1 Basics of Statistical Learning

1.1 Introduction

Data science deals with the problem of decision making in the context of data. In particular, this means that given some outcome and some related data, can we calculate an optimal choice (given suitable constraints/requirements) accounting for randomness in the data (e.g. measurement error).

Note, in the following document, I follow the convention that all vectors are by default column vectors, and $\mathbf{X} \in R^{n\times(d+1)}$ i.e. is formed by vertical concatenation of $\mathbf{x}_i^T$.

1.2 Linear Regression

The goal of linear regression is to predict the value of some output, $y \in R$ given some input data, $\mathbf{x} \in R^d$. In practice, these data take the form of a set of labelled pairs, $D := \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ which we use to estimate some function $f(\mathbf{x}_i)$ that aims to approximate the output y_i. To do this, we assume a linear model of the form $f(\mathbf{x}; \mathbf{w}) = \langle \mathbf{w}, \mathbf{x}\rangle + w_0$ and define a loss function $L = \sum_{i\in D_0}[y_i - f(\mathbf{x}_i; \mathbf{w})]^2$ (where $D_0 \subset D$ is defined as the 'training set').

This reduces the problem of estimating f to the problem of estimating values of $\mathbf{w}_{LS}$ such that $\mathbf{w}_{LS} := \text{argmin}_{\mathbf{w}} \sum_{i=0}^n (y_i - f(\mathbf{x}_i; \mathbf{w}))^2$.

By solving this minimization problem we find that

$$\mathbf{w}_{LS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

. (See A.1.1 for proof).

We note that this solution is only valid for cases where $n > d$ since for $n < d$ the problem is undetermined and we cannot find a unique solution for the coefficients.

The choice of loss function in this procedure feels quite arbitrary. Fortunately, we can derive it by considering the problem from a probabilistic viewpoint:

- Assume y_i is conditionally Normal with variance σ^2, i.e. that $p(y_i \mid \mathbf{x}_i, \mathbf{w}) = N_{y_i}(f(\mathbf{x}_i; \mathbf{w}), \sigma^2)$.
- Assume all pairs of data $(\mathbf{x}_i, y_i)$ are IID, s.t. $p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{i=1}^n N_{y_i}(f(\mathbf{x}_i, \mathbf{w}), \sigma^2)$.

To compute the optimal values of $\mathbf{w}$ we choose to maximise the probability of the data given the parameters (known as the *likelihood*). This is known as *Maximum Likelihood Estimation* (MLE). In actuality, we maximize the log-likelihood as it transforms the product of probabilities into a sum of log-probabilities.

It turns out that if we carry out this procedure we reproduce the least-squares formulation of the problem and hence the same solution, $\mathbf{w}_{LS}$. We also note that by following this logic we can also make a MLE of the conditional variance: $\sigma_{ML}^2 = \frac{1}{n}[y - f(\mathbf{x}; \mathbf{w}_{ML})]^2$ and therefore we can get a view of the uncertainty of our predictions simultaneously.

Using the Normal distribution to model $p(y \mid \mathbf{x}, \mathbf{w}, \sigma)$ can sometimes be a bad idea, for instance when

- The target variable is discrete, not continuous, e.g. integer counts. A possible solution would be the Poisson distribution.

- The (conditional) target distribution is not unimodal. Possible solution would be a Gaussian mixture model.

1.2.1 Feature transforms

The formulation of standard least squares regression assumes that the relationship between y and $\mathbf{x}$ is linear. If this is not the case, we can extend the model to allow polynomial transformations of our features. For example, consider a map $\mathbf{\Phi}(\mathbf{x}) : R^d \to R^{db}$ that transforms an input in the folowing way $\mathbf{\Phi}(x) := [x, x^2, x^3, ..., x^b]$. We can then reformulate all of the above with a new (still linear) model $g(\mathbf{x}; \mathbf{w}) = \langle \mathbf{w}, \mathbf{\Phi}(\mathbf{x}) \rangle + w_0$ and derive the corresponding solution $\mathbf{w}_{LS} = (\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^T \mathbf{y}$ in the same way as the usual LS solution.

