Chapter 4 Linear Models for Classification

Input vector $\mathbf{x} \in \mathbb{R}^D$

Goal: assign $\mathbf{x}$ to one of K discrete classes C_k, $k = 1 \ldots K$

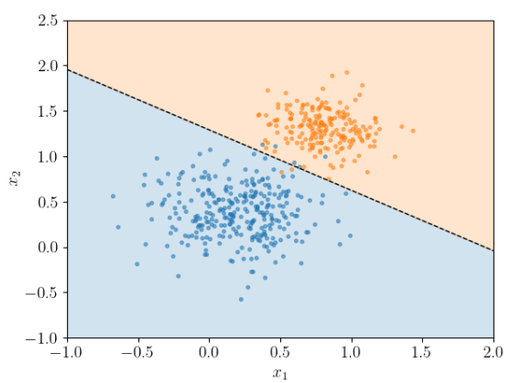
So the input space is divided into decision regions (R_k)

whose boundaries are called decision boundaries or decision surfaces

Linear models for classification mean that the decision boundaries

are linear functions of input vector $\mathbf{x}$

(i.e. $D-1$ dimensional hyperplane)



Probablistic models

Two class problem (binary representation)

single target variable $t \in \{0, 1\}$ s.t $t = 1$ represents class C_1 and $t = 0$ represents class C_2

The value of t can be interpreted as the probability that class is C_1

$K > 2$ classes problem (multiclass)

K-dimensional vector $\mathbf{t}$: one hot vector (1-of-K coding)

If the class is C_j, then all elements t_k of $\mathbf{t}$ are zero except t_j, $t_j = 1$. $(1,0,0)$, $(0,1,0)$,

Again we can interprete t_k as the probability that the class is C_k.

In the linear model, the model prediction $y(x, w)$ was given by a linear function of w.

For classification problem, we need to predict discrete class or more generally posterior probabilities.

⇒ generalize the model in which we transform the linear function of w using a nonlinear function $f(\cdot)$ so that

$$y(x) = f(w^T x + w_0)$$

$f(\cdot)$ is known as an 'activation function'. Its inverse is called a 'link function'

The decision boundaries correspond to $y(\mathbf{x}) =$ constant (i.e. $\mathbf{w}^T\mathbf{x} + w_0 =$ constant) and hence decision boundaries are linear function of $\mathbf{x}$.

In contrast to the models used for regression, classifications are not linear in $\mathbf{w}$ due to $f(\cdot)$

As regression models, we can use a fixed nonlinear transformation with a vector of basis functions $\boldsymbol{\phi}(\mathbf{x})$.

We begin by considering classification directly in the original input space $\mathbf{x}$.

4.1 Discriminant functions

Discriminant function takes $\mathbf{x}$ and assigns it one of K classes C_k

We restrict attention to linear discriminants (decision boundaries are hyperplanes)

4.1.1 Two classes

The simplest linear discriminant function

$$y(\mathbf{x}) := \mathbf{w}^T \mathbf{x} + w_0$$

where $\mathbf{w}$ is the weight vector and w_0 is a bias.

An input $\mathbf{x}$ is assigned to C_1 if $y(\mathbf{x}) \geq 0$ and

is assigned to C_2 if $y(\mathbf{x}) < 0$

$\Rightarrow$ decision boundary is defined by $y(\mathbf{x}) = 0$.

$$\| $$

$$\mathbf{w}^T \mathbf{x} + w_0$$

Arbitrary point $\mathbf{x}$ and let $\mathbf{x}_\perp$ be its orthogonal projection onto decision surface so that

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$$

where $r = y(\mathbf{x})/\|\mathbf{w}\|$

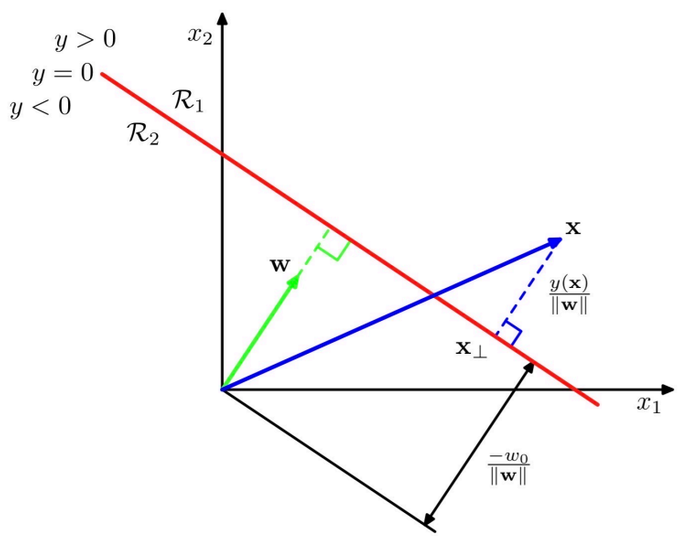


Figure 4.1 Illustration of the geometry of a linear discriminant function in two dimensions. The decision surface, shown in red, is perpendicular to $\mathbf{w}$, and its displacement from the origin is controlled by the bias parameter w_0. Also, the signed orthogonal distance of a general point $\mathbf{x}$ from the decision surface is given by $y(\mathbf{x})/\|\mathbf{w}\|$.

We can use dummy input $x_0 = 1$ and then define $\widetilde{w}_j := (w_0, w_j)$ and $\widetilde{x} := (x_0, x)$ so that

$$y(x) = \widetilde{w}_j^T \widetilde{x}$$

In this case the decision boundaries are D-dimensional hyperplanes passing through the origin of D+1 dim input space.

4.1.2 Multiple classes

Now consider the extension of linear discriminants to $K > 2$ classes

One-versus-the-rest classifier

Use $K-1$ classifiers each of which solves a two-class problem separating class C_k from other class

This method leads to regions that are ambiguously classified

One - versus - one classifier KC_2

Use $K(K-1)/2$ discriminant functions, one for every possible pair of classes.

This too run into the problem of ambiguous regions.

We need too many classifiers

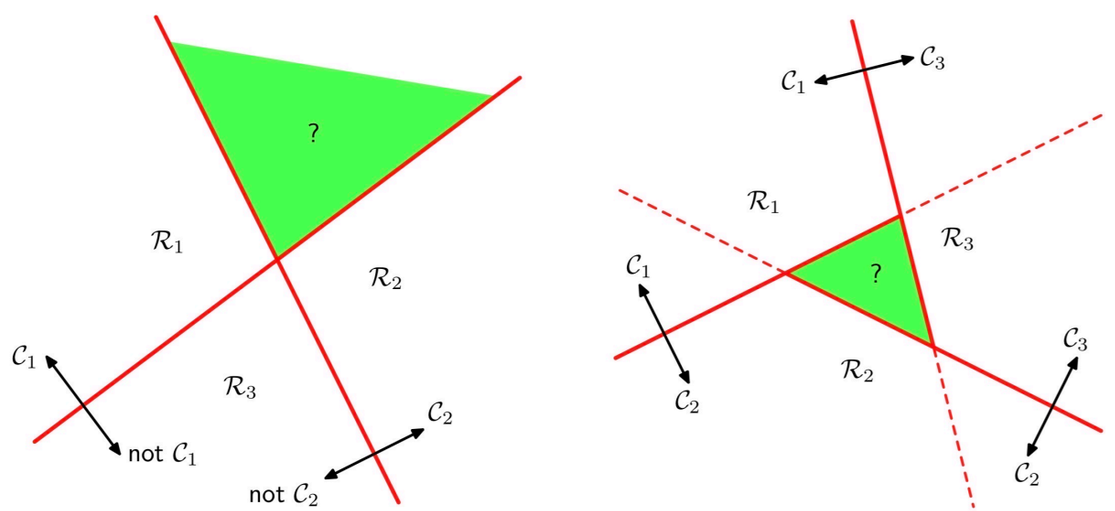


Figure 4.2 Attempting to construct a K class discriminant from a set of two class discriminants leads to ambiguous regions, shown in green. On the left is an example involving the use of two discriminants designed to distinguish points in class $\mathcal{C}_k$ from points not in class $\mathcal{C}_k$. On the right is an example involving three discriminant functions each of which is used to separate a pair of classes $\mathcal{C}_k$ and $\mathcal{C}_j$.

Consider a single K-class discriminant comprising K linear functions of the form

$$y_k(\mathbf{x}) := \mathbf{w}_k^T \mathbf{x} + w_{k0}$$

$\mathbf{w}_k \in \mathbb{R}^D$ weight vector

$w_{k0} \in \mathbb{R}$ bias

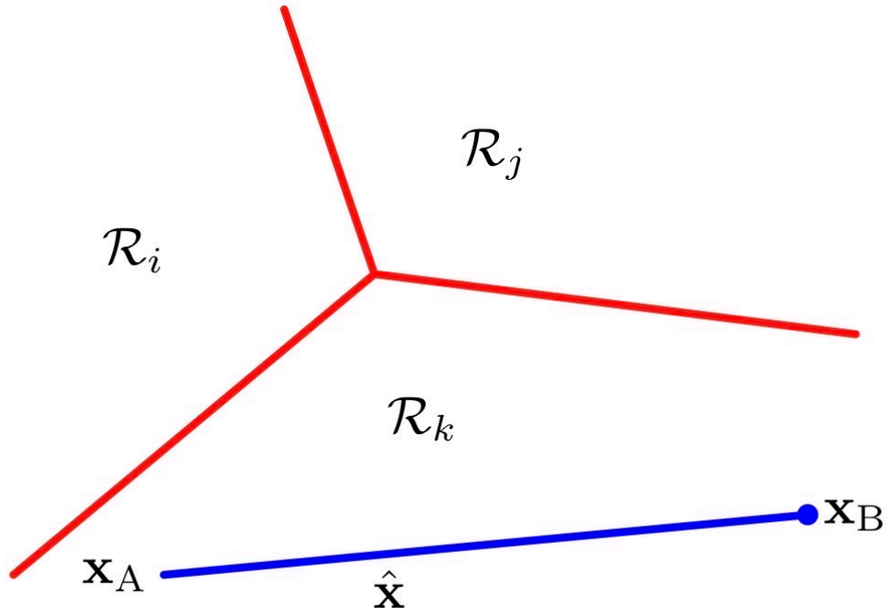
Assign a point $\mathbf{x}$ to class C_k if $y_k(\mathbf{x}) > y_j(\mathbf{x}) \quad \forall j \neq k$.

