Bayesian Linear Classification 7

7.1 Introduction

The goal in (Bayesian) Linear Classification is (as the name suggests) to learn linear models for classification, meaning models in which the decision boundaries of the input space are linear functions of the input points.

This scenario is somehow more complicated than Bayesian Linear Regression, because, due to a different model for the noise in the observations, the posterior distribution is not analytically tractable. Hence the challenge is to find a good approximation of the posterior of interest.

Consider a dataset $D = (x_n, y_n)$ with $n = 1, \dots, N$, where y_n are categorical. There are various ways of representing class labels y_n, depending on the problem at hand:

- 2-class problems: $y_n \in \{0, 1\}$.
- K-class problem ($K > 2$): $y_n = (y_{nj})_{j=1,\dots,K}$ such that $y_{nj} \in \{0, 1\}$, $\sum_j y_{nj} = 1$. This is called **one-hot-encoding** convention, essentially y_n is represented as a boolean vector of K numbers, with the constraint that only one entry is 1 and all the others are 0.

We have three possible approaches to classification:

- **Discriminant function** $f(x) \in \{1, \dots, K\}$: the goal is to learn a function that maps each input to a specific class (e.g. random forest classification is based on this approach).
- **Discriminative approach** $p(C_k|x) = f(h(x))$: the goal is to model explicitly the class posterior (e.g. logistic regression, where $p(C_k|x) = f(w^T\phi(x))$). In this context f is called *activation function*, f^{-1} is called *link function*.
- **Generative approach** $p(C_k|x) = \frac{p(x|C_k)p(C_k)}{p(x)}$: the goal is to model the class conditional probability $p(x|C_k)$ from data, then, considering a prior over classes $p(C_k)$, plug the Bayes' theorem to compute class posterior $p(C_k|x)$.

7.2 Logistic Regression

As already mentioned, **Logistic Regression** is a discriminative model.

Given data $(x_n, y_n), n = 1, \dots, N$, we want to learn $p(C_k|x) = f(w^T\phi(x))$, where $\phi(x) = (\phi_0(x), \dots, \phi_{M-1}(x))$ are the *basis functions*.

The activation function is usually chosen to be either the **Logit** (logistic, sigmoid) function:

$$\sigma(a) = \frac{1}{1 + e^{-a}}$$

or the **Probit** function (i.e. the cumulative distribution function of the standard Gaussian distribution):

$$\psi(a) = \int_{-\infty}^{a} \mathcal{N}(\theta|0,1)d\theta$$

Consider the binary classification scenario, i.e. $y_n \in \{0,1\}$, with logit activation function. Call $s_n = \sigma(w^T \phi(x_n)) = p(C_1|x_n)$ (i.e. probability of assigning input x to class 1). For a fixed w, we use a Bernoulli model of noise:

$$p(y_n|x_n) = s_n^{y_n}(1-s_n)^{1-y_n}$$

Hence the likelihood is:

$$p(\underline{y}|\underline{x}) = \prod_{n=1}^{N} s_n^{y_n}(1-s_n)^{1-y_n}$$

Once we have the likelihood, we can compute the cross-entropy error function as:

$$E(w) = -\frac{1}{N}\log p(\underline{y}|\underline{x}) = -\frac{1}{N}\sum_{n=1}^{N} y_n \log s_n + (1-y_n)\log(1-s_n)$$

Remark: in this case the dependency on the weights w is highly non-linear, indeed it is through $\log s_n$, being s_n the sigmoid function.

The gradient of the cross-entropy reads as:

$$\nabla E(w) = \frac{1}{N}\sum_{n=1}^{N}(s_n - y_n)\phi(x_n)$$

The equation $\nabla E(w) = 0$ has no analytic solution, hence we need to resort to a numerical optimization method in order to find the maximum likelihood solution $w_{ML} = \operatorname{argmin}_w E(w)$ (note that $E(w)$ is a convex function).

One possibility is to use stochastic gradient descent for online training, using the update rule for w:

$$w_{n+1} = w_n - \eta_n \nabla E(w_n)$$

where η_n is called *learning rate*.

Remark: if the data are linearly separable in the feature space, then any separating hyperplane $w_{ML}^T\phi(x) = 0$ is a solution, hence we have ∞-many solutions and the optimization problem is ill-defined. To overcome this issue, we typically introduce a penalty term in the function that should be optimized (forcing the weights to be as small as possible), such that the problem remains convex, e.g. we might minimize $E(w) + \alpha w^T w$.

