Multivariate Analysis
Discriminant Analysis and Classification

Shaobo Jin

Department of Mathematics

Intended Learning Outcome

Through this chapter, you should be able to

1. apply different classifiers,
2. compare different classifiers.

Motivation

Discrimination and classification are used to separate observations and allocate new subjects to previously defined classes. They are called supervised learning in machine learning.

1. Discrimination: use labeled observations to build a classification rule (classifier).
2. Classification: allocate new subjects to classes using classifiers.

Some examples are

1. given the medical history, whether a patient has cancer or not.
2. given your credit history, whether you can pay back the mortgage loan or not.
3. given your hotel booking history, which hotel brand you will choose, Scandic, Best Western, or Radisson.

Two-Class Problem

Suppose that there are in total two classes (pass or fail, healthy or sick, positive or negative). The classes are labeled as π_1 and π_2. For each subject, we observe a $p \times 1$ vector of covariates $\boldsymbol{X}$.

- We need to partition the sample space into two disjoint region R_1 and R_2, such that $R_1 \cup R_2 = \Omega$, where Ω is the sample space.
- If a new vector $\boldsymbol{x}_0 \in R_1$ ($\boldsymbol{x}_0 \in R_2$), then the subject is classified to class π_1 (π_2).

How we partition and classify the subjects can be either probabilistic or deterministic.

Misclassification

A good classifier should result in few misclassifications or low costs associated with misclassification. Let $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ be the density functions associated with the $p \times 1$ vector of covariates $\boldsymbol{X}$ for the populations π_1 and π_2. Then,

$$P(i \mid j) := P(\text{Classify as } \pi_i \mid \text{truth } \pi_j) = \int_{\boldsymbol{x} \in R_i} f_j(\boldsymbol{x})\, d\boldsymbol{x}.$$

Hence, for $j \neq i$,

$$\begin{aligned} P(\text{Correctly classified as } \pi_i) &= P(\text{Classify as } \pi_i \mid \text{truth } \pi_i)\, P(\text{truth } \pi_i), \\ P(\text{Misclassified as } \pi_i) &= P(\text{Classify as } \pi_i \mid \text{truth } \pi_j)\, P(\text{truth } \pi_j), \end{aligned}$$

where $P(\text{truth } \pi_i)$ is some prior probability such that

$$P(\text{truth } \pi_1) + P(\text{truth } \pi_2) = 1.$$

Classification Table

A classification table or a confusion matrix cross-classifies the binary response y and the predicted binary response.

	Observed	
Predicted	1 (TRUE)	2 (FALSE)
1 (TRUE)	m_{11} (TP)	m_{12} (FP)
2 (FALSE)	m_{21} (FN)	m_{22} (TN)

From the classification table, we can compute

$$\begin{aligned}
\text{sensitivity (true positive rate)} &= \frac{m_{11}}{m_{11}+m_{21}}, \\
\text{specificity (true negative rate)} &= \frac{m_{22}}{m_{12}+m_{22}}, \\
\text{false positive rate} &= \frac{m_{12}}{m_{12}+m_{22}}, \\
\text{false negative rate} &= \frac{m_{21}}{m_{11}+m_{21}}.
\end{aligned}$$

F-Score of Binary Classification

	Observed	
Predicted	1 (TRUE)	2 (FALSE)
1 (TRUE)	m_{11}	m_{12}
2 (FALSE)	m_{21}	m_{22}

In machine learning,

$$\begin{aligned} \text{Precision} &= \frac{m_{11}}{m_{11}+m_{12}}, \\ \text{Recall} &= \frac{m_{11}}{m_{11}+m_{21}}. \end{aligned}$$

The F-score is

$$\text{F-score} = 2\frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} = \frac{2m_{11}}{2m_{11}+m_{12}+m_{21}} \in [0,1].$$

Cost of Misclassification

Let $c(i \mid j)$ be the cost of classifying π_j to π_i. The cost matrix tabulates the costs of misclassification:

	Observed	
Predicted	1 (TRUE)	2 (FALSE)
1 (TRUE)	0	$c(1 \mid 2)$
2 (FALSE)	$c(2 \mid 1)$	0

The expected cost of misclassification (ECM) is

$$\text{ECM} = c(2 \mid 1) P(2 \mid 1) p_1 + c(1 \mid 2) P(1 \mid 2) p_2,$$

where $p_i = P(\text{truth } \pi_i)$. A reasonable classification rule should have a small ECM.

Loosely speaking, since it requires the prior probability p_i, it can be viewed as Bayesian discrimination.

Minimizing ECM

Result 11.1: Known Prior Probabilities

The regions R_1 and R_2 that minimize the ECM are defined by the values $\boldsymbol{x}$ for which the following inequalities hold

$$R_1 = \left\{ \boldsymbol{x}; \quad \frac{f_1(\boldsymbol{x})}{f_2(\boldsymbol{x})} \geq \frac{c(1 \mid 2)}{c(2 \mid 1)} \frac{p_2}{p_1} \right\},$$

$$R_2 = \left\{ \boldsymbol{x}; \quad \frac{f_1(\boldsymbol{x})}{f_2(\boldsymbol{x})} < \frac{c(1 \mid 2)}{c(2 \mid 1)} \frac{p_2}{p_1} \right\}.$$

An Example Using ECM

Known Prior Probabilities

Suppose that $\boldsymbol{X} \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ in π_1, and $\boldsymbol{X} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ in π_2. It is known that $p_1 = 0.8$ and $p_2 = 0.2$. Let $c(2 \mid 1) = 5$ and $c(1 \mid 2) = 10$. Then,

$$\frac{c(1 \mid 2)}{c(2 \mid 1)} \frac{p_2}{p_1} = \frac{10}{5} \frac{0.2}{0.8} = 0.5.$$

If a new subject with $\boldsymbol{X} = \boldsymbol{x}_0$ satisfies

$$\frac{f_1(\boldsymbol{x}_0)}{f_2(\boldsymbol{x}_0)} = \frac{\exp\left\{-\frac{1}{2}\boldsymbol{x}_0^T \boldsymbol{\Sigma}^{-1} \boldsymbol{x}_0\right\}}{\exp\left\{-\frac{1}{2}(\boldsymbol{x}_0 - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}_0 - \boldsymbol{\mu})\right\}} \leq 0.5,$$

then it will be classified to π_1. Otherwise, it is classified to π_2.

Special Case I: Highest Likelihood

Suppose that

$$\frac{c\left(1 \mid 2\right)}{c\left(2 \mid 1\right)}\frac{p_2}{p_1} \quad = \quad 1.$$

Then the classifier that minimizes ECM is

$$\begin{aligned} R_1 &= \left\{\boldsymbol{x};\quad f_1\left(\boldsymbol{x}\right) \geq f_2\left(\boldsymbol{x}\right)\right\}, \\ R_2 &= \left\{\boldsymbol{x};\quad f_1\left(\boldsymbol{x}\right) < f_2\left(\boldsymbol{x}\right)\right\}. \end{aligned}$$

The densities $f_1\left(\boldsymbol{x}\right)$ and $f_2\left(\boldsymbol{x}\right)$ can be viewed as conditional density (or likelihood)

$$f_i\left(\boldsymbol{x}\right) \quad = \quad f\left(\boldsymbol{x} \mid \text{truth } \pi_i\right).$$