Note that if $\mathbf{\Phi}(\mathbf{X})$ is symmetric ($\mathbf{\Phi}^T = \mathbf{\Phi}$) and invertible, then the solution reduces to $\mathbf{\Phi}(\mathbf{X})^{-1}\mathbf{y}$ (A.1.2).

Feature transforms can be powerful ways to capture non-linearity in a linear modelling framework and can be extended beyond the example here with interaction terms (e.g. $\mathbf{x}_1\mathbf{x}_2$). They have several downsides, however, in that they increase the risk of overfitting, increase the computational complexity of the calculation of $\mathbf{w}_{LS}$ (see A.1.3) and are more likely to suffer from the ‘curse of dimensionality’.

1.3 Overfitting

When we build a statistical model, we desire it to have the property of *generalization*, i.e. it can predict unseen data with as much accuracy as the training data. We can easily see this behaviour by measuring the error in our training and test sets simultaneously as we increase the number of parameters in our model. We see that we can achieve arbitrarily low training error with sufficient parameters, but our test error will have a turning point. The region after this turning point is where the model is considered to be *overfitting*.

1.3.1 Regularization

If we are solving a problem that requires the expressiveness of a high parameter model, we can attempt to mitigate the overfitting problem by *regularizing*. We implement this by adding a penalty term to the loss function to constrain the values of the coefficients. This penalty term usually takes the form of the L_1 norm $\|\mathbf{x}\|_1$ or squared Euclidean (L_2) norm $\|\mathbf{x}\|_2^2$. For L_2 we carry out the new optimization: $\mathbf{w}_{LS-R} = \text{argmin}_{\mathbf{w}} \sum_{i \in D} [y_i - f(\mathbf{x}_i; \mathbf{w})]^2 + \lambda \mathbf{w}^T \mathbf{w}$ to find the new solution

$$\mathbf{w}_{LS-R} = (\mathbf{\Phi}^T \mathbf{\Phi} + \lambda \mathbf{I})^{-1} \mathbf{\Phi}^T \mathbf{y}$$

(Appendix A.1.4). As λ increases, coefficient values must shrink and this in turn reduces overfitting. Note that if we make λ *too* large we will end up *underfitting* our model and losing too much expressiveness.

1.3.2 Cross-validation

In the previous section we introduced a parameter λ that determined the degree of regularization applied. In order to pick a suitable value for this parameter that neither over nor underfits our data, we can tune it via a procedure known as *cross-validation* that can be defined as the following process, designed to optimise some parameter $\theta \in R$ given a model $f(\theta)$ and loss function $L(\theta)$.

```
Algorithm 1: K-fold Cross-Validation
1. Split dataset D into K disjoint sets $D_1, ..., D_K$
2. Define a search space for $\theta$, e.g. a logarithmically spaced grid.
3. for $i{=}1{:}K$ do
       Fit $f(\mathbf{X}^{(i)}_{train}; \theta)$ with $\mathbf{X}^{(i)}_{train} = \{\mathbf{x} \in \bigcup_{j \neq i}^{K} D_j\}$
       Compute $L(\mathbf{X}_{test}; \theta)$ with $\mathbf{X}_{test} = \{\mathbf{x} \in D_i\}$
4. Select $\theta$ such that $\sum_i L_i(\theta)/K$ is minimized.
```

Cross-validation requires the data to be IID to be valid. Note that choosing the number of folds K should be dictated by the bias-variance trade-off of our data and model.

1.3.3 Curse of Dimensionality

The *Curse of Dimensionality* refers to the fact that many statistical problems have high-dimensional data and that the number of points needed to solve the problem can increase exponentially as the number of dimensions increases. For example, consider a polynomial transform similar to that mentioned earlier, but including symmetric cross terms up to order d. We find in this case that the dimension of our transformed dataset grows from $R^d \to R^{db+\binom{d}{1}+\binom{d}{2}+\cdots+\binom{d}{d}}$ and we know that $\binom{d}{1} + \binom{d}{2} + \cdots + \binom{d}{d} = 2^d$; hence exponential growth.

