0.1 Generative classifier

$$p(y=c|\vec{x},\vec{\theta}) = \frac{p(y=c|\vec{\theta})p(\vec{x}|y=c,\vec{\theta})}{\sum_{c'} p(y=c'|\vec{\theta})p(\vec{x}|y=c',\vec{\theta})} \tag{0.1}$$

এই অংশটি বিভিন্ন classifier এবং machine learning এ prior knowledge এর ভূমিকা নিয়ে আলোচনা করে:

- Generative classifier: এটি এমন একটি classifier যা feature গুলির distribution কে model করে যখন class label দেওয়া হয়; এখানে "generative" term ব্যবহার করা হচ্ছে কারণ মডেল এখানে x feature generate করছে প্রত্যেকটা ক্লাস c এর জন্যে।

- $p(y=c|)$ class label এর উপরে prior বোঝায়, এবং $p(x|y=c,)$ টার্ম class c-এর জন্য class conditional density বলা হয়, উপরের ইকুয়েশন অনুযায়ী ।

- Discriminative classifier: Data generation process এর model করার পরিবর্তে, এটি সরাসরি class এর posterior $p(y=c|\vec{x})$ মডেল তৈরি করছে, অর্থাৎ, data দেওয়া হলে class label এর সম্ভাব্যতা model করে।

0.2 Bayesian concept learning

Concept learning এর ধারণাটি বা concept-টি ব্যাখ্যা করা যায়, শব্দের অর্থ শেখার সাথে; এটাকে binary classification এর সাথেও তুলনা করা যায়। উদাহরণস্বরূপ, যদি একটি feature $\vec{x}$ কোন concept C এর উদাহরণ হয়, তাহলে আমরা $f(\vec{x}) = 1$ নির্ধারণ করি; অন্যথায়, আমরা $f(\vec{x}) = 0$ নির্ধারণ করি। লক্ষ্য হল function f শেখা, যা একটি indicator হিসাবে কাজ করে এবং নির্ধারণ করে যে কোন feature concept সেট C তে অন্তর্ভুক্ত।

0.2.1 Likelihood

This crucial equation embodies what Tenenbaum calls the size principle, which means the model favours the simplest (smallest) hypothesis consistent with the data. This is more commonly known as Occam's razor[1].

$$p(\mathcal{D}|h) \triangleq \left(\frac{1}{\text{size}(h)}\right)^N = \left(\frac{1}{|h|}\right)^N \tag{0.2}$$

- □ এই ইকুয়েশনে দেখানো হচ্ছে h হাইপোথিসিসের ওপর ভিত্তি করে, $\mathcal{D}$ ডেটা সেটের সম্ভাবনা।
- □ size(h) বা $|h|$: বোঝায় হাইপোথিসিসের সাইজ।
- □ N: হলো ডেটার সংখ্যা।
- □ ইকুয়েশনে $\left(\frac{1}{|h|}\right)^N$ দ্বারা বোঝানো হচ্ছে h-এর আকার যত বাড়বে, সম্ভাবনা এর মান ততো কমবে।

This crucial equation embodies what Tenenbaum calls the size principle, which means the model favours the simplest (smallest) hypothesis consistent with the data. This is more commonly known as Occam's razor[2].

0.2.2 Prior

Prior হল background knowledge বা data পর্যবেক্ষণের আগে করা অনুমান। এটি subjective, অর্থাৎ এটি ব্যক্তিভেদে পরিবর্তিত হতে পারে। উদাহরণস্বরূপ, একটি শিশু এবং একজন গণিত অধ্যাপকের মধ্যে ভিন্ন priors থাকবে তাদের জ্ঞানের পার্থক্যের কারণে।

The prior is decided by human, not machines, so it is subjective. The subjectivity of the prior is controversial. For example, that a child and a math professor will reach different answers. In fact, they presumably not only have different priors, but also different hypothesis spaces. However, we can finesse that by defining the hypothesis space of the child and the math professor to be the same, and then setting the child's prior weight to be zero on certain "advanced" concepts. Thus there is no sharp distinction between the prior and the hypothesis space.

However, the prior is the mechanism by which background knowledge can be brought to bear on a problem. Without this, rapid learning (i.e., from small samples sizes) is impossible.