Special Case II: Total Probability of Misclassification

The total probability of misclassification is

$$\begin{aligned} \text{TPM} &= P\left(\text{Misclassifying a observation}\right) \\ &= P\left(\text{Classify as } \pi_2, \text{truth } \pi_1\right) + P\left(\text{Classify as } \pi_1, \text{truth } \pi_2\right) \\ &= P\left(2 \mid 1\right) p_1 + P\left(1 \mid 2\right) p_2, \end{aligned}$$

which is equivalent to ECM when $c\left(1 \mid 2\right) = c\left(2 \mid 1\right)$. Hence, the regions R_1 and R_2 that minimize the TPM are

$$\begin{aligned} R_1 &= \left\{ \boldsymbol{x}; \quad \frac{f_1\left(\boldsymbol{x}\right)}{f_2\left(\boldsymbol{x}\right)} \geq \frac{p_2}{p_1} \right\}, \\ R_2 &= \left\{ \boldsymbol{x}; \quad \frac{f_1\left(\boldsymbol{x}\right)}{f_2\left(\boldsymbol{x}\right)} < \frac{p_2}{p_1} \right\}. \end{aligned}$$

Special Case III: Highest Posterior Probability

The densities $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ can be viewed as conditional density (or likelihood)

$$f_i(\boldsymbol{x}) = f(\boldsymbol{x} \mid \text{truth } \pi_i).$$

By the Bayes rule,

$$\begin{aligned} P(\text{truth } \pi_i \mid \text{observe } \boldsymbol{x}_0) &= \frac{P(\text{observe } \boldsymbol{x}_0 \mid \text{truth } \pi_i) P(\text{truth } \pi_i)}{\sum_{j=1}^{2} P(\text{observe } \boldsymbol{x}_0 \mid \text{truth } \pi_j) P(\text{truth } \pi_j)} \\ &= \frac{f_i(\boldsymbol{x}_0) p_i}{f_1(\boldsymbol{x}_0) p_1 + f_2(\boldsymbol{x}_0) p_2}. \end{aligned}$$

The Naive Bayes classifier classifies $\boldsymbol{x}_0$ to π_i if

$$P(\text{truth } \pi_i \mid \text{observe } \boldsymbol{x}_0) > P(\text{truth } \pi_j \mid \text{observe } \boldsymbol{x}_0).$$

This is equivalent to the classifier that minimizes TPM.

Naive Bayes: Gaussian Discriminant Analysis

So far, we assume $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ are known. If they are unknown, we need to estimate them from the data. We illustrate the idea with the naive Bayes classifier.

- Let $Z \sim \text{Bernoulli}(\phi)$ be a random variable that indicates the class, where $Z = 1$ or 0:

$$\boldsymbol{X} \mid Z = 1 \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}), \quad \boldsymbol{X} \mid Z = 0 \sim N(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}).$$

- The likelihood function is

$$L(\boldsymbol{\mu}_1, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}, \phi) = \prod_{j=1}^{n} P(\boldsymbol{x}_j, z_j) = \prod_{j=1}^{n} [f_1(\boldsymbol{x}_j)\phi]^{z_j} [f_0(\boldsymbol{x}_j)(1-\phi)]^{1-z_j}.$$

- The MLEs are

$$\hat{\phi} = n^{-1} \sum_{j=1}^{n} z_j, \ \hat{\boldsymbol{\mu}}_1 = \frac{\sum_{j=1}^{n} z_j \boldsymbol{x}_j}{\sum_{j=1}^{n} z_j}, , \ \hat{\boldsymbol{\mu}}_0 = \frac{\sum_{j=1}^{n} (1-z_j)\boldsymbol{x}_j}{\sum_{j=1}^{n} (1-z_j)},$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\left[\sum_{j=1}^{n} z_j (\boldsymbol{x}_j - \hat{\boldsymbol{\mu}}_1)(\boldsymbol{x}_j - \hat{\boldsymbol{\mu}}_1)^T + \sum_{j=1}^{n} (1-z_j)(\boldsymbol{x}_j - \hat{\boldsymbol{\mu}}_0)(\boldsymbol{x}_j - \hat{\boldsymbol{\mu}}_0)^T\right].$$

Gaussian Discriminant Analysis: Classification

We classify $\boldsymbol{x}_0$ to class 1, if $P(Z=1 \mid \boldsymbol{x}_0) \geq P(Z=0 \mid \boldsymbol{x}_0)$, where

$$P(Z=1 \mid \boldsymbol{x}_0) = \frac{\frac{\hat{\phi}}{(2\pi)^{p/2}\sqrt{\det(\hat{\boldsymbol{\Sigma}})}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_1)^T \hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_1)\right\}}{P(\boldsymbol{x}_0)},$$

$$P(Z=0 \mid \boldsymbol{x}_0) = \frac{\frac{1-\hat{\phi}}{(2\pi)^{p/2}\sqrt{\det(\hat{\boldsymbol{\Sigma}})}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_0)^T \hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_0)\right\}}{P(\boldsymbol{x}_0)}.$$

It is equivalent to

$$\frac{\hat{\phi}\exp\left\{-\frac{1}{2}(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_1)^T \hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_1)\right\}}{\left(1-\hat{\phi}\right)\exp\left\{-\frac{1}{2}(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_0)^T \hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_0)\right\}} \geq 1.$$

Gaussian Discriminant Analysis: Decision Boundary

We can also say that we classify $\boldsymbol{x}_0$ to class 1, if

$$\begin{aligned}\log\left(\frac{\hat{\phi}}{1-\hat{\phi}}\right) &\geq \frac{1}{2}\left(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_1\right)^T\hat{\boldsymbol{\Sigma}}^{-1}\left(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_1\right)\\ &\quad -\frac{1}{2}\left(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_0\right)^T\hat{\boldsymbol{\Sigma}}^{-1}\left(\boldsymbol{x}_0-\hat{\boldsymbol{\mu}}_2\right)\\ &= \left(\hat{\boldsymbol{\mu}}_2-\hat{\boldsymbol{\mu}}_1\right)^T\hat{\boldsymbol{\Sigma}}^{-1}\boldsymbol{x}_0+\frac{1}{2}\left(\hat{\boldsymbol{\mu}}_1^T\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\mu}}_1-\hat{\boldsymbol{\mu}}_0^T\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\mu}}_0\right),\end{aligned}$$

which is a quadratic function in $\boldsymbol{x}_0$. Hence, the decision boundary is linear in $\boldsymbol{x}_0$.

We have assume $\boldsymbol{\Sigma}$ to be the same for different groups. You can use other assumptions, e.g., different $\boldsymbol{\Sigma}$, diagonal $\boldsymbol{\Sigma}$, etc.

Fisher's Linear Discriminant Analysis

We need to know or estimate $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ in order to use the classifiers based on ECM. Fisher's linear discriminant analysis (LDA) does not assume any distributional assumption. The idea is that a linear combination of $\boldsymbol{x}$ ($y = \boldsymbol{a}^T\boldsymbol{x}$) should carry enough information for discrimination. If we observe

1. $\boldsymbol{x}_{11}, ..., \boldsymbol{x}_{1n_1}$ from π_1,
2. $\boldsymbol{x}_{21}, ..., \boldsymbol{x}_{2n_2}$ from π_2,

then we should find $\boldsymbol{a}$ such that the linear combination of the first class should be as far from the linear combination of the second class as possible. We need to assume two classes have the same covariance matrix, i.e., $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$.