1.3.4 Probalistic regression

We saw that we can address overfitting with regularization constraints added to our loss function. This is generally considered the ‘frequentist’ approach. Alternatively we can reformulate the regression problem in terms of inferring the posterior probability distribution of the parameters given the data, using Bayes theorem. i.e. $p(\mathbf{w} \mid D) = \frac{p(D \mid \mathbf{w})p(\mathbf{w})}{p(D)}$. If we decide to maximise this probability w.r.t $\mathbf{w}$ we can obtain a point estimate solution as we did for the regularized version and indeed we find that the resulting solution (known as the *Maximum A Posteriori* (MAP)) is equivalent to the (L_2) regularized solution, with $\lambda = \frac{\sigma^2}{\sigma^2_{\mathbf{w}}}$. (Appendix A.1.5)

We can do better than this, however, and calculate the predictive distributon $p(\hat{y} \mid \mathbf{x}, D) = \int p(\hat{y} \mid \mathbf{x}, \mathbf{w}) p(\mathbf{w} \mid D) d\mathbf{w}$ by marginalizing out the parameter $\mathbf{w}$. What we find (A.1.6), is that

$$p(\hat{y} \mid \mathbf{x}, D) = N_{\hat{y}} \left[f(\mathbf{x}; \mathbf{w}_{LS-R}), \sigma^2 + \mathbf{\Phi} \sigma^2 \left(\mathbf{\Phi}^T \mathbf{\Phi} + \frac{\sigma^2}{\sigma^2_{\mathbf{w}}} \mathbf{I} \right)^{-1} \mathbf{\Phi}^T \right]$$

.

We can use this to calculate expected outcomes and their associated variance, i.e. $E_{p(\hat{y} \mid \mathbf{x}, D)}[\hat{y} \mid \mathbf{x}]$ and $\text{var}_{p(\hat{y} \mid \mathbf{x}, D)}[\hat{y} \mid \mathbf{x}]$, giving us a measure of the uncertainty in our predictions again.

1.4 Binary classification

When we have a task that involves making discrete decisions, we call it a *classification* problem. For tasks with 2 possible outputs, we label our output data as $y \in \{+1, -1\}$. In this scenario, we are really trying to estimate a function $f(\mathbf{x})$ that separates our space into two regions R_+ and R_- with a *decision boundary* defined by $f(\mathbf{x}) = 0$. To estimate $f(\mathbf{x})$ we again need to define a suitable error function to minimize. In this case, we can use false positive and false negative points, i.e. points we have incorrectly labelled as (correspondingly) +1 when they should be -1 and vice versa.

1.4.1 Bayes optimal classifier

By defining the probability of making false positive or false negative mistakes as $P(\mathbf{x} \text{ is FP or FN} \mid f) = \int_{R_+} p(\mathbf{x}, y = -1) d\mathbf{x} + \int_{R_-} p(\mathbf{x}, y = +1) d\mathbf{x}$ we can derive a *Bayes optimal classifier*, $f^*(\mathbf{x}) = p(\mathbf{x}, y = +1) - p(\mathbf{x}, y = -1)$ which minimizes this quantity, as derived in the Appendix A.1.7. Note that this is an optimal solution *in principal* but in practice it is not necessarily straightforward to obtain the joint distribution $p(\mathbf{x}, y)$.

1.4.2 Risk minimization

In real problems, we may find that we want to weight the significance of false positives and false negatives differently, depending on their real-world implication (e.g. misdiagnosis of a patient). In this case, we can define a *loss matrix* that encodes our view of the relative importance of these outcomes. We can then calculate the expected loss of making a wrong decison as $\text{argmin}_{y_0} E_{p(y \mid \mathbf{x})}[L(y, y_0) \mid \mathbf{x}]$.

To actually compute these things we need to estimate $p(y \mid \mathbf{x})$. In practice, this means we use the calculable object $p(y \mid \mathbf{x}, D)$ instead and estimate it one of two ways:

Discriminative - we estimate $p(y \mid \mathbf{x}, D)$ directly from $p(y \mid \mathbf{x}, \mathbf{w})$. This method aims purely to directly solve the problem of estimating the probability of y given the data.

