Bayesian Logistic Classification

3F8: Inference Coursework
Theo Brown
Selwyn College, University of Cambridge

March 20, 2022

Abstract

Abstract goes here

1 Introduction

2 Laplace approximation for logistic regression

2.1 Justification

In the previous report, the maximum-likelihood estimate for the model weights was used in classification[1]. A full posterior distribution of the weights is required to perform fully Bayesian classification:

$$p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y}) = \frac{p(\boldsymbol{X}|\boldsymbol{w},\boldsymbol{y})p(\boldsymbol{w})}{p(\boldsymbol{X}|\boldsymbol{y})} \tag{1}$$

Calculating the model evidence $p(\boldsymbol{X}|\boldsymbol{y})$ requires integrating a product of logistic functions, which is intractable. The Laplace approximation can be used to find an approximate posterior distribution and an approximate predictive distribution for the logistic classifier, while avoiding the difficult integral. The derivation of these results is found in Appendix A.

In this application, the prior distribution of the model weights is chosen to be Gaussian:

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; 0, \boldsymbol{I}\sigma_0^2) \tag{2}$$

2.1.1 Finding the MAP estimate

The Laplace approximation requires the location of a maximum in the distribution to be approximated, which in this case will be the MAP estimate of $\boldsymbol{w}$. This can be found by applying gradient ascent to the posterior. Firstly, the gradient of the log-posterior is calculated:

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{w}} \log p(\boldsymbol{w}|\tilde{\boldsymbol{X}},\boldsymbol{y}) &= \frac{\partial}{\partial \boldsymbol{w}} \log p(\tilde{\boldsymbol{X}}|\boldsymbol{w},\boldsymbol{y}) + \frac{\partial}{\partial \boldsymbol{w}} \log p(\boldsymbol{w}) \\ &= \tilde{\boldsymbol{X}}(\mathbf{y} - \sigma(\tilde{\boldsymbol{X}}^T\boldsymbol{w})) + \frac{1}{\sigma_0^2}\boldsymbol{w} \end{aligned} \tag{3}$$

This can then be used with a standard gradient-based solver to find a value for $\boldsymbol{w}_{\text{MAP}}$.

2.1.2 Results of the Laplace approximation

Applying the Laplace approximation gives following results:

- Approximate posterior distribution of $\boldsymbol{w}$:

$$p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y}) \approx \mathcal{N}(\boldsymbol{w}; \boldsymbol{w}_{\text{MAP}}, \boldsymbol{S}_N) \tag{4}$$

[1]For the model definition, and definitions of $\boldsymbol{X}$, $\boldsymbol{y}$, $\tilde{\boldsymbol{x}}$, etc, see the previous report.

- Approximate predictive distribution for new points $\boldsymbol{x}_*$:

$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{y}, \boldsymbol{X}) \approx \sigma\left(\frac{\mu_p}{\sqrt{1+\sigma_p^2\lambda^2}}\right) \tag{5}$$

- Approximate model evidence:

$$p(\boldsymbol{X}|\boldsymbol{y}) = \sqrt{\frac{(2\pi)^N}{\det \boldsymbol{S}_N^{-1}}} \exp\left(\mathcal{L}(\boldsymbol{w}_{\text{MAP}}) + \mathcal{P}(\boldsymbol{w}_{\text{MAP}})\right) \tag{6}$$

where

$$\boldsymbol{S}_N = \left(\frac{1}{\sigma_0^2}\boldsymbol{I} + \tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^T\boldsymbol{\sigma}(1-\boldsymbol{\sigma})\right)^{-1} \tag{7}$$

$$\boldsymbol{\sigma} = \sigma(\tilde{\boldsymbol{X}}^T\boldsymbol{w}) \tag{8}$$

$$\mu_p = \tilde{\boldsymbol{x}}_*^T\boldsymbol{w}_{\text{MAP}} \tag{9}$$

$$\sigma_p^2 = \tilde{\boldsymbol{x}}_*^T\boldsymbol{S}_N\tilde{\boldsymbol{x}}_* \tag{10}$$

$$\lambda^2 = \frac{\pi}{8} \tag{11}$$

$$\mathcal{L}(\boldsymbol{w}_{\text{MAP}}) = \log p(\boldsymbol{X}|\boldsymbol{w},\boldsymbol{y})\big|_{\boldsymbol{w}=\boldsymbol{w}_{\text{MAP}}} \tag{12}$$

$$\mathcal{P}(\boldsymbol{w}_{\text{MAP}}) = \log p(\boldsymbol{w})\big|_{\boldsymbol{w}=\boldsymbol{w}_{\text{MAP}}} \tag{13}$$

and N is the number of data points $\boldsymbol{x}_n$.

