, as shown in Figure 8.7(b). However, this model has some advantages over logistic regression, as we will see.

9.4.1 ML/MAP estimation using gradient-based optimization

We can find the MLE for probit regression using standard gradient methods. Let $\mu_i = \mathbf{w}^T\mathbf{x}_i$, and let $\tilde{y}_i \in \{-1, +1\}$. Then the gradient of the log-likelihod for a specific case is given by

$$\mathbf{g}_i \triangleq \frac{d}{d\mathbf{w}} \log p(\tilde{y}_i|\mathbf{w}^T\mathbf{x}_i) = \frac{d\mu_i}{d\mathbf{w}} \frac{d}{d\mu_i} \log p(\tilde{y}_i|\mathbf{w}^T\mathbf{x}_i) = \mathbf{x}_i \frac{\tilde{y}_i \phi(\mu_i)}{\Phi(\tilde{y}_i\mu_i)} \tag{9.95}$$

where ϕ is the standard normal pdf, and Φ is its cdf. Similarly, the Hessian for a single case is given by

$$\mathbf{H}_i = \frac{d}{d\mathbf{w}^2} \log p(\tilde{y}_i|\mathbf{w}^T\mathbf{x}_i) = -\mathbf{x}_i \left(\frac{\phi(\mu_i)^2}{\Phi(\tilde{y}_i\mu_i)^2} + \frac{\tilde{y}_i\mu_i\phi(\mu_i)}{\Phi(\tilde{y}_i\mu_i)} \right) \mathbf{x}_i^T \tag{9.96}$$

We can modify these expressions to compute the MAP estimate in a straightforward manner. In particular, if we use the prior $p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \mathbf{V}_0)$, the gradient and Hessian of the penalized log likelihood have the form $\sum_i \mathbf{g}_i + 2\mathbf{V}_0^{-1}\mathbf{w}$ and $\sum_i \mathbf{H}_i + 2\mathbf{V}_0^{-1}$. These expressions can be passed to any gradient-based optimizer. See `probitRegDemo` for a demo.

9.4.2 Latent variable interpretation

We can interpret the probit (and logistic) model as follows. First, let us associate each item $\mathbf{x}_i$ with two latent utilities, u_{0i} and u_{1i}, corresponding to the possible choices of $y_i = 0$ and $y_i = 1$. We then assume that the observed choice is whichever action has larger utility. More precisely, the model is as follows:

$$u_{0i} \triangleq \mathbf{w}_0^T\mathbf{x}_i + \delta_{0i} \tag{9.97}$$

$$u_{1i} \triangleq \mathbf{w}_1^T\mathbf{x}_i + \delta_{1i} \tag{9.98}$$

$$y_i = \mathbb{I}(u_{1i} > u_{10}) \tag{9.99}$$

where δ's are error terms, representing all the other factors that might be relevant in decision making that we have chosen not to (or are unable to) model. This is called a **random utility model** or **RUM** (McFadden 1974; Train 2009).

Since it is only the difference in utilities that matters, let us define $z_i = u_{1i} - u_{0i} + \epsilon_i$, where $\epsilon_i = \delta_{1i} - \delta_{0i}$. If the δ's have a Gaussian distribution, then so does ϵ_i. Thus we can write

$$z_i \triangleq \mathbf{w}^T\mathbf{x}_i + \epsilon_i \tag{9.100}$$

$$\epsilon_i \sim \mathcal{N}(0, 1) \tag{9.101}$$

$$y_i = 1 = \mathbb{I}(z_i \geq 0) \tag{9.102}$$

Following (Fruhwirth-Schnatter and Fruhwirth 2010), we call this the difference RUM or **dRUM** model.

When we marginalize out z_i, we recover the probit model:

$$p(y_i = 1|\mathbf{x}_i, \mathbf{w}) = \int \mathbb{I}(z_i \geq 0)\mathcal{N}(z_i|\mathbf{w}^T\mathbf{x}_i, 1)dz_i \tag{9.103}$$

$$= p(\mathbf{w}^T\mathbf{x}_i + \epsilon \geq 0) = p(\epsilon \geq -\mathbf{w}^T\mathbf{x}_i) \tag{9.104}$$

$$= 1 - \Phi(-\mathbf{w}^T\mathbf{x}_i) = \Phi(\mathbf{w}^T\mathbf{x}_i) \tag{9.105}$$

where we used the symmetry of the Gaussian.[3] This latent variable interpretation provides an alternative way to fit the model, as discussed in Section 11.4.6.

Interestingly, if we use a Gumbel distribution for the δ's, we induce a logistic distibution for ϵ_i, and the model reduces to logistic regression. See Section 24.5.1 for further details.

