Machine Learning

Classification 3 and 4

Hiroshi Shimodaira and Hao Tang

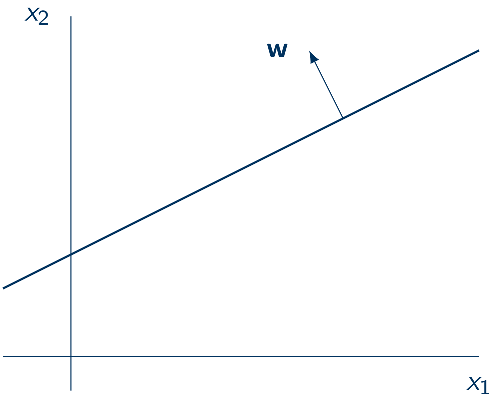
2026 *Ver. 1.0*

Classification with a linear classifier

- $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$: data set
  - $\boldsymbol{x}_i = \begin{bmatrix} x_{i1} & \cdots & x_{id} \end{bmatrix}^\top$, $i = 1, \ldots, N$: input, feature vector, *features*
  - y_i: *label*, ground truth, gold reference, for $\boldsymbol{x}_i$.
- $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$: *linear separator*, *linear predictor*
  - $\boldsymbol{w} = \begin{bmatrix} w_1 & \cdots & w_d \end{bmatrix}^\top$: weights, weight vector
  - $b \in \mathbb{R}$: bias
  - $\{\boldsymbol{w}, b\}$: parameters $\cdots$ $(\boldsymbol{\theta} = [b\ \boldsymbol{w}^\top]^\top)$
- $h(\boldsymbol{x}) = \mathrm{sgn}(f(x))$, where $\mathrm{sgn}(z) = \begin{cases} -1 & \text{if } z < 0 \\ +1 & \text{if } z \geq 0 \end{cases}$

NB: This is a non-standard definition of a sign function

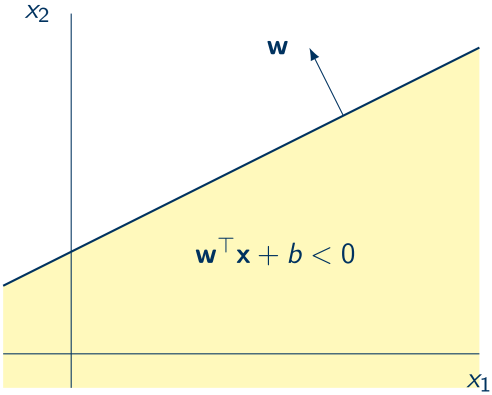
Geometry of linear classification



$$w_1 x_1 + w_2 x_2 + b = 0$$

$$\mathbf{w}^\top \mathbf{x} + b = 0 \quad \text{where } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \ \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

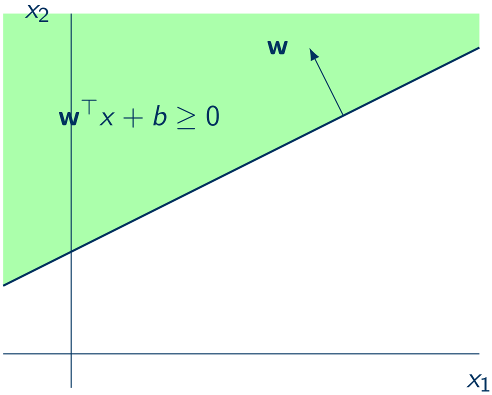
Geometry of linear classification



$$w_1 x_1 + w_2 x_2 + b = 0$$

$$\mathbf{w}^\top \mathbf{x} + b = 0 \quad \text{where } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \ \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

Geometry of linear classification



$$w_1 x_1 + w_2 x_2 + b = 0$$

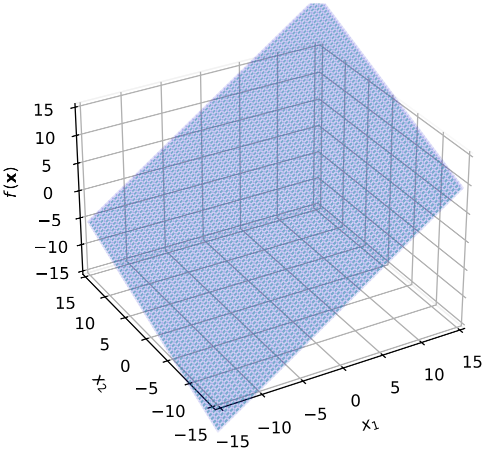
$$\mathbf{w}^\top \mathbf{x} + b = 0 \quad \text{where } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \ \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

$\cdots$ **hyperplane**, **decision boundary**, splitting the space into **decision regions**

NB: $\boldsymbol{w}$ is a normal vector of the hyperplane. b is not the x_2 intercept.

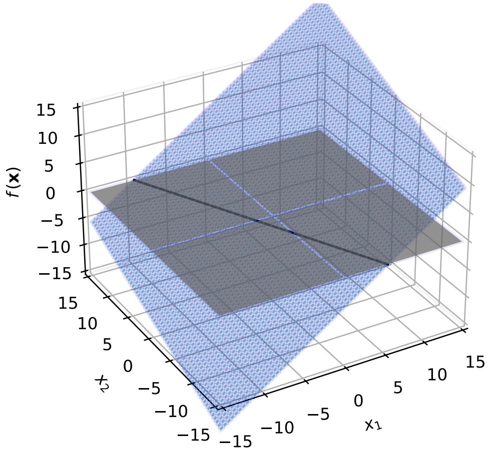
Geometry of linear classification *(cont.)*