0.2.3 Posterior

The posterior is simply the likelihood times the prior, normalized.

$$p(h|\mathcal{D}) \triangleq \frac{p(\mathcal{D}|h)p(h)}{\sum_{h'\in\mathcal{H}} p(\mathcal{D}|h')p(h')} = \frac{\mathbb{I}(\mathcal{D}\in h)p(h)}{\sum_{h'\in\mathcal{H}} \mathbb{I}(\mathcal{D}\in h')p(h')} \tag{0.3}$$

- □ এই ইকুয়েশনে Bayes' theorem ব্যবহার করে h-এর posterior probability বের করা হয়েছে, যখন $\mathcal{D}$ ডেটা দেয়া থাকেবে আগে থেকে।
- □ $p(h|\mathcal{D})$: হলো posterior probability।
- □ $p(\mathcal{D}|h)$: হলো likelihood।
- □ $p(h)$: হলো prior probability।
- □ denominator: সব possible hypothesis-এর probability এর যোগফল $\sum_{h'\in\mathcal{H}} p(\mathcal{D}|h')p(h')$, যেটি সম্পূর্ণ প্রবাবিলিটিকে নরমালাইজ করতে ব্যবহার করা হচ্ছে

এখানে $\mathbb{I}(\mathcal{D}\in h)p(h)$ হল 1 iff (if and only if) শুধুমাত্র তখনই যখন সমস্ত ডেটা hypothesis h এর extension এর মধ্যে থাকে।

[1] http://en.wikipedia.org/wiki/Occam%27s_razor

[2] http://en.wikipedia.org/wiki/Occam%27s_razor

সাধারণভাবে, যখন আমাদের কাছে পর্যাপ্ত ডেটা থাকে, posterior $p(h|\mathcal{D})$ একটি নির্দিষ্ট concept এর উপর বেশি কেন্দ্রীভূত (peaked) হয়ে থাকে, যেটাকে MAP estimate বলে; অর্থাৎ-

$$p(h|\mathcal{D}) \to \hat{h}^{MAP} \tag{0.4}$$

where $\hat{h}^{MAP}$ is the posterior mode,

□ এই ইকুয়েশনটি বোঝায় যে, যেই h-এর posterior probability সবচেয়ে বেশি, সেটিই হবে $\hat{h}^{MAP}$, অর্থাৎ Maximum A Posteriori (MAP) estimate।

$$\begin{aligned}\hat{h}^{MAP} &\triangleq \arg\max_h p(h|\mathcal{D}) = \arg\max_h p(\mathcal{D}|h)p(h) \\ &= \arg\max_h [\log p(\mathcal{D}|h) + \log p(h)]\end{aligned} \tag{0.5}$$

□ এই ইকুয়েশন দেখাচ্ছে কিভাবে MAP estimate বের করা হয়। এটা $p(h|\mathcal{D})$ বা posterior probability এর সর্বোচ্চ মান বের করে

□ $p(\mathcal{D}|h)p(h)$ likelihood এবং prior এর গুণফল maximize করা হচ্ছে MAP estimate পাওয়ার জন্যে।

□ optimization সহজ করার জন্য product sum এর লগারিদম করা হচ্ছে ।

যেহেতু likelihood term N এর উপর exponent আকারে নির্ভরশীল, এবং prior একই(constant) থাকে, বেশি বেশি ডাটা পাওয়ার সাথে সাথে MAP estimate ধীরে ধীরে maximum likelihood estimate বা MLE এর দিকে মিলিত (converge) হয়-

$$\hat{h}^{MLE} \triangleq \arg\max_h p(\mathcal{D}|h) = \arg\max_h \log p(\mathcal{D}|h) \tag{0.6}$$

□ এখানে $\hat{h}^{MLE}$ হলো Maximum Likelihood Estimate (MLE), যা শুধু $p(\mathcal{D}|h)$-এর ওপর ভিত্তি করে বের করা হয়, $p(h)$ বা prior এখানে ধরা হয় না।

অন্য কথায়, যদি আমাদের কাছে যথেষ্ট ডেটা থাকে, আমরা দেখতে পাই যে ডেটা prior কে অতিক্রম করে দেয়।