MANOVA-Like Idea

Let $\bar{\boldsymbol{x}}_i$ be the sample mean of class π_i, and $\bar{\boldsymbol{x}}$ be the sample mean ignoring classes. Then,

$$\begin{aligned}
\sum_{i=1}^{2}\sum_{j=1}^{n_j}\left(\boldsymbol{a}^T\boldsymbol{x}_{ij}-\boldsymbol{a}^T\bar{\boldsymbol{x}}\right)^2 &= \sum_{i=1}^{2}\sum_{j=1}^{n_j}\boldsymbol{a}^T\left(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}}\right)\left(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}}\right)^T\boldsymbol{a} \\
&= \boldsymbol{a}^T\left[\sum_{i=1}^{2}\sum_{j=1}^{n_j}\left(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}}\right)\left(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}}\right)^T\right]\boldsymbol{a} \\
&= \boldsymbol{a}^T\left[\boldsymbol{W}+\boldsymbol{B}\right]\boldsymbol{a},
\end{aligned}$$

where

$$\begin{aligned}
\boldsymbol{W} &= \sum_{i=1}^{2}\sum_{j=1}^{n_i}\left(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}}_i\right)\left(\boldsymbol{x}_{ij}-\bar{\boldsymbol{x}}_i\right)^T, \\
\boldsymbol{B} &= \sum_{i=1}^{2} n_i\left(\bar{\boldsymbol{x}}_i-\bar{\boldsymbol{x}}\right)\left(\bar{\boldsymbol{x}}_i-\bar{\boldsymbol{x}}\right)^T.
\end{aligned}$$

Within Versus Between Variation

In LDA, we want to maximize

$$\frac{\boldsymbol{a}^T \boldsymbol{B} \boldsymbol{a}}{\boldsymbol{a}^T \boldsymbol{W} \boldsymbol{a}}.$$

The maximizer $\boldsymbol{a}$ is the eigenvector of $\boldsymbol{W}^{-1}\boldsymbol{B}$ corresponding to the largest eigenvalue of $\boldsymbol{W}^{-1}\boldsymbol{B}$. In this case

$$\boldsymbol{a} \quad = \quad \boldsymbol{W}^{-1} \left(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2 \right),$$

which is called Fisher's linear discriminant function.

Fisher's LDA

We classify $\boldsymbol{x}_0$ to π_1 if $\boldsymbol{a}^T\boldsymbol{x}_0$ is closer to the mean of $\boldsymbol{a}^T\boldsymbol{X}_1$ (first class) than the mean of $\boldsymbol{a}^T\boldsymbol{X}_2$ (second class). That is,

$$\left|(\bar{\boldsymbol{x}}_1-\bar{\boldsymbol{x}}_2)^T\boldsymbol{W}^{-1}(\boldsymbol{x}_0-\bar{\boldsymbol{x}}_1)\right| \quad < \quad \left|(\bar{\boldsymbol{x}}_1-\bar{\boldsymbol{x}}_2)^T\boldsymbol{W}^{-1}(\boldsymbol{x}_0-\bar{\boldsymbol{x}}_2)\right|.$$

This is equivalent to classifying $\boldsymbol{x}_0$ to π_1 if

$$(\bar{\boldsymbol{x}}_1-\bar{\boldsymbol{x}}_2)^T\boldsymbol{W}^{-1}\boldsymbol{x}_0 \quad \geq \quad \frac{1}{2}(\bar{\boldsymbol{x}}_1-\bar{\boldsymbol{x}}_2)^T\boldsymbol{W}^{-1}(\bar{\boldsymbol{x}}_1+\bar{\boldsymbol{x}}_2).$$

The decision boundary is linear in $\boldsymbol{x}_0$. Hence, it is called Fisher's linear discriminant analysis (LDA).

Two Normal Populations

Now we assume that $\boldsymbol{X} \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ in π_1, and $\boldsymbol{X} \sim N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ in π_2. We will derive classifiers for two multivariate normal populations.

1. Case I: $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$,
2. Case II: $\boldsymbol{\Sigma}_1 \neq \boldsymbol{\Sigma}_2$.

Case I: $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$

Suppose that

1. $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$,
2. $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, and $\boldsymbol{\Sigma}$ are known.

The classifier that minimizes ECM is the one that classifies $\boldsymbol{x}_0$ to π_1 if

$$\frac{f_1(\boldsymbol{x}_0)}{f_2(\boldsymbol{x}_0)} = \frac{\exp\left\{-\frac{1}{2}(\boldsymbol{x}_0-\boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_0-\boldsymbol{\mu}_1)\right\}}{\exp\left\{-\frac{1}{2}(\boldsymbol{x}_0-\boldsymbol{\mu}_2)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_0-\boldsymbol{\mu}_2)\right\}} \leq \frac{c(1\mid 2)}{c(2\mid 1)}\frac{p_2}{p_1}.$$

It is equivalent to classifying $\boldsymbol{x}_0$ to π_1 if

$$(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}_0 - \frac{1}{2}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1+\boldsymbol{\mu}_2) \leq \log\left[\frac{c(1\mid 2)}{c(2\mid 1)}\frac{p_2}{p_1}\right].$$

Estimated $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$

$\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, and $\boldsymbol{\Sigma}$ are typically unknown. We need to estimate them in order to classify $\boldsymbol{x}_0$.

- If we observe $\boldsymbol{x}_{11}$, ..., $\boldsymbol{x}_{1n_1}$ from π_1 and $\boldsymbol{x}_{21}$, ..., $\boldsymbol{x}_{2n_2}$ from π_2, then

$$\hat{\boldsymbol{\mu}}_1 = \bar{\boldsymbol{x}}_1, \quad \hat{\boldsymbol{\mu}}_2 = \bar{\boldsymbol{x}}_2,$$
$$\boldsymbol{S}_{\text{pooled}} = \frac{(n_1 - 1)\,\boldsymbol{S}_1 + (n_2 - 1)\,\boldsymbol{S}_2}{n_1 + n_2 - 2}.$$

- We allocate $\boldsymbol{x}_0$ to π_1 if

$$(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{S}_{\text{pooled}}^{-1} \boldsymbol{x}_0 - \frac{1}{2}(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{S}_{\text{pooled}}^{-1} (\bar{\boldsymbol{x}}_1 + \bar{\boldsymbol{x}}_2)$$
$$\geq \log\left[\frac{c\,(1 \mid 2)}{c\,(2 \mid 1)} \frac{p_2}{p_1}\right].$$

Connection to Fisher's LDA

Let $y = \boldsymbol{a}^T \boldsymbol{x}_0$ and $\hat{y} = \hat{\boldsymbol{a}}^T \boldsymbol{x}_0$, where $\boldsymbol{a} = \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$ and $\hat{\boldsymbol{a}} = \boldsymbol{S}_{\text{pooled}}^{-1} (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)$. Consider a special case where

$$\frac{c(1 \mid 2)}{c(2 \mid 1)} \frac{p_2}{p_1} = 1.$$

We allocate $\boldsymbol{x}_0$ to π_1 if

$$\underbrace{(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{S}_{\text{pooled}}^{-1}}_{\hat{\boldsymbol{a}}^T} \boldsymbol{x}_0 \geq \frac{1}{2} \underbrace{(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{S}_{\text{pooled}}^{-1}}_{\hat{\boldsymbol{a}}^T} (\bar{\boldsymbol{x}}_1 + \bar{\boldsymbol{x}}_2).$$

Note that

$$\boldsymbol{W} = \sum_{i=1}^{2} \sum_{j=1}^{n_i} (\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i) (\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)^T,$$

$$\boldsymbol{S}_{\text{pooled}} = \frac{(n_1 - 1) \boldsymbol{S}_1 + (n_2 - 1) \boldsymbol{S}_2}{n_1 + n_2 - 2} = \frac{\boldsymbol{W}}{n_1 + n_2 - 2}.$$

Hence, it is identical to Fisher's LDA.