$f(\boldsymbol{x}) = w_1 x_1 + w_2 x_2 + b$

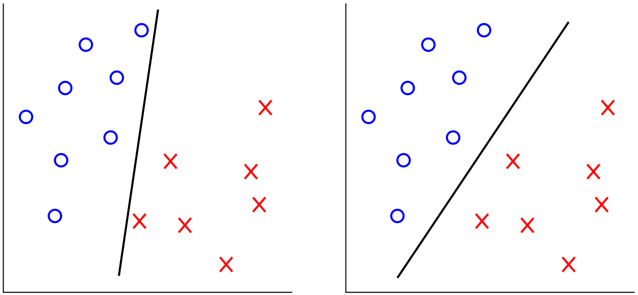


Geometry of linear classification *(cont.)*

$f(\boldsymbol{x}) = w_1 x_1 + w_2 x_2 + b$

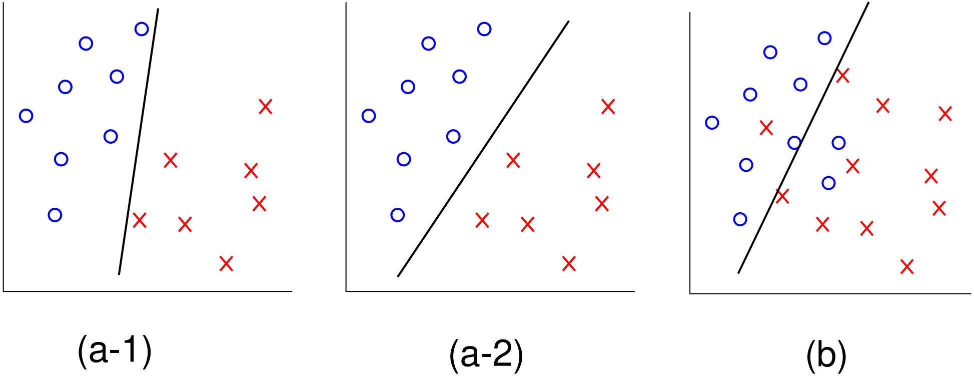


Linearly separable vs linearly non-separable

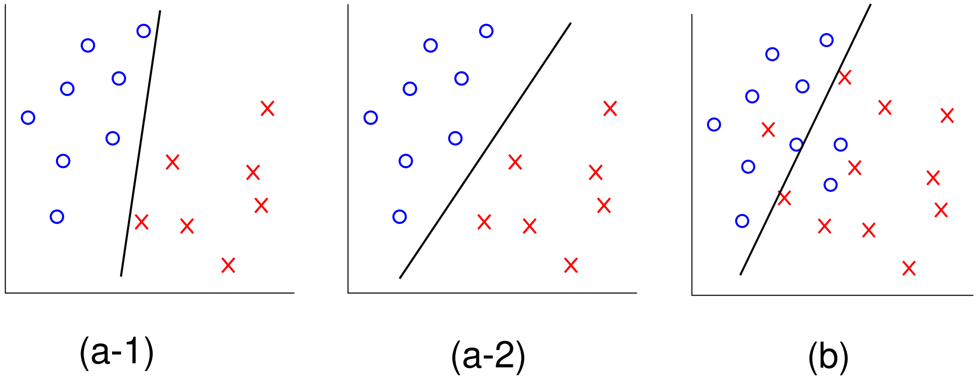


(a-1) (a-2)

Linearly separable vs linearly non-separable



Linearly separable vs linearly non-separable



Linearly separable **Linearly non-separable**

Binary classification with discriminative classifier

$$h(\boldsymbol{x}) = \begin{cases} -1 & \text{if } \boldsymbol{w}^\top \boldsymbol{x} + b < 0 \\ +1 & \text{if } \boldsymbol{w}^\top x + b \geq 0 \end{cases} \tag{1}$$

- The hyperplane $\boldsymbol{w}^\top \boldsymbol{x} + b = 0$ separates the two classes.

Binary classification with discriminative classifier

$$h(\boldsymbol{x}) = \begin{cases} -1 & \text{if } \boldsymbol{w}^\top \boldsymbol{x} + b < 0 \\ +1 & \text{if } \boldsymbol{w}^\top x + b \geq 0 \end{cases} \tag{1}$$

- The hyperplane $\boldsymbol{w}^\top \boldsymbol{x} + b = 0$ separates the two classes.
- The function h labels one class as -1 and the other class as $+1$.

Binary classification with discriminative classifier

$$h(\boldsymbol{x}) = \begin{cases} -1 & \text{if } \boldsymbol{w}^\top \boldsymbol{x} + b < 0 \\ +1 & \text{if } \boldsymbol{w}^\top x + b \geq 0 \end{cases} \tag{1}$$

- The hyperplane $\boldsymbol{w}^\top \boldsymbol{x} + b = 0$ separates the two classes.
- The function h labels one class as -1 and the other class as $+1$.
- The task is called *binary classification*, because there are two classes.

Binary classification with discriminative classifier

$$h(\boldsymbol{x}) = \begin{cases} -1 & \text{if } \boldsymbol{w}^\top \boldsymbol{x} + b < 0 \\ +1 & \text{if } \boldsymbol{w}^\top x + b \geq 0 \end{cases} \tag{1}$$

- The hyperplane $\boldsymbol{w}^\top \boldsymbol{x} + b = 0$ separates the two classes.
- The function h labels one class as -1 and the other class as $+1$.
- The task is called *binary classification*, because there are two classes.
- Why not finding the model parameters $\{\boldsymbol{w}, b\}$ directly based on a misclassification *loss*?

$$\min_{\boldsymbol{w},b} \sum_{i=1}^{N} \ell(\hat{y}_i, y_i), \qquad \text{where } \hat{y}_i = h(\boldsymbol{x}_i)$$

Zero-one loss

$$\ell_{01}(\hat{y}, y) = \begin{cases} 1 & \text{if } \hat{y} \neq y \\ 0 & \text{otherwise} \end{cases} = \mathbb{1}_{\hat{y} \neq y} \tag{2}$$

- Think $\hat{y}$ as the prediction and y as the label.
- We suffer a loss of 1 if we predict the label wrong.
- In the binary case, $\ell_{01}(\hat{y}, y) = \mathbb{1}_{\hat{y}y<0}$.

Discriminative training of a classifier

- Given $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$, find $\boldsymbol{\theta}$ such that the **zero-one loss**

$$L = \frac{1}{N} \sum_{i=1}^{N} \ell_{01}(h(\boldsymbol{x}_i), y_i) \tag{3}$$

is minimised. NB: L is called a **cost function**.