So the decision boundary between C_k and C_j is given by

$$(\mathbf{w}_k - \mathbf{w}_j)^T \mathbf{x} + (w_{k0} - w_{j0}) = 0.$$

i.e. $D-1$ dimensional hyperplane.

Figure 4.3 Illustration of the decision regions for a multiclass linear discriminant, with the decision boundaries shown in red. If two points $\mathbf{x}_\mathrm{A}$ and $\mathbf{x}_\mathrm{B}$ both lie inside the same decision region $\mathcal{R}_k$, then any point $\widehat{\mathbf{x}}$ that lies on the line connecting these two points must also lie in $\mathcal{R}_k$, and hence the decision region must be singly connected and convex.



Decision regions are singly connected and convex.

Let x_A and x_B be in R_k. Let $\hat{x}$ be as

$$\hat{x} := \lambda x_A + (1-\lambda) x_B \qquad 0 \le \lambda \le 1$$

Since the discriminant functions are linear, we obtain

$$y_k(\hat{x}) = \lambda y_k(x_A) + (1-\lambda) y_k(x_B)$$

Because x_A and x_B lie inside R_k, $y_k(x_A) > y_j(x_A)$ and $y_k(x_B) > y_j(x_B)$ $\forall j \neq k$, hence $y_k(\hat{x}) > y_j(\hat{x})$ $\forall j \neq k$

4.1.3 Least squares for classification

Consider a classification problem with K classes with 1-of-K scheme for the target vector $\mathbf{t}$.

The minimization of SSE function is the method that it approximates the conditional expectation $\mathbb{E}[\mathbf{t} \mid \mathbf{x}]$ of the target values given the input $\mathbf{x}$.

Each class C_k, $k = 1, 2, \ldots K$

$$y_k(x) = w_k^T x + w_{k0} = (\tilde{w}_k^T \tilde{x})$$

We can group these linear models using vector notation

$$y(x) := \tilde{W}^T \tilde{x} = \begin{pmatrix} y_1(x) \\ \vdots \\ y_K(x) \end{pmatrix} \quad K\text{-dim}$$

예측값

where $\tilde{x}$ is the augmented input vector $(1, x)^T$ and

$(D+1) \times K$

$$\tilde{W} = (\tilde{w}_1, \tilde{w}_2, \ldots, \tilde{w}_K) \quad , \quad \tilde{w}_K = (w_{k0}, w_k)^T$$

Input $\mathbf{x}$ is assigned to the class for which the output

$y_k = \tilde{\mathbf{w}}_k^T \tilde{\mathbf{x}}$ $(= \mathbf{w}_k^T \mathbf{x} + w_{k0})$ is largest

Determine the parameter matrix $\tilde{\mathbf{W}}$ by minimizing a SSE.

Consider a training data set $\{\mathbf{x}_n, \mathbf{t}_n\}$ $n = 1, \ldots N$

$\mathbf{x}_n \in \mathbb{R}^D$, $\mathbf{t}_n \in \mathbb{R}^k$ (one hot vector)

Define a matrix $\mathbf{T}$ ($N \times K$) whose n^{th} row is the vector $\mathbf{t}_n^T$

" $\tilde{\mathbf{X}}$ ($N \times (D+1)$) whose n^{th} row is the vector $\tilde{\mathbf{x}}_n^T = (1, \mathbf{x}_n)^T$

$$SSE = \frac{1}{2} \sum_{n=1}^{N} \| \underbrace{\tilde{\mathbf{W}}^T \tilde{\mathbf{x}}_n}_{\mathbf{y}(\mathbf{x}_n)} - \mathbf{t}_n \|^2$$

예측 vector, 실제 target

SSE function can be written as

$$E_D(\tilde{W}) = \frac{1}{2} Tr\{(\tilde{X}\tilde{W} - T)^T (\tilde{X}\tilde{W} - T)\}$$

Set the gradient w.r.t $\tilde{W}$ to zero vector. So we obtain the minimizing solution of $E_D(\tilde{W})$ for $\tilde{W}$ as follows

$$\tilde{W} = \underline{(\tilde{X}^T \tilde{X})^{-1} \tilde{X}^T} T = \tilde{X}^+ T$$

where $\tilde{X}^+$ is the pseudo-inverse of $\tilde{X}$. The discriminant function is given by

$$y(x) = \tilde{W}^T \tilde{x} = T^T (\tilde{X}^+)^T \tilde{x}$$

W : the parameter matrix whose k^{th} column is w_k

X : the matrix whose n^{th} row is x_n^T

Then, $$E_D(\tilde{W}) = \frac{1}{2} \mathrm{Tr}\left\{ (X W + \mathbb{1} w_0^T - T)^T (X W + \mathbb{1} w_0^T - T) \right\}$$

where k-dim $\mathbb{1} := (1, 1, .. 1)^T$, $w_0 := (w_{10}, w_{20}, .. w_{k0})^T$

Calculate the derivative of $E_D(W)$ w.r.t w_0

$$\nabla_{w_0} E_D(\tilde{W}) = \underset{k\times 1}{2N w_0} + \underset{N\times k}{2 (\underset{N\times D}{X} \overset{D\times k}{W} - T)^T} \underset{k\times 1}{\mathbb{1}}$$

$N \times k$

$k \times N$

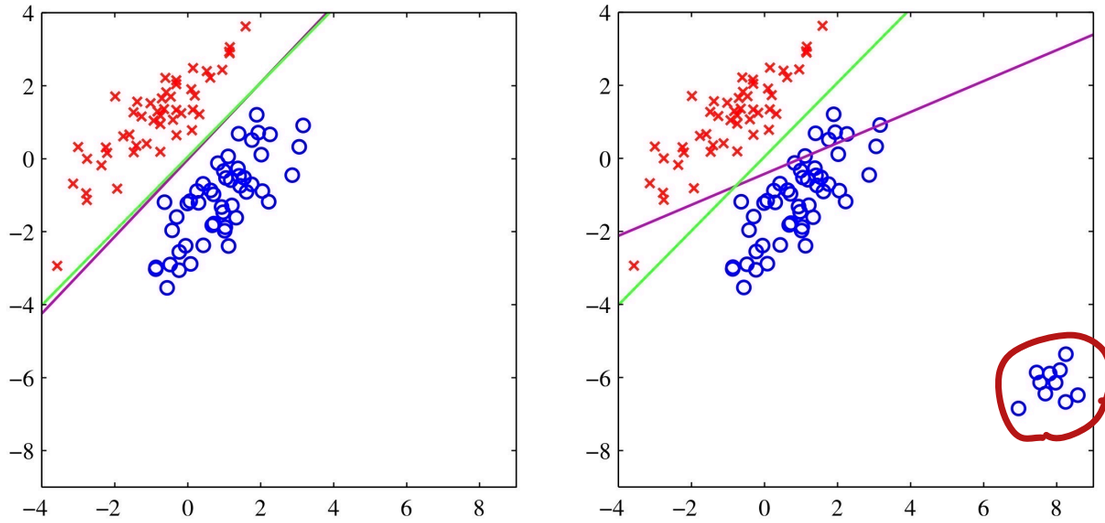
(See later !)

We have obtained the discriminant function using least square approach.

$$y(\mathbf{x}) = \tilde{\mathbf{W}}^T \tilde{\mathbf{x}} = \mathbf{T}^T (\tilde{\mathbf{X}}^\dagger)^T \tilde{\mathbf{x}}$$

where $\tilde{\mathbf{X}}^\dagger = (\tilde{\mathbf{X}}^T \tilde{\mathbf{X}})^{-1} \tilde{\mathbf{X}}^T$.

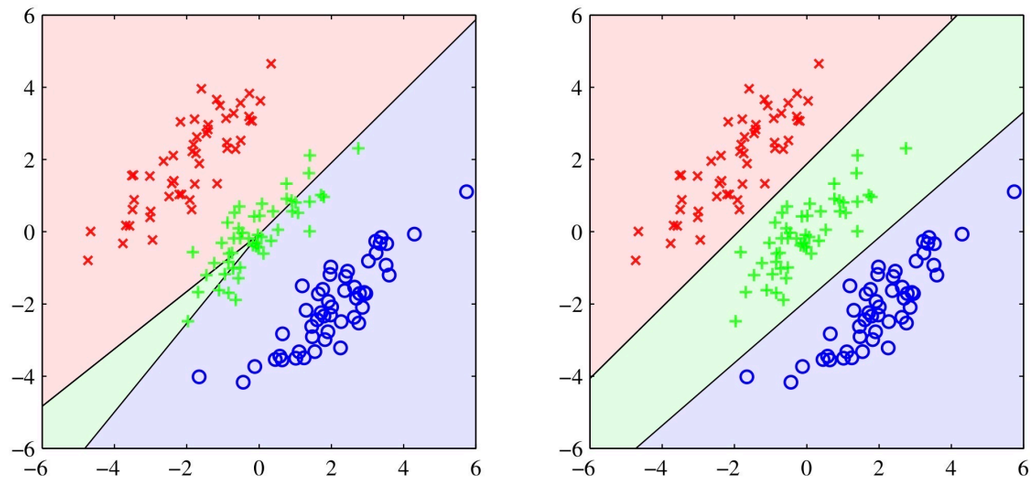
This discriminant function does not have any probabilistic interpretation and is not robust to outliers (least square)



SSE penalizes predictions that are too correct in that such points (far from boundary)

Figure 4.4 The left plot shows data from two classes, denoted by red crosses and blue circles, together with the decision boundary found by least squares (magenta curve) and also by the logistic regression model (green curve), which is discussed later in Section 4.3.2. The right-hand plot shows the corresponding results obtained when extra data points are added at the bottom left of the diagram, showing that least squares is highly sensitive to outliers, unlike logistic regression.