Connection to Naive Bayes

Consider another special case where

$$\frac{c\,(1 \mid 2)}{c\,(2 \mid 1)} \quad = \quad 1.$$

We allocate $\boldsymbol{x}_0$ to π_1 if

$$(\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{S}_{\text{pooled}}^{-1} \boldsymbol{x}_0 - \frac{1}{2} (\bar{\boldsymbol{x}}_1 - \bar{\boldsymbol{x}}_2)^T \boldsymbol{S}_{\text{pooled}}^{-1} (\bar{\boldsymbol{x}}_1 + \bar{\boldsymbol{x}}_2) \quad \geq \quad \log \left(\frac{\hat{p}_2}{\hat{p}_1} \right).$$

This is equivalent to the Gaussian discriminant analysis.

Case II: $\boldsymbol{\Sigma}_1 \neq \boldsymbol{\Sigma}_2$

Suppose that $\boldsymbol{\Sigma}_1 \neq \boldsymbol{\Sigma}_2$. The minimum ECM classifier allocates $\boldsymbol{x}_0$ to π_1, if

$$\frac{\left[\det\left(\boldsymbol{\Sigma}_1\right)\right]^{-p/2} \exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_0-\boldsymbol{\mu}_1\right)^T \boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}_0-\boldsymbol{\mu}_1\right)\right\}}{\left[\det\left(\boldsymbol{\Sigma}_2\right)\right]^{-p/2} \exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_0-\boldsymbol{\mu}_2\right)^T \boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}_0-\boldsymbol{\mu}_2\right)\right\}} \geq \frac{c\left(1 \mid 2\right)}{c\left(2 \mid 1\right)} \frac{p_2}{p_1}.$$

This is equivalent to allocating $\boldsymbol{x}_0$ to π_1, if

$$-\frac{1}{2}\boldsymbol{x}_0^T\left(\boldsymbol{\Sigma}_1^{-1}-\boldsymbol{\Sigma}_2^{-1}\right)\boldsymbol{x}_0+\left(\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}_1^{-1}-\boldsymbol{\mu}_2^T\boldsymbol{\Sigma}_2^{-1}\right)\boldsymbol{x}_0-k \geq \log\left[\frac{c\left(1 \mid 2\right)}{c\left(2 \mid 1\right)}\frac{p_2}{p_1}\right],$$

where

$$k = \frac{p}{2}\log\left[\frac{\det\left(\boldsymbol{\Sigma}_1\right)}{\det\left(\boldsymbol{\Sigma}_2\right)}\right]+\frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1-\frac{1}{2}\boldsymbol{\mu}_2^T\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\mu}_2.$$

The decision boundary is a quadratic function of $\boldsymbol{x}$. Hence, this is called quadratic discriminant analysis (QDA).

Logistic Model For Two Populations

Let $Y_j \mid \boldsymbol{X}_j \sim \text{Bernoulli}\,(\phi_j)$ be a random variable that indicates the class of subject j, where $Y_j = 1$ for class π_1 and $Y_j = 0$ for class π_0. In logistic regression, we model ϕ_j as

$$\log\left(\frac{\phi_j}{1-\phi_j}\right) = \alpha + \boldsymbol{\beta}^T \boldsymbol{x}_j,$$

that is

$$\phi_j = P\left(Y_j = 1 \mid \boldsymbol{x}_j\right) = \frac{\exp\left(\alpha + \boldsymbol{\beta}^T \boldsymbol{x}_j\right)}{1 + \exp\left(\alpha + \boldsymbol{\beta}^T \boldsymbol{x}_j\right)},$$

$$1 - \phi_j = P\left(Y_j = 0 \mid \boldsymbol{x}_j\right) = \frac{1}{1 + \exp\left(\alpha + \boldsymbol{\beta}^T \boldsymbol{x}_j\right)}.$$

Maximum Likelihood Estimator

We often assume that $\{Y_j\}$ are mutually independent given $\boldsymbol{X}$. Then, the log-likelihood function is

$$\begin{aligned}\ell(\alpha, \boldsymbol{\beta}) &= \sum_{j=1}^{n} \left[I(y_j = 1) \log \phi_j + I(y_j = 0) \log(1 - \phi_j) \right] \\ &= \sum_{j=1}^{n} \left\{ I(y_j = 1) \left(\alpha + \boldsymbol{\beta}^T \boldsymbol{x}_j\right) - \log\left[1 + \exp\left(\alpha + \boldsymbol{\beta}^T \boldsymbol{x}_j\right)\right] \right\},\end{aligned}$$

where $I(y_j = k) = 1$ if $y_j = k$, and 0 otherwise. The estimator $\left(\hat{\alpha}, \hat{\boldsymbol{\beta}}\right)$ maximizes $L(\alpha, \boldsymbol{\beta})$ using numerical methods.
We often allocate $\boldsymbol{x}_0$ to π_1 if

$$\frac{\exp\left(\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{x}_0\right)}{1 + \exp\left(\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{x}_0\right)} \geq 0.5.$$

It is equivalent to $\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{x}_0 \geq 0$, a linear decision boundary.

Penalized Logistic Regression

It is also popular to consider penalized logistic regression. Instead of maximizing the log-likelihood $\ell(\alpha, \boldsymbol{\beta})$, we minimize

$$\min_{\alpha,\beta} \underbrace{-\sum_{j=1}^{n} \left\{ I(y_j = 1)(\alpha + \boldsymbol{\beta}^T \boldsymbol{x}_j) - \log\left[1 + \exp(\alpha + \boldsymbol{\beta}^T \boldsymbol{x}_j)\right] \right\}}_{-1 \times \ell(\alpha, \boldsymbol{\beta})} + \text{penalty term}.$$

Two popular penalty terms are

$$\text{LASSO}: \quad \lambda \sum_{i=1}^{p} |\beta_j|,$$

$$\text{Ridge}: \quad \lambda \boldsymbol{\beta}^T \boldsymbol{\beta},$$

where $\lambda \geq 0$ is a tuning parameter, often selected by cross validation or information criterion. The decision boundary remains linear.

Limitation

A limitation of classic logistic regression is that we need to specify the form of the covariates $\boldsymbol{x}$ in $\boldsymbol{\beta}^T\boldsymbol{x}$ ourselves.
To overcome the limitation, we need to know how $\boldsymbol{\beta}$ is estimated. The gradient of the log-likelihood function is

$$\frac{\partial L(\alpha, \boldsymbol{\beta})}{\partial \beta_k} = \sum_{j=1}^{n} (y_j - \phi_j) x_{jk}.$$

If the gradient descend algorithm is used to update $\boldsymbol{\beta}$, the initial guess $\boldsymbol{\beta}^{(0)} = \mathbf{0}$ of $\boldsymbol{\beta}$ is improved by

$$\boldsymbol{\beta}^{(1)} = \boldsymbol{\beta}^{(0)} - \sum_{j=1}^{n} \left(y_j - \phi_j^{(0)}\right) \boldsymbol{x}_j,$$

$$\boldsymbol{\beta}^{(2)} = \boldsymbol{\beta}^{(1)} - \sum_{j=1}^{n} \left(y_j - \phi_j^{(1)}\right) \boldsymbol{x}_j = \boldsymbol{\beta}^{(0)} - \sum_{j=1}^{n} \left(\phi_j^{(1)} - \phi_j^{(0)}\right) \boldsymbol{x}_j,$$

until no updates can be made.

Euclidean Inner Product

- Hence, we can expect the final estimate of $\boldsymbol{\beta}$ to be a linear combination of $\boldsymbol{x}_j$, say

$$\hat{\boldsymbol{\beta}} = \sum_{j=1}^{n} d_j \boldsymbol{x}_j,$$

  for some coefficients d_j.
- We allocate $\boldsymbol{x}_0$ to π_1 if $\hat{\alpha} + \hat{\boldsymbol{\beta}}^T \boldsymbol{x}_0 \geq 0$. It is equivalent to

$$\hat{\alpha} + \sum_{j=1}^{n} d_j \boldsymbol{x}_j^T \boldsymbol{x}_0 \geq 0,$$

  which depends on the Euclidean inner product $\boldsymbol{x}_j^T \boldsymbol{x}_0 = \langle \boldsymbol{x}_j, \boldsymbol{x}_0 \rangle$.