Discriminative training of a classifier

- Given $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$, find $\boldsymbol{\theta}$ such that the **zero-one loss**

$$L = \frac{1}{N} \sum_{i=1}^{N} \ell_{01}(h(\boldsymbol{x}_i), y_i) \tag{3}$$

  is minimised. NB: L is called a **cost function**.
- The act of finding the model parameter $\boldsymbol{\theta}$ is called *training*.
  (We also say "fit the model on the training data" to mean the training)

Discriminative training of a classifier

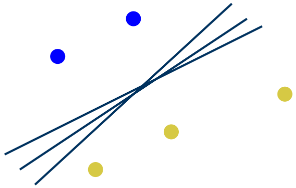
- Given $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$, find $\boldsymbol{\theta}$ such that the **zero-one loss**

$$L = \frac{1}{N} \sum_{i=1}^{N} \ell_{01}(h(\boldsymbol{x}_i), y_i) \tag{3}$$

  is minimised. NB: L is called a **cost function**.
- The act of finding the model parameter $\boldsymbol{\theta}$ is called *training*.
  (We also say "fit the model on the training data" to mean the training)
- In the binary case,

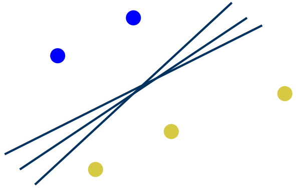
$$L = \frac{1}{N} \sum_{i=1}^{N} \ell_{01}(\text{sgn}(\boldsymbol{w}^\top \boldsymbol{x}_i + b), y_i) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{y_i(\text{sgn}(\boldsymbol{w}^\top \boldsymbol{x}_i + b)) < 0} \tag{4}$$

Training based on the zero-one loss



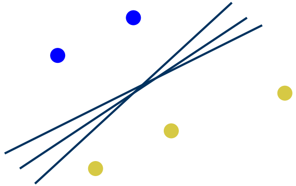
- Slightly changing $\boldsymbol{w}$ and b does not change the loss.

Training based on the zero-one loss



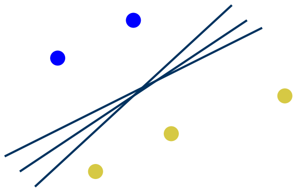
- Slightly changing $\boldsymbol{w}$ and b does not change the loss.
- The loss value only changes when the hyperplane flips the sign of a data point, and it either increases by 1 or none at all.

Training based on the zero-one loss



- Slightly changing $\boldsymbol{w}$ and b does not change the loss.
- The loss value only changes when the hyperplane flips the sign of a data point, and it either increases by 1 or none at all.
- The loss function (with respect to $\boldsymbol{w}$ and b) is like step functions, flat everywhere with discontinuity when the value changes.

Training based on the zero-one loss



- Slightly changing $\boldsymbol{w}$ and b does not change the loss.
- The loss value only changes when the hyperplane flips the sign of a data point, and it either increases by 1 or none at all.
- The loss function (with respect to $\boldsymbol{w}$ and b) is like step functions, flat everywhere with discontinuity when the value changes.
- Finding the optimal $\boldsymbol{w}$ and b is inherently combinatorial and hard.

What about minimising the squared error?

$$\min_{\boldsymbol{w},b} \sum_{i=1}^{N} \Big((\boldsymbol{w}^\top \boldsymbol{x}_i + b) - y_i \Big)^2 , \quad y_i \in \{-1, +1\}$$

What about minimising the squared error?

$$\min_{\boldsymbol{w},b} \sum_{i=1}^{N} \Big((\boldsymbol{w}^\top \boldsymbol{x}_i + b) - y_i \Big)^2 , \quad y_i \in \{-1, +1\}$$

- We will discuss this in the lecture on linear regression.

What about minimising the squared error?

$$\min_{\boldsymbol{w},b} \sum_{i=1}^{N} \left((\boldsymbol{w}^\top \boldsymbol{x}_i + b) - y_i \right)^2 , \quad y_i \in \{-1, +1\}$$

- We will discuss this in the lecture on linear regression.
- We know we can find a solution in closed form.

What about minimising the squared error?

$$\min_{\boldsymbol{w},b} \sum_{i=1}^{N} \left((\boldsymbol{w}^\top \boldsymbol{x}_i + b) - y_i \right)^2 , \quad y_i \in \{-1, +1\}$$

- We will discuss this in the lecture on linear regression.
- We know we can find a solution in closed form.
- Training samples far from the decision boundary influence the solution than those near it.

Types of linear classifiers

- Linear Discriminant Analysis (LDA)
- Template-based matching with Euclidean distance
- Fisher's linear discriminant
- Logistic regression
- Support Vector Machine (linear version)
- Perceptron (original version)
- Single-layer neural networks with no hidden nodes

$\vdots$

Q: Which of the above are from a generative approach?

A probabilistic approach

- The range of $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b : (-\infty, +\infty)$
- We want to squeeze the range into $[0, 1]$ with a function $g(s)$ so that it can be treated as a probability.

$$g(f(\boldsymbol{x})) = g(\boldsymbol{w}^\top \boldsymbol{x} + b) \;\rightarrow\; p(y = +1 \mid \boldsymbol{x})$$

A probabilistic approach

- The range of $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$: $(-\infty, +\infty)$
- We want to squeeze the range into $[0, 1]$ with a function $g(s)$ so that it can be treated as a probability.

$$g(f(\boldsymbol{x})) = g(\boldsymbol{w}^\top \boldsymbol{x} + b) \ \rightarrow \ p(y = +1 \mid \boldsymbol{x})$$

- A candidate for $g(s)$ is the **logistic (sigmoid) function**:

$$g(s) = \frac{e^s}{1 + e^s} = \frac{1}{1 + e^{-s}} \tag{5}$$

A probabilistic approach

- The range of $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$: $(-\infty, +\infty)$
- We want to squeeze the range into $[0, 1]$ with a function $g(s)$ so that it can be treated as a probability.

$$g(f(\boldsymbol{x})) = g(\boldsymbol{w}^\top \boldsymbol{x} + b) \ \rightarrow \ p(y = +1 \mid \boldsymbol{x})$$

- A candidate for $g(s)$ is the **logistic (sigmoid) function**:

$$g(s) = \frac{e^s}{1 + e^s} = \frac{1}{1 + e^{-s}} \tag{5}$$

- *Logistic regression model*:

$$p(y = +1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{6}$$

$$p(y = -1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = 1 - p(y = +1 \mid \boldsymbol{x}) \tag{7}$$

A probabilistic approach

- The range of $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$: $(-\infty, +\infty)$
- We want to squeeze the range into $[0, 1]$ with a function $g(s)$ so that it can be treated as a probability.

$$g(f(\boldsymbol{x})) = g(\boldsymbol{w}^\top \boldsymbol{x} + b) \rightarrow p(y = +1 \mid \boldsymbol{x})$$