The same ideas described so far can be recasted to the case of multi-class classification. In this scenario data are (x_n, y_n) with $y_n = (y_{n1}, \ldots, y_{nK})$, i.e. one-hot-encoding over K classes, and class-conditional distributions are:

$$p(C_k|x) = \sigma_k(W^T \phi(x))$$

where W^T is a $K \times M$ matrix, and

$$\sigma_k(\underline{a}) = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

is called **softmax** function (intuitively it turns a vector of real numbers into a vector of probabilities).

Explicitly, this corresponds to:

$$\begin{cases} a_1 = w_1^T \phi(x) \\ \dots \\ a_k = w_k^T \phi(x) \end{cases}$$

and

$$p(C_k|x) = \sigma_k(W^T \phi(x)) = \sigma_k(a_1, ..., a_k)$$

In this case the cross-entropy is:

$$E(w) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{j=1}^{K} y_{nj} \log s_{nj}$$

with $s_{nj} = \sigma_j(W^T \phi(x_n))$.

Hence the gradient for class j weights is:

$$\nabla_{w_j} E(w) = \frac{1}{N} \sum_{n=1}^{N} (s_{nj} - y_{nj}) \phi(x_n)$$

7.3 Laplace Approximation

The idea behind **Laplace Approximation** is to approximate an (unknown) distribution with a Gaussian. Notice that it is a local approximation and does not capture the properties of the global distribution. Intuitively this technique consists in centering a Gaussian in a mode of the distribution, and use information from the Hessian (i.e. we match the curvature) in order to identify the variance of the Gaussian.

This approximation is often used when we want to approximate some posterior distribution, which is known up to a normalization constant.

In the 1-dimensional case, the form of the distribution that we want to approximate is $p(z) = \frac{1}{Z} f(z)$, with $Z = \int f(z)dz$. The idea is to:

- find a mode z_0 of $f(z)$, i.e. a point such that $\frac{d}{dz} f(z_0) = 0$ and z_0 is a point of maximum;
- match the curvature of f at z_0 with a normal distribution.

We can rely on the fact that the logarithm of a Gaussian density is a quadratic function and Taylor expand $\log f(z)$ around z_0:

$$\log f(z) \approx \log f(z_0) - \frac{1}{2}A(z - z_0)^2$$

with $A = -\frac{d^2}{dz^2} \log f(z_0)$, $A > 0$ (since z_0 is a mode).

Hence taking the exponential:

$$f(z) \approx f(z_0) \cdot \exp\Big(- \frac{1}{2}A(z - z_0)^2\Big)$$

Since we seek to approximate $p(z)$ with a Gaussian $q(z)$, this is given by:

$$q(z) \sim \mathcal{N}(z|z_0, A^{-1})$$

i.e. A takes the role of the precision of the approximating Gaussian distribution. More explicitly:

$$q(z) = \Big(\frac{A}{2\pi}\Big)^{\frac{1}{2}} \exp\Big(- \frac{1}{2}A(z - z_0)^2\Big)$$

Since $p(z) = \frac{1}{Z}f(z) \approx \frac{1}{Z}f(z_0)\exp\big(- \frac{1}{2}A(z - z_0)^2\big)$, we also have an approximation of the marginal likelihood:

$$Z \approx f(z_0)\Big(\frac{A}{2\pi}\Big)^{-\frac{1}{2}}$$

In the n-dimensional case we proceed in the same way: given a density $p(z) = \frac{1}{Z}f(z)$, we find a mode z_0 (s.t. $\nabla \log f(z_0) = 0$) and approximate $\log f(z)$ around z_0 by Taylor expansion:

$$\log f(z) \approx \log f(z_0) - \frac{1}{2}(z - z_0)^T A(z - z_0)$$

with $A = -\nabla\nabla \log f(z_0)$.

This gives a Gaussian approximation around z_0 by:

$$q(z) = \mathcal{N}(z|z_0, A^{-1})$$

Furthermore the normalization constant can be approximated as:

$$Z = \frac{(2\pi)^{\frac{1}{2}}}{|A|^{\frac{1}{2}}} f(z_0)$$

and for the multivariate case:

$$Z = \frac{(2\pi)^{\frac{k}{2}}}{|A|^{\frac{1}{2}}} f(z_0)$$

Remark: if the distribution p is very skewed, the Laplace approximation is not very effective; if the distribution p is multimodal, we should take the dominant mode (if present) as mean of the approximating Gaussian.