2.2 Implementation in Python

Firstly, the data is loaded and split into training and test sets.

```python
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.utils import shuffle

X_data, y_data = shuffle(np.loadtxt('data/X.txt'),
                         np.loadtxt('data/y.txt'))

X_train, X_test, y_train, y_test = train_test_split(X_data, y_data, train_size=800)
```

Functions are defined to expand the data through a set of radial basis functions centred on the training points, and prepend a column of ones:

```python
def prepend_ones(M):
    return np.column_stack((np.ones(M.shape[0]), M))

def expand_rbf(l, X, Z=X_train):
    X2 = np.sum(X**2, 1)
    Z2 = np.sum(Z**2, 1)
    ones_Z = np.ones(Z.shape[ 0 ])
    ones_X = np.ones(X.shape[ 0 ])
    r2 = np.outer(X2, ones_Z) - 2 * np.dot(X, Z.T) + np.outer(ones_X, Z2)
    return prepend_ones(np.exp(-0.5 / l**2 * r2))
```

A numerical approximation of the MAP estimate for the weights is found:

```python
from scipy.optimize import fmin_l_bfgs_b as minimise

def logistic(x):
    return 1 / (1 + np.exp(-x))
```

```
def log_prior(w, variance):
    return -1 / (2 * variance) * (w.T @ w)

def log_likelihood(w, X, y):
    sigma = logistic(X @ w)
    return np.sum(y * np.log(sigma)
                  + (1 - y) * np.log(1 - sigma))

def negative_log_posterior(w, X, y, prior_variance):
    return -(log_likelihood(w, X, y) + log_prior(w, prior_variance))

def negative_posterior_gradient(w, X, y, prior_variance):
    return -((y - logistic(X @ w)) @ X - w / prior_variance)

def find_w_map(X, y, w0=None, prior_variance=1):
    if w0 is None:
        w0 = np.random.normal(size=X.shape[1])
    w_map, posterior_at_wmap, d = minimise(negative_log_posterior,
                                           w0,
                                           negative_posterior_gradient,
                                           args=[X, y, prior_variance])
    return w_map

expanded_training_set = expand_rbf(rbf_width, X_train)
w_map = find_w_map(expanded_training_set, y_train)
```

Note that scipy.optimize.fmin_l_bfgs_b is a minimisation function, so to maximise the posterior we have to work with the negative posterior and negative posterior gradient. Once $\boldsymbol{w}_{\text{MAP}}$ is found, the Laplace approximation for the model evidence and the predictive distribution can be found:

```
def S_N_inv(w, rbf_width, prior_variance):
    M = w.shape[0]
    X_tilde = expand_rbf(rbf_width, X_train)
    sigma = logistic(X_tilde @ w)
    return np.identity(M) / prior_variance + X_tilde @ X_tilde.T @ sigma @ (1 - sigma)

def log_evidence(w, X, y, rbf_width, prior_variance):
    d = np.linalg.det(S_N_inv(w, rbf_width, prior_variance))
    return (M/2)*np.log(2*np.pi) - 0.5*np.log(d) + log_likelihood(w, X, y) + log_prior(w, prior_

def laplace_prediction(inputs, weights, rbf_width, prior_variance):
    X_tilde = expand_rbf(rbf_width, X_train)
    sigma = logistic(X_tilde @ weights)
    S_N = np.linalg.inv(S_N_inv(weights, rbf_width, prior_variance))
    predictive_mean = inputs @ weights
    predictive_variance = np.array([x.T @ C_N @ x for x in inputs])
    return logistic(predictive_mean / np.sqrt(1 + predictive_variance*np.pi/8))
```

3 Performance of the Laplace Approximation

A Deriving Bayesian Logistic Classification using the Laplace Approximation

A.1 Laplace approximation

To define a probability distribution function (PDF) $p(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^N$ it is necessary to integrate a function to find the normalising constant K:

$$p(\boldsymbol{x}) := \frac{f(\boldsymbol{x})}{\int f(\boldsymbol{x})d\boldsymbol{x}} = \frac{1}{K} f(\boldsymbol{x}) \tag{14}$$

In many cases, the integral $\int f(\boldsymbol{x})d\boldsymbol{x}$ is intractable or does not have a closed-form solution.