Least square approach



Logistic regression

Figure 4.5 Example of a synthetic data set comprising three classes, with training data points denoted in red ($\times$), green ($+$), and blue ($\circ$). Lines denote the decision boundaries, and the background colours denote the respective classes of the decision regions. On the left is the result of using a least-squares discriminant. We see that the region of input space assigned to the green class is too small and so most of the points from this class are misclassified. On the right is the result of using logistic regressions as described in Section 4.3.2 showing correct classification of the training data.

4.1.4 Fisher's Linear discriminant

Consider a linear classification in terms of dimensionality reduction

Input $\mathbf{x} \in \mathbb{R}^D$.

Consider a projection to one dimension using

$$y := \mathbf{w}^T \mathbf{x}$$

Threshold w_0 on y. So if $y \geq w_0$ then $\mathbf{x}$ is classified as class C_1 otherwise class C_2

Considerable loss of information and overlapping in one dimension

Goal: determine w or select projection maximizing the class separation.

Consider a two classes problem with N_1 points of class C_1 and N_2 points of class of C_2

The mean vectors of the two classes

$$m_1 := \frac{1}{N_1} \sum_{n \in C_1} x_n \quad , \quad m_2 := \frac{1}{N_2} \sum_{n \in C_2} x_n$$

First, choose w to maximize the difference of projected means

$$m_2 - m_1 = w^T (m_2 - m_1) \quad \text{where} \quad m_k := w^T m_k$$

We constrain w to have unit length, i.e. $\sum_i w_i^2 = 1$

Using a Lagrange multiplier, we find

$$w \propto (m_2 - m_1) \qquad \text{(see Fig 4.6)}$$

Second, consider a small variance within each class

The within-class variance of the transformed (projected) data from class C_k is given by

$$s_k^2 := \sum_{n \in C_k} (y_n - m_k)^2$$

where $y_n = w^T x_n$ and $m_k := w^T m_k$

The Fisher criterion : maximize

$$J(\mathbf{w}) := \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \tag{4.26}$$

between-class variance

total within-class variance

$$J(\mathbf{w}) = \frac{\| \mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1) \|^2}{\sum_{n \in C_1} (\mathbf{w}^T \mathbf{x}_n - m_1)^2 + \sum_{n \in C_2} (\mathbf{w}^T \mathbf{x}_n - m_2)^2}$$

$$\| \mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1) \|^2 = [\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1)] [\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1)]^T = \mathbf{w}^T \mathbf{S}_B \mathbf{w}$$

where $\mathbf{S}_B := (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T$ (4.27)

$D \times 1$ $1 \times D$

$$s_1^2 + s_2^2 = \sum_{n \in C_1} [w_1^T (x_n - m_1)]^2 + \sum_{n \in C_2} [w_1^T (x_n - m_2)]^2$$

$$= w_1^T S_w^1 w_1 + w_1^T S_w^2 w_1 = w_1^T S_w w_1$$

where $S_w^k := \sum_{n \in C_k} (x_n - m_k)(x_n - m_k)^T, \quad k = 1, 2$

$$S_w := \sum_{C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{C_2} (x_n - m_2)(x_n - m_2)^T \qquad (4.28)$$

Thus $J(w_1) = \dfrac{w_1^T S_B w_1}{w_1^T S_w w_1}$

S_B : between-class covariance matrix

S_w : within-class covariance matrix

w_1 D-dim

1×D D×D D×1

Differentiating $J(w)$ w.r.t w, we found $J(w)$ is maximized when

$$(w^T S_B w)\, S_w\, w = (w^T S_w\, w)\, S_B\, w \qquad (4.29)$$

($w^T S_B w$: scalar; $w^T S_w w$: scalar)

We have used $\nabla_w (w^T S w) = w^T (S + S^T)$.

By (4.27) (def of S_B), $S_B w$ is in direction of $(m_2 - m_1)$

$$(m_2 - m_1)\underbrace{(m_2 - m_1)^T w}_{1\times D \quad D\times 1}$$

$$(m_2 - m_1)(m_2 - m_1)^T =: S_B$$

And drop the scalar factors $(w^T S_B w)$ and $(w^T S_w w)$

Multiplying both sides of (4.29) by S_W^{-1}, we then obtain

$$w \propto S_W^{-1}(m_2 - m_1) \qquad (4.30)$$

Note that if within-class covariance is isotropic ($S_W = \lambda I$) then solution w is proportional to $m_2 - m_1$

(4.30) is known as Fisher's linear discriminant. This is the direction for projection of the data down to 1-dimension.

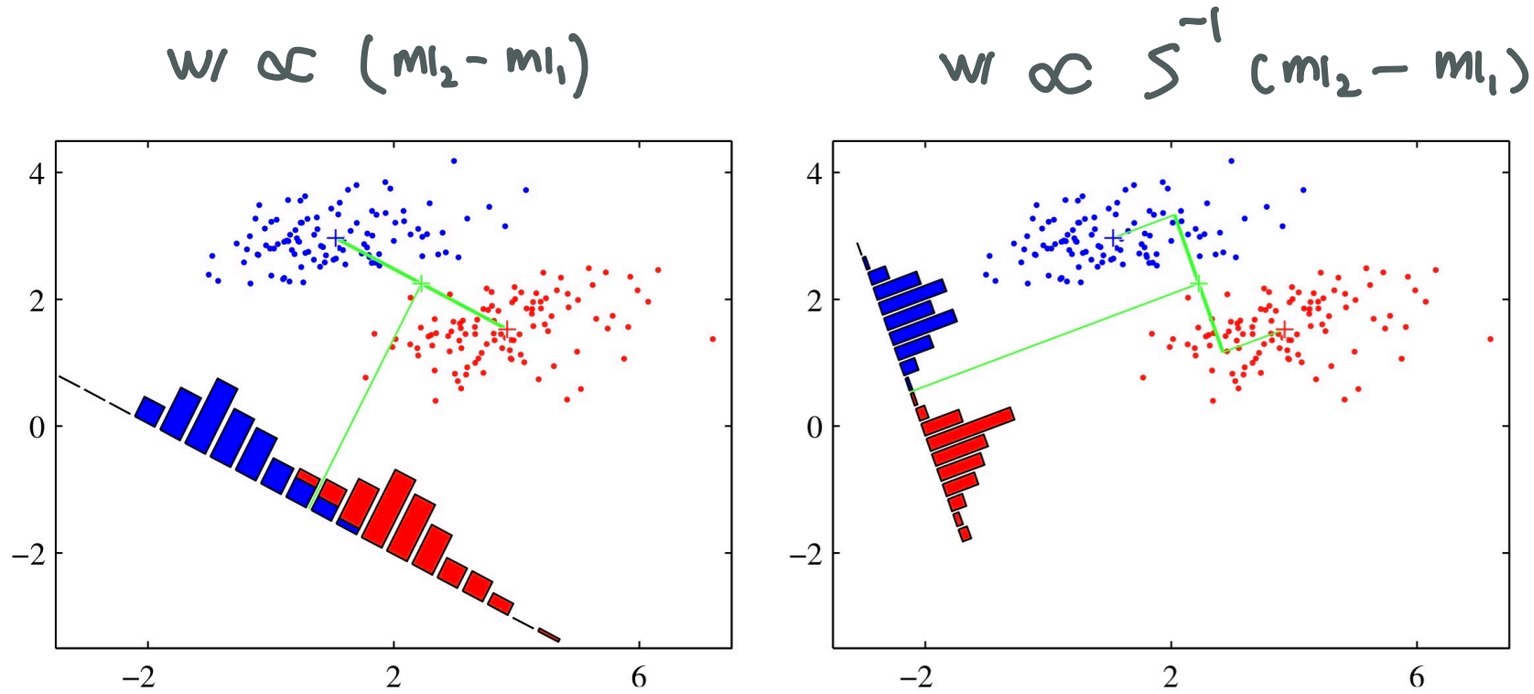


Figure 4.6 The left plot shows samples from two classes (depicted in red and blue) along with the histograms resulting from projection onto the line joining the class means. Note that there is considerable class overlap in the projected space. The right plot shows the corresponding projection based on the Fisher linear discriminant, showing the greatly improved class separation.

4.1.5 Relation to least squares

Two approaches of linear discriminants for two-class problem

The least squares make the model predictions as close as possible as to a set of target values.

$$\text{minimize} \quad \sum_{n=1}^{N} \| \underbrace{W^T \tilde{x}_n}_{y(x_n)} - t_n \|^2$$

The Fisher criterion was derived by requiring maximum class separation in the 1-dim output space

Let us see the relationship between these two approaches.

We will show that the Fisher criterion can be obtained as a special case of least squares.

Let N_1 (resp N_2) be # of patterns in class C_1 (resp. C_2)

Take the target values for class C_1 to be N/N_1

This target value approximates the reciprocal (inverse) of the prior probability for class C_1.