New Features

If we are not satisfied with the results using $\boldsymbol{x}$, we can create a vector of new features $\boldsymbol{\delta}(\boldsymbol{x}_j)$ and fit a new logistic model

$$\log\left(\frac{\phi_j}{1-\phi_j}\right) = \psi + \boldsymbol{\gamma}^T \boldsymbol{\delta}(\boldsymbol{x}_j).$$

Then, we allocate $\boldsymbol{x}_0$ to π_1 if

$$\hat{\psi} + \sum_{j=1}^{n} d_j^* \langle \boldsymbol{\delta}(\boldsymbol{x}_j), \boldsymbol{\delta}(\boldsymbol{x}_0) \rangle \geq 0,$$

for some coefficients d_j^*.

Kernel Function and Kernel Matrix

A function $\kappa(\boldsymbol{x}, \boldsymbol{z})$ is a kernel function if

1. it is symmetric, $\kappa(\boldsymbol{x}, \boldsymbol{z}) = \kappa(\boldsymbol{z}, \boldsymbol{x})$,
2. the kernel matrix $\boldsymbol{K}$ with (i, j)th entry $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is positive semi-definite for all $\boldsymbol{x}_1$, ..., $\boldsymbol{x}_n$.

If $\kappa(\boldsymbol{x}, \boldsymbol{z})$ is a kernel function, then you must be able to find a function $\boldsymbol{\delta}()$ such that

$$\kappa(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{\delta}^T(\boldsymbol{x})\boldsymbol{\delta}(\boldsymbol{z}).$$

As a kernel function, we will have an eigen-decomposition

$$\kappa(\boldsymbol{x}, \boldsymbol{z}) = \sum_{m=1}^{\infty} \rho_m e_m(\boldsymbol{x}) e_m(\boldsymbol{z}),$$

for some eigenvalues ρ_k and eigenfunctions $e_m(\boldsymbol{x})$.

Kernel Trick

Kernel trick is a commonly used trick to create new features from your original observed features.

- If the new features are created using $\boldsymbol{\delta}(\boldsymbol{x})$, then we allocate $\boldsymbol{x}_0$ to π_1 if

$$\hat{\psi} + \sum_{j=1}^{n} d_j^* \langle \boldsymbol{\delta}(\boldsymbol{x}_j), \boldsymbol{\delta}(\boldsymbol{x}_0) \rangle \quad \geq \quad 0.$$

- If κ is the corresponding kernel function, the classifier is equivalent to

$$\hat{\psi} + \sum_{j=1}^{n} d_j^* \kappa(\boldsymbol{x}_0, \boldsymbol{x}_j) \quad \geq \quad 0.$$

- This means that we only need to choose the kernel function κ, if we need to create new features from the raw covariates $\boldsymbol{x}$.

Benefits of Kernel Trick

The eigen-decomposition implies that

$$\begin{aligned}\sum_{j=1}^{n} d_j^* \kappa\left(\boldsymbol{x}_0, \boldsymbol{x}_j\right) &= \sum_{j=1}^{n} d_j^* \sum_{m=1}^{\infty} \rho_m e_m\left(\boldsymbol{x}_0\right) e_m\left(\boldsymbol{x}_j\right) \\ &= \sum_{m=1}^{\infty} \underbrace{\left(\sum_{j=1}^{n} d_j^* \sqrt{\rho_m} e_m\left(\boldsymbol{x}_j\right)\right)}_{\text{our new coefficients } d_j^*} \underbrace{\sqrt{\rho_m} e_m\left(\boldsymbol{x}_0\right)}_{\text{our new feature } \phi_m(\boldsymbol{x})}\end{aligned}$$

That is, we are using the vector of new features $\boldsymbol{\delta}\left(\boldsymbol{x}\right)$, possibly of infinite dimension, to model ϕ_j.

Reproducing Kernel Hilbert Space Regression

- The kernel trick means that we first choose a kernel function, compute the kernel matrix $\boldsymbol{K}$ using the data matrix $\boldsymbol{X}$, and use $\boldsymbol{K}$ as the data matrix to fit our logistic regression model.
- However, when the data matrix $\boldsymbol{X}$ is of dimension $n \times p$, then the kernel matrix $\boldsymbol{K}$ is $n \times n$ and we have n regression coefficients to estimate. We often cannot obtain a unique estimator.
- Hence, we often use penalized maximum likelihood: the coefficients can be estimated by minimizing

$$-1 \times \text{log-likelihood} + \lambda \boldsymbol{\beta}^T \boldsymbol{K} \boldsymbol{\beta},$$

where λ is the tuning parameter. It is often chosen by cross validation.

Classify New Observation

- Suppose that $\kappa(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{\delta}^T(\boldsymbol{x})\,\boldsymbol{\delta}(\boldsymbol{z})$. Then, it is equivalent to transforming our original data $\boldsymbol{x}$ to new data $\boldsymbol{\delta}(\boldsymbol{x})$.
- Our model is then

$$P(Y_j = 1 \mid \boldsymbol{x}_j) = \frac{\exp\left[\psi + \boldsymbol{\gamma}^T \boldsymbol{\delta}(\boldsymbol{x}_j)\right]}{1 + \exp\left[\psi + \boldsymbol{\gamma}^T \boldsymbol{\delta}(\boldsymbol{x}_j)\right]},$$

- We classify a new observation $\boldsymbol{x}_0$ to $y_0 = 1$ if

$$\begin{aligned} 0 &\leq \hat{\boldsymbol{\gamma}}^T \boldsymbol{\delta}(\boldsymbol{x}_0) + \hat{\psi} \\ &= \left[\sum_{j=1}^{n} d_j^* \boldsymbol{\delta}(\boldsymbol{x}_j)\right]^T \boldsymbol{\delta}(\boldsymbol{x}_0) + \hat{\psi} \\ &= \sum_{j=1}^{n} d_j^* \left[\boldsymbol{\delta}^T(\boldsymbol{x}_j)\,\boldsymbol{\delta}(\boldsymbol{x}_0)\right] + \hat{\psi} \\ &= \sum_{j=1}^{n} d_j^* \kappa(\boldsymbol{x}_j, \boldsymbol{x}_0) + \hat{\psi}. \end{aligned}$$

Motivation: Margin

Suppose that we code Y as $\{-1, 1\}$. We define a linear classifier as

$$\hat{y} = \begin{cases} 1, & \text{if } \boldsymbol{w}^T\boldsymbol{x} + b \geq 0, \\ -1, & \text{if } \boldsymbol{w}^T\boldsymbol{x} + b < 0. \end{cases}$$

The hyperplane $\boldsymbol{w}^T\boldsymbol{x} + b = 0$ is our decision boundary. The distance of a point $\boldsymbol{x}_0$ to the decision boundary is

$$\frac{\left|\boldsymbol{w}^T\boldsymbol{x}_0 + b\right|}{\|\boldsymbol{w}\|} = \frac{y_0\left(\boldsymbol{w}^T\boldsymbol{x}_0 + b\right)}{\|\boldsymbol{w}\|}.$$

We can choose the decision boundary such that the distance (margin) is maximized.