0.2.4 Posterior predictive distribution

The concept of posterior predictive distribution[3] is normally used in a Bayesian context, where it makes use of the entire posterior distribution of the parameters given the observed data to yield a probability distribution over an interval rather than simply a point estimate.

$$p(\tilde{\vec{x}}|\mathcal{D}) \triangleq \mathbb{E}_{h|\mathcal{D}}[p(\tilde{\vec{x}}|h)] = \begin{cases} \sum_h p(\tilde{\vec{x}}|h)p(h|\mathcal{D}) \\ \int p(\tilde{\vec{x}}|h)p(h|\mathcal{D})\mathrm{d}h \end{cases} \tag{0.7}$$

This is just a weighted average of the predictions of each individual hypothesis and is called Bayes model averaging(Hoeting et al. 1999).

0.3 The beta-binomial model

0.3.1 Likelihood

Given $X \sim \text{Bin}(\theta)$, the likelihood of $\mathcal{D}$ is given by

$$p(\mathcal{D}|\theta) = \text{Bin}(N_1|N,\theta) \tag{0.8}$$

0.3.2 Prior

$$\text{Beta}(\theta|a,b) \propto \theta^{a-1}(1-\theta)^{b-1} \tag{0.9}$$

The parameters of the prior are called hyper-parameters.

0.3.3 Posterior

$$\begin{aligned}p(\theta|\mathcal{D}) &\propto \text{Bin}(N_1|N_1+N_0,\theta)\text{Beta}(\theta|a,b) \\ &= \text{Beta}(\theta|N_1+a,N_0b)\end{aligned} \tag{0.10}$$

Note that updating the posterior sequentially is equivalent to updating in a single batch. To see this, suppose we have two data sets $\mathcal{D}_a$ and $\mathcal{D}_b$ with sufficient statistics N_1^a, N_0^a and N_1^b, N_0^b. Let $N_1 = N_1^a + N_1^b$ and $N_0 = N_0^a + N_0^b$ be the sufficient statistics of the combined datasets. In batch mode we have

$$\begin{aligned}p(\theta|\mathcal{D}_a,\mathcal{D}_b) &= p(\theta,\mathcal{D}_b|\mathcal{D}_a)p(\mathcal{D}_a) \\ &\propto p(\theta,\mathcal{D}_b|\mathcal{D}_a) \\ &= p(\mathcal{D}_b,\theta|\mathcal{D}_a) \\ &= p(\mathcal{D}_b|\theta)p(\theta|\mathcal{D}_a) \\ &\text{Combine Equation ?? and ??} \\ &= \text{Bin}(N_1^b|\theta,N_1^b+N_0^b)\text{Beta}(\theta|N_1^a+a,N_0^a+b) \\ &= \text{Beta}(\theta|N_1^a+N_1^b+a,N_0^a+N_0^b+b)\end{aligned}$$

This makes Bayesian inference particularly well-suited to online learning, as we will see later.

0.3.3.1 Posterior mean and mode

From Table ??, the posterior mean is given by

[3] http://en.wikipedia.org/wiki/Posterior_predictive_distribution

$$\bar{\theta} = \frac{a+N_1}{a+b+N} \tag{0.11}$$

The mode is given by

$$\hat{\theta}_{MAP} = \frac{a+N_1-1}{a+b+N-2} \tag{0.12}$$

If we use a uniform prior, then the MAP estimate reduces to the MLE,

$$\hat{\theta}_{MLE} = \frac{N_1}{N} \tag{0.13}$$

We will now show that the posterior mean is convex combination of the prior mean and the MLE, which captures the notion that the posterior is a compromise between what we previously believed and what the data is telling us.

0.3.3.2 Posterior variance

The mean and mode are point estimates, but it is useful to know how much we can trust them. The variance of the posterior is one way to measure this. The variance of the Beta posterior is given by

$$\mathrm{var}(\theta|\mathcal{D}) = \frac{(a+N_1)(b+N_0)}{(a+N_1+b+N_0)^2(a+N_1+b+N_0+1)} \tag{0.14}$$

We can simplify this formidable expression in the case that $N \gg a,b$, to get

$$\mathrm{var}(\theta|\mathcal{D}) \approx \frac{N_1 N_0}{NNN} = \frac{\hat{\theta}_{MLE}(1-\hat{\theta}_{MLE})}{N} \tag{0.15}$$

0.3.4 Posterior predictive distribution

So far, we have been focusing on inference of the unknown parameter(s). Let us now turn our attention to prediction of future observable data.