Generative - we try and estimate $p(\mathbf{x}|y, D)$ and use that $p(y|\mathbf{x}, D) \propto p(\mathbf{x}|y, D)p(y)$ and that we can just use an empirical estimate of $p(y)$ (i.e. the actual proportion of labelled classes in D). This method allows one to simulate(/generate) new data $\mathbf{x}_{new}$ given a label y.

For cases where it's particularly important to avoid misclassifications, we can implement a rejection rule as a means of avoiding making uncertain classifications by disgarding data within some threshold region where the probabilities of belonging to the separate classes is similar.

For regression problems with continuous outputs we must replace the loss matrix with a loss function. With a squared loss like we used for LS, we can show that $\hat{y} = E_{p(y \mid \mathbf{x})}[y]$ (A.1.8), i.e. the mean of the distribution. If we were to change our loss function to the absolute value risk, $L(y, y_0) := |y - y_0|$ we would instead find that the solution is instead the *median* value of $p(y \mid \mathbf{x})$ (A.1.9).

Appendices

A Appendix

A.1 Linear Regression

A.1.1 Linear LS Regression solution

Prove $\mathbf{w}_{LS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

$$\mathbf{w} := \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=0}^{n} (y_i - f(\mathbf{x}_i; \mathbf{w}))^2$$

$$\text{given that } f(\mathbf{x}; \mathbf{w}) := \langle \mathbf{w}, \mathbf{x} \rangle + w_0$$

First, we rewrite the problem in matrix notation, with the assumption that all vectors are by default column vectors, and $\mathbf{X} \in R^{n\times(d+1)}$ i.e. is formed by vertical concatenation of $\mathbf{x}_i^T$ Then we have

$$\begin{aligned} \underset{\mathbf{w}}{\operatorname{argmin}}\ & (\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w}) \\ & = \mathbf{y}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - \mathbf{w}^T\mathbf{X}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\mathbf{w} \end{aligned}$$

Transpose of a scalar does nothing:

$$= \mathbf{y}^T\mathbf{y} + \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{y}$$

Differentiate both sides w.r.t $\mathbf{w}$:

$$\partial_{\mathbf{w}} : 0 = 0 + 2\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{X}^T\mathbf{y}$$

$$\implies \mathbf{w}_{LS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

A.1.2 Linear LS solution with symmetric feature transform

Prove $\mathbf{w}_{LS} = [\mathbf{\Phi}(\mathbf{X})]^{-1}\mathbf{y}$ if $\mathbf{\Phi}(\mathbf{X})$ is symmetric and invertible.

A matrix $\mathbf{X}$ is symmetric if $\mathbf{X}^T = \mathbf{X}$. A matrix $\mathbf{X}$ is invertible, if $\det \mathbf{X} \neq 0$. Since we know the $\mathbf{w}_{LS}$ solution already, we can substitute these properties:

$$\begin{aligned} \mathbf{w}_{LS} &= (\mathbf{\Phi}(\mathbf{X})^T\Phi(\mathbf{X}))^{-1}\mathbf{\Phi}(\mathbf{X})^T\mathbf{y} \\ &= (\mathbf{\Phi}(\mathbf{X})\mathbf{\Phi}(\mathbf{X}))^{-1}\mathbf{\Phi}(\mathbf{X})\mathbf{y} \\ &= \mathbf{\Phi}(\mathbf{X})^{-1}\mathbf{\Phi}(\mathbf{X})^{-1}\mathbf{\Phi}(\mathbf{X})\mathbf{y} \\ &= \mathbf{\Phi}(\mathbf{X})^{-1}\mathbf{y} \end{aligned}$$

A.1.3 Computational complexity of feature transforms

If we increase b of $\phi(\mathbf{x})$ by 2-fold, by how many folds will the computation time of $\mathbf{w}_{LS}$ increase?