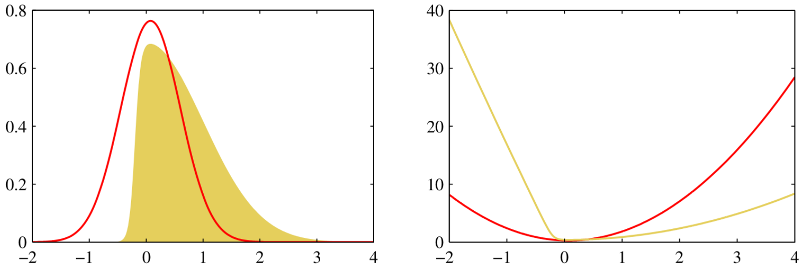

Figure 7.1: Illustration of the Laplace approximation applied to the distribution $p(z) \propto \exp(-z^2/2)\sigma(20z+4)$ where $\sigma(z)$ is the logistic sigmoid function. The left plot shows the normalized distribution $p(z)$ in yellow, together with the Laplace approximation centred on the mode z_0 of $p(z)$ in red. The right plot shows the negative logarithms of the corresponding curves.

7.3.1 Laplace approximation for model comparison

It is possible to use Laplace approximation for the marginal likelihood in a model comparison framework.

Consider data D and a parametric model M depending on parameters θ. We fix a prior $p(\theta)$ over θ, and we plug the Bayes' theorem to get $p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}$.

Typically the marginal likelihood $p(D) = \int p(D|\theta)p(\theta)d\theta$ is hard to compute. This fits the previous framework if we set $f(\theta) = p(D|\theta)p(\theta)$ and $Z = p(D)$.

By Laplace approximation around the maximum-a-posteriori (MAP) estimate θ_{MAP} we get:

$$p(D) \approx \frac{(2\pi)^{\frac{M}{2}}}{|A|^{\frac{1}{2}}} f(\theta_{MAP}) = \frac{(2\pi)^{\frac{M}{2}}}{|A|^{\frac{1}{2}}} p(D|\theta_{MAP})p(\theta_{MAP})$$

So,

$$\log p(D) \approx \log p(D|\theta_{MAP}) + \log p(\theta_{MAP}) + \frac{M}{2}\log(2\pi) - \frac{1}{2}\log|A|$$

with $M = |\theta|$ (number of parameters) and $A = -\nabla\nabla\big[\log p(D|\theta_{MAP}) + \log p(\theta_{MAP})\big]$

Remark: the marginal likelihood is a trade off between model complexity and fit to the data (indeed the last three terms in the previous sum penalize the log-likelihood in terms of model complexity).

The **Bayesian Information Content (BIC)** index is defined as:

$$\log p(D) \approx \log p(D|\theta_{MAP}) - \frac{1}{2}M\log N$$

and it is a further approximation of the marginal likelihood. It can be used to penalize log-likelihood w.r.t. model complexity, to compare different models.

7.4 Bayesian Logistic Regression

Given observations (x_n, y_n) with $n = 1, \dots, N$, consider a set of basis functions $\phi(x) = \phi_0(x), \dots, \phi_{M-1}(x)$ and the logit activation function $\sigma(w^T \phi(x))$.

To recast logistic regression in a Bayesian framework, we place a Gaussian prior over w, $p(w) = \mathcal{N}(w|m_0, S_0)$, with m_0, S_0 fixed or computed via marginal likelihood optimization. The posterior is then:

$$p(w|\underline{x}, \underline{y}) = \frac{p(\underline{y}|w, \underline{x})p(w)}{p(\underline{y}|\underline{x})} \propto p(\underline{y}|w, \underline{x})p(w)$$

Recall that, in the 2-class problem with a Bernoulli model of noise, the likelihood reads as $p(\underline{y}|w, \underline{x}) = \prod_{i=1}^N s_i^{y_i}(1 - s_i)^{1-y_i}$, being $s_i = s_i(w) = \sigma(w^T \phi(x_i))$.

Hence the log-posterior is now:

$$\begin{aligned}\log p(w|\underline{y}, \underline{x}) &= \log p(w) + \log p(\underline{y}|w) + C \\ &= -\frac{1}{2}(w - m_0)^T S_0^{-1}(w - m_0) + \\ &+ \sum_{i=1}^N \left[y_i \log s_i(w) + (1 - y_i)\log(1 - s_i(w))\right] + C\end{aligned}$$

Remark: as already mentioned, $s_i(w)$ is not quadratic on w, it is actually an exponential dependency on w (hence not analytically tractable).