Consider predicting the probability of heads in a single future trial under a $\mathrm{Beta}(a,b)$posterior. We have

$$\begin{aligned} p(\tilde{x}|\mathcal{D}) &= \int_0^1 p(\tilde{x}|\theta)p(\theta|\mathcal{D})\mathrm{d}\theta \\ &= \int_0^1 \theta \mathrm{Beta}(\theta|a,b)\mathrm{d}\theta \\ &= \mathbb{E}[\theta|\mathcal{D}] = \frac{a}{a+b} \end{aligned} \tag{0.16}$$

0.3.4.1 Overfitting and the black swan paradox

Let us now derive a simple Bayesian solution to the problem. We will use a uniform prior, so $a = b = 1$. In this case, plugging in the posterior mean gives Laplace's rule of succession

$$p(\tilde{x}|\mathcal{D}) = \frac{N_1+1}{N_0+N_1+1} \tag{0.17}$$

This justifies the common practice of adding 1 to the empirical counts, normalizing and then plugging them in, a technique known as add-one smoothing. (Note that plugging in the MAP parameters would not have this smoothing effect, since the mode becomes the MLE if $a = b = 1$, see Section ??.)

0.3.4.2 Predicting the outcome of multiple future trials

Suppose now we were interested in predicting the number of heads, $\tilde{x}$, in M future trials. This is given by

$$p(\tilde{x}|\mathcal{D}) = \int_0^1 \mathrm{Bin}(\tilde{x}|M,\theta)\mathrm{Beta}(\theta|a,b)\mathrm{d}\theta \tag{0.18}$$

$$= \binom{M}{\tilde{x}} \frac{1}{B(a,b)} \int_0^1 \theta^{\tilde{x}}(1-\theta)^{M-\tilde{x}}\theta^{a-1}(1-\theta)^{b-1}\mathrm{d}\theta \tag{0.19}$$

We recognize the integral as the normalization constant for a $\mathrm{Beta}(a+\tilde{x},M\tilde{x}+b)$ distribution. Hence

$$\int_0^1 \theta^{\tilde{x}}(1-\theta)^{M-\tilde{x}}\theta^{a-1}(1-\theta)^{b-1}\mathrm{d}\theta = B(\tilde{x}+a,M-\tilde{x}+b) \tag{0.20}$$

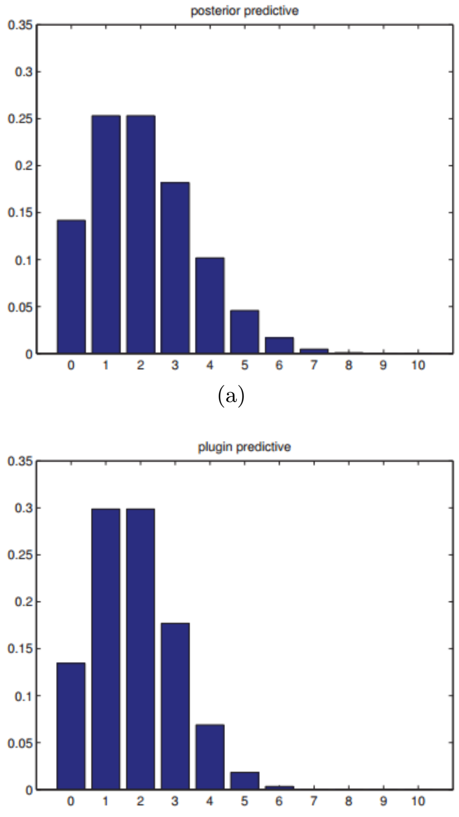
Thus we find that the posterior predictive is given by the following, known as the (compound) beta-binomial distribution:

$$Bb(x|a,b,M) \triangleq \binom{M}{x} \frac{B(x+a,M-x+b)}{B(a,b)} \tag{0.21}$$

This distribution has the following mean and variance

$$\mathrm{mean} = M\frac{a}{a+b}\ ,\ \mathrm{var} = \frac{Mab}{(a+b)^2}\frac{a+b+M}{a+b+1} \tag{0.22}$$