For class C_2, take the targets to be $-N/N_2$

The sum-of-squares error function can be written

$$E = \frac{1}{2} \sum_{n=1}^{N} (\mathbf{w}^T \mathbf{x}_n + w_0 - t_n)^2$$

$\mathbf{x}_n \in \mathbb{R}^D$, $\mathbf{w} = (w_1 \dots w_D)^T$

Set the derivative of E w.r.t w_0 and $\mathbf{w}$ to zero,

$$\frac{\partial E}{\partial w_0} = \sum_{n=1}^{N} (\mathbf{w}^T \mathbf{x}_n + w_0 - t_n) = 0$$

$$\nabla_{\mathbf{w}} E = \sum_{n=1}^{N} (\mathbf{w}^T \mathbf{x}_n + w_0 - t_n) \mathbf{x}_n = 0 \qquad (4.33)$$

Thus,

$$w_0 = -w_1^T m_1 \qquad (4.34)$$

where m_1 is the mean of total input data set and is given by

$$m_1 = \frac{1}{N} \sum_{n=1}^{N} x_n = \frac{1}{N} (N_1 m_1 + N_2 m_2) .$$

To obtain (4.34) we have used

$$\sum_{n=1}^{N} t_n = N_1 \frac{N}{N_1} - N_2 \frac{N}{N_2} = 0$$

(4.33) can be written (Excercise 4.6)

$$\left(S_w + \frac{N_1 N_2}{N} S_B \right) w = N (m_1 - m_2)$$

where S_w is defined by (4.28) and S_B is defined by (4.27).

Since $S_B w$ is always in direction of $(m_2 - m_1)$, we can write

$$w \propto S_w^{-1} (m_2 - m_1)$$

where we have ignored irrelevant scale factors.

We have also found an expression for the bias value $w_0 = w^T m_1$. It means that x is classified as belonging to class C_1 if $y(x) = w^T (x - m_1) > 0$.

4.1.6 Fisher's discriminant for multiple classes ($K > 2$)

WLOG, assume $D > K$

The generalization of within-covariance matrix to the case of K classes follows from (4.28) to give

$$S_W = \sum_{k=1}^{K} S_k \qquad \text{(input space)}$$

where

$$S_k := \sum_{n \in C_k} (x_n - m_k)(x_n - m_k)^T$$

$$m_k = \frac{1}{N_k} \sum_{n \in C_k} x_n$$

where N_k is # of patterns in C_k

Consider the total covariance matrix

$$S_T := \sum_{n=1}^{N} (x_n - m)(x_n - m)^T \qquad \text{(input space)}$$

where m is the mean of the total data set.

This S_T can be decomposed into the sum of the within-class covariance matrix S_W and an additional matrix S_B

$$S_T = S_W + S_B$$

We identify S_B as a measure of the between-class covariance

$$S_B = \sum_{k=1}^{K} N_k (m_k - m)(m_k - m)^T \quad (4.46) \qquad \text{(input space)}$$

Next we introduce $D' > 1$ linear 'features' $y_k := w_k^T x$ where $k = 1, \ldots D'$. (D-dim weight w_k)

The weight vectors $\{w_k\}$ can be considered to be the columns of a matrix W ($D \times D'$) so that

$$y = W^T x \qquad D' \text{-dim}$$

$$D \xrightarrow{W^T} D'$$

Now define similar matrices in the project D'-dim y-space

$$S_W = \sum_{k=1}^{K} \sum_{n \in C_k} (y_n - \mu_k)(y_n - \mu_k)^T \qquad (D' \times D' \text{ matrix})$$

and

$$S_B = \sum_{k=1}^{K} N_k (\mu_k - \mu)(\mu_k - \mu)^T \qquad (D' \times D' \text{ matrix})$$

where

$$\mu_k := \frac{1}{N_k} \sum_{n \in C_k} y_n \,, \qquad \mu = \frac{1}{N} \sum_{k=1}^{K} N_k \mu_k = \frac{1}{N} \sum_{n=1}^{N} y_n$$

To determine W, we need to define a scalar (benefit) which is large when S_B is large and when S_W is small.

Consider

$$J(W) := \mathrm{Tr}\{S_W^{-1} S_B\}$$

feature space

This criterion can be written as an explicit function in the form

$$J(W) = \mathrm{Tr}\{(W^T S_W W)^{-1} (W^T S_B W)\}$$

$D' \times D$ $D \times D$ $D \times D'$

input space

Remark

- From (4.46) (def of S_B), S_B is the sum of K matrices and each of which is of rank 1.
- Because of the definition of m_1, only (K-1) of these matrices are independent
- Thus, S_B has rank at most (K-1) and so there are at most (K-1) eigenvalues.
- So the projection onto the (K-1) dim subspace spanned by the eigenvectors of S_B does not change $J(W)$. More than (K-1) linear features are meaningless

4.1.7 The Perceptron algorithm (linear discriminant model)

Two-class classification

$\mathbf{x}$ input vector

$\phi(\mathbf{x})$ its feature vector for a fixed nonlinear function $\phi(\cdot)$.

Linear model of the form

$$y(\mathbf{x}) := f(\mathbf{w}^T \phi(\mathbf{x}))$$ parameter vector $\mathbf{w}$

where the nonlinear activation function $f(\cdot)$ is given by a step function of the form

$$f(a) := \begin{cases} 1 & a \geq 0 \\ -1 & a < 0 \end{cases}$$

Here $\Phi(x)$ include a bias component $\phi_0(x) = 1$.

For the perceptron,

target values $t = 1$ for C_1, $t = -1$ for C_2

How to determine w ?

How to define error function of w ?

Misclassification rate ?

Perceptron criterion

Idea: if x_n is in class C_1, then $w^T \phi(x_n) > 0$ ($t_n = 1$)

" " " C_2, then $w^T \phi(x_n) < 0$ ($t_n = -1$)

Using $t \in \{-1, 1\}$ target coding, we are seeking w s.t

$$w^T \phi(x_n) t_n > 0$$

The perceptron criterion is given by

$$E_p(w) := -\sum_{n \in \mathcal{M}} w^T \phi(x_n) t_n \qquad (4.54)$$

where $\mathcal{M}$ denotes the set of all misclassified patterns.

So the total error function is piecewise linear for $\mathbf{w}$.

If $\mathbf{x}$ is correctly classified then the contribution to the error is zero.

The stochastic gradient descent algorithm to this error

$$\mathbf{w}^{(\tau+1)} := \mathbf{w}^{(\tau)} - \eta \nabla E_p(\mathbf{w}) = \mathbf{w}^{(\tau)} + \eta \boldsymbol{\phi}(\mathbf{x}_n) t_n$$

where η is the learning rate parameter. (put $\eta = 1$)

If the pattern is correctly classified then $\mathbf{w}$ remains unchanged.

In case it is incorrectly classified.

for C_1, we add $\phi(\mathbf{x}_n)$ onto the current estimate of $\mathbf{w}$

while for C_2, we subtract $\phi(\mathbf{x}_n)$ from $\mathbf{w}$.

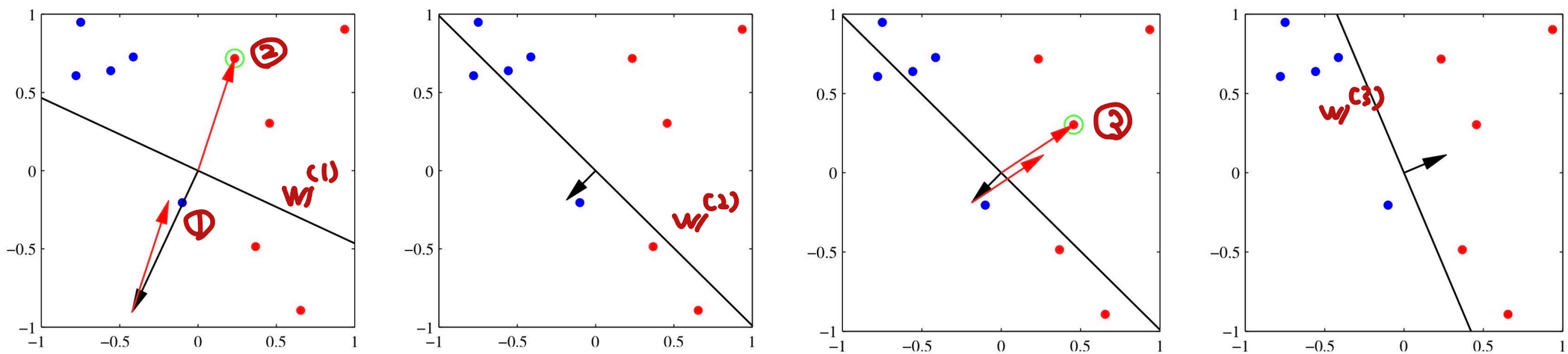


Figure 4.7 Illustration of the convergence of the perceptron learning algorithm, showing data points from two classes (red and blue) in a two-dimensional feature space (ϕ_1, ϕ_2). The top left plot shows the initial parameter vector $\mathbf{w}$ shown as a black arrow together with the corresponding decision boundary (black line), in which the arrow points towards the decision region which classified as belonging to the red class. The data point circled in green is misclassified and so its feature vector is added to the current weight vector, giving the new decision boundary shown in the top right plot. The bottom left plot shows the next misclassified point to be considered, indicated by the green circle, and its feature vector is again added to the weight vector giving the decision boundary shown in the bottom right plot for which all data points are correctly classified.

Remark

- In view of (4.54) and (4.55), the contribution to the error from a misclassified pattern will be reduced

$$-\mathbf{w}^{(\tau+1)T}\boldsymbol{\phi}(\mathbf{x}_n)t_n = -\mathbf{w}^{(\tau)T}\boldsymbol{\phi}(\mathbf{x}_n)t_n - (\boldsymbol{\phi}(\mathbf{x}_n)t_n)^T\boldsymbol{\phi}(\mathbf{x}_n)t_n$$

single component of E_P

$$< -\mathbf{w}^{(\tau)T}\boldsymbol{\phi}(\mathbf{x}_n)t_n$$

where we have set $\eta = 1$ and used $\|\boldsymbol{\phi}(\mathbf{x}_n)t_n\|^2 > 0$

- This does not the contribution to the error function from all misclassified patterns (other)

- The change in w_r may have caused some previously correctly classified patterns to become misclassified.
- In case that the training data set is linearly separable, perceptron learning algorithm is guarrenteed to find an exact solution in a finite number of steps (by perceptron convergence theorem)
- Perceptron does not provide probabilistic output
- Can not generalize $K > 2$ classes
- based on linear combinations of fixed basis functions.