We can perform Laplace approximation of the posterior, the steps are the following:

1. find $w_{MAP} = \text{argmax}_w \log p(w|\underline{y}) = \text{argmax}_w \log p(w) + \log p(\underline{y}|w)$ by running a numerical optimization (we can ignore the constant which does not change the location of the maximum).
2. Compute the Hessian at w_{MAP} and invert it: this will give the precision matrix of the Gaussian

$$S_N^{-1} = S_0^{-1} + \sum_{n=1}^N s_n(w_{MAP})(1 - s_n(w_{MAP}))\phi(x_n)\phi^T(x_n)$$

Hence, the Laplace approximation of the posterior is $p(w|\underline{y}) \approx q(w)$ with

$$q(w) = \mathcal{N}(w|w_{MAP}, S_N)$$

Given this posterior, we need to marginalize it to compute the **predictive distribution** (which will allow us to do model averaging).

In the binary classification scenario, the predictive distribution for class C_1 is given by (plugging the previous approximation):

$$p(C_1|x^\star, \underline{y}, \underline{x}) = \int p(C_1|x^\star, w, \underline{x}, \underline{y})p(w|\underline{y}, \underline{x})dw \approx \int \sigma(w^T \phi(x^\star))q(w)dw$$

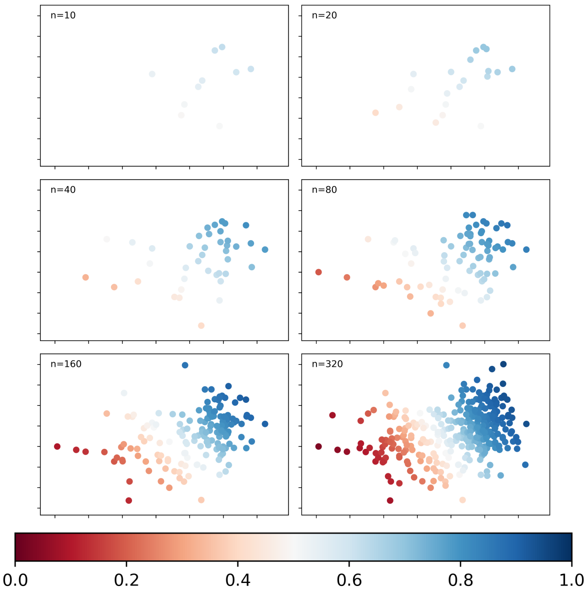

Figure 7.2: These plots show the predictive distribution for an increasing number of points in the training set. The more intense the colour, the more confident is the prediction (close to 0 or 1). As a consequence of using a Bayesian approach, the confidence of predictions increases with the number of observations.

This is in principle an M-dimensional integral. However, σ depends on w only through the 1-dimensional projection $a = w^T\phi(x^\star)$, which is a linear combination of Gaussians, because w are normally distributed and ϕ are fixed. Hence q restricted to the dimension identified by a is still a Gaussian distribution: $q(a) \sim \mathcal{N}(a|\mu_a, \sigma_a^2)$, with $\mu_a = w_{MAP}^T\phi(x^\star)$ and $\sigma_a^2 = \phi^T(x^\star)S_N\phi(x^\star)$, so that:

$$p(C_1|x^\star, \underline{y}, \underline{x}) = \int \sigma(a)q(a)da$$

At this point we can use the **probit approximation** trick, i.e. we can approximate the previous integral by approximating the logistic function with the probit $\sigma(a) \approx \psi(\lambda a)$, such that $\lambda^2 = \frac{\pi}{8}$ and $\sigma'(0) = \psi'(0)$

Hence:

$$\int \psi(\lambda a)q(a)da = \int \psi(\lambda a)\mathcal{N}(a|\mu_a, \sigma_a^2)da = \Psi\left(\frac{\mu_a}{(\lambda^{-2} + \sigma_a^2)^{\frac{1}{2}}}\right)$$

so that, approximating back to the logistic, we get:

$$p(C_1|x^\star, \underline{y}, \underline{x}) \approx \sigma(\kappa(\sigma_a^2)\mu_a)$$

being $\kappa(\sigma_a^2) = \left(1 + \pi\frac{\sigma_a^2}{8}\right)^{-\frac{1}{2}}$

In this way, the predictive distribution depends from μ_a but it is rescaled by the variance:

- if $\sigma_a^2 = 0$ then $p(C_1|x^\star, \underline{y}, \underline{x}) \approx \sigma(\mu_a)$
- if $\sigma_a^2 \gg 0$ then $p(C_1|x^\star, \underline{y}, \underline{x}) \longrightarrow \frac{1}{2}$ which represents the maximum level of uncertainty on the prediction

Remark: the dominating cost of this procedure is identifying the MAP.