This process is illustrated in Figure ??. We start with a $\mathrm{Beta}(2,2)$ prior, and plot the posterior predictive density after seeing $N_1 = 3$ heads and $N_0 = 17$ tails. Figure ??(b) plots a plug-in approximation using a MAP estimate. We see that the Bayesian prediction has longer tails, spreading its probability mass more widely, and is therefore less prone to overfitting and blackswan type paradoxes.



(a)

(b)

Figure 0.1: (a) Posterior predictive distributions after seeing $N_1 = 3, N_0 = 17$. (b) MAP estimation.

0.4 The Dirichlet-multinomial model

In the previous section, we discussed how to infer the probability that a coin comes up heads. In this section, we generalize these results to infer the probability that a dice with K sides comes up as face k.

0.4.1 Likelihood

Suppose we observe N dice rolls, $\mathcal{D} = \{x_1, x_2, \cdots, x_N\}$, where $x_i \in \{1, 2, \cdots, K\}$. The likelihood has the form

$$p(\mathcal{D}|\vec{\theta}) = \binom{N}{N_1 \cdots N_k} \prod_{k=1}^{K} \theta_k^{N_k} \quad \text{where } N_k = \sum_{i=1}^{N} \mathbb{I}(y_i = k) \tag{0.23}$$

almost the same as Equation ??.

0.4.2 Prior

$$\text{Dir}(\vec{\theta}|\vec{\alpha}) = \frac{1}{B(\vec{\alpha})} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1} \mathbb{I}(\vec{\theta} \in S_K) \tag{0.24}$$

0.4.3 Posterior

$$p(\vec{\theta}|\mathcal{D}) \propto p(\mathcal{D}|\vec{\theta}) p(\vec{\theta}) \tag{0.25}$$

$$\propto \prod_{k=1}^{K} \theta_k^{N_k} \theta_k^{\alpha_k - 1} = \prod_{k=1}^{K} \theta_k^{N_k + \alpha_k - 1} \tag{0.26}$$

$$= \text{Dir}(\vec{\theta}|\alpha_1 + N_1, \cdots, \alpha_K + N_K) \tag{0.27}$$

From Equation ??, the MAP estimate is given by

$$\hat{\theta}_k = \frac{N_k + \alpha_k - 1}{N + \alpha_0 - K} \tag{0.28}$$

If we use a uniform prior, $\alpha_k = 1$, we recover the MLE:

$$\hat{\theta}_k = \frac{N_k}{N} \tag{0.29}$$

0.4.4 Posterior predictive distribution

The posterior predictive distribution for a single multinoulli trial is given by the following expression:

$$p(X = j|\mathcal{D}) = \int p(X = j|\vec{\theta}) p(\vec{\theta}|\mathcal{D}) \mathrm{d}\vec{\theta} \tag{0.30}$$

$$= \int p(X = j|\theta_j) \left[\int p(\vec{\theta}_{-j}, \theta_j|\mathcal{D}) \mathrm{d}\vec{\theta}_{-j} \right] \mathrm{d}\theta_j \tag{0.31}$$

$$= \int \theta_j p(\theta_j|\mathcal{D}) \mathrm{d}\theta_j = \mathbb{E}[\theta_j|\mathcal{D}] = \frac{\alpha_j + N_j}{\alpha_0 + N} \tag{0.32}$$

where $\vec{\theta}_{-j}$ are all the components of except θ_j.

The above expression avoids the zero-count problem. In fact, this form of Bayesian smoothing is even more important in the multinomial case than the binary case, since the likelihood of data sparsity increases once we start partitioning the data into many categories.

0.5 Naive Bayes classifiers

Assume the features are conditionally independent given the class label, then the class conditional density has the following form

$$p(\vec{x}|y=c,\vec{\theta}) = \prod_{j=1}^{D} p(x_j|y=c,\vec{\theta}_{jc}) \tag{0.33}$$

The resulting model is called a naive Bayes classifier(NBC).