9.4.3 Ordinal probit regression *

One advantage of the latent variable interpretation of probit regression is that it is easy to extend to the case where the response variable is ordinal, that is, it can take on C discrete values which can be ordered in some way, such as low, medium and high. This is called **ordinal regression**. The basic idea is as follows. We introduce $C+1$ thresholds γ_j and set

$$y_i = j \quad \text{if} \quad \gamma_{j-1} < z_i \leq \gamma_j \tag{9.106}$$

where $\gamma_0 \leq \cdots \leq \gamma_C$. For identifiability reasons, we set $\gamma_0 = -\infty$, $\gamma_1 = 0$ and $\gamma_C = \infty$. For example, if $C = 2$, this reduces to the standard binary probit model, whereby $z_i < 0$ produces $y_i = 0$ and $z_i \geq 0$ produces $y_i = 1$. If $C = 3$, we partition the real line into 3 intervals: $(-\infty, 0]$, $(0, \gamma_2]$, (γ_2, ∞). We can vary the parameter γ_2 to ensure the right relative amount of probability mass falls in each interval, so as to match the empirical frequencies of each class label.

Finding the MLEs for this model is a bit trickier than for binary probit regression, since we need to optimize for $\mathbf{w}$ and $\boldsymbol{\gamma}$, and the latter must obey an ordering constraint. See e.g., (Kawakatsu and Largey 2009) for an approach based on EM. It is also possible to derive a simple Gibbs sampling algorithm for this model (see e.g., (Hoff 2009, p216)).

9.4.4 Multinomial probit models *

Now consider the case where the response variable can take on C unordered categorical values, $y_i \in \{1, \ldots, C\}$. The **multinomial probit** model is defined as follows:

$$z_{ic} = \mathbf{w}^T \mathbf{x}_{ic} + \epsilon_{ic} \tag{9.107}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \tag{9.108}$$

$$y_i = \arg\max_c z_{ic} \tag{9.109}$$

See e.g., (Dow and Endersby 2004; Scott 2009; Fruhwirth-Schnatter and Fruhwirth 2010) for more details on the model and its connection to multinomial logistic regression. (By defining $\mathbf{w} = [\mathbf{w}_1, \ldots, \mathbf{w}_C]$, and $\mathbf{x}_{ic} = [\mathbf{0}, \ldots, \mathbf{0}, \mathbf{x}_i, \mathbf{0}, \ldots, \mathbf{0}]$, we can recover the more familiar formulation $z_{ic} = \mathbf{x}_i^T \mathbf{w}_c$.) Since only relative utilities matter, we constrain $\mathbf{R}$ to be a correlation matrix. If instead of setting $y_i = \text{argmax}_c\, z_{ic}$ we use $y_{ic} = \mathbb{I}(z_{ic} > 0)$, we get a model known as **multivariate probit**, which is one way to model C correlated binary outcomes (see e.g., (Talhouk et al. 2011)).

[3] Note that the assumption that the Gaussian noise term is zero mean and unit variance is made without loss of generality. To see why, suppose we used some other mean μ and variance σ^2. Then we could easily rescale $\mathbf{w}$ and add an offset term without changing the likelihood. since $P(\mathcal{N}(0, 1) \geq -\mathbf{w}^T\mathbf{x}) = P(\mathcal{N}(\mu, \sigma^2) \geq -(\mathbf{w}^T\mathbf{x} + \mu)/\sigma)$.

9.5 Multi-task learning

Sometimes we want to fit many related classification or regression models. It is often reasonable to assume the input-output mapping is similar across these different models, so we can get better performance by fitting all the parameters at the same time. In machine learning, this setup is often called **multi-task learning** (Caruana 1998), **transfer learning** (e.g., (Raina et al. 2005)), or **learning to learn** (Thrun and Pratt 1997). In statistics, this is usually tackled using hierarchical Bayesian models (Bakker and Heskes 2003), as we discuss below, although there are other possible methods (see e.g., (Chai 2010)).

9.5.1 Hierarchical Bayes for multi-task learning

Let y_{ij} be the response of the i'th item in group j, for $i = 1 : N_j$ and $j = 1 : J$. For example, j might index schools, i might index students within a school, and y_{ij} might be the test score, as in Section 5.6.2. Or j might index people, and i might index purchaes, and y_{ij} might be the identity of the item that was purchased (this is known as **discrete choice modeling** (Train 2009)). Let $\mathbf{x}_{ij}$ be a feature vector associated with y_{ij}. The goal is to fit the models $p(y_j|\mathbf{x}_j)$ for all j.

Although some groups may have lots of data, there is often a long tail, where the majority of groups have little data. Thus we can't reliably fit each model separately, but we don't want to use the same model for all groups. As a compromise, we can fit a separate model for each group, but encourage the model parameters to be similar across groups. More precisely, suppose $\mathbb{E}[y_{ij}|\mathbf{x}_{ij}] = g(\mathbf{x}_{ij}^T\boldsymbol{\beta}_j)$, where g is the link function for the GLM. Furthermore, suppose $\boldsymbol{\beta}_j \sim \mathcal{N}(\boldsymbol{\beta}_*, \sigma_j^2\mathbf{I})$, and that $\boldsymbol{\beta}_* \sim \mathcal{N}(\boldsymbol{\mu}, \sigma_*^2\mathbf{I})$. In this model, groups with small sample size borrow statistical strength from the groups with larger sample size, because the $\boldsymbol{\beta}_j$'s are correlated via the latent common parents $\boldsymbol{\beta}_*$ (see Section 5.5 for further discussion of this point). The term σ_j^2 controls how much group j depends on the common parents and the σ_*^2 term controls the strength of the overall prior.