The Laplace approximation finds a Gaussian, $q(\boldsymbol{x})$, that provides a good approximation to $p(\boldsymbol{x})$ near a local maximum of $p(\boldsymbol{x})$. As $q(\boldsymbol{x})$ is Gaussian, its normalising constant is easy to find, so the problem of solving $\int f(\boldsymbol{x})d\boldsymbol{x}$ is avoided. To find $q(\boldsymbol{x})$, we start with the truncated Taylor expansion of $\ln f(\boldsymbol{x})$ around a local maximum $\boldsymbol{x}_0$:

$$\ln f(\boldsymbol{x}) \approx \ln f(\boldsymbol{x}_0) + \nabla \ln f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x}_0}(\boldsymbol{x}-\boldsymbol{x}_0) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_0)^T \nabla^2 \ln f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x}_0}(\boldsymbol{x}-\boldsymbol{x}_0)$$

At a maximum of $f(\boldsymbol{x})$, $\nabla \ln f(\boldsymbol{x}) = 0$ as the logarithm is a monotonic function. Hence, close to $\boldsymbol{x}_0$:

$$\begin{aligned} \ln f(\boldsymbol{x}) &\approx \ln f(\boldsymbol{x}_0) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_0)^T \nabla^2 \ln f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x}_0}(\boldsymbol{x}-\boldsymbol{x}_0) \\ f(\boldsymbol{x}) &\approx f(\boldsymbol{x}_0) \exp\left(\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_0)^T \nabla^2 \ln f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x}_0}(\boldsymbol{x}-\boldsymbol{x}_0)\right) \end{aligned} \tag{15}$$

Equation 15 is of the form of an un-normalised Gaussian. Let $\boldsymbol{P} = -\nabla^2 \ln f(\boldsymbol{x})\big|_{\boldsymbol{x}=\boldsymbol{x}_0}$, and normalise Equation 15 to get an approximation for $p(\boldsymbol{x})$:

$$\begin{aligned} p(\boldsymbol{x}) &\approx \frac{1}{(2\pi)^{\frac{M}{2}} \det \boldsymbol{P}^{-\frac{1}{2}}} \exp\left(\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_0)^T \boldsymbol{P}(\boldsymbol{x}-\boldsymbol{x}_0)\right) \\ &= \mathcal{N}(\boldsymbol{x}; \boldsymbol{x}_0, \boldsymbol{P}^{-1}) \end{aligned} \tag{16}$$

For this to hold, $\boldsymbol{P}$ must be positive definite, i.e. $\boldsymbol{x}_0$ must be a maximum.

This method can be used instead to find an approximate value of the normalising constant K. Substituting Equation 15 into the definition of K:

$$\begin{aligned} K = \int f(\boldsymbol{x})d\boldsymbol{x} &\approx f(\boldsymbol{x}_0) \int \exp\left(\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_0)^T \boldsymbol{P}(\boldsymbol{x}-\boldsymbol{x}_0)\right) d\boldsymbol{x} \\ &= \frac{(2\pi)^{\frac{N}{2}}}{\det \boldsymbol{P}^{\frac{1}{2}}} f(\boldsymbol{x}_0) \end{aligned} \tag{17}$$

A.2 Bayesian logistic regression

A.2.1 Posterior distribution

Given that the posterior Laplace approximation will be Gaussian, a Gaussian prior for the model weights is chosen:

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{m}_0, \boldsymbol{S}_0) \tag{18}$$

In order to apply the Laplace approximation to the posterior, the location of a maximum in the posterior is required (this will be the MAP estimate). Using the results for the likelihood of the logistic model in the previous report and the log of Equation 18, the log-posterior of the model weights is:

$$\begin{aligned} \ln p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) = &\sum_{n=1}^{N} y_n \log \sigma(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_n) + (1-y_n) \log \sigma(-\boldsymbol{w}^T \tilde{\boldsymbol{x}}_n) \\ &+ \frac{1}{2}(\boldsymbol{w}-\boldsymbol{w}_0)^T \boldsymbol{S}_0^{-1}(\boldsymbol{w}-\boldsymbol{w}_0) \\ &+ \text{const} \end{aligned} \tag{19}$$

The value of $\boldsymbol{w}$ at the maximum, $\boldsymbol{w}_{\text{MAP}}$, can be found by setting the derivative of Equation 19 to zero. The mean of $q(\boldsymbol{w})$ will be set to $\boldsymbol{w}_{\text{MAP}}$.

Using Equation 16, the covariance matrix of $q(\boldsymbol{w})$ can be found:

$$\begin{aligned}\boldsymbol{S}_N^{-1} &= -\nabla^2 \ln p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y})\big|_{\boldsymbol{w}=\boldsymbol{w}_{\text{MAP}}} \\ &= \boldsymbol{S}_0^{-1} + \sum_{n=1}^{N} \sigma(\boldsymbol{w}^T \tilde{\boldsymbol{x}}_n)\sigma(-\boldsymbol{w}^T \tilde{\boldsymbol{x}}_n)\tilde{\boldsymbol{x}}_n \tilde{\boldsymbol{x}}_n^T \end{aligned} \tag{20}$$