$$
\begin{aligned}
\mathbf{\Phi}(\mathbf{X}) &\in R^{n\times(b+1)} \\
O(\mathbf{\Phi}(\mathbf{X})^T\mathbf{\Phi}(\mathbf{X})) &\sim O(n(b+1)^2) \sim O(nb^2) \\
\mathbf{C} &= \mathbf{\Phi}(\mathbf{X})^T\mathbf{\Phi}(\mathbf{X}) \\
O(\mathbf{C}^{-1}) &\sim b^3 \\
O(\mathbf{C}^{-1}\mathbf{\Phi}(\mathbf{X})^T) &\sim 2nb^2 + b^3 \\
b &\to 2b \\
O(\mathbf{C}^{-1}\mathbf{\Phi}(\mathbf{X})^T) &\to 8nb^2 + 8b^3
\end{aligned}
$$

Since computational complexity depends on both dimensions of $\mathbf{X}$ due to matrix multiplication operations, the total increase will depend on the relative size of n and b. For example, consider the following cases:

$$n \sim b \implies O(3b^3) \to O(16b^3) \implies > 5 \text{ fold} \tag{1}$$

$$n \sim 2b \implies O(5b^3) \to O(24b^3) \implies \sim 5 \text{ fold} \tag{2}$$

$$n \sim 10b \implies O(21b^3) \to O(88b^3) \implies \sim 4 \text{ fold} \tag{3}$$

As $\frac{n}{b}$ grows, the increase tends to 4. So for any reasonable dataset, you'd expect the increase to be in the range of $4 - 5$.

A.1.4 Regularized LS

Prove $\mathbf{w}_{LS-R} = (\mathbf{\Phi}^T\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^T\mathbf{y}$

$$
\begin{aligned}
L &= (\mathbf{y} - \mathbf{\Phi}\mathbf{w})^T(\mathbf{y} - \mathbf{\Phi}\mathbf{w}) + \lambda\mathbf{w}^T\mathbf{w} \\
&= \mathbf{y}^T\mathbf{y} + \mathbf{w}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w} - 2\mathbf{w}^T\mathbf{\Phi}^T\mathbf{y} + \lambda\mathbf{w}^T\mathbf{w} \\
\partial_{\mathbf{w}}L &= 2\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w} - 2\mathbf{\Phi}^T\mathbf{y} + 2\lambda\mathbf{w} \\
0 &= (\mathbf{\Phi}^T\mathbf{\Phi} + \lambda\mathbf{I})\mathbf{w} - \mathbf{\Phi}^T\mathbf{y} \\
\implies \mathbf{w}_{LS-R} &= (\mathbf{\Phi}^T\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^T\mathbf{y}
\end{aligned}
$$

A.1.5 Maximum A Posteriori solution

Prove $\mathbf{w}_{MAP} = \mathbf{w}_{LS-R}$ with $\lambda = \frac{\sigma^2}{\sigma^2_{\mathbf{w}}}$

$$
\begin{aligned}
\mathbf{w}_{MAP} &= \underset{\mathbf{w}}{\operatorname{argmax}}\, p(\mathbf{w} \mid D) \\
&= \underset{\mathbf{w}}{\operatorname{argmax}} \prod_{i \in D} N_{y_i}(f(\mathbf{\Phi}_i; \mathbf{w}), \sigma^2) \cdot N_{\mathbf{w}}(0, \sigma_{\mathbf{w}}^2 \mathbf{I}) \\
L &= \log p(\mathbf{w} \mid D) \\
&= -\log Z - \log Z_{\mathbf{w}} - \frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{\Phi}\mathbf{w})^T(\mathbf{y} - \mathbf{\Phi}\mathbf{w}) - \frac{1}{2\sigma_{\mathbf{w}}^2}\mathbf{w}^T\mathbf{w} \\
\partial_{\mathbf{w}} L = 0 &= \frac{1}{\sigma^2}\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w} - \frac{1}{\sigma^2}\mathbf{\Phi}^T\mathbf{y} + \frac{1}{\sigma_{\mathbf{w}}^2}\mathbf{w} \\
\implies \mathbf{w}_{MAP} &= \left(\mathbf{\Phi}^T\mathbf{\Phi} + \frac{\sigma^2}{\sigma_{\mathbf{w}}^2}\mathbf{I}\right)^{-1} \mathbf{\Phi}^T\mathbf{y}
\end{aligned}
$$

where Z denotes the normalisation constants of all the Likelihood distributions combined and $Z_{\mathbf{w}}$ denotes the prior Likelihood normalisation constant. Both terms vanish when taking the derivative so are not necessary to specify.