Suppose, for simplicity, that $\boldsymbol{\mu} = \mathbf{0}$, and that σ_j^2 and σ_*^2 are all known (e.g., they could be set by cross validation). The overall log probability has the form

$$\log p(\mathcal{D}|\boldsymbol{\beta}) + \log p(\boldsymbol{\beta}) = \sum_j \left[\log p(\mathcal{D}_j|\boldsymbol{\beta}_j) - \frac{||\boldsymbol{\beta}_j - \boldsymbol{\beta}_*||^2}{2\sigma_j^2} \right] - \frac{||\boldsymbol{\beta}_*||^2}{2\sigma_*^2} \tag{9.110}$$

We can perform MAP estimation of $\boldsymbol{\beta} = (\boldsymbol{\beta}_{1:J}, \boldsymbol{\beta}_*)$ using standard gradient methods. Alternatively, we can perform an iterative optimization scheme, alternating between optimizing the $\boldsymbol{\beta}_j$ and the $\boldsymbol{\beta}_*$; since the likelihood and prior are convex, this is guaranteed to converge to the global optimum. Note that once the models are trained, we can discard $\boldsymbol{\beta}_*$, and use each model separately.

9.5.2 Application to personalized email spam filtering

An interesting application of multi-task learning is **personalized spam filtering**. Suppose we want to fit one classifier per user, $\boldsymbol{\beta}_j$. Since most users do not label their email as spam or not, it will be hard to estimate these models independently. So we will let the $\boldsymbol{\beta}_j$ have a common prior $\boldsymbol{\beta}_*$, representing the parameters of a generic user.

In this case, we can emulate the behavior of the above model with a simple trick (Daume 2007b; Attenberg et al. 2009; Weinberger et al. 2009): we make two copies of each feature $\mathbf{x}_i$, one concatenated with the user id, and one not. The effect will be to learn a predictor of the form

$$\mathbb{E}\left[y_i|\mathbf{x}_i, u\right] = (\boldsymbol{\beta}_*, \mathbf{w}_1, \cdots, \mathbf{w}_J)^T[\mathbf{x}_i, \mathbb{I}(u=1)\mathbf{x}_i, \cdots, \mathbb{I}(u=J)\mathbf{x}_i] \tag{9.111}$$

where u is the user id. In other words,

$$\mathbb{E}\left[y_i|\mathbf{x}_i, u=j\right] = (\boldsymbol{\beta}_*^T + \mathbf{w}_j)^T\mathbf{x}_i \tag{9.112}$$

Thus $\boldsymbol{\beta}_*$ will be estimated from everyone's email, whereas $\mathbf{w}_j$ will just be estimated from user j's email.

To see the correspondence with the above hierarchical Bayesian model, define $\mathbf{w}_j = \boldsymbol{\beta}_j - \boldsymbol{\beta}_*$. Then the log probability of the original model can be rewritten as

$$\sum_j \left[\log p(\mathcal{D}_j|\boldsymbol{\beta}_* + \mathbf{w}_j) - \frac{||\mathbf{w}_j||^2}{2\sigma_j^2}\right] - \frac{||\boldsymbol{\beta}_*||^2}{2\sigma_*^2} \tag{9.113}$$

If we assume $\sigma_j^2 = \sigma_*^2$, the effect is the same as using the augmented feature trick, with the same regularizer strength for both $\mathbf{w}_j$ and $\boldsymbol{\beta}_*$. However, one typically gets better performance by not requiring that σ_j^2 be equal to σ_*^2 (Finkel and Manning 2009).

9.5.3 Application to domain adaptation

Domain adaptation is the problem of training a set of classifiers on data drawn from different distributions, such as email and newswire text. This problem is obviously a special case of multi-task learning, where the tasks are the same.

(Finkel and Manning 2009) used the above hierarchical Bayesian model to perform domain adaptation for two NLP tasks, namely named entity recognition and parsing. They report reasonably large improvements over fitting separate models to each dataset, and small improvements over the approach of pooling all the data and fitting a single model.

9.5.4 Other kinds of prior

In multi-task learning, it is common to assume that the prior is Gaussian. However, sometimes other priors are more suitable. For example, consider the task of **conjoint analysis**, which requires figuring out which features of a product customers like best. This can be modelled using the same hierarchical Bayesian setup as above, but where we use a sparsity-promoting prior on $\boldsymbol{\beta}_j$, rather than a Gaussian prior. This is called **multi-task feature selection**. See e.g., (Lenk et al. 1996; Argyriou et al. 2008) for some possible approaches.

It is not always reasonable to assume that all tasks are all equally similar. If we pool the parameters across tasks that are qualitatively different, the performance will be worse than not using pooling, because the inductive bias of our prior is wrong. Indeed, it has been found experimentally that sometimes multi-task learning does worse than solving each task separately (this is called **negative transfer**).