Support Vector: Brief Intro

We want to maximize the margin such that we can correct classify all data points:

$$\max_{w,b} \left[\min_j \frac{y_j \left(\boldsymbol{w}^T \boldsymbol{x}_j + b \right)}{\|\boldsymbol{w}\|} \right], \quad \text{s.t.} \quad y_j \left(\boldsymbol{w}^T \boldsymbol{x}_j + b \right) \geq 0, \ \forall j.$$

It is equivalent to

$$\min_{w,b} \boldsymbol{w}^T \boldsymbol{w}, \quad \text{s.t.} \quad y_j \left(\boldsymbol{w}^T \boldsymbol{x}_j + b \right) \geq 1, \ \forall j.$$

Some data points must satisfy $y_j \left(\boldsymbol{w}^T \boldsymbol{x}_j + b \right) = 1$. These points are called support vectors.

Hinge Loss: Brief Intro

In practice, we often cannot correctly classify all points. Hence, we change our optimization problem and allow some small errors:

$$\min_{w,b,\xi_j} \boldsymbol{w}^T\boldsymbol{w} + C\sum_{j=1}^{n}\xi_j, \quad \text{s.t.} \quad y_j\left(\boldsymbol{w}^T\boldsymbol{x}_j + b\right) \geq 1-\xi_j, \ , \xi_j \geq 0, \ \forall j,$$

where $\{\xi_j\}$ are known as the slack variables, and C is a tuning parameter.

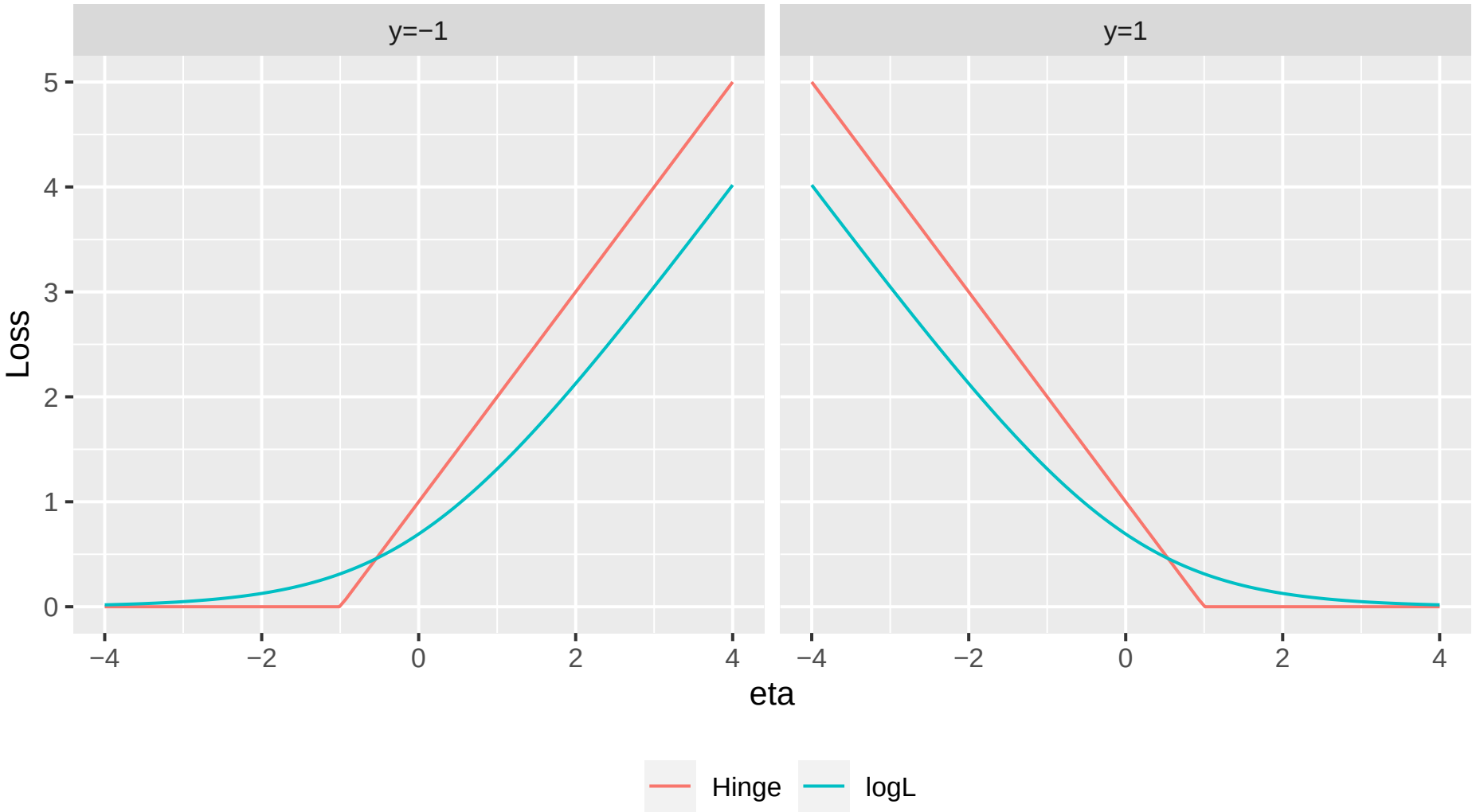
It is equivalent to

$$\min_{w,b} \underbrace{\sum_{j=1}^{n}\max\left\{0, \ 1-y_j\left(\boldsymbol{w}^T\boldsymbol{x}_j+b\right)\right\}}_{\text{Hinge loss}} + C^{-1}\boldsymbol{w}^T\boldsymbol{w}.$$

In contrast, the penalized logistic regression uses the log-likelihood loss

$$\min_{\alpha,\beta} -\sum_{j=1}^{n}\left\{I\left(y_j=1\right)\left(\alpha+\boldsymbol{\beta}^T\boldsymbol{x}_j\right) - \log\left[1+\exp\left(\alpha+\boldsymbol{\beta}^T\boldsymbol{x}_j\right)\right]\right\} + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}.$$

Hinge Loss Vs Log-Likelihood Loss



Kernel Trick

The solution of

$$\min_{w,b} \sum_{j=1}^{n} \max\left\{0,\ 1 - y_j\left(\boldsymbol{w}^T\boldsymbol{x}_j + b\right)\right\} + C^{-1}\boldsymbol{w}^T\boldsymbol{w}$$

is of the form

$$\hat{\boldsymbol{w}} \quad = \quad \sum_{j=1}^{n} d_j y_j \boldsymbol{x}_j$$

for some coefficients d_j. We classify a new observation $\boldsymbol{x}_0$ to $y_0 = 1$ if

$$0 \quad \leq \quad \hat{\boldsymbol{w}}^T\boldsymbol{x}_0 + \hat{b} = \sum_{j=1}^{n} d_j y_j \left(\boldsymbol{x}_j^T\boldsymbol{x}_0\right) + \hat{b}.$$

Hence, the support vector machine often uses the kernel trick that replaces $\boldsymbol{x}_j^T\boldsymbol{x}_0$ by some kernel $\kappa\left(\boldsymbol{x}_j, \boldsymbol{x}_0\right)$.