- A candidate for $g(s)$ is the **logistic (sigmoid) function**:

$$g(s) = \frac{e^s}{1 + e^s} = \frac{1}{1 + e^{-s}} \tag{5}$$

- *Logistic regression model*:

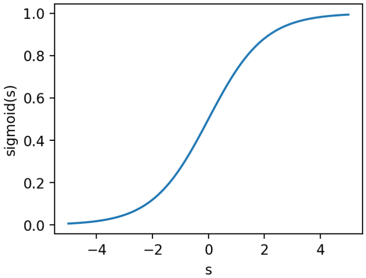
$$p(y = +1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{6}$$

$$p(y = -1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = 1 - p(y = +1 \mid \boldsymbol{x}) \tag{7}$$

$$= \frac{\exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{8}$$

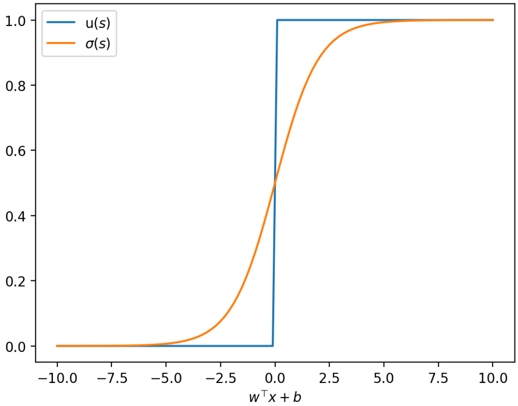
Sigmoid function

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



- When $s \to \infty$, $\sigma(s) \to 1$.
- When $s \to -\infty$, $\sigma(s) \to 0$.

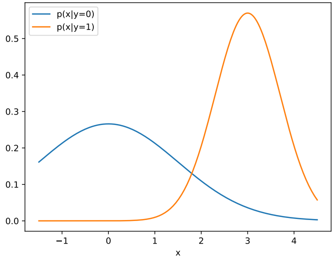
Sigmoid function vs step function



Step function: $u(s) = \begin{cases} 0 & \text{if } s < 0 \\ 1 & \text{if } s \geq 0 \end{cases}$

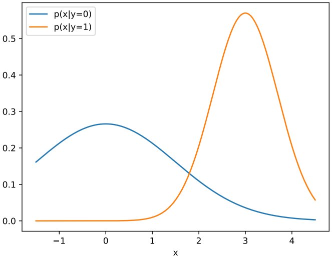
Interpretation of the logistic regression model

Data distributions $p(x \mid y)$

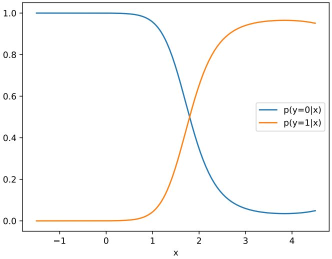


Interpretation of the logistic regression model

Data distributions $p(x \mid y)$

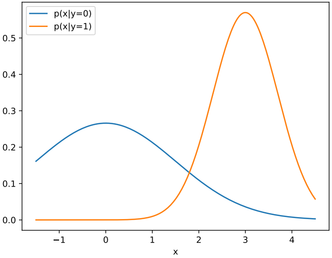


Posterior prob. $p(y \mid x)$

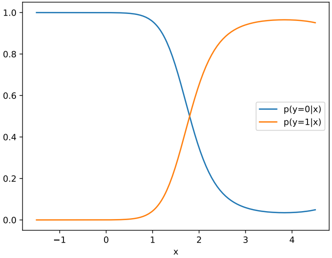


Interpretation of the logistic regression model

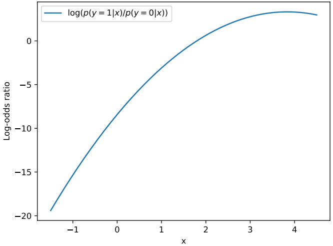
Data distributions $p(x \mid y)$



Posterior prob. $p(y \mid x)$

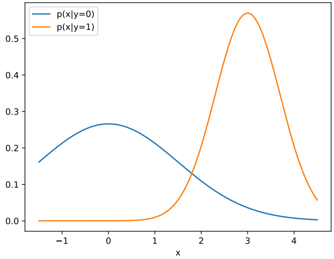


$\log \frac{p(y=1 \mid x)}{p(y=0 \mid x)}$

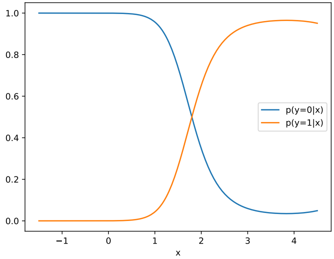


Interpretation of the logistic regression model

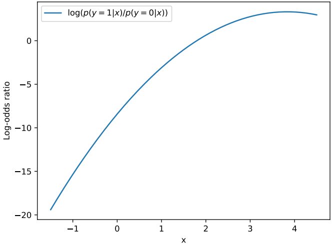
Data distributions $p(x \mid y)$



Posterior prob. $p(y \mid x)$



$\log \frac{p(y=1 \mid x)}{p(y=0 \mid x)}$



Model the log odds ratio with a line: $\log \frac{p(y=1 \mid x)}{p(y=0 \mid x)} = \boldsymbol{w}^\top \boldsymbol{x} + b$

Classification with the logistic regression model

For a test input $\boldsymbol{x}$,

1. calculate the posterior probability with the model.

$$p(y=1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))}$$

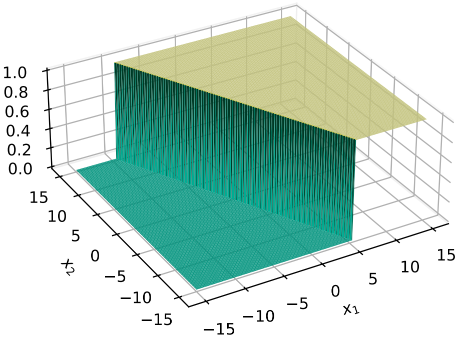
2. make a prediction:

$$\hat{y} = \begin{cases} +1 & p(y=+1 \mid \boldsymbol{x}, \boldsymbol{\theta}) > \text{threshold}, \\ -1 & p(y=+1 \mid \boldsymbol{x}, \boldsymbol{\theta}) \leq \text{threshold} \end{cases} \tag{9}$$

NB: threshold $= 0.5$ normally – it gives a minimum misclassification rate.