Linear models for classification

Discriminant function

- Least squared $y(x) = \tilde{W}^T \tilde{x}$
- Fisher $y(x) = W^T x$
- Perceptron $y(x) = f(w_i^T \phi(x))$

Probablistic models

- generative
- discriminative

$p(x|C_1)$ 가정

$\downarrow$

$p(x)$

$\leftarrow$

$p(C_1|x)$

4.2 Probabilistic Generative model

Discriminative and generative approaches to classification.

Consider the case of two classes.

$$p(C_1|x) = \frac{p(x|C_1)\,p(C_1)}{p(x|C_1)\,p(C_1) + p(x|C_2)\,p(C_2)} \tag{4.57}$$

$$= \frac{1}{1+\exp(-a)} =: \sigma(a)$$

logistic sigmoid

where we have defined

$$a = \ln \frac{p(x|C_1)\,p(C_1)}{p(x|C_2)\,p(C_2)} \tag{4.58}$$

Remark of sigmoid

- Bounded function
- Symmetry property $\sigma(-a) = 1 - \sigma(a)$
- The inverse of the logistic sigmoid is given by

$$a = \ln\left(\frac{\sigma}{1-\sigma}\right)$$ logit function

$K > 2$ classes.

$$p(C_k|x) = \frac{p(x|C_k)\,p(C_k)}{\sum_j p(x|C_j)\,p(C_j)} = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

softmax function

which is known as the normalized exponential (multiclass generalization of the logistic sigmoid. Here a_k are defined by

$$a_k := \ln(p(x|C_k)\,p(C_k)).$$

4.2.1 Continuous inputs

$x \in \mathbb{R}^D$ continuous vector

$p(x \mid C_k)$

Assume the class-conditional densities are Gaussian and all classes share the same covariance matrix. (only different mean vector)

I.e.

$$p(x \mid C_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_k)^T \Sigma^{-1} (x-\mu_k)\right\}$$

Here Σ is independent of class C_k.

Consider the case of two classes. From (4.57) and (4.58),

$$p(C_1|\mathbf{x}) = \sigma(\mathbf{w}_1^T\mathbf{x} + w_0)$$

결과

$$= \sigma(a)$$

$$a = \ln \frac{p(\mathbf{x}|C_1)\,p(C_1)}{p(\mathbf{x}|C_2)\,p(C_2)}$$

where we have defined

$$\mathbf{w}_1 := \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

$$w_0 := -\frac{1}{2}\boldsymbol{\mu}_1^T\Sigma^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^T\Sigma^{-1}\boldsymbol{\mu}_2 + \ln\frac{p(C_1)}{p(C_2)}$$

Because of the assumption of common covariance matrices, it becomes a linear function of $\mathbf{x}$ in the argument of the logistic sigmoid.

Thus, decision boundary ($\mathbf{x}$ s.t $p(C_k|\mathbf{x}) = c$) is a linear function of $\mathbf{x}$.

The prior $p(C_k)$ enter only through the bias parameter w_0.

For the general case of $K > 2$ classes under the assumption shared covariance matrix of $p(x \mid C_k)$

$$a_k(x) := w_k^T x + w_{k0}$$

$$p(C_k \mid x) = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

where we have defined

$$a_k = \ln\left(p(x \mid C_k)\, p(C_k)\right)$$

$$w_k := \Sigma^{-1}(\mu_k)$$

$$w_{k0} := -\frac{1}{2}\mu_k^T \Sigma^{-1} \mu_k + \ln p(C_k)$$

$a_k(x)$ is again linear function of x.

If each class-conditional density $p(\mathbf{x}|C_k)$ has its own covariance matrix Σ_k, then the cancellations of quadratic form of $\mathbf{x}$ will no longer occur.

So we obtain quadratic functions of $\mathbf{x}$ giving rise to a quaratic discriminant.

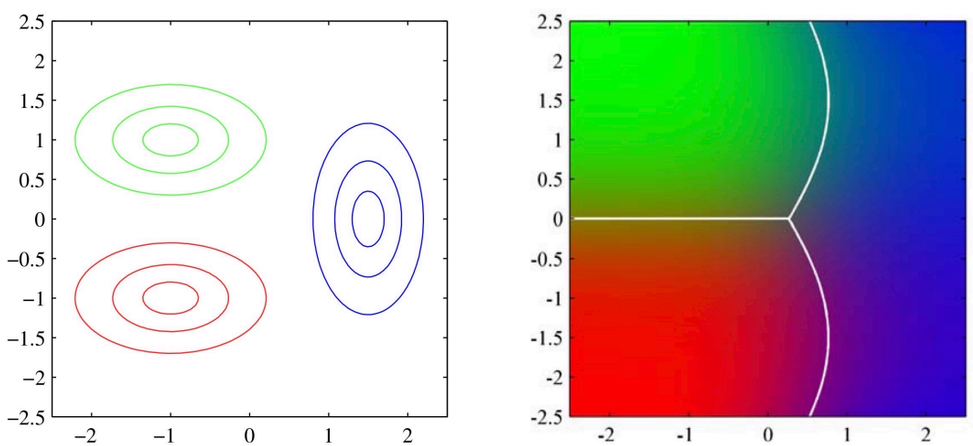


Figure 4.11 The left-hand plot shows the class-conditional densities for three classes each having a Gaussian distribution, coloured red, green, and blue, in which the red and green classes have the same covariance matrix. The right-hand plot shows the corresponding posterior probabilities, in which the RGB colour vector represents the posterior probabilities for the respective three classes. The decision boundaries are also shown. Notice that the boundary between the red and green classes, which have the same covariance matrix, is linear, whereas those between the other pairs of classes are quadratic.

4.2.2 Maximum likelihood solution

Two classes classification

Data set $\{x_n, t_n\}$, $n = 1 \dots N$. $x_n \in \mathbb{R}^D$, $t_n = 1$ or 0

$t_n = 1$ denotes class C_1 and $t = 0$ denotes class C_2

Gaussian class conditional density with a shared covariance matrix

Denote the prior class probability $p(C_1) = \pi$. so that $p(C_2) = 1 - \pi$.

x_n from class C_1, $t_n = 1$ hence

$$p(x_n, C_1) = p(C_1)\, p(x_n \mid C_1) = \pi\, \mathcal{N}(x_n \mid \mu_1, \Sigma)$$

Shared covariance

Similarly, for class C_2, $t_n = 0$

$$p(x_n, C_2) = p(C_2)\, p(x_n \mid C_2) = (1 - \pi)\, \mathcal{N}(x_n \mid \mu_2, \Sigma)$$

Thus the likelihood function is given by

$$p(\mathbf{t}, \mathbf{X} \mid \pi, \mu_1, \mu_2, \Sigma) = \prod_{n=1}^{N} [\pi \mathcal{N}(\mathbf{x}_n \mid \mu_1, \Sigma)]^{t_n} [(1-\pi)\mathcal{N}(\mathbf{x}_n \mid \mu_2, \Sigma)]^{1-t_n}$$

where $\mathbf{t} = (t_1, \ldots t_N)^T$, $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N)^T$

As usual, we maximize the log of the likelihood function.

Consider first π. The log likelihood function of π is

$$\sum_{n=1}^{N} \{ t_n \ln \pi + (1-t_n) \ln (1-\pi) \}$$

Setting the derivative w.r.t π equal to 0. So we obtain

$$\pi = \frac{1}{N}\sum_{n=1}^{N} t_n = \frac{N_1}{N} = \frac{N_1}{N_1 + N_2}$$

C_1 에 속할

where N_1 (resp. N_2) is # of points in C_1 (resp. C_2)

Thus MLE for π is simply the fraction of points in C_1

Now consider the maximization w.r.t μ_1.

The terms of loglikelihood function depending on μ_1

$$\sum_{n=1}^{N} t_n \ln \mathcal{N}(x_n \mid \mu_1, \Sigma) = -\frac{1}{2} \sum_{n=1}^{N} t_n (x_n - \mu_1)^T \Sigma^{-1} (x_n - \mu_1) + \text{constant}$$

Setting the derivative w.r.t μ_1 to 0, we obtain

$$\mu_1 = \frac{1}{N_1} \sum_{n=1}^{N} t_n x_n$$

which is simply the mean of vectors x_n assigned to C_1.

Similarly we can obtain the result for μ_2 as

$$\mu_2 = \frac{1}{N_2} \sum_{n=1}^{N} (1 - t_n) x_n$$

which again is the mean of vectors x_n assigned to C_2.