For these solutions (MAP and LS-R) we also find that the solution is now still valid for underdetermined systems where $n < d$.

A.1.6 Full probabilistic approach

Prove

$$\int p(\hat{y} \mid \mathbf{x}; \mathbf{w}) p(\mathbf{w} \mid D) d\mathbf{w} = N_{\hat{y}} \left[f(\mathbf{x}; \mathbf{w}_{LS-R}), \sigma^2 + \mathbf{\Phi}\sigma^2 \left(\mathbf{\Phi}^T\mathbf{\Phi} + \frac{\sigma^2}{\sigma_{\mathbf{w}}^2}\mathbf{I}\right)^{-1} \mathbf{\Phi}^T \right]$$

First calculate $p(\mathbf{w} \mid D)$ and demonstrate that it is another Normal distribution.

$$
\begin{aligned}
p(\hat{y} \mid \mathbf{x}; \mathbf{w}) &= N_{\hat{y}}(f(\mathbf{x}; \mathbf{w}), \sigma^2) \\
p(\mathbf{w} \mid D) &= \prod_{i \in D} N_y(f(\mathbf{x}; \mathbf{w}), \sigma^2) \cdot N_{\mathbf{w}}(0, \sigma_{\mathbf{w}}^2 \mathbf{I})
\end{aligned}
$$

$$p(\mathbf{w} \mid D) = \frac{1}{Z} \exp\left[-\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{\Phi}\mathbf{w})^2 + \frac{1}{2\sigma_{\mathbf{w}}^2}\mathbf{w}^T\mathbf{w} \right]$$

where Z is the normalisation factor of all the probabilities combined together

Multiplying out the exponent quadratics we get:

$$-\frac{1}{2\sigma^2}(\mathbf{y}^T\mathbf{y} + \mathbf{w}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w} - 2\mathbf{w}^T\mathbf{\Phi}^T\mathbf{y}) - \frac{1}{2\sigma_{\mathbf{w}}^2}\mathbf{w}^T\mathbf{w}$$

and combining terms in $\mathbf{w}^T\mathbf{w}$ and $\mathbf{w}$:

$$-\frac{1}{2\sigma^2}(\mathbf{y}^T\mathbf{y}) + \frac{1}{2\sigma^2}\mathbf{w}^T \left[\mathbf{\Phi}^T\mathbf{\Phi} + \frac{\sigma^2}{\sigma_{\mathbf{w}}^2} \right] \mathbf{w} + \frac{1}{\sigma^2}\mathbf{w}^T\mathbf{\Phi}^T\mathbf{y}$$

Now, defining $\mathbf{\Sigma} = \sigma^2 \left[\mathbf{\Phi}^T\mathbf{\Phi} + \frac{\sigma^2}{\sigma_{\mathbf{w}}^2} \right]^{-1}$ and $\mathbf{b} = \frac{1}{\sigma^2}\mathbf{\Phi}^T\mathbf{y}$ to simplify the expression and completing the square for $\mathbf{w}$:

$$-\frac{1}{2}(\mathbf{w}-\boldsymbol{\Sigma}\mathbf{b})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\Sigma}\mathbf{b})-\mathbf{b}^T\boldsymbol{\Sigma}\mathbf{b}$$

which is clearly a Gaussian in $\mathbf{w}$, so we have that

$$p(\mathbf{w}\mid D)=N_{\mathbf{w}}\left(\frac{1}{\sigma^2}\boldsymbol{\Sigma}\mathbf{b},\boldsymbol{\Sigma}\right)=N_{\mathbf{w}}\left(\left[\boldsymbol{\Phi}^T\boldsymbol{\Phi}+\frac{\sigma^2}{\sigma^2_{\mathbf{w}}}\right]^{-1}\boldsymbol{\Phi}^T\mathbf{y},\sigma^2\left[\boldsymbol{\Phi}^T\boldsymbol{\Phi}+\frac{\sigma^2}{\sigma^2_{\mathbf{w}}}\right]^{-1}\right)$$