Defining $\boldsymbol{\sigma} = \sigma(\tilde{\boldsymbol{X}}^T \boldsymbol{w})$, this can be written in vector form as:

$$\boldsymbol{S}_N^{-1} = \boldsymbol{S}_0^{-1} + \tilde{\boldsymbol{X}}\tilde{\boldsymbol{X}}^T \boldsymbol{\sigma}(1 - \boldsymbol{\sigma}) \tag{21}$$

Hence, using Equation 16, the posterior distribution is:

$$p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) \approx \mathcal{N}(\boldsymbol{w}; \boldsymbol{w}_{\text{MAP}}, \boldsymbol{S}_N) \tag{22}$$

And using Equation 17, the normalizing constant is:

$$p(\boldsymbol{X}|\boldsymbol{y}) = \frac{(2\pi)^{\frac{N}{2}}}{\det \boldsymbol{S}_N^{-\frac{1}{2}}} p(\boldsymbol{X}|\boldsymbol{w} = \boldsymbol{w}_{\text{MAP}}, \boldsymbol{y}) p(\boldsymbol{w} = \boldsymbol{w}_{\text{MAP}}) \tag{23}$$

A.3 Predictive distribution

The predictive distribution can also be approximated using the Laplace method:

$$\begin{aligned} p(y^* = 1|\boldsymbol{x}^*, \boldsymbol{y}, \boldsymbol{X}) &= \int p(y^* = 1|\boldsymbol{x}^*, \boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{X}) d\boldsymbol{w} \\ &\approx \int \sigma(\boldsymbol{w}^T \boldsymbol{x}^*) q(\boldsymbol{w}) d\boldsymbol{w} \end{aligned} \tag{24}$$

Using the sifting property of the delta function:

$$\sigma(\boldsymbol{w}^T \boldsymbol{x}) = \int \delta(a - \boldsymbol{w}^T \boldsymbol{x})\sigma(a) da \tag{25}$$

Hence:

$$\begin{aligned} p(y^* = 1|\boldsymbol{x}^*, \boldsymbol{y}, \boldsymbol{X}) &\approx \int \int \delta(a - \boldsymbol{w}^T \boldsymbol{x}^*)\sigma(a) q(\boldsymbol{w}) d\boldsymbol{w} da \\ &= \int \sigma(a) \int \delta(a - \boldsymbol{x}^{*T} \boldsymbol{w}) q(\boldsymbol{w}) d\boldsymbol{w} da \end{aligned}$$

The inner integral applies a linear constraint to $q(\boldsymbol{w})$, as the argument of the delta function is 0 unless $a = \boldsymbol{x}^{*T}\boldsymbol{w}$. Hence, the approximate predictive distribution is:

$$\begin{aligned} p(y^* = 1|\boldsymbol{x}^*, \boldsymbol{y}, \boldsymbol{X}) &\approx \int \sigma(a) \mathcal{N}(a; \boldsymbol{x}^{*T} \boldsymbol{w}_{\text{MAP}}, \boldsymbol{x}^{*T} \boldsymbol{S}_N \boldsymbol{x}^{\boldsymbol{*}}) da \\ &= \int \sigma(a) \mathcal{N}(a; \mu_p, \sigma_p^2) da \end{aligned} \tag{26}$$

Where

$$\mu_p = \boldsymbol{x}^{*T} \boldsymbol{w}_{\text{MAP}} \tag{27}$$

$$\sigma_p^2 = \boldsymbol{x}^{*T} \boldsymbol{S}_N \boldsymbol{x}^{\boldsymbol{*}} \tag{28}$$

This integral cannot be expressed analytically, so another approximation is required. The logistic function can be approximated well by a probit function scaled such that the gradient of the two functions at the origin are equal. It can be shown that this gives:

$$\sigma(x) \approx \Phi^{-1}\left(\sqrt{\frac{\pi}{8}} x\right) = \Phi^{-1}(\lambda x) \tag{29}$$

Substituting into Equation 26 and evaluating using properties of the probit function gives:

$$\begin{aligned} p(y^* = 1|\boldsymbol{x}^*, \boldsymbol{y}, \boldsymbol{X}) &\approx \int \Phi^{-1}(\lambda x)\mathcal{N}(a; \mu_p, \sigma_p)da \\ &= \Phi^{-1}\left(\frac{\mu_p}{\sqrt{\lambda^{-2} + \sigma_p^2}}\right) \end{aligned} \tag{30}$$

Using Equation 29, this can be converted back into a logistic function:

$$p(y^* = 1|\boldsymbol{x}^*, \boldsymbol{y}, \boldsymbol{X}) \approx \sigma\left(\frac{\mu_p}{\sqrt{1 + \sigma_p^2\lambda^2}}\right) \tag{31}$$