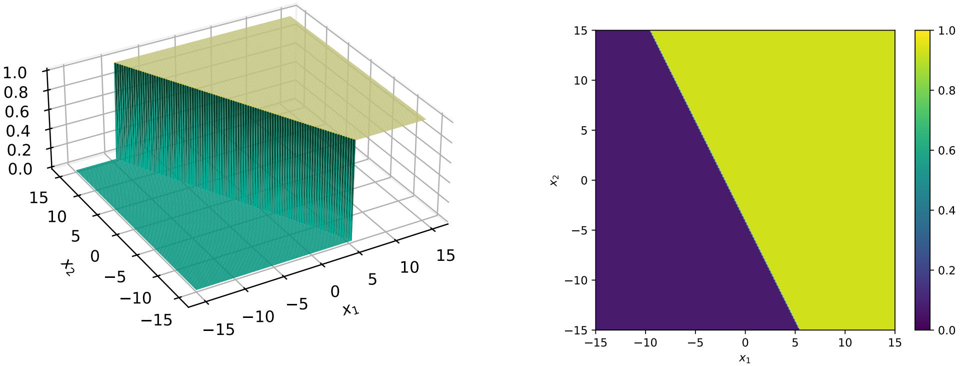
Decision surface - step function version

$$\mathrm{u}(\boldsymbol{w}^\top \boldsymbol{x} + b)$$



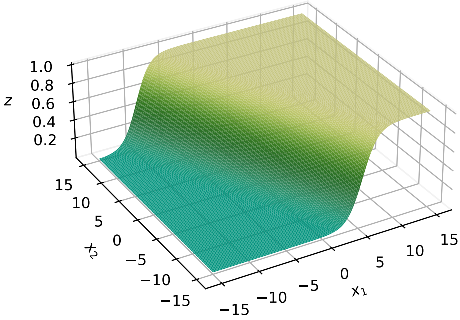
Decision surface - step function version

$$\mathrm{u}(\boldsymbol{w}^\top \boldsymbol{x} + b)$$



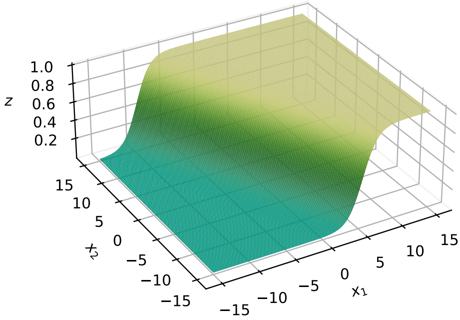
Decision surface - sigmoid function version

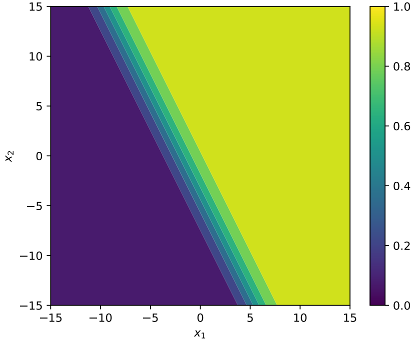
$$\sigma(\boldsymbol{w}^\top \boldsymbol{x} + b)$$



Decision surface - sigmoid function version

$$\sigma(\boldsymbol{w}^\top \boldsymbol{x} + b)$$





A logistic regression model

$$p(y=+1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1+\exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{10}$$

$$p(y=-1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = 1 - \frac{1}{1+\exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} = \frac{\exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))}{1+\exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{11}$$

$$= \frac{1}{\exp(\boldsymbol{w}^\top \boldsymbol{x} + b) + 1} \tag{12}$$

Thus,

$$p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1+\exp(-y(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{13}$$

How to train the logistic regression model?

- Apply the *maximum likelihood estimation (MLE)*:

  Given a data set $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$,
  maximise the likelihood L of $\boldsymbol{w}$ and b.

$$\max_{\boldsymbol{w}, b} L \tag{14}$$

$$L = \log \prod_{i=1}^{N} p(y_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}) = \sum_{i=1}^{N} \log \frac{1}{1 + \exp(-y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b))} \tag{15}$$

$$= \sum_{i=1}^{N} -\log \left(1 + \exp(-y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b)) \right) \tag{16}$$

How to find the optimal solutions w and b?

- The zero-one loss $\sum_{i=1}^{N} \mathbb{1}_{y_i(\boldsymbol{w}^\top \boldsymbol{x}_i+b)<0}$ is flat, and is hard to optimise.

- The log likelihood of the logistic regression model
$L = \sum_{i=1}^{N} -\log(1 + \exp(-y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b))$ is differentiable.

- However,

$$\frac{\partial L}{\partial w_i} = 0, \; i = 1, \ldots, d \quad \text{and} \quad \frac{\partial L}{\partial b} = 0 \tag{17}$$

do not have *closed-form* solutions.
$\rightarrow$ employ *gradient ascent*.

- We will come back to this in a lecture on optimisation.

What if we use 0/1 labels instead of -1/+1?

- $y \in \{0, 1\}$ instead of $\{-1, +1\}$

What if we use 0/1 labels instead of -1/+1?

- $y \in \{0, 1\}$ instead of $\{-1, +1\}$

$$p(y=1 \mid \boldsymbol{x}) = \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{18}$$

$$p(y=0 \mid \boldsymbol{x}) = 1 - \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{19}$$

What if we use 0/1 labels instead of -1/+1?

- $y \in \{0, 1\}$ instead of $\{-1, +1\}$

$$p(y = 1 \mid \boldsymbol{x}) = \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{18}$$

$$p(y = 0 \mid \boldsymbol{x}) = 1 - \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))} \tag{19}$$

$$p(y \mid \boldsymbol{x}) = \left(\frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))}\right)^y \left(1 - \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))}\right)^{1-y} \tag{20}$$

$$= s^y (1 - s)^{1-y} \tag{21}$$

$$\text{where } s = \frac{1}{1 + \exp(-(\boldsymbol{w}^\top \boldsymbol{x} + b))}.$$

What if we use 0/1 labels instead of -1/+1? *(cont.)*

Training with MLE,

$$L = \log \prod_{i=1}^{N} p(y_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}) \tag{22}$$

$$= \log \prod_{i=1}^{N} s_i^{y_i} (1 - s_i)^{1-y_i} \tag{23}$$

$$= \sum_{i=1}^{N} y_i \log s_i + (1 - y_i) \log(1 - s_i) \tag{24}$$

$$= -\sum_{i=1}^{N} H(y_i, s_i) \tag{25}$$

where $H(p, q) = -\sum_x p(x) \log q(x)$ is a cross entropy between the two probability distributions p and q. For a binary case, $H(p, q) = -(p \log q + (1 - p) \log(1 - q))$.