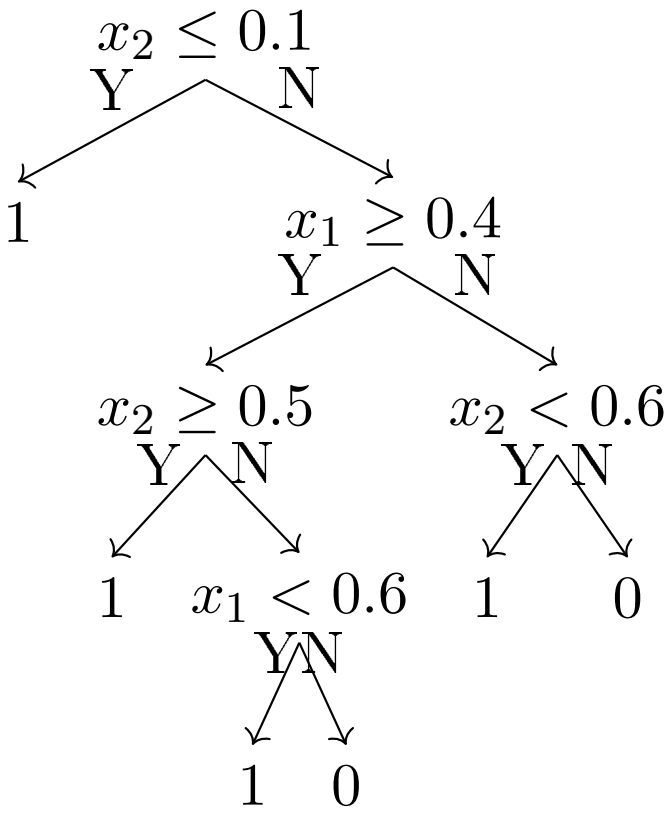
Support Vector Machine

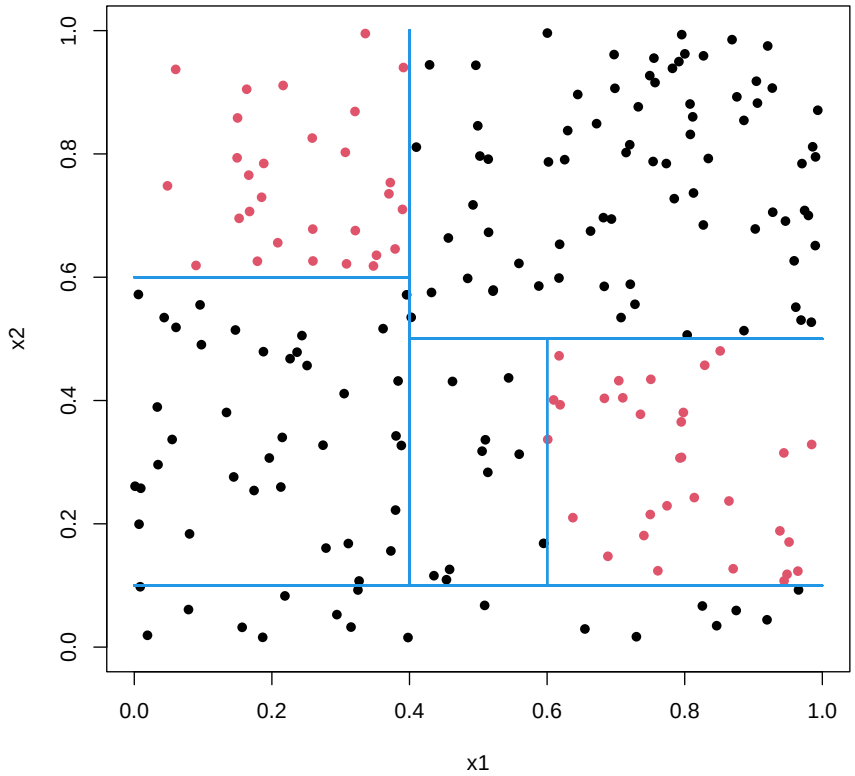
- The kernel trick means that we first choose a kernel function, compute the kernel matrix $\boldsymbol{K}$ using the data matrix $\boldsymbol{X}$, and use $\boldsymbol{K}$ as the data matrix to train our support vector machine.
- Suppose that $\kappa(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{\delta}^T(\boldsymbol{x})\boldsymbol{\delta}(\boldsymbol{z})$. Then, it is equivalent to transforming our original data $\boldsymbol{x}$ to new data $\boldsymbol{\delta}(\boldsymbol{x})$.
- We classify a new observation $\boldsymbol{x}_0$ to $y_0 = 1$ if

$$\begin{aligned} 0 &\leq \hat{\boldsymbol{w}}^T \boldsymbol{\delta}(\boldsymbol{x}_0) + \hat{b} \\ &= \left[\sum_{j=1}^{n} d_j^* y_j \boldsymbol{\delta}(\boldsymbol{x}_j)\right]^T \boldsymbol{\delta}(\boldsymbol{x}_0) + \hat{b} \\ &= \sum_{j=1}^{n} d_j^* y_j \left[\boldsymbol{\delta}^T(\boldsymbol{x}_j)\boldsymbol{\delta}(\boldsymbol{x}_0)\right] + \hat{b} \\ &= \sum_{j=1}^{n} d_j^* y_j \kappa(\boldsymbol{x}_j, \boldsymbol{x}_0) + \hat{b}. \end{aligned}$$

Classification Tree

A classification tree is a hierarchical method of partitioning the sample space. Every partition corresponds to a class.





Best Partition

The main idea of a tree method is that, at each step, we pick a binary split that best partitions the data. For classification trees, we often use the Gini index

$$G = \sum_{k=1}^{K} \hat{p}_{mk} \left(1 - \hat{p}_{mk}\right),$$

or the cross entropy

$$G = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk},$$

to measure what is the best partition, where $\hat{p}_{mk}$ is the proportion of observations in the mth region that are from the kth class.

- If $\hat{p}_{mk}$ are close to 0 or 1, then G is small.
- We want to minimize G.

How To Grow A Tree

We still need to decide

1. tree depth: when to stop split the tree,
2. leaf size: number of observations in a leaf (node).

A tree can overfit the data. Hence, we can

- grow a large tree until each terminal node has few observations or the tree is tall enough.
- use cross validation to decide when we can stop growing the tree.

A Tree Versus A Forest

A random forest grows multiple trees and combine them together to make the final decision.

Algorithm 1: Random forest

```
1 Specify an integer B ;
2 for each integer i from 1 to B do
3 |   Sample with replacement a subset of the original data ;
4 |   Sample without replacement a subset of covariates/features ;
5 |   Grow a classification tree ;
6 end
7 Compute the proprotion of trees that classify an object to a class.
```

Other Methods

There are many more classification methods such as

- Boosting methods,
- XGboost (state-of-the art),
- Neural networks.

Imbalanced Groups

Suppose that we have observed

Yes	No
100	9900

We can simply say we assign all classes to "No", then the accuracy is 99%.

1. Downsampling: We sample a subset from "No".
2. Oversampling: "duplicate" the group "Yes" until the proportions are approximately equal.
3. Weight the observations.

ECM for Several Populations

It is possible to generalize the above methods from two populations to several populations.

The ECM for two classes is

$$\text{ECM} = c(2 \mid 1) P(2 \mid 1) p_1 + c(1 \mid 2) P(1 \mid 2) p_2.$$

Suppose that we have more than two groups. The ECM of misclassifying π_i to a wrong class is

$$\text{ECM}(i) = \sum_{k \neq i} c(k \mid i) P(k \mid i) p_i.$$

The overall ECM for g classes is

$$\text{ECM} = \sum_{i=1}^{g} \left[\sum_{k \neq i} c(k \mid i) P(k \mid i) \right] p_i.$$

Minimizing ECM

Result 11.5: Min ECM for Several Classes

The classification regions that minimize the overall ECM are by allocating $\boldsymbol{x}_0$ to class π_k, $k = 1, ..., g$, for which $\sum_{k \neq i} p_i f_i(\boldsymbol{x}_0) c(k \mid i)$ is smallest.