One way around this problem is to use a more flexible prior, such as a mixture of Gaussians. Such flexible priors can provide robustness against prior mis-specification. See e.g., (Xue et al. 2007; Jacob et al. 2008) for details. One can of course combine mixtures with sparsity-promoting priors (Ji et al. 2009). Many other variants are possible.

9.6 Generalized linear mixed models *

Suppose we generalize the multi-task learning scenario to allow the response to include information at the group level, $\mathbf{x}_j$, as well as at the item level, $\mathbf{x}_{ij}$. Similarly, we can allow the parameters to vary across groups, $\boldsymbol{\beta}_j$, or to be tied across groups, $\boldsymbol{\alpha}$. This gives rise to the following model:

$$\mathbb{E}\left[y_{ij}|\mathbf{x}_{ij}, \mathbf{x}_j\right] = g\left(\boldsymbol{\phi}_1(\mathbf{x}_{ij})^T\boldsymbol{\beta}_j + \boldsymbol{\phi}_2(\mathbf{x}_j)^T\boldsymbol{\beta}'_j + \boldsymbol{\phi}_3(\mathbf{x}_{ij})^T\boldsymbol{\alpha} + \boldsymbol{\phi}_4(\mathbf{x}_j)^T\boldsymbol{\alpha}'\right) \tag{9.114}$$

where the $\boldsymbol{\phi}_k$ are basis functions. This model can be represented pictorially as shown in Figure 9.2(a). (Such figures will be explained in Chapter 10.) Note that the number of $\boldsymbol{\beta}_j$ parameters grows with the number of groups, whereas the size of $\boldsymbol{\alpha}$ is fixed.

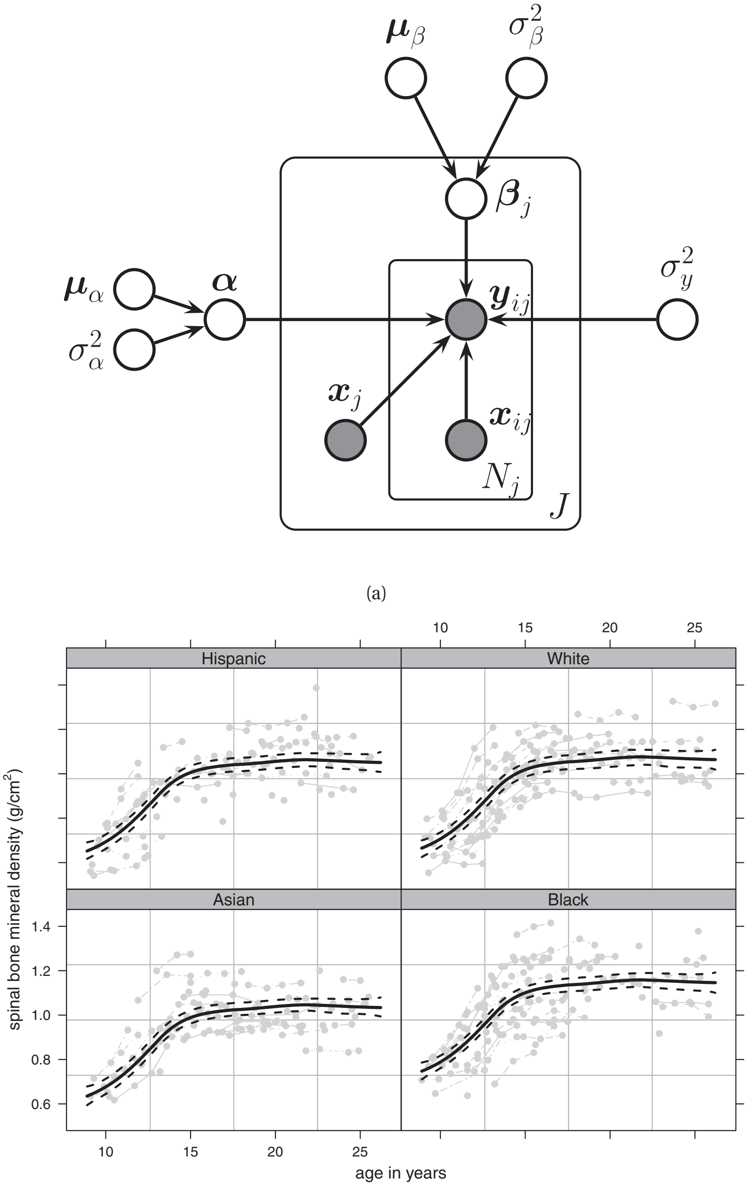
Frequentists call the terms $\boldsymbol{\beta}_j$ **random effects**, since they vary randomly across groups, but they call $\boldsymbol{\alpha}$ a **fixed effect**, since it is viewed as a fixed but unknown constant. A model with both fixed and random effects is called a **mixed model**. If $p(y|\mathbf{x})$ is a GLM, the overall model is called a **generalized linear mixed effects model** or **GLMM**. Such models are widely used in statistics.