Classification losses

Suppose we have a labelled data point $(\boldsymbol{x}, y)$.

- Zero-one loss

$$\mathbb{1}_{y(\boldsymbol{w}^\top \boldsymbol{x}+b)<0} \tag{26}$$

- Log loss (logistic loss)

$$-\log p(y \mid \boldsymbol{x}) = \log(1 + \exp(-y(\boldsymbol{w}^\top \boldsymbol{x} + b)) \tag{27}$$

NB: this is the negative log likelihood

Notation caveat

- The log loss notation $-\log p(y \mid \boldsymbol{x})$ can be misleading.
- Is y the ground truth or is it a free variable?
- What it really means is $-\log p(y = y^* \mid \boldsymbol{x})$ given a pair $(x, \boldsymbol{y}^*)$.
- Or $-\log p(y = y_i \mid \boldsymbol{x}_i)$ given a pair $(\boldsymbol{x}_i, y_i)$ in a data set.

Multiclass classification with logistic regression

Replace the sigmoid $g(z)$ with the **softmax function** $\boldsymbol{g}(\boldsymbol{a}) = [g_1(\boldsymbol{a}) \; \cdots \; g_K(\boldsymbol{a})]^\top$

$$g(a) = \frac{\exp(a)}{1+\exp(a)} \quad \longrightarrow \quad \begin{aligned} g_1(\boldsymbol{a}) &= \frac{\exp(a_1)}{\sum_{k'=1}^K \exp(a'_k)} \\ g_2(\boldsymbol{a}) &= \frac{\exp(a_2)}{\sum_{k'=1}^K \exp(a'_k)} \\ &\vdots \\ g_K(\boldsymbol{a}) &= \frac{\exp(a_K)}{\sum_{k'=1}^K \exp(a_{k'})} \end{aligned} \qquad \boldsymbol{a} = [a_1 \; a_2 \ldots \; a_K]^\top$$

Redefining $\boldsymbol{x} = [1 \; x_1 \; x_2 \; \cdots \; x_d]^\top$ and $\boldsymbol{w} = [w_0 \; w_1 \; \cdots \; w_d]^\top$, logistic regression is given as:

$$p(y=k \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{w}_k^\top \boldsymbol{x})}{\sum_{k'=1}^K \exp(\boldsymbol{w}_{k'}^\top \boldsymbol{x})} \tag{28}$$

$$\hat{y} = \arg\max_k \frac{\exp(\boldsymbol{w}_k^\top \boldsymbol{x})}{\sum_{k'=1}^K \exp(\boldsymbol{w}_{k'}^\top \boldsymbol{x})} = \arg\max_k \exp(\boldsymbol{w}_k^\top \boldsymbol{x})$$

Softmax for binary classification

$$p(y=+1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{w}_{+1}^\top \boldsymbol{x})}{\exp(\boldsymbol{w}_{+1}^\top \boldsymbol{x}) + \exp(\boldsymbol{w}_{-1}^\top \boldsymbol{x})} \tag{29}$$

$$= \frac{1}{1+\exp(-(\boldsymbol{w}_{+1} - \boldsymbol{w}_{+1})^\top \boldsymbol{x})} = \frac{1}{1+\exp(-\boldsymbol{w}^\top \boldsymbol{x})} \tag{30}$$

$$p(y=-1 \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{w}_{-1}^\top \boldsymbol{x})}{\exp(\boldsymbol{w}_{+1}^\top \boldsymbol{x}) + \exp(\boldsymbol{w}_{-1}^\top \boldsymbol{x})} \tag{31}$$

$$= \frac{\exp(-(\boldsymbol{w}_{+1} - \boldsymbol{w}_{-1})^\top \boldsymbol{x})}{1+\exp(-(\boldsymbol{w}_{+1} - \boldsymbol{w}_{-1})^\top \boldsymbol{x})} = \frac{\exp(-\boldsymbol{w}^\top \boldsymbol{x})}{1+\exp(-\boldsymbol{w})^\top \boldsymbol{x})} \tag{32}$$

where $\boldsymbol{w} = \boldsymbol{w}_{+1} - \boldsymbol{w}_{-1}$.

$\rightarrow$ the same as the sigmoid.

Training of the multiclass logistic regression model

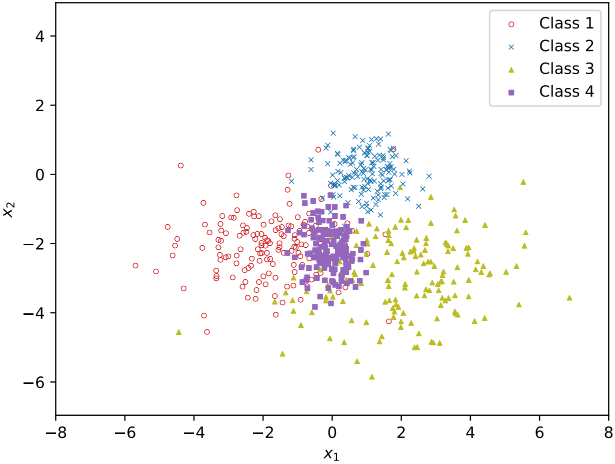
The log likelihood for a training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$:

$$L = \sum_{i=1}^{N} \log g_{y_i}(\boldsymbol{x}_i; \boldsymbol{\theta}) \tag{33}$$

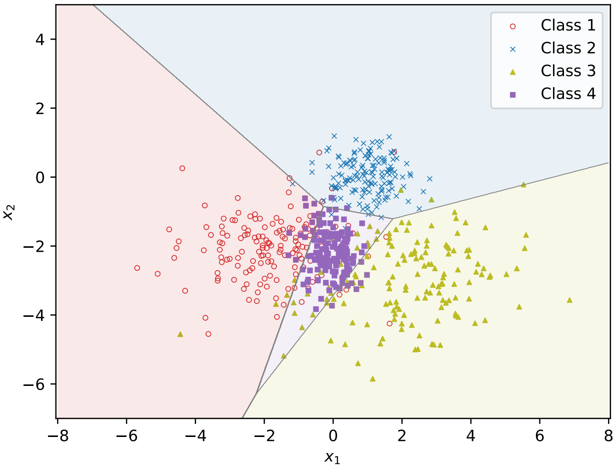
$$= \sum_{i=1}^{N} \left(\boldsymbol{w}_{y_i}^\top \boldsymbol{x}_i - \log \left(\sum_{k=1}^{K} \exp(\boldsymbol{w}_k^\top \boldsymbol{x}_i) \right) \right) \tag{34}$$

We can apply the maximum likelihood estimation (MLE).