Finally, consider the MLE solution for Σ. Pick out the terms in the log likelihood function depending on Σ, we have

$$-\frac{1}{2} \sum_{n=1}^{N} t_n \ln |\Sigma| - \frac{1}{2} \sum_{n=1}^{N} t_n (x_n - \mu_1)^T \Sigma^{-1} (x_n - \mu_1)$$

$$-\frac{1}{2} \sum_{n=1}^{N} (1 - t_n) \ln |\Sigma| - \frac{1}{2} \sum_{n=1}^{N} (1 - t_n) (x_n - \mu_2)^T \Sigma^{-1} (x_n - \mu_2)$$

$$= -\frac{N}{2}\ln|\Sigma| - \frac{N}{2}\mathrm{Tr}\{\Sigma^{-1}S\}$$

where we have defined

$$S = \frac{N_1}{N}S_1 + \frac{N_2}{N}S_2$$

$$S_1 = \frac{1}{N_1}\sum_{n\in C_1}(x_n - \mu_{1,1})(x_n - \mu_{1,1})^T$$

$$S_2 = \frac{1}{N}\sum_{n\in C_2}(x_n - \mu_{1,2})(x_n - \mu_{1,2})^T$$

Using the standard result for MLE solution for a Gaussian distribution, we see that $\Sigma = S$

4.2.3 Discrete features

Consider the case of discrete feature value x_i

For simplicity, assume $x_i \in \{0, 1\}$ and D-dim vector $\mathbf{x}$

$$\mathbf{x} = (x_1, x_2, \ldots x_D)^T$$

Here we will make the naive Bayes assumption (the feature values are treated as independent, conditioned on C_k)

I.e. $p(x_1, x_2 \mid C_k) = p(x_1 \mid C_k)\, p(x_2 \mid C_k)$

Thus class-conditional distributions are given by

$$p(\mathbf{x} \mid C_k) = \prod_{i=1}^{D} p(x_i \mid C_k) = \prod_{i=1}^{D} \mu_{ki}^{x_i} (1 - \mu_{ki})^{1 - x_i} \qquad (4.81)$$

which contain D independent parameters for each class.

$$p(x \mid C_k) = \mu_k^{x} (1-\mu_k)^{1-x}$$

μ_k : C_k 라는 가정하에 $x = 1$ 일 확률

$x \in \{0, 1\}$

Substituting into (4.63) ($a_k = \ln(p(x \mid C_k) p(C_k))$)

$$a_k(x) = \sum_{i=1}^{D} \{ x_i \ln \mu_{ki} + (1-x_i) \ln(1-\mu_{ki}) \} + \ln p(C_k)$$

which are linear linear functions of x_i

4.3 Probabilistic Discriminative Models

Finding the parameters of a generalized linear model

Generative model (indirect) vs discriminative model (direct)

Generative model : Fitting class conditional densities $p(x|C_k)$ and class priors separately and then applying Baye's Theorem.

Discriminative model : Maximizing a likelihood function defined through the conditional distribution $p(C_k|x)$

Remarks of two approaches

4.3.1 Fixed basis functions

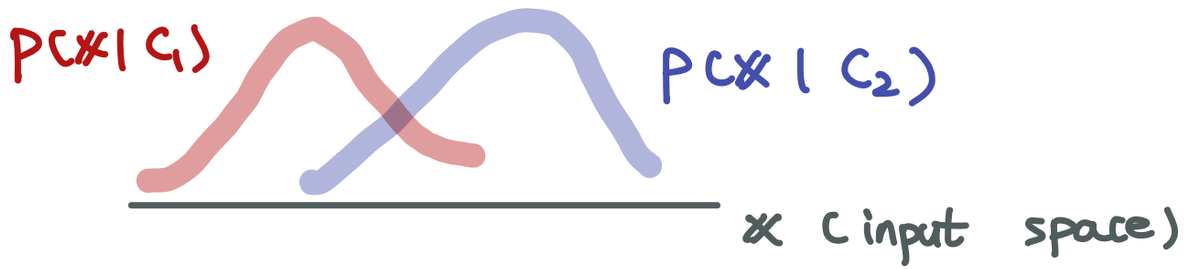
$\Phi(\cdot)$: vector of basis functions $\{\phi_0, \cdots \phi_{M-1}\}$. $\phi_0(x) = 1$

We make a fixed nonlinear transformation of the inputs.

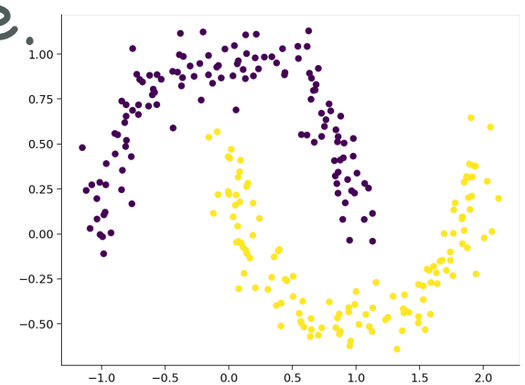
The resulting decision boundaries will be nonlinear in the original input x space (linear in the feature space)

We shall include a fixed basis function transformation $\Phi(x)$.

For many problems in the real world, there is a significant overlap between the class-conditional densities $p(x|C_k)$.



Note that nonlinear transformation cannot remove such class overlap. i.e. this transformation can make it possible to separate points that are not linearly separable. Suitable choices of nonlinearity can make the process of modelling the posterior probabilities easier.



4.3.2 Logistic regression

Two-class classification

In section 4.2 (4.57), we saw that under rather general assumptions, the posterior probability of class C_1 can be written as

$$p(C_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma(\mathbf{w}^T\boldsymbol{\phi})$$

with $p(C_2|\boldsymbol{\phi}) = 1 - p(C_1|\boldsymbol{\phi})$. Here $\sigma(\cdot)$ is the logistic sigmoid function and $\boldsymbol{\phi}$ is the feature vector i.e. $\boldsymbol{\phi} = \boldsymbol{\phi}(\mathbf{x})$

For M-dim feature space, this model has M adjustable parameters (w).

linearly

By contrast, Gaussian class conditional densities model using maximum likelihood method needs $2M$ parameters for mean vectors and $M(M+1)/2$ parameters for shared covariance matrix. Together with class prior this gives a total of $M(M+5)/2 + 1$ parameters

quadratically

Determine the parameters w of the logistic regression model.

Use maximum likelihood method.

For a data set $\{\phi_n, t_n\}$ where $t_n \in \{0, 1\}$ and $\phi_n = \phi(x_n)$ with $n = 1, 2, \ldots N$, the likelihood function can be written

$$p(\mathbf{t} \mid w) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n} \qquad t_n = 0 \text{ or } 1$$

where $\mathbf{t} := (t_1, t_2, \ldots t_N)^T$ and $y_n = p(C_1 \mid \phi) = \sigma(w^T \phi)$

$p(t_n \mid w) = y_n^{t_n} (1 - y_n)^{1-t_n}$, $t_n = 0$ or 1, its prediction y_n or $1 - y_n$ $\quad t_n = 1 \quad t_n = 0$

Negative logarithm of the likelihood which gives the cross-entropy error function in the form

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{ t_n \ln y_n + (1-t_n) \ln(1-y_n) \}$$

where $y_n = \sigma(a_n)$, $a_n = \mathbf{w}^T \boldsymbol{\phi}_n$ with $\boldsymbol{\phi}_n := \boldsymbol{\phi}(\mathbf{x}_n)$.

The gradient of the error function w.r.t $\mathbf{w}$ is given by

$$\nabla_{\mathbf{w}} E(\mathbf{w}) = \sum_{n=1}^{N} \underbrace{(y_n - t_n)}_{\text{error}} \boldsymbol{\phi}_n \qquad \text{basis vector} \qquad (4.91)$$

We have used $\frac{d\sigma}{da} = \sigma(1-\sigma)$

From (4.91), we can use a sequential algorithm. The weight vector $\mathbf{w}$ is updated in which ∇E_n is the n^{th} term in (4.91).

4.3.3 Iterated reweighted least squares

In the case of the linear regression model, MLE solution, on the assumption of a Gaussian noise model, leads to a closed-form solution.

For logistic regression, there is no longer a closed-form solution. However the error function $E(\mathbf{w})$ is convex. Hence there is a unique minimum

The error function can be minimized by an iterative technique based on the Newton - Raphson iterative optimization scheme.

(Fletcher 1987 ; Bishop and Nabney 2008)

$$\mathbf{w}^{(new)} = \mathbf{w}^{(old)} - \mathbf{H}^{-1} \nabla E(\mathbf{w})$$

where $\mathbf{H}$ is the Hessian matrix $\nabla\nabla E(\mathbf{w})$ w.r.t $\mathbf{w}$.

First, apply Newton - Raphson to the linear regression model

$$\nabla_{\mathbf{w}} E(\mathbf{w}) = \sum_{n=1}^{N} (\mathbf{w}^T \boldsymbol{\phi}_n - t_n) \boldsymbol{\phi}_n = \boldsymbol{\Phi}^T \boldsymbol{\Phi} \mathbf{w} - \boldsymbol{\Phi}^T \mathbf{t}$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \boldsymbol{\Phi}^T \boldsymbol{\Phi} \quad (\text{indep of } \mathbf{w})$$

where Φ is the $N \times M$ design matrix whose n^{th} row is given by Φ_n^T. So

$$\begin{aligned} w^{(new)} &= w^{(old)} - H^{-1} \nabla E(w^{(old)}) \\ &= w^{(old)} - (\Phi^T \Phi)^{-1} \{ \Phi^T \Phi w^{(old)} - \Phi^T t \} \\ &= (\Phi^T \Phi)^{-1} \Phi^T t \end{aligned}$$

is the standard least-squares solution. Since the SSE is the quadratic form of w, Newton-Raphson formula gives the exact solution in one step

Second, apply Newton - Raphson to the logistic regression model with cross - entropy error function

$$\nabla_{w} E(w) = \sum_{n=1}^{N} (y_n - t_n)\,\phi_n = \Phi^T (y - t)$$

$$H = \nabla\nabla E(w) = \sum_{n=1}^{N} y_n (1 - y_n)\,\phi_n \phi_n^T = \Phi^T R \Phi$$

where R is the $N \times N$ diagonal matrix with elements

$$R_{nn} = y_n (1 - y_n)$$

$y_n = \sigma(w^T \phi_n)$

Here H is not independent of w. So the error function is not quadratic form of w.