9.6.1 Example: semi-parametric GLMMs for medical data

Consider the following example from (Wand 2009). Suppose y_{ij} is the amount of spinal bone mineral density (SBMD) for person j at measurement i. Let x_{ij} be the age of person, and let x_j be their ethnicity, which can be one of: White, Asian, Black, or Hispanic. The primary goal is to determine if there are significant differences in the mean SBMD among the four ethnic groups, after accounting for age. The data is shown in the light gray lines in Figure 9.2(b). We see that there is a nonlinear effect of SBMD vs age, so we will use a **semi-parametric model** which combines linear regression with non-parametric regression (Ruppert et al. 2003). We also see that there is variation across individuals within each group, so we will use a mixed effects model. Specifically, we will use $\boldsymbol{\phi}_1(\mathbf{x}_{ij}) = 1$ to account for the random effect of each person; $\boldsymbol{\phi}_2(x_{ij}) = 0$ since no other coefficients are person-specific; $\boldsymbol{\phi}_3(x_{ij}) = [b_k(x_{ij})]$, where b_k is the k'th spline basis functions (see Section 15.4.6.2), to account for the nonlinear effect of age; and $\boldsymbol{\phi}_4(x_j) = [\mathbb{I}(x_j = w), \mathbb{I}(x_j = a), \mathbb{I}(x_j = b), \mathbb{I}(x_j = h)]$ to account for the effect of the different ethnicities. Furthermore, we use a linear link function. The overall model is therefore

$$\begin{aligned}\mathbb{E}\left[y_{ij}|x_{ij}, x_j\right] &= \beta_j + \boldsymbol{\alpha}^T\mathbf{b}(x_{ij}) + \epsilon_{ij} & (9.115)\\ &\quad +\alpha'_w\mathbb{I}(x_j = w) + \alpha'_a\mathbb{I}(x_j = a) + \alpha'_b\mathbb{I}(x_j = b) + \alpha'_h\mathbb{I}(x_j = h) & (9.116)\end{aligned}$$

where $\epsilon_{ij} \sim \mathcal{N}(0, \sigma_y^2)$. $\boldsymbol{\alpha}$ contains the non-parametric part of the model related to age, $\boldsymbol{\alpha}'$ contains the parametric part of the model related to ethnicity, and β_j is a random offset for person j. We endow all of these regression coefficients with separate Gaussian priors. We can then perform posterior inference to compute $p(\boldsymbol{\alpha}, \boldsymbol{\alpha}', \boldsymbol{\beta}, \boldsymbol{\sigma}^2|\mathcal{D})$ (see Section 9.6.2 for



(a)

(b)

Figure 9.2 (a) Directed graphical model for generalized linear mixed effects model with J groups. (b) Spinal bone mineral density vs age for four different ethnic groups. Raw data is shown in the light gray lines. Fitted model shown in black (solid is the posterior predicted mean, dotted is the posterior predictive variance). From Figure 9 of (Wand 2009). Used with kind permission of Matt Wand computational details). After fitting the model, we can compute the prediction for each group. See Figure 9.2(b) for the results. We can also perform significance testing, by computing $p(\alpha_g - \alpha_w|\mathcal{D})$ for each ethnic group g relative to some baseline (say, White), as we did in Section 5.2.3.

9.6.2 Computational issues

The principle problem with GLMMs is that they can be difficult to fit, for two reasons. First, $p(y_{ij}|\boldsymbol{\theta})$ may not be conjugate to the prior $p(\boldsymbol{\theta})$ where $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\beta})$. Second, there are two levels of unknowns in the model, namely the regression coefficients $\boldsymbol{\theta}$ and the means and variances of the priors $\boldsymbol{\eta} = (\boldsymbol{\mu}, \boldsymbol{\sigma})$.

One approach is to adopt fully Bayesian inference methods, such as variational Bayes (Hall et al. 2011) or MCMC (Gelman and Hill 2007). We discuss VB in Section 21.5, and MCMC in Section 24.1.

An alternative approach is to use empirical Bayes, which we discuss in general terms in Section 5.6. In the context of a GLMM, we can use the EM algorithm (Section 11.4), where in the E step we compute $p(\boldsymbol{\theta}|\boldsymbol{\eta}, \mathcal{D})$, and in the M step we optimize $\boldsymbol{\eta}$. If the linear regression setting, the E step can be performed exactly, but in general we need to use approximations. Traditional methods use numerical quadrature or Monte Carlo (see e.g., (Breslow and Clayton 1993)). A faster approach is to use variational EM; see (Braun and McAuliffe 2010) for an application of variational EM to a multi-level discrete choice modeling problem.

In frequentist statistics, there is a popular method for fitting GLMMs called **generalized estimating equations** or **GEE** (Hardin and Hilbe 2003). However, we do not recommend this approach, since it is not as statistically efficient as likelihood-based methods (see Section 6.4.3). In addition, it can only provide estimates of the population parameters $\boldsymbol{\alpha}$, but not the random effects $\boldsymbol{\beta}_j$, which are sometimes of interest in themselves.

