What we will cover today

We will introduce the concept of a linear classifier with two examples.

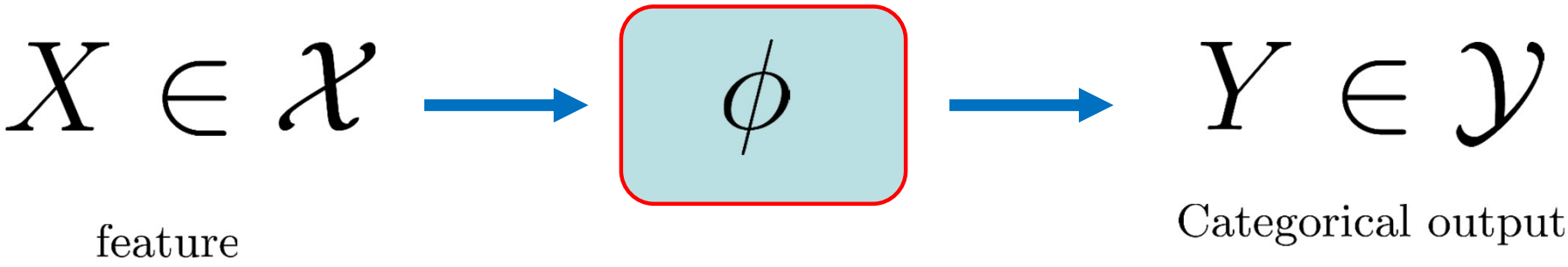
1. Linear discriminant analysis (LDA):
   - LDA models the joint distributions as a mixture of Gaussians.

2. Logistic regression:
   - Logistic regression models the class conditional distribution directly.

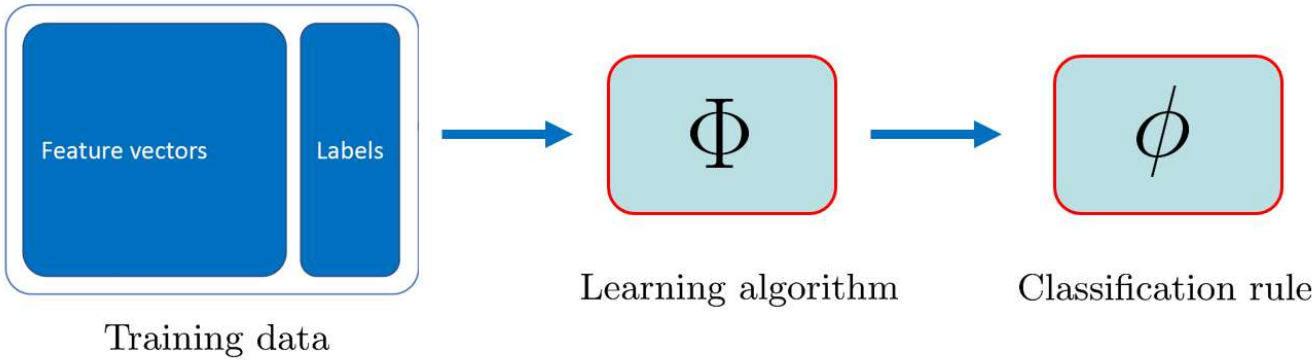
Linear classification

Classification and supervised learning

In machine learning, we want to learn a function $\phi : \mathcal{X} \to \mathcal{Y}$, which takes as input a feature vector $X \in \mathcal{X}$, and returns a categorical variable $\phi(X) \in \mathcal{Y}$ which is also known as a label.



Supervised learning: The training data is then passed to a learning algorithm. The output of the learning algorithm is the classification rule that we want.



In this lecture, we will consider learning algorithms for linear classification.

Linear classification

Let's suppose we want to learn a binary classifier $\phi : \mathcal{X} \to \{0, 1\}$

Suppose we have d continuous features $X = (X^1, \cdots, X^d) \in \mathcal{X} = \mathbb{R}^d$

Example: Penguin classification

We want to predict penguin species (Adelie or Gentoo) based on $X = (X^1, X^2)$ where

X^1 = the weight of the penguin (grams)

X^2 = the flipper length of the penguin (mm)

Goal: to find a function ϕ (classifier, or classification rule) that map a feature vector to a species.

Different types of classifiers are available. One important class of classifiers are linear classifiers (see the next slide).