The form of the class-conditional density depends on the type of each feature. We give some possibilities below:

- In the case of real-valued features, we can use the Gaussian distribution: $p(\vec{x}|y,\vec{\theta}) = \prod_{j=1}^{D} \mathcal{N}(x_j|\mu_{jc},\sigma_{jc}^2)$, where μ_{jc} is the mean of feature j in objects of class c, and σ_{jc}^2 is its variance.
- In the case of binary features, $x_j \in \{0,1\}$, we can use the Bernoulli distribution: $p(\vec{x}|y,\vec{\theta}) = \prod_{j=1}^{D} \text{Ber}(x_j|\mu_{jc})$, where μ_{jc} is the probability that feature j occurs in class c. This is sometimes called the multivariate Bernoulli naive Bayes model. We will see an application of this below.
- In the case of categorical features, $x_j \in \{a_{j1}, a_{j2}, \cdots, a_{jS_j}\}$, we can use the multinoulli distribution: $p(\vec{x}|y,\vec{\theta}) = \prod_{j=1}^{D} \text{Cat}(x_j|\vec{\mu}_{jc})$, where $\vec{\mu}_{jc}$ is a histogram over the K possible values for x_j in class c.

Obviously we can handle other kinds of features, or use different distributional assumptions. Also, it is easy to mix and match features of different types.

0.5.1 Optimization

We now discuss how to "train" a naive Bayes classifier. This usually means computing the MLE or the MAP estimate for the parameters. However, we will also discuss how to compute the full posterior, $p(\vec{\theta}|\mathcal{D})$.

0.5.1.1 MLE for NBC

The probability for a single data case is given by

$$\begin{aligned} p(\vec{x}_i, y_i|\vec{\theta}) &= p(y_i|\vec{\pi})\prod_j p(x_{ij}|\vec{\theta}_j) \\ &= \prod_c \pi_c^{\mathbb{I}(y_i=c)} \prod_j \prod_c p(x_{ij}|\vec{\theta}_{jc})^{\mathbb{I}(y_i=c)} \end{aligned} \tag{0.34}$$

Hence the log-likelihood is given by

$$p(\mathcal{D}|\vec{\theta}) = \sum_{c=1}^{C} N_c \log \pi_c + \sum_{j=1}^{D}\sum_{c=1}^{C}\sum_{i:y_i=c} \log p(x_{ij}|\vec{\theta}_{jc}) \tag{0.35}$$

where $N_c \triangleq \sum_i \mathbb{I}(y_i=c)$ is the number of feature vectors in class c.

We see that this expression decomposes into a series of terms, one concerning $\vec{\pi}$, and DC terms containing the θ_{jc}'s. Hence we can optimize all these parameters separately.

From Equation ??, the MLE for the class prior is given by

$$\hat{\pi}_c = \frac{N_c}{N} \tag{0.36}$$

The MLE for θ_{jc}'s depends on the type of distribution we choose to use for each feature.

In the case of binary features, $x_j \in \{0,1\}$, $x_j|y=c \sim \text{Ber}(\theta_{jc})$, hence

$$\hat{\theta}_{jc} = \frac{N_{jc}}{N_c} \tag{0.37}$$

where $N_{jc} \triangleq \sum_{i:y_i=c} \mathbb{I}(y_i=c)$ is the number that feature j occurs in class c.

In the case of categorical features, $x_j \in \{a_{j1}, a_{j2}, \cdots, a_{jS_j}\}$, $x_j|y=c \sim \text{Cat}(\vec{\theta}_{jc})$, hence

$$\hat{\vec{\theta}}_{jc} = (\frac{N_{j1c}}{N_c}, \frac{N_{j2c}}{N_c}, \cdots, \frac{N_{jS_j}}{N_c})^T \tag{0.38}$$

where $N_{jkc} \triangleq \sum_{i=1}^{N} \mathbb{I}(x_{ij}=a_{jk}, y_i=c)$ is the number that feature $x_j = a_{jk}$ occurs in class c.

0.5.1.2 Bayesian naive Bayes

Use a $\text{Dir}(\vec{\alpha})$ prior for $\vec{\pi}$.