Decision regions with a multiclass logistic regression model



Decision regions with a multiclass logistic regression model



Adapting a binary classifier to multiclass classification

- one-versus-rest (one-against-all)
- one-versus-one
-

One-versus-rest

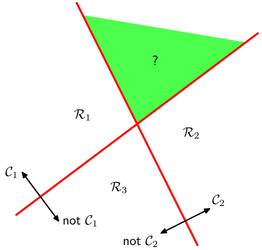
$$\hat{y}(\boldsymbol{x}) = \arg\max_k g_k(\boldsymbol{x})$$

Discriminant function	+1 class	-1 class
$g_1(\boldsymbol{x})$	C_1	$C_2, \ldots, C_K$
$g_2(\boldsymbol{x})$	C_2	$C_1, C_3, \ldots, C_K$
$\vdots$	$\vdots$	$\vdots$
$g_{K-1}(\boldsymbol{x})$	C_{K-1}	$C_1, \ldots, C_{K-2}, C_K$
$g_K(\boldsymbol{x})$	C_K	$C_1, \ldots, C_{K-1}$

One-versus-rest

$$\hat{y}(\boldsymbol{x}) = \arg\max_k g_k(\boldsymbol{x})$$

Discriminant function	+1 class	-1 class
$g_1(\boldsymbol{x})$	C_1	$C_2, \ldots, C_K$
$g_2(\boldsymbol{x})$	C_2	$C_1, C_3, \ldots, C_K$
$\vdots$	$\vdots$	$\vdots$
$g_{K-1}(\boldsymbol{x})$	C_{K-1}	$C_1, \ldots, C_{K-2}, C_K$
$g_K(\boldsymbol{x})$	C_K	$C_1, \ldots, C_{K-1}$



Issues:

- Ambiguous decision regions
- Separate training of each $g_k(\boldsymbol{x})$ from the others - no global training
- Imbalance training data set - negative classes are much larger than positive ones

One-versus-one

$$\{g_{kk'}(\boldsymbol{x})\} \qquad k' > k,\; k, k' = 1, \ldots, K \quad \cdots \; K(K-1)/2 \text{ discriminants}$$

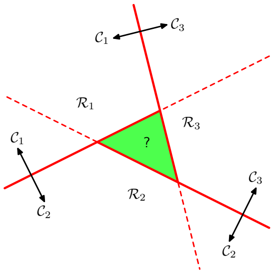
Discriminant function	+1 class	-1 class
$g_{12}(\boldsymbol{x})$	C_1	C_2
$\vdots$	$\vdots$	$\vdots$
$g_{23}(\boldsymbol{x})$	C_2	C_3
$\vdots$	$\vdots$	$\vdots$
$g_{K-1,K}(\boldsymbol{x})$	C_{K-1}	C_K

$\rightarrow$ Classification by voting: the class that wins the most is chosen

One-versus-one

$$\{g_{kk'}(\boldsymbol{x})\} \qquad k' > k,\ k, k' = 1, \dots, K \quad \cdots\ K(K-1)/2 \text{ discriminants}$$

Discriminant function	+1 class	-1 class
$g_{12}(\boldsymbol{x})$	C_1	C_2
$\vdots$	$\vdots$	$\vdots$
$g_{23}(\boldsymbol{x})$	C_2	C_3
$\vdots$	$\vdots$	$\vdots$
$g_{K-1,K}(\boldsymbol{x})$	C_{K-1}	C_K



$\rightarrow$ Classification by voting: the class that wins the most is chosen

Issues:

- Ambiguous decision regions
- Not scalable in K

Practical issues with logistic regression

- Linear classifier – what if the data set is not linearly separable?

Practical issues with logistic regression

- Linear classifier – what if the data set is not linearly separable?

  $\rightarrow$ We will discuss this at the lecture on 'features and kernels'

Practical issues with logistic regression

- Linear classifier – what if the data set is not linearly separable?

  $\rightarrow$ We will discuss this at the lecture on 'features and kernels'

- Overfitting - the model overfits on to the training set and does not generalise

Practical issues with logistic regression

- Linear classifier – what if the data set is not linearly separable?

  $\rightarrow$ We will discuss this at the lecture on 'features and kernels'

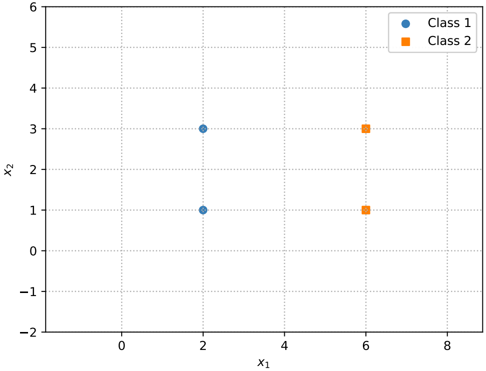
- Overfitting - the model overfits on to the training set and does not generalise

  $\rightarrow$ Employ 'regularisation' (or a penalty) in the cost function – this will be discussed in 'optimisation'.