Since $y_n = \sigma(w^T \phi(x_n))$,

$$0 < y_n < 1 \quad \text{and} \quad u^T H u > 0 \quad \text{for } \forall u \in \mathbb{R}^M$$

So Hessian matrix H is positive definite. Hence the error function E is a convex function of w and $\exists!$ minimum

$$w^{(new)} = w^{(old)} - (\Phi^T R \Phi)^{-1} \Phi^T (y - t)$$

$$= (\Phi^T R \Phi)^{-1} \{ \Phi^T R \Phi w^{(old)} - \Phi^T (y - t) \}$$

$$= (\Phi^T R \Phi)^{-1} \Phi^T R z \qquad (4.99)$$

where z is the N-dimensional vector with

$$z := \Phi w^{(old)} - R^{-1}(y - t)$$

// Recall the MLE solution for w of the linear regression (least-squared)

$$w_{ML} = (\Phi^T \Phi)^{-1} \Phi^T t$$

//

(4.99) is the form of a set of normal equations for a weighted least-squares problem.

The weighting matrix R is not constant but depends on w.

So we must apply the normal equations iteratively.

For this reason, the algorithm is known as IRLS iterative reweighted least squares.

4.3.4 Multiclass logistic regression

In section 4.2, we discussed the generative models for multiclass classification. The posterior probabilities are given by a softmax transformation of linear functions of feature variables

$$p(C_k \mid \Phi) = y_k(\Phi) = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

where the activations a_k are given by

$$a_k = w_k^T \Phi \qquad \Phi = (\phi_0(x), \ldots \phi_{M-1}(x))^T$$

There we used MLE to determine separately the Gaussian class-conditional densities and the class priors and then found the corresponding posterior probabilities, thereby implicitly determining the parameters $\{w_k\}$

Here we consider the use of maximum likelihood to determine the parameters $\{w_k\}$ of this model directly

t_n : one hot vector , Φ_n : feature vector

The likelihood function is given by

$$p(T \mid w_1 \dots w_K) = \prod_{n=1}^{N} \prod_{k=1}^{K} p(C_k \mid \Phi)^{t_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}^{t_{nk}}$$

where $y_{nk} := y_k(\Phi_n)$ and T is an $N \times K$ matrix of target variables with elements t_{nk}

//

$$p(t \mid w_1, \dots w_K) = \prod_{k=1}^{K} p(C_k \mid \Phi)^{t_k}$$

t = one hot vector

이 weight vector 들로 결정 target t 가 나올 확률

For some fixed n, $\sum_k t_{nk} = 1$ $\sum_k y_{nk} = 1$ //

Taking the negative logarithm then gives

$$E(w_1, \ldots w_K) = -\ln p(T \mid w_1 \ldots w_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_{nk}$$

which is known as the cross-entropy error function for the multiclass classification problem

Note that the derivatives of y_k w.r.t all a_j.

$$\frac{\partial y_k}{\partial a_j} = y_k (I_{kj} - y_j) \qquad \left(y_k = \frac{\exp(a_k)}{\sum_j \exp(a_j)} \right)$$

where I_{kj} are the elements of identity matrix

We now take the gradient of the error function w.r.t one of the parameter vectors w_j.

$$\nabla_{w_j} E(w_1 \dots w_K) = \sum_{n=1}^{N} \underbrace{(y_{nj} - t_{nj})}_{\text{prediction error}} \underbrace{\phi_n}_{\text{basis function}} \qquad (4.109)$$

where we have used $\sum_k t_{nk} = 1$.

Note that we see the same form arising for the gradient as was found for SSE with linear regression and the cross-entropy error for the logistic regression model.

So we can use this to formulate a sequential algorithm.

In this case, each of the weight vectors is updated using

$$\mathbf{w}_j^{(\tau+1)} = \mathbf{w}_j^{(\tau)} - \eta \nabla E_n \qquad (3.22)$$

Now to find a batch algorithm, we appeal to the Newton-Raphson update to obtain the corresponding IRLS algorithm. The Hessian matrix that comprises blocks of size $MK \times MK$ (MK x MK) in which block j, k is given by

$$\nabla_{\mathbf{w}_j} \nabla_{\mathbf{w}_k} E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = \sum_{n=1}^{N} y_{nk}(I_{kj} - y_{nj}) \boldsymbol{\phi}_n \boldsymbol{\phi}_n^{\mathrm{T}}$$

This Hessian matrix for the multiclass logistic regression model is positive definite and so the error function again has a unique minimum.

4.3.5 Probit regression

Consider the two-class classification and the framework of generalized linear models so that

$$p(t=1 \mid a) = f(a)$$

where $a = \mathbf{w}^T \boldsymbol{\phi}$ and $f(\cdot)$ is the activation function.

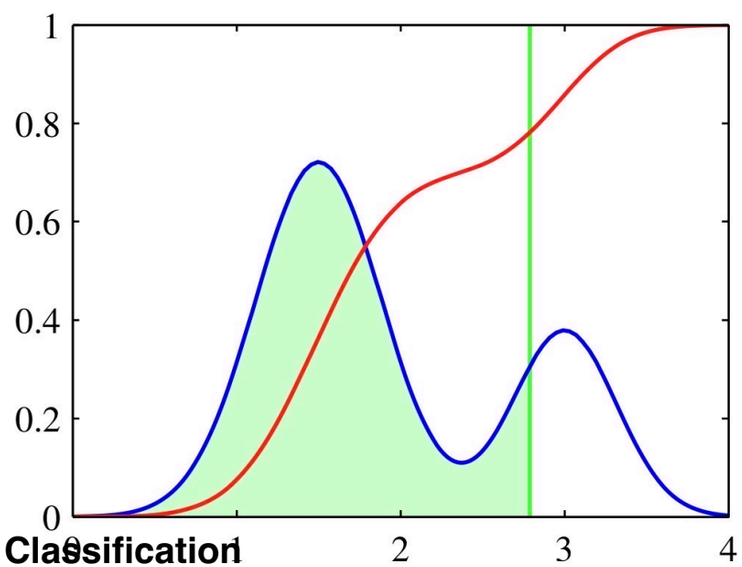
Consider a noisy threshold model. For each input $\boldsymbol{\phi}_n = \boldsymbol{\phi}(\mathbf{x}_n)$, we evaluate $a_n = \mathbf{w}^T \boldsymbol{\phi}_n$ and then set the target value according to

$$\begin{cases} t_n = 1 & \text{if } a_n \geq \theta \\ t_n = 0 & \text{otherwise} \end{cases}$$

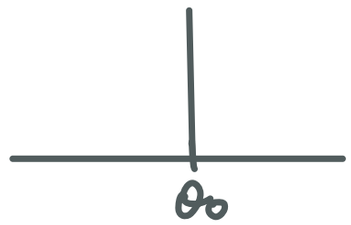
If the value $\theta \sim p(\theta)$, then the corresponding activation function f will be given by

$$f(a) = \int_{-\infty}^{a} p(\theta)\, d\theta$$

Figure 4.13 Schematic example of a probability density $p(\theta)$ shown by the blue curve, given in this example by a mixture of two Gaussians, along with its cumulative distribution function $f(a)$, shown by the red curve. Note that the value of the blue curve at any point, such as that indicated by the vertical green line, corresponds to the slope of the red curve at the same point. Conversely, the value of the red curve at this point corresponds to the area under the blue curve indicated by the shaded green region. In the stochastic threshold model, the class label takes the value $t = 1$ if the value of $a = \mathbf{w}^{\mathrm{T}}\phi$ exceeds a threshold, otherwise it takes the value $t = 0$. This is equivalent to an activation function given by the cumulative distribution function $f(a)$.



As a specific example $p(\theta) = \delta_{\theta_0}(\theta)$



$\Rightarrow \quad f(a) = 1 \quad$ if $\quad a \geq \theta_0 \quad$ otherwise $\quad f(a) = 0.$

In addition, $p(\theta) = \mathcal{N}(\theta \mid 0, 1)$ the corresponding cumulative distribution function is given by

$$\Phi(a) := \int_{-\infty}^{a} \mathcal{N}(\theta \mid 0, 1)\, d\theta$$

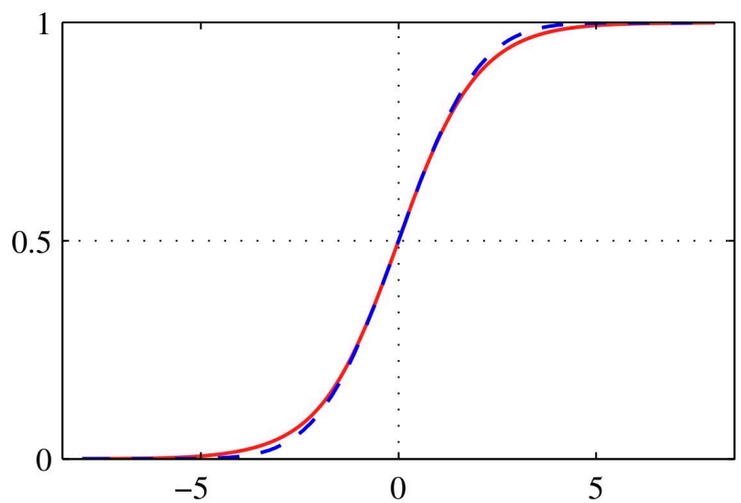
which is known as the inverse probit function

Remark

- It has sigmoidal shape
- The use of a general Gaussian does not change the model
- erf function

$$\text{erf}(a) := \frac{2}{\sqrt{\pi}} \int_0^a \exp(-\theta^2)\, d\theta$$

Figure 4.9 Plot of the logistic sigmoid function $\sigma(a)$ defined by (4.59), shown in red, together with the scaled probit function $\Phi(\lambda a)$, for $\lambda^2 = \pi/8$, shown in dashed blue, where $\Phi(a)$ is defined by (4.114). The scaling factor $\pi/8$ is chosen so that the derivatives of the two curves are equal for $a = 0$.



erf function is related to the inverse probit function by

$$\Phi(a) := \frac{1}{2}\left\{1 + \text{erf}\left(\frac{a}{\sqrt{2}}\right)\right\}$$

The generalized linear model based on an inverse probit activation function is known as probit regression.