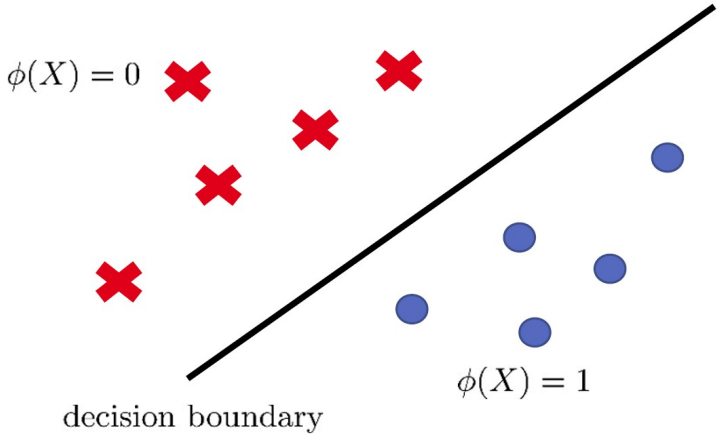
Linear classification

Let's suppose we want to learn a binary classifier $\phi : \mathcal{X} \to \{0, 1\}$

Suppose we have d continuous features $X = (X^1, \cdots, X^d) \in \mathcal{X} = \mathbb{R}^d$

Linear classifiers are a special type of classifiers that make classification decisions based on a linear combination of input (features).

More specifically, a linear classifier $\phi : \mathcal{X} \to \{0, 1\}$ cuts the feature space into two with a linear hyperplane. Given a feature vector $X \in \mathcal{X}$, the output of ϕ is completely decided by which side of the hyperplane that X lies in.



If $d = 2$ then the feature space is divided by a line.

In general, if $d \geq 2$, the feature space is divided by a $(d-1)$ dimensional hyper-plane (also called the decision boundary)

Linear classification

Let's suppose we want to learn a binary classifier $\phi : \mathcal{X} \to \{0, 1\}$

Linear classifiers are a special type of classifiers that make classification decisions based on a linear combination of input (features).

A linear classifier $\phi : \mathcal{X} \to \{0, 1\}$ is of the form

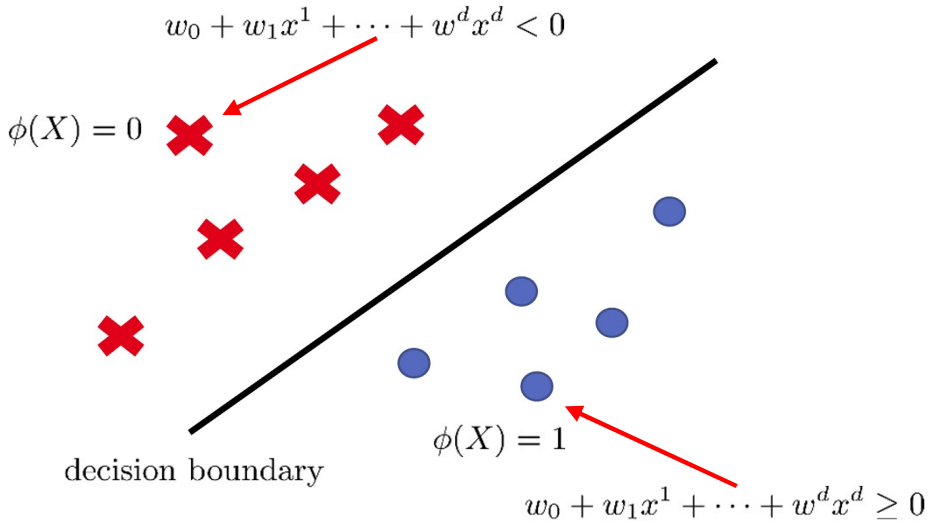
$$\phi(x) = \begin{cases} 0 & \text{if } w_0 + w_1 x^1 + \cdots + w^d x^d < 0, \\ 1 & \text{if } w_0 + w_1 x^1 + \cdots + w^d x^d \geq 0, \end{cases}$$

Note: here $w_0 + w_1 x^1 + \cdots + w^d x^d$ is a linear combination of the features.

where $w^0, \cdots, w^d$ are parameters of the classifier ϕ.

weight: $\omega := (w^1, \cdots, w^d)$

bias: w^0



Linear classification

Let's suppose we want to learn a binary classifier $\phi : \mathcal{X} \to \{0, 1\}$

Linear classifiers are a special type of classifiers that make classification decisions based on a linear combination of input (features).