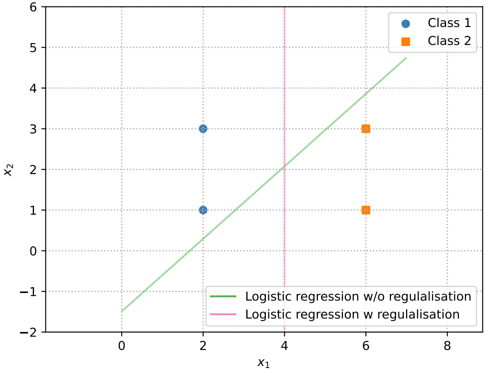
$$L = \sum_{i=1}^{N} -\log\left(1 + \exp(-y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b))\right) - \text{regulariser} \tag{35}$$

$$\text{A regulariser} = \lambda \|\boldsymbol{\theta}\|_2^2$$

Overfitting



Overfitting



Summary

- Log loss in the binary case

$$\sum_{i=1}^{N} \log\left(1 + \exp(-y_i \boldsymbol{w}^\top \boldsymbol{x}_i)\right) \tag{36}$$

- Log loss in the multiclass case

$$\sum_{i=1}^{N} -\boldsymbol{w}_{y_i}^\top \boldsymbol{x}_i + \log\left(\sum_{y' \in \mathcal{Y}} \exp(\boldsymbol{w}_{y'}^\top \boldsymbol{x}_i)\right) \tag{37}$$

Logistic regression vs LDA

- Logistic regression:

$$p(y = k \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{w}_k^\top \boldsymbol{x})}{\sum_{k'=1}^{K} \exp(\boldsymbol{w}_{k'}^\top \boldsymbol{x})}$$

- LDA

$$g_k(\boldsymbol{x}) = \log p(y = k \mid \boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{w}_k^\top \boldsymbol{x} + w_{k0} + \text{const} \tag{38}$$

$$p(y = k \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{\exp\left(\boldsymbol{w}_k^\top \boldsymbol{x} + w_{k0}\right)}{\sum_{k'} \exp\left(\boldsymbol{w}_{k'}^\top \boldsymbol{x} + w_{k'0}\right)} \tag{39}$$

where $\boldsymbol{w}_k^\top = \boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1}$ $w_{k0} = -\frac{1}{2}\boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k + \log p(C_k)$

Summary (*cont.*)

binary classification	multiclass classification
$h(\boldsymbol{x}) = \begin{cases} -1 & \text{if } \boldsymbol{w}^\top \boldsymbol{x} < 0 \\ +1 & \text{if } \boldsymbol{w}^\top \boldsymbol{x} \geq 0 \end{cases}$	$h(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \boldsymbol{w}_y^\top \boldsymbol{x}$
$p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{1 + \exp(-y \boldsymbol{w}^\top \boldsymbol{x})}$	$p(y \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{w}_y^\top \boldsymbol{x})}{\sum_{y' \in \mathcal{Y}} \exp(\boldsymbol{w}_{y'}^\top \boldsymbol{x})}$

Appendix – softmax

$$\text{softmax}\left(\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}\right) = \begin{bmatrix} \frac{\exp(a_1)}{\sum_{i=1}^{n} \exp(a_i)} \\ \frac{\exp(a_2)}{\sum_{i=1}^{n} \exp(a_i)} \\ \vdots \\ \frac{\exp(a_n)}{\sum_{i=1}^{n} \exp(a_i)} \end{bmatrix} \tag{40}$$

Appendix – softmax *(cont.)*

- $\mathrm{softmax}(\begin{bmatrix}1 & 2 & 3\end{bmatrix}^\top) = \begin{bmatrix}0.09 & 0.24 & 0.67\end{bmatrix}^\top$
- $\mathrm{softmax}(\begin{bmatrix}100 & 200 & 300\end{bmatrix}^\top) = \begin{bmatrix}10^{-87} & 10^{-44} & 1.0\end{bmatrix}^\top$
- Softmax always returns a probability distribution.
- When the dynamic range of the input is large, the result of softmax becomes "sharp."

Appendix – softmax *(cont.)*

- Claim: $\frac{\exp(a_{\max}/\tau)}{\sum_{i=1}^{n}\exp(a_i/\tau)} \to 1$ when $\tau \to 0$.
- That means $\frac{\exp(a_j/\tau)}{\sum_{i=1}^{n}\exp(a_i/\tau)} \to 0$ when $\tau \to 0$ for any a_j that is not the max.
- We have

$$\frac{\exp(a_m/\tau)}{\sum_{i=1}^{n}\exp(a_i/\tau)} = \frac{\exp(a_m/\tau)}{\exp(a_m/\tau) + \sum_{i\neq m}\exp(a_i/\tau)} \tag{41}$$

$$= \frac{1}{1 + \sum_{i\neq m}\exp((a_i - a_m)/\tau)} \to 1 \tag{42}$$

when $\tau \to 0$ because a_m is the largest and $a_i - a_m < 0$.

Quizzes

1. Consider two column vectors such that $\mathbf{a} = (1, 2, 3)^\top$ and $\mathbf{b} = (-3, 3, -1)^\top$.
    - Find $\mathbf{a} + \mathbf{b}$.
    - Find $\mathbf{a} - \mathbf{b}$.
    - Find $\|\mathbf{a}\|$, $\|\mathbf{b}\|$, and $\|\mathbf{a} - \mathbf{b}\|$.
    - Find $\mathbf{a}^\top \mathbf{b}$.
    - Find $\mathbf{a}\mathbf{b}^\top$.
    - What is the geometric relationship between $\mathbf{a}$ and $\mathbf{b}$?
2. Considering a classification problem of two classes, whose discriminant function takes the form, $y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + w_0$.
    - Show that the decision boundary is a straight line when $D = 2$.
    - Show that the weight vector $\mathbf{w}$ is a normal vector to the decision boundary.
3. Derive a formula for the Euclidean distance between the origin $(0, 0)$ and a line $y = ax + b$, where a and b are arbitrary constants.

Quizzes *(cont.)*

4. Considering a linear classifier of binary classification in a two-dimensional vector space, such that the points $(-2, -3)$ and $(4, 1)$ are on the decision boundary, and the point $(2, -3)$ lies in the -1 class region.
   - Find the parameters $(\boldsymbol{w}, b)$ of the classifier.
   - Find the unit normal vector of $\boldsymbol{w}$.
5. Consider the following logistic regression model:

$$p(y = +1 \mid x) = \frac{1}{1 + \exp(-(wx + b))}$$

   Plot $p(y = +1 \mid x)$ for each of the following cases, where you use a fixed plotting range or show all the plots on a single graph for comparison, and report your findings.
   - $w = 1, b = 0$
   - $w = 1, b = 1$
   - $w = -1, b = 1$
   - $w = 0.5, b = 1$
   - $w = 2, b = 1$

Quizzes *(cont.)*

6. Consider the logistic sigmoid function.

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

- Based on the graph of $\sigma(x)$, make an educated guess about the shape of the derivative $\sigma'(x)$ without performing any calculations and illustrate it by hand.
- Find the derivative of $\sigma(x)$.
- Plot the derivative on a graph.