In the case of binary features, use a $\text{Beta}(\beta 0, \beta 1)$ prior for each θ_{jc}; in the case of categorical features, use a $\text{Dir}(\vec{\alpha})$ prior for each $\vec{\theta}_{jc}$. Often we just take $\vec{\alpha} = \vec{1}$ and $\vec{\beta} = \vec{1}$, corresponding to add-one or Laplace smoothing.

0.5.2 Using the model for prediction

The goal is to compute

$$\begin{aligned} y = f(\vec{x}) &= \arg\max_{c} P(y=c|\vec{x},\vec{\theta}) \\ &= P(y=c|\vec{\theta})\prod_{j=1}^{D} P(x_j|y=c,\vec{\theta}) \end{aligned} \tag{0.39}$$

We can the estimate parameters using MLE or MAP, then the posterior predictive density is obtained by simply plugging in the parameters $\bar{\vec{\theta}}$(MLE) or $\hat{\vec{\theta}}$(MAP).

Or we can use BMA, just integrate out the unknown parameters.

0.5.3 The log-sum-exp trick

when using generative classifiers of any kind, computing the posterior over class labels using Equation ?? can fail due to numerical underflow. The problem is that $p(\vec{x}|y=c)$ is often a very small number, especially if is a high-dimensional vector. This is because we require that $\sum_{\vec{x}} p(\vec{x}|y)=1$, so the probability of observing any particular high-dimensional vector is small. The obvious solution is to take logs when applying Bayes rule, as follows:

$$\log p(y=c|\vec{x},\vec{\theta}) = b_c - \log\left(\sum_{c'} e^{b_{c'}}\right) \tag{0.40}$$

where $b_c \triangleq \log p(\vec{x}|y=c,\vec{\theta}) + \log p(y=c|\vec{\theta})$.

We can factor out the largest term, and just represent the remaining numbers relative to that. For example,

$$\begin{aligned} \log(e^{-120}+e^{-121}) &= \log(e^{-120}(1+e^{-1})) \\ &= \log(1+e^{-1}) - 120 \end{aligned} \tag{0.41}$$

In general, we have

$$\sum_{c} e^{b_c} = \log\left[(\sum e^{b_c-B})e^{B}\right] = \log\left(\sum e^{b_c-B}\right) + B \tag{0.42}$$

where $B \triangleq \max\{b_c\}$.

This is called the log-sum-exp trick, and is widely used.

0.5.4 Feature selection using mutual information

Since an NBC is fitting a joint distribution over potentially many features, it can suffer from overfitting. In addition, the run-time cost is $O(D)$, which may be too high for some applications.

One common approach to tackling both of these problems is to perform feature selection, to remove "irrelevant" features that do not help much with the classification problem. The simplest approach to feature selection is to evaluate the relevance of each feature separately, and then take the top K,whereKis chosen based on some tradeoff between accuracy and complexity. This approach is known as variable ranking, filtering, or screening.

One way to measure relevance is to use mutual information (Section ??) between feature X_j and the class label Y

$$\mathbb{I}(X_j,Y) = \sum_{x_j}\sum_{y} p(x_j,y)\log\frac{p(x_j,y)}{p(x_j)p(y)} \tag{0.43}$$

If the features are binary, it is easy to show that the MI can be computed as follows

$$\mathbb{I}_j = \sum_{c}\left[\theta_{jc}\pi_c\log\frac{\theta_{jc}}{\theta_j} + (1-\theta_{jc})\pi_c\log\frac{1-\theta_{jc}}{1-\theta_j}\right] \tag{0.44}$$

where $\pi_c = p(y=c)$, $\theta_{jc} = p(x_j=1|y=c)$, and $\theta_j = p(x_j=1) = \sum_c \pi_c\theta_{jc}$.

0.5.5 Classifying documents using bag of words

Document classification is the problem of classifying text documents into different categories.