9.7 Learning to rank *

In this section, we discuss the **learning to rank** or **LETOR** problem. That is, we want to learn a function that can rank order a set of items (we will be more precise below). The most common application is to information retrieval. Specifically, suppose we have a query q and a set of documents $d^1, \ldots, d^m$ that might be relevant to q (e.g., all documents that contain the string q). We would like to sort these documents in decreasing order of relevance and show the top k to the user. Similar problems arise in other areas, such as collaborative filtering. (Ranking players in a game or tournament setting is a slightly different kind of problem; see Section 22.5.5.)

Below we summarize some methods for solving this problem, following the presentation of (Liu 2009). This material is not based on GLMs, but we include it in this chapter anyway for lack of a better place.

A standard way to measure the relevance of a document d to a query q is to use a probabilistic language model based on a bag of words model. That is, we define $sim(q, d) \triangleq p(q|d) = \prod_{i=1}^{n} p(q_i|d)$, where q_i is the i'th word or term, and $p(q_i|d)$ is a multinoulli distribution estimated from document d. In practice, we need to smooth the estimated distribution, for example by using a Dirichlet prior, representing the overall frequency of each word. This can be estimated from all documents in the system. More precisely, we can use

$$p(t|d) = (1-\lambda)\frac{\text{TF}(t,d)}{\text{LEN}(d)} + \lambda p(t|\text{background}) \tag{9.117}$$

where TF(t,d) is the frequency of term t in document d, LEN(d) is the number of words in d, and $0 < \lambda < 1$ is a smoothing parameter (see e.g., Zhai and Lafferty (2004) for details).

However, there might be many other signals that we can use to measure relevance. For example, the PageRank of a web document is a measure of its authoritativeness, derived from the web's link structure (see Section 17.2.4 for details). We can also compute how often and where the query occurs in the document. Below we discuss how to learn how to combine all these signals.[4]

9.7.1 The pointwise approach

Suppose we collect some training data representing the relevance of a set of documents for each query. Specifically, for each query q, suppose that we retrieve m possibly relevant documents d_j, for $j = 1 : m$. For each query document pair, we define a feature vector, $\mathbf{x}(q, d)$. For example, this might contain the query-document similarity score and the page rank score of the document. Furthermore, suppose we have a set of labels y_j representing the degree of relevance of document d_j to query q. Such labels might be binary (e.g., relevant or irrelevant), or they may represent a degree of relevance (e.g., very relevant, somewhat relevant, irrelevant). Such labels can be obtained from query logs, by thresholding the number of times a document was clicked on for a given query.

If we have binary relevance labels, we can solve the problem using a standard binary classification scheme to estimate, $p(y = 1|\mathbf{x}(q,d))$. If we have ordered relevancy labels, we can use ordinal regression to predict the rating, $p(y = r|\mathbf{x}(q,d))$. In either case, we can then sort the documents by this scoring metric. This is called the **pointwise approach** to LETOR, and is widely used because of its simplicity. However, this method does not take into account the location of each document in the list. Thus it penalizes errors at the end of the list just as much as errors at the beginning, which is often not the desired behavior. In addition, each decision about relevance is made very myopically.

9.7.2 The pairwise approach

There is evidence (e.g., (Carterette et al. 2008)) that people are better at judging the relative relevance of two items rather than absolute relevance. Consequently, the data might tell us that d_j is more relevant than d_k for a given query, or vice versa. We can model this kind of data using a binary classifier of the form $p(y_{jk}|\mathbf{x}(q,d_j), \mathbf{x}(q,d_k))$, where we set $y_{jk} = 1$ if rel$(d_j, q) >$ rel(d_k, q) and $y_{jk} = 0$ otherwise.

One way to model such a function is as follows:

$$p(y_{jk} = 1|\mathbf{x}_j, \mathbf{x}_k) = \text{sigm}(f(\mathbf{x}_j) - f(\mathbf{x}_k)) \tag{9.118}$$

[4] Rather surprisingly, Google does not (or at least, did not as of 2008) using such learning methods in its search engine. Source: Peter Norvig, quoted in `http://anand.typepad.com/datawocky/2008/05/are-human-experts-less-prone-to-catastrophic-errors-than-machine-learned-models.html`.

where $f(\mathbf{x})$ is a scoring function, often taken to be linear, $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$. This is a special kind of neural network known as **RankNet** (Burges et al. 2005) (see Section 16.5 for a general discussion of neural networks). We can find the MLE of $\mathbf{w}$ by maximizing the log likelihood, or equivalently, by minimizing the cross entropy loss, given by

$$\begin{aligned} L &= \sum_{i=1}^{N}\sum_{j=1}^{m_i}\sum_{k=j+1}^{m_i} L_{ijk} && (9.119) \\ -L_{ijk} &= \mathbb{I}(y_{ijk}=1)\log p(y_{ijk}=1|\mathbf{x}_{ij},\mathbf{x}_{ik},\mathbf{w}) \\ &\quad +\mathbb{I}(y_{ijk}=0)\log p(y_{ijk}=0|\mathbf{x}_{ij},\mathbf{x}_{ik},\mathbf{w}) && (9.120) \end{aligned}$$

This can be optimized using gradient descent. A variant of RankNet is used by Microsoft's Bing search engine.[5]

9.7.3 The listwise approach

The pairwise approach suffers from the problem that decisions about relevance are made just based on a pair of items (documents), rather than considering the full context. We now consider methods that look at the entire list of items at the same time.