Now we can use the relations given in [1], (equations 2.113, 2.114 and 2.115) which states the following relations:

$$\begin{aligned}p(\mathbf{x})&=N(\boldsymbol{\mu},\boldsymbol{\Lambda}^{-1})\\ p(y\mid\mathbf{x})&=N(\mathbf{A}\mathbf{x}+\mathbf{b},\mathbf{L}^{-1})\\ p(y)&=\int p(y\mid\mathbf{x})p(\mathbf{x})d\mathbf{x}\\ &=N(\mathbf{A}\boldsymbol{\mu}+\mathbf{b},\mathbf{L}^{-1}+\mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^T)\end{aligned}$$

We can associate each of these quantities with the quantities we already have, as follows:

$$\begin{aligned}p(\mathbf{x})&=p(\mathbf{w}\mid D)\\ p(y\mid\mathbf{x})&=p(\hat{y}\mid\mathbf{x},\mathbf{w})\\ p(y)&=p(\hat{y}\mid\mathbf{x},D)\end{aligned}$$

Hence, comparing the mean and variance for each of the distributions, we get that

$$\begin{aligned}\boldsymbol{A}&=\boldsymbol{\Phi}\\ \boldsymbol{b}&=0\\ \boldsymbol{L}^{-1}&=\sigma^2\\ \boldsymbol{\mu}&=\boldsymbol{\Sigma}\boldsymbol{\Phi}^T\mathbf{y}\\ \boldsymbol{\Lambda}^{-1}&=\sigma^2\boldsymbol{\Sigma}\end{aligned}$$

Putting these all together, we get that

$$\begin{aligned}p(\hat{y}\mid\mathbf{x},D)&=N_{\hat{y}}(\boldsymbol{\Phi}\boldsymbol{\Sigma}\boldsymbol{\Phi}^T\mathbf{y},\sigma^2+\sigma^2\boldsymbol{\Phi}\boldsymbol{\Sigma}\boldsymbol{\Phi}^T)\\ &=N_{\hat{y}}\left[f(\mathbf{x};\mathbf{w}_{LS-R}),\sigma^2+\boldsymbol{\Phi}\sigma^2\left(\boldsymbol{\Phi}^T\boldsymbol{\Phi}+\frac{\sigma^2}{\sigma^2_{\mathbf{w}}}\mathbf{I}\right)^{-1}\boldsymbol{\Phi}^T\right]\end{aligned}$$

Notice that given a Normally distributed Likelihood and prior, we obtain a Normally distributed posterior and Normal predictive distribution.

A.1.7 Bayes optimal classifier

$$P(\mathbf{x}\text{ is FP or FN}\mid f)=\int_{R_+}p(\mathbf{x},y=-1)d\mathbf{x}+\int_{R_-}p(\mathbf{x},y=+1)d\mathbf{x}$$

Prove $P(\text{FP or FN}\mid f)=(P(error))$ is minimised when $f(\mathbf{x})=p(\mathbf{x},y=+1)-p(\mathbf{x},y=-1)$

$f(\mathbf{x})$ is effectively defined by the location we will call $\mathbf{u}$ where the function is zero. This is because it defines a boundary and we only care about the sign of the function in the regions either side of $\mathbf{u}$. Therefore, demonstrating the conditions at which the $P(error)$ is minimised in terms of $\mathbf{u}$, equivalently tells us a condition for $f(\mathbf{x})$ to minimise $P(error)$.