Remark

- The probit model is significantly more sensitive to outliers.
- sigmoid $\exp(-x)$ vs inverse probit $\exp(-x^2)$

  as $x \to +\infty$.

4.4 The Laplace Approximation

In section 4.5 we will discuss the Bayesian treatment of logistic regression. We cannot integrate exactly over the parameter vector $\mathbf{w}$ since the posterior distribution is no longer Gaussian. So it is necessary to introduce some form of approximation.

Now we introduce the Laplace approximation, that aims to find a Gaussian approximation to unknown prob. density defined over a set of continuous variables.

z: single continuous variable

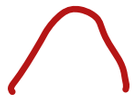
Suppose the distribution $p(z)$ is defined by

$$p(z) = \frac{1}{Z} f(z)$$

where Z is a normalization constant and assumed to be unknown.

In the Laplace method, the goal is to find a Gaussian approximation $q(z)$ which is centered on a mode of $p(z)$

First find a mode $p(z)$, i.e. z_0 s.t $f'(z_0) = 0$



$$\left.\frac{df(z)}{dz}\right|_{z=z_0} = 0$$

Note that the logarithm of Gaussian distribution is a quadratic form of variables. Therefore a Taylor expension of $\ln f(z)$ centered on the mode z_0 is given by

$$\ln f(z) \simeq \ln f(z_0) - \frac{1}{2} A (z - z_0)^2$$

where $$A = -\left.\frac{d^2}{dz} \ln f(z)\right|_{z=z_0}$$

Taking the exponential we obtain

$$f(z) \simeq f(z_0) \exp\left\{-\frac{A}{2}(z - z_0)^2\right\}$$

We can then obtain a normalized distribution $q(z)$ so that

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

Note that it will only be well defined if its precision

$A > 0$ (z_0 must be local maximum or $f''(z_0) < 0$)

$\mathbf{z}$: M - dim vector

Extend the Laplace method to approximate $p(\mathbf{z}) = f(\mathbf{z})/Z$.

At a stationary point $\mathbf{z}_0$, $\nabla f(\mathbf{z})$ will vanish. Expanding around this stationary point $\mathbf{z}_0$ we have

$$\ln f(\mathbf{z}) \simeq \ln f(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^T A (\mathbf{z}-\mathbf{z}_0)$$

where $M \times M$ Hessian matrix A is defined by

$$A = -\nabla\nabla \ln f(\mathbf{z})\big|_{\mathbf{z}=\mathbf{z}_0}$$

Taking exponential we obtain

$$f(\mathbf{z}) \simeq f(\mathbf{z}_0) \exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^T \mathbf{A} (\mathbf{z}-\mathbf{z}_0)\right\}$$

Thus

$$q(\mathbf{z}) = \frac{|\mathbf{A}|^{1/2}}{(2\pi)^{M/2}} \exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^T \mathbf{A} (\mathbf{z}-\mathbf{z}_0)\right\} = \mathcal{N}(\mathbf{z} | \mathbf{z}_0, \mathbf{A}^{-1})$$

where $|\mathbf{A}|$ denotes the determinant of $\mathbf{A}$.

As before, this Gaussian will be well defined if $\mathbf{A}$ is positive definite.

Remark

- Need to find a mode θ_0 and evaluate Hessian matrix.
- In practice, a mode will be found by running some form of numerical optimization algorithm
- Limitations of multimodal case
- Normalization constant Z does not need to be known.

As well as approximating the distribution $p(\mathbf{z})$, we can obtain an approximation to Z

$$Z = \int f(\mathbf{z})\, d\mathbf{z} \simeq f(\mathbf{z}_0) \int \exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathrm{T}} \mathbf{A} (\mathbf{z}-\mathbf{z}_0)\right\} d\mathbf{z}$$

$$= f(\mathbf{z}_0) \frac{(2\pi)^{M/2}}{|\mathbf{A}|^{1/2}}$$

4.5 Bayesian Logistic Regression

The evaluation of the posterior distribution over w would require normalization of the product of a prior distribution and a likelihood function. Note that the likelihood function comprises a product of logistic sigmoid (by our assumption)

i.e. $p(t \mid w) = \prod_n y_n^{t_n} (1 - y_n)^{1-t_n}$, $y_n = \sigma(w^T \phi_n)$.

Evaluation of the predictive distribution is similarly intractable. Here we consider the application of the Laplace approximation to the problem of Bayesian logistic regression

4.5.1 Laplace approximation

We need the evaluation of the second derivatives of the log posterior (finding the Hessian matrix)

Because we seek a Gaussian representation (approximation) for the posterior distribution, we introduce a Gaussian prior.

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w} \mid \mathbf{m}_0, \mathbf{S}_0)$$

where $\mathbf{m}_0$, $\mathbf{S}_0$ are fixed hyperparameters.

The posterior distribution over $\mathbf{w}$ is given by

prior likelihood

$$p(\mathbf{w}|\mathbf{t}) \propto p(\mathbf{w})\, p(\mathbf{t}|\mathbf{w})$$

Gaussian product of sigmoid

where $\mathbf{t} := (t_1, \ldots t_N)^T$. Taking the log of both sides.

$$\ln p(\mathbf{w}|\mathbf{t}) = -\frac{1}{2}(\mathbf{w} - \mathbf{m}_0)^T \mathbf{S}_0^{-1} (\mathbf{w} - \mathbf{m}_0)$$

$$+ \sum_{n=1}^{N} \{ t_n \ln y_n + (1 - t_n) \ln (1 - y_n) \} + \text{constant}$$

where $y_n = \sigma(\mathbf{w}^T \boldsymbol{\phi}_n)$.

To obtain a Gaussian approximation to the posterior distribution, we first maximize the posterior distribution to give the MAP (maximum a posterior) solution w_{MAP} defining the mean (mode) of Gaussian. The covariance is then given by

$$S_N^{-1} = -\nabla\nabla \ln p(w \mid t) = S_0^{-1} + \sum_{n=1}^{N} y_n(1-y_n)\phi_n \phi_n^T$$

The Gaussian approximation to the posterior distribution

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

4.5.2 Predictive distribution

There remains the take of marginalizing w.r.t $q(w)$ to make prediction. Let $\phi = \phi(x)$ be the feature vector.

The predictive distribution for C_1 is obtained by marginalizing w.r.t $p(w|t)$, which is itself approximated by a Gaussian distribution $q(w)$ so that

$$p(C_1|\phi, t) = \int p(C_1|\phi, w)\, p(w|t)\, dw \simeq \int \sigma(w^T\phi)\, q(w)\, dw$$

i.e. $p(C_2|\phi, t) = 1 - p(C_1|\phi, t)$

Let $\delta(\cdot)$ be the Dirac delta function. Then we have

$$\sigma(w^T\phi) = \int \delta(a - w^T\phi)\sigma(a)\,da$$

From this

$$\int \sigma(w^T\phi)\,q(w)\,dw = \int \sigma(a)\,p(a)\,da$$

$$= \mathbb{E}_a[\sigma]$$

where

$$p(a) = \int \delta(a - w^T\phi)\,q(w)\,dw.$$

Gaussian

new prob. distribution

The Dirac delta imposes a linear constraint on w.

So $p(a)$ forms a marginal distribution from the joint distribution $q(w)$ by integrating out all directions orthogonal to Φ. It follows that $p(a)$ is Gaussian.

$$\mu_a = \mathbb{E}[a] = \int p(a)\, a \, da = \iint \delta(a - w^T \Phi)\, q(w)\, dw\, a\, da$$

$$= \iint \delta(a - w^T \Phi)\, a\, da\, q(w)\, dw$$

$$= \int w^T \Phi\, q(w)\, dw$$

$$= w_{MAP}\, \Phi$$

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

Similarly,

$$\sigma_a^2 = \mathrm{var}[a] = \int p(a) \{a^2 - \mathbb{E}[a]\}^2 \, da$$

$$= \int q(w) \{(w^T \phi)^2 - (m_N^T \phi)^2\} \, dw = \phi^T S_N \phi$$

We have used $q(w) = \mathcal{N}(w \mid w_{MAP}, S_N)$.

Thus, the variational approximation to the predictive distribution becomes

(4.151)

$$p(C_1 \mid \phi) \simeq \int \sigma(a) p(a) \, da = \int \sigma(a) \mathcal{N}(a \mid \mu_a, \sigma_a^2) \, da$$

This integral cannot be analytically. So we approximate $\sigma(a)$ by $\Phi(\lambda a)$ with suitable value λ (say $\lambda^2 = \pi/8$)

The advantage of using an inverse profit function is that the below integral (convolution) can be expressed analytically in terms of another inverse profit function.

$$\int \Phi(\lambda a)\, \mathcal{N}(a \mid \mu, \sigma^2)\, da = \Phi\left(\frac{\mu}{(\lambda^{-2} + \sigma^2)^{1/2}}\right)$$

(Spiegelhalter and Lauritzen 1990; Mackay 1992b; Barber and Bishop, 1998a)

We apply the approximation $\sigma(a) \simeq \Phi(\lambda a)$ and it leads to the following approximation

$$\int \sigma(a)\, \mathcal{N}(a \mid \mu, \sigma^2)\, da \simeq \sigma(\kappa(\sigma^2)\mu)$$

where we defined

$$\kappa(\sigma^2) = (1 + \pi\sigma^2/8)^{-1/2}$$

Applying this result to (4.151), we obtain the approximate predictive distribution in the form

$$p(C_1 \mid \boldsymbol{\phi}, \boldsymbol{t}) \simeq \sigma(\kappa(\sigma_a^2)\mu_a)$$