We can define a total order on a list by specifying a permutation of its indices, $\boldsymbol{\pi}$. To model our uncertainty about $\boldsymbol{\pi}$, we can use the **Plackett-Luce** distribution, which derives its name from independent work by (Plackett 1975) and (Luce 1959). This has the following form:

$$p(\boldsymbol{\pi}|\mathbf{s}) = \prod_{j=1}^{m} \frac{s_j}{\sum_{u=j}^{m} s_u} \tag{9.121}$$

where $s_j = s(\pi^{-1}(j))$ is the score of the document ranked at the j'th position.

To understand Equation 9.121, let us consider a simple example. Suppose $\boldsymbol{\pi} = (A, B, C)$. Then we have that $p(\boldsymbol{\pi})$ is the probability of A being ranked first, times the probability of B being ranked second given that A is ranked first, times the probabilty of C being ranked third given that A and B are ranked first and second. In other words,

$$p(\boldsymbol{\pi}|\mathbf{s}) = \frac{s_A}{s_A+s_B+s_C} \times \frac{s_B}{s_B+s_C} \times \frac{s_C}{s_C} \tag{9.122}$$

To incorporate features, we can define $s(d) = f(\mathbf{x}(q,d))$, where we often take f to be a linear function, $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$. This is known as the **ListNet** model (Cao et al. 2007). To train this model, let $\mathbf{y}_i$ be the relevance scores of the documents for query i. We then minimize the cross entropy term

$$-\sum_i \sum_{\boldsymbol{\pi}} p(\boldsymbol{\pi}|\mathbf{y}_i)\log p(\boldsymbol{\pi}|\mathbf{s}_i) \tag{9.123}$$

Of course, as stated, this is intractable, since the i'th term needs to sum over $m_i!$ permutations. To make this tractable, we can consider permutations over the top k positions only:

$$p(\boldsymbol{\pi}_{1:k}|\mathbf{s}_{1:m}) = \prod_{j=1}^{k} \frac{s_j}{\sum_{u=1}^{m} s_u} \tag{9.124}$$

[5] Source: `http://www.bing.com/community/site_blogs/b/search/archive/2009/06/01/user-needs-features-and-the-science-behind-bing.aspx`.

There are only $m!/(m-k)!$ such permutations. If we set $k=1$, we can evaluate each cross entropy term (and its derivative) in $O(m)$ time.

In the special case where only one document from the presented list is deemed relevant, say $y_i = c$, we can instead use multinomial logistic regression:

$$p(y_i = c|\mathbf{x}) = \frac{\exp(s_c)}{\sum_{c'=1}^{m} \exp(s_{c'})} \tag{9.125}$$

This often performs at least as well as ranking methods, at least in the context of collaborative filtering (Yang et al. 2011).

9.7.4 Loss functions for ranking

There are a variety of ways to measure the performance of a ranking system, which we summarize below.

- **Mean reciprocal rank (MRR)**. For a query q, let the rank position of its first relevant document be denoted by $r(q)$. Then we define the **mean reciprocal rank** to be $1/r(q)$. This is a very simple performance measure.
- **Mean average precision (MAP)**. In the case of binary relevance labels, we can define the **precision at k** of some ordering as follows:

  $$\text{P@k}(\boldsymbol{\pi}) \triangleq \frac{\text{num. relevant documents in the top } k \text{ positions of } \boldsymbol{\pi}}{k} \tag{9.126}$$

  We then define the average precision as follows:

  $$\text{AP}(\boldsymbol{\pi}) \triangleq \frac{\sum_k \text{P@k}(\boldsymbol{\pi}) \cdot I_k}{\text{num. relevant documents}} \tag{9.127}$$

  where I_k is 1 iff document k is relevant. For example, if we have the relevancy labels $\mathbf{y} = (1,0,1,0,1)$, then the AP is $\frac{1}{3}(\frac{1}{1} + \frac{2}{3} + \frac{3}{5}) \approx 0.76$. Finally, we define the **mean average precision** as the AP averaged over all queries.
- **Normalized discounted cumulative gain (NDCG)**. Suppose the relevance labels have multiple levels. We can define the **discounted cumulative gain** of the first k items in an ordering as follows:

  $$\text{DCG@k}(\mathbf{r}) = r_1 + \sum_{i=2}^{k} \frac{r_i}{\log_2 i} \tag{9.128}$$

  where r_i is the relevance of item i and the $\log_2$ term is used to discount items later in the list. Table 9.3 gives a simple numerical example. An alternative definition, that places stronger emphasis on retrieving relevant documents, uses

  $$\text{DCG@k}(\mathbf{r}) = \sum_{i=1}^{k} \frac{2^{r_i} - 1}{\log_2(1+i)} \tag{9.129}$$