$$\begin{aligned}R_+ &= \{\mathbf{x} \in R \mid f(\mathbf{x}) \geq 0\}\\ R_- &= \{\mathbf{x} \in R \mid f(\mathbf{x}) \leq 0\}\end{aligned}$$

$$\begin{aligned}f(\mathbf{u}) &= 0\\ P(error) &= \int_R p(\mathbf{x}, y = +1) \cdot H(\mathbf{u}) + p(\mathbf{x}, y = -1)(1 - H(\mathbf{u}))d\mathbf{x}\\ \partial_{\mathbf{u}} P(error) &= \int_R p(\mathbf{x}, y = +1) \cdot \delta(\mathbf{u}) + p(\mathbf{x}, y = -1)(1 - \delta(\mathbf{u})d\mathbf{x}\\ 0 &= p(\mathbf{u}, y = +1) - p(\mathbf{u}, y = -1)\\ &\implies p(\mathbf{u}, y = +1) = p(\mathbf{u}, y = -1)\end{aligned}$$

where $H(x)$ is the Heaviside step function, and $\partial_x H(x) := \delta(x)$ where $\delta(x)$ is the Dirac delta function.

A.1.8 Optimal prediction with squared loss

$$\begin{aligned}L(y, y_0) = (y - y_0)^2&\\ \hat{y} &= \underset{y_0}{\operatorname{argmin}} E_{p(y \mid \mathbf{x})}[L(y, y_0) \mid \mathbf{x}]\\ \hat{y} &= \underset{y_0}{\operatorname{argmin}} E_{p(y \mid \mathbf{x})}[(y - y_0)^2 \mid \mathbf{x}]\end{aligned}$$

Prove $\hat{y} = E_{p(y \mid \mathbf{x})}[y]$

$$\begin{aligned}&\underset{y_0}{\operatorname{argmin}} E_{p(y \mid \mathbf{x})}[(y - y_0)^2 \mid \mathbf{x}]\\ &= \underset{y_0}{\operatorname{argmin}} E_{p(y \mid \mathbf{x})}[y^2 + y_0^2 - 2yy_0 \mid \mathbf{x}]\\ &= \partial_{y_0} \int p(y \mid \mathbf{x})[y^2 + y_0^2 - 2yy_0 \mid \mathbf{x}]dy\\ 0 &= \int p(y \mid \mathbf{x})[y_0 - y]dy\\ \implies \hat{y} &= \int p(y \mid \mathbf{x})ydy = E_{p(y \mid \mathbf{x})}[y]\end{aligned}$$

A.1.9 Absolute value risk function

$$\begin{aligned} L(y, y_0) &= |y - y_0| \\ \hat{y} &= \underset{y_0}{\operatorname{argmin}} E_{p(y \mid \mathbf{x})}[L(y, y_0) \mid \mathbf{x}] \\ \hat{y} &= \underset{y_0}{\operatorname{argmin}} E_{p(y \mid \mathbf{x})}[|y - y_0| \mid \mathbf{x}] \end{aligned}$$

Prove $\hat{y} = \text{median}(p(y \mid \mathbf{x}))$
where median(.) is defined as the value m s.t. $\int_{-\infty}^{m} p(y \mid \mathbf{x}) dy = \int_{m}^{\infty} p(y \mid \mathbf{x}) dy$.

$$\begin{aligned} &\underset{y_0}{\operatorname{argmin}} E_{p(y \mid \mathbf{x})}[|y - y_0| \mid \mathbf{x}] \\ &= \partial_{y_0} \left[\int_{-\infty}^{y_0} p(y \mid \mathbf{x})(y - y_0) dy - \int_{y_0}^{\infty} p(y \mid \mathbf{x})(y - y_0) dy \right] \end{aligned}$$

where we have split the integral at the point $y = y_0$ and flipped the sign of the second integrand, since for that region $y_0 > y$.

Hence,

$$\begin{aligned} 0 &= -\int_{-\infty}^{y_0} p(y \mid \mathbf{x}) dy + \int_{y_0}^{\infty} p(y \mid \mathbf{x}) dy \\ \implies & \int_{-\infty}^{m} p(y \mid \mathbf{x}) dy = \int_{m}^{\infty} p(y \mid \mathbf{x}) dy \end{aligned}$$

for $m = \hat{y}$ the solution of the minimization problem.

References

[1] Christopher M Bishop. *Pattern recognition and machine learning.* Information science and statistics. Springer, New York, NY, 2006. Softcover published in 2016.