- If all $p_i c(k \mid i)$ are equal, then it becomes the highest likelihood classifier: we allocate $\boldsymbol{x}_0$ to π_k, if $f_k(\boldsymbol{x}_0) > f_i(\boldsymbol{x}_0)$, $\forall i \neq k$.
- If all misclassification costs are equal, then it becomes the minimum TPM classifier: we allocate $\boldsymbol{x}_0$to π_k, if

$$p_k f_k(\boldsymbol{x}_0) \quad > \quad p_i f_i(\boldsymbol{x}_0), \qquad \forall i \neq k.$$

- The minimum TPM classifier is identical to the naive Bayes classifier that maximizes the posterior,

$$P(\pi_k \mid \boldsymbol{x}_0) \quad = \quad \frac{f(\boldsymbol{x}_0 \mid \pi_k) P(\pi_k)}{P(\boldsymbol{x}_0)} = \frac{f_k(\boldsymbol{x}_0) p_k}{P(\boldsymbol{x}_0)} \propto f_k(\boldsymbol{x}_0) p_k.$$

Example: Classification with Normal Populations

Suppose that $f_i(\boldsymbol{x})$ is the density function of $N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma})$. The naive Bayes classifier allocates $\boldsymbol{x}$ to π_k if

$$\log[p_k f_k(\boldsymbol{x})] = \max_i \log[p_i f_i(\boldsymbol{x})].$$

Here,

$$\begin{aligned}\log[p_k f_k(\boldsymbol{x})] &= \log p_k - \frac{p}{2}\log(2\pi) - \frac{1}{2}\log\det(\boldsymbol{\Sigma}) \\ &\quad -\frac{1}{2}\left(\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k - \boldsymbol{\mu}_k^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{\mu}_k^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k\right).\end{aligned}$$

We call

$$d_i(\boldsymbol{x}) = \boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i + \log p_i$$

the linear discrimination score. We allocate $\boldsymbol{x}$ to π_k, if $d_k(\boldsymbol{x})$ is the largest of $d_1(\boldsymbol{x})$, ..., $d_g(\boldsymbol{x})$. If we do not know $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}$, they are estimated by $\bar{\boldsymbol{x}}_i$ and $\boldsymbol{S}_{\text{pooled}}$, respectively.

Example: Classification with Normal Populations

Suppose that $f_i(\boldsymbol{x})$ is the density function of $N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$. The naive Bayes classifier allocates $\boldsymbol{x}$ to π_k if

$$\log\left[p_k f_k(\boldsymbol{x})\right] = \max_i \log\left[p_i f_i(\boldsymbol{x})\right].$$

Here,

$$\begin{aligned}\log\left[p_k f_k(\boldsymbol{x})\right] &= \log p_k - \frac{p}{2}\log(2\pi) - \frac{1}{2}\log\det(\boldsymbol{\Sigma}_k) \\ &\quad -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k).\end{aligned}$$

We call

$$d_i^Q(\boldsymbol{x}) = -\frac{1}{2}\log\det(\boldsymbol{\Sigma}_i) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i) + \log p_i$$

the quadratic discrimination score. We allocate $\boldsymbol{x}$ to π_k, if $d_k^Q(\boldsymbol{x})$ is the largest of $d_1^Q(\boldsymbol{x})$, ..., $d_g^Q(\boldsymbol{x})$. If we do not know $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$, they are estimated by $\bar{\boldsymbol{x}}_i$ and $\boldsymbol{S}_i$, respectively.

Fisher's Discriminant Analysis

Fisher's discriminant analysis can be generalized to more than two groups. We still assume $\boldsymbol{\Sigma}_1 = \cdots = \boldsymbol{\Sigma}_g = \boldsymbol{\Sigma}$.

Similar to LDA for two classes, we want to maximize

$$\frac{\boldsymbol{a}^T \boldsymbol{B} \boldsymbol{a}}{\boldsymbol{a}^T \boldsymbol{W} \boldsymbol{a}},$$

where

$$\begin{aligned} \boldsymbol{W} &= \sum_{\ell=1}^{g} \sum_{j=1}^{n_\ell} (\boldsymbol{x}_{\ell j} - \bar{\boldsymbol{x}}_\ell) (\boldsymbol{x}_{\ell j} - \bar{\boldsymbol{x}}_\ell)^T, \\ \boldsymbol{B} &= \sum_{\ell=1}^{g} n_\ell (\bar{\boldsymbol{x}}_\ell - \bar{\boldsymbol{x}}) (\bar{\boldsymbol{x}}_\ell - \bar{\boldsymbol{x}})^T. \end{aligned}$$

The maximizer $\boldsymbol{a}$ is the eigenvector of $\boldsymbol{W}^{-1}\boldsymbol{B}$ corresponding to the largest eigenvalue of $\boldsymbol{W}^{-1}\boldsymbol{B}$.

Fisher's Linear Discriminants

If we have more than two classes, using only one linear combination is often not enough. We often need to use $r > 1$ linear combinations. Let $(\lambda_i, \boldsymbol{e}_i)$ be the eigenvalue and eigenvector of $\boldsymbol{W}^{-1}\boldsymbol{B}$, where $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$.

1. Fisher's first linear discriminant is $\boldsymbol{e}_1^T \boldsymbol{x}_0$,
2. Fisher's second linear discriminant is $\boldsymbol{e}_2^T \boldsymbol{x}_0$,
3. Fisher's kth linear discriminant is $\boldsymbol{e}_k^T \boldsymbol{x}_0$.

We allocate $\boldsymbol{x}_0$ to π_k if the linear combinations $\left\{\boldsymbol{e}_j^T \boldsymbol{x}_0\right\}$ are closer to the mean of the linear combinations of the kth class. That is,

$$\sum_{j=1}^{r} \left(\boldsymbol{e}_j^T \boldsymbol{x}_0 - \boldsymbol{e}_j^T \bar{\boldsymbol{x}}_k\right)^2 \leq \sum_{j=1}^{r} \left(\boldsymbol{e}_j^T \boldsymbol{x}_0 - \boldsymbol{e}_j^T \bar{\boldsymbol{x}}_i\right)^2, \quad \forall i \neq k.$$

Linear Discriminants

If r equal to the rank of $\boldsymbol{W}^{-1}\boldsymbol{B}$, then it can be shown that

$$\begin{aligned} & \sum_{j=1}^{r}\left(\boldsymbol{e}_j^T\boldsymbol{x}_0-\boldsymbol{e}_j^T\bar{\boldsymbol{x}}_i\right)^2 \\ = \quad & \left(\boldsymbol{x}_0-\bar{\boldsymbol{x}}_i\right)^T\boldsymbol{W}^{-1}\left(\boldsymbol{x}_0-\bar{\boldsymbol{x}}_i\right) \\ = \quad & -2\left[\bar{\boldsymbol{x}}_i^T\boldsymbol{W}^{-1}\boldsymbol{x}_0-\frac{1}{2}\bar{\boldsymbol{x}}_i^T\boldsymbol{W}^{-1}\bar{\boldsymbol{x}}_i+\log\left(p_i\right)\right]+\boldsymbol{x}_0^T\boldsymbol{W}^{-1}\boldsymbol{x}_0+2\log\left(p_i\right). \end{aligned}$$

Let

$$d_i\left(\boldsymbol{x}\right) \quad = \quad \bar{\boldsymbol{x}}_i^T\boldsymbol{W}^{-1}\boldsymbol{x}-\frac{1}{2}\bar{\boldsymbol{x}}_i^T\boldsymbol{W}^{-1}\bar{\boldsymbol{x}}_i+\log\left(p_i\right).$$

It is the same as allocating $\boldsymbol{x}_0$ to π_k if $d_k\left(\boldsymbol{x}\right)$ is the largest of $d_1\left(\boldsymbol{x}\right)$, ..., $d_g\left(\boldsymbol{x}\right)$.