  The trouble with DCG is that it varies in magnitude just because the length of a returned list may vary. It is therefore common to normalize this measure by the ideal DCG, which is

i	1	2	3	4	5	6
r_i	3	2	3	0	1	2
$\log_2 i$	0	1	1.59	2.0	2.32	2.59
$\frac{r_i}{\log_2 i}$	N/A	2	1.887	0	0.431	0.772

Table 9.3 Illustration of how to compute NDCG, from `http://en.wikipedia.org/wiki/Discounted_cumulative_gain`. The value r_i is the relevance score of the item in position i. From this, we see that DCG@6 $= 3 + (2 + 1.887 + 0 + 0.431 + 0.772) = 8.09$. The maximum DCG is obtained using the ordering with scores 3, 3, 2, 2, 1, 0. Hence the ideal DCG is 8.693, and so the normalized DCG is 8.09 / 8.693 = 0.9306.

the DCG obtained by using the optimal ordering: IDCG@k$(\mathbf{r}) = \text{argmax}_{\boldsymbol{\pi}}$ DCG@k$(\mathbf{r})$. This can be easily computed by sorting $r_{1:m}$ and then computing DCG@k. Finally, we define the **normalized discounted cumulative gain** or **NDCG** as DCG/IDCG. Table 9.3 gives a simple numerical example. The NDCG can be averaged over queries to give a measure of performance.

- **Rank correlation**. We can measure the correlation between the ranked list, $\boldsymbol{\pi}$, and the relevance judgement, $\boldsymbol{\pi}^*$, using a variety of methods. One approach, known as the (weighted) **Kendall's** τ statistics, is defined in terms of the weighted pairwise inconsistency between the two lists:

$$\tau(\boldsymbol{\pi}, \boldsymbol{\pi}^*) = \frac{\sum_{u<v} w_{uv} \left[1 + \text{sgn}(\pi_u - \pi_v)\text{sgn}(\pi_u^* - \pi_v^*)\right]}{2 \sum_{u<v} w_{uv}} \tag{9.130}$$

A variety of other measures are commonly used.

These loss functions can be used in different ways. In the Bayesian approach, we first fit the model using posterior inference; this depends on the likelihood and prior, but not the loss. We then choose our actions at test time to minimize the expected future loss. One way to do this is to sample parameters from the posterior, $\boldsymbol{\theta}^s \sim p(\boldsymbol{\theta}|\mathcal{D})$, and then evaluate, say, the precision@k for different thresholds, averaging over $\boldsymbol{\theta}^s$. See (Zhang et al. 2010) for an example of such an approach.

In the frequentist approach, we try to minimize the empirical loss on the training set. The problem is that these loss functions are not differentiable functions of the model parameters. We can either use gradient-free optimization methods, or we can minimize a surrogate loss function instead. Cross entropy loss (i.e., negative log likelihood) is an example of a widely used surrogate loss function.

Another loss, known as **weighted approximate-rank pairwise** or **WARP** loss, proposed in (Usunier et al. 2009) and extended in (Weston et al. 2010), provides a better approximation to the precision@k loss. WARP is defined as follows:

$$\text{WARP}(\mathbf{f}(\mathbf{x}, :), y) \triangleq L(\text{rank}(\mathbf{f}(\mathbf{x}, :), y)) \tag{9.131}$$

$$\text{rank}(\mathbf{f}(\mathbf{x}, :), y) = \sum_{y' \neq y} \mathbb{I}(f(\mathbf{x}, y') \geq f(\mathbf{x}, y)) \tag{9.132}$$

$$L(k) \triangleq \sum_{j=1}^{k} \alpha_j, \quad \text{with } \alpha_1 \geq \alpha_2 \geq \cdots \geq 0 \tag{9.133}$$

Here $\mathbf{f}(x, :) = [f(\mathbf{x}, 1), \ldots, f(\mathbf{x}, |y|)]$ is the vector of scores for each possible output label, or, in IR terms, for each possible document corresponding to input query $\mathbf{x}$. The expression rank$(\mathbf{f}(\mathbf{x}, :), y)$ measures the rank of the true label y assigned by this scoring function. Finally, L transforms the integer rank into a real-valued penalty. Using $\alpha_1 = 1$ and $\alpha_{j>1} = 0$ would optimize the proportion of top-ranked correct labels. Setting $\alpha_{1:k}$ to be non-zero values would optimize the top k in the ranked list, which will induce good performance as measured by MAP or precision@k. As it stands, WARP loss is still hard to optimize, but it can be further approximated by Monte Carlo sampling, and then optimized by gradient descent, as described in (Weston et al. 2010).

Exercises

Exercise 9.1 Conjugate prior for univariate Gaussian in exponential family form

Derive the conjugate prior for μ and $\lambda = 1/\sigma^2$ for a univariate Gaussian using the exponential family, by analogy to Section 9.2.5.5. By suitable reparameterization, show that the prior has the form $p(\mu, \lambda) = \mathcal{N}(\mu|\gamma, \lambda(2\alpha - 1))\mathrm{Ga}(\lambda|\alpha, \beta)$, and thus only has 3 free parameters.

Exercise 9.2 The MVN is in the exponential family

Show that we can write the MVN in exponential family form. Hint: use the information form defined in Section 4.3.3.