0.5.5.1 Bernoulli product model

One simple approach is to represent each document as a binary vector, which records whether each word is present or not, so $x_{ij}=1$ iff word j occurs in document i, otherwise $x_{ij}=0$. We can then use the following class conditional density:

$$\begin{aligned} p(\vec{x}_i|y_i=c,\vec{\theta}) &= \prod_{j=1}^{D}\text{Ber}(x_{ij}|\theta_{jc}) \\ &= \prod_{j=1}^{D}\theta_{jc}^{x_{ij}}(1-\theta_{jc})^{1-x_{ij}} \end{aligned} \tag{0.45}$$

This is called the Bernoulli product model, or the binary independence model.

0.5.5.2 Multinomial document classifier

However, ignoring the number of times each word occurs in a document loses some information (McCallum and Nigam 1998). A more accurate representation counts the number of occurrences of each word. Specifically, let $\vec{x}_i$ be a vector of counts for document i, so $x_{ij} \in \{0, 1, \cdots, N_i\}$, where N_i is the number of terms in document i(so $\sum_{j=1}^{D} x_{ij} = N_i$). For the class conditional densities, we can use a multinomial distribution:

$$p(\vec{x}_i|y_i = c, \vec{\theta}) = \mathrm{Mu}(\vec{x}_i|N_i, \vec{\theta}_c) = \frac{N_i!}{\prod_{j=1}^{D} x_{ij}!} \prod_{j=1}^{D} \theta_{jc}^{x_{ij}} \tag{0.46}$$

where we have implicitly assumed that the document length N_i is independent of the class. Here $_{jc}$ is the probability of generating word j in documents of class c; these parameters satisfy the constraint that $\sum_{j=1}^{D} \theta_{jc} = 1$ for each class c.

Although the multinomial classifier is easy to train and easy to use at test time, it does not work particularly well for document classification. One reason for this is that it does not take into account the burstiness of word usage. This refers to the phenomenon that most words never appear in any given document, but if they do appear once, they are likely to appear more than once, i.e., words occur in bursts.

The multinomial model cannot capture the burstiness phenomenon. To see why, note that Equation ?? has the form $\theta_{jc}^{x_{ij}}$, and since $\theta_{jc} \ll 1$ for rare words, it becomes increasingly unlikely to generate many of them. For more frequent words, the decay rate is not as fast. To see why intuitively, note that the most frequent words are function words which are not specific to the class, such as "and", "the", and "but"; the chance of the word "and" occuring is pretty much the same no matter how many time it has previously occurred (modulo document length), so the independence assumption is more reasonable for common words. However, since rare words are the ones that matter most for classification purposes, these are the ones we want to model the most carefully.

0.5.5.3 DCM model

Various ad hoc heuristics have been proposed to improve the performance of the multinomial document classifier (Rennie et al. 2003). We now present an alternative class conditional density that performs as well as these ad hoc methods, yet is probabilistically sound (Madsen et al. 2005).

Suppose we simply replace the multinomial class conditional density with the Dirichlet Compound Multinomial or DCM density, defined as follows:

$$\begin{aligned} p(\vec{x}_i|y_i = c, \vec{\alpha}) &= \int \mathrm{Mu}(\vec{x}_i|N_i, \vec{\theta}_c)\mathrm{Dir}(\vec{\theta}_c|\vec{\alpha}_c) \\ &= \frac{N_i!}{\prod_{j=1}^{D} x_{ij}!} \prod_{j=1}^{D} \frac{B(\vec{x}_i + \vec{\alpha}_c)}{B(\vec{\alpha}_c)} \end{aligned} \tag{0.47}$$

(This equation is derived in Equation TODO.) Surprisingly this simple change is all that is needed to capture the burstiness phenomenon. The intuitive reason for this is as follows: After seeing one occurence of a word, say wordj, the posterior counts on □j gets updated, making another occurence of wordjmore likely. By contrast, if□j is fixed, then the occurences of each word are independent. The multinomial model corresponds to drawing a ball from an urn with Kcolors of ball, recording its color, and then replacing it. By contrast, the DCM model corresponds to drawing a ball, recording its color, and then replacing it with one additional copy; this is called the Polya urn.

Using the DCM as the class conditional density gives much better results than using the multinomial, and has performance comparable to state of the art methods, as described in (Madsen et al. 2005). The only disadvantage is that fitting the DCM model is more complex; see (Minka 2000e; Elkan 2006) for the details.