A linear classifier $\phi : \mathcal{X} \to \{0, 1\}$ is of the form

$$\phi(x) = \begin{cases} 0 & \text{if } w_0 + w_1 x^1 + \cdots + w^d x^d < 0, \\ 1 & \text{if } w_0 + w_1 x^1 + \cdots + w^d x^d \geq 0, \end{cases}$$

weight: $\omega := (w^1, \cdots, w^d)$

bias: w^0

We can rewrite ϕ into the following form:

For vectors $a := (a^1, \cdots, a^d)$ and $b := (b^1, \cdots, b^d)$, define $ab^T := a^1 b^1 + \cdots + a^d b^d$.

Then the classifier is rewritten as

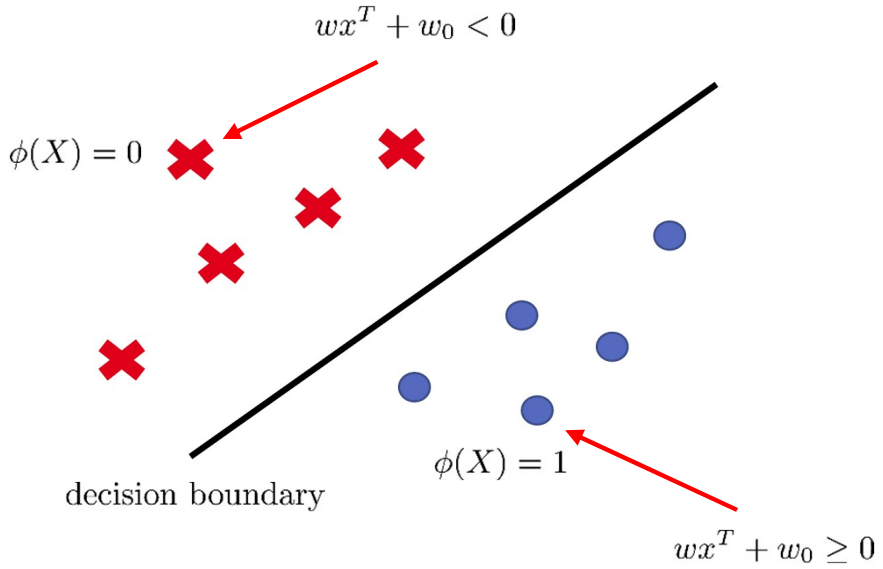
$$\phi(x) = \mathbb{1}\Big\{wx^T + w_0 \geq 0\Big\}$$

Linear classification

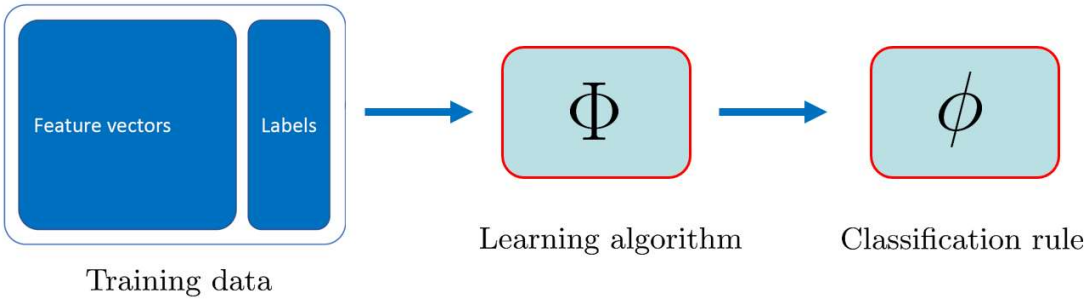
Let's suppose we want to learn a binary classifier $\phi : \mathcal{X} \to \{0, 1\}$

Linear classifiers are a special type of classifiers that make classification decisions based on a linear combination of input (features).

$$\phi(x) = \mathbb{1}\left\{wx^T + w_0 \geq 0\right\}$$



To learn ϕ we need to decide the parameters ω, ω_0.



Linear classification

Let's suppose we want to learn a binary classifier $\phi : \mathcal{X} \to \{0, 1\}$

Linear classifiers are a special type of classifiers that make classification decisions based on a linear combination of input (features).

$$\phi(x) = \mathbb{1}\Big\{wx^T + w_0 \geq 0\Big\}$$

To learn ϕ we need to decide the parameters ω, ω_0.

What's a good learning algorithm for linear classification?

- Linear discriminant analysis
- Logistic regression
- Others $\cdots$

Linear discriminant analysis

Linear discriminant analysis

To find a linear classifier, we can use the idea of linear discriminant analysis.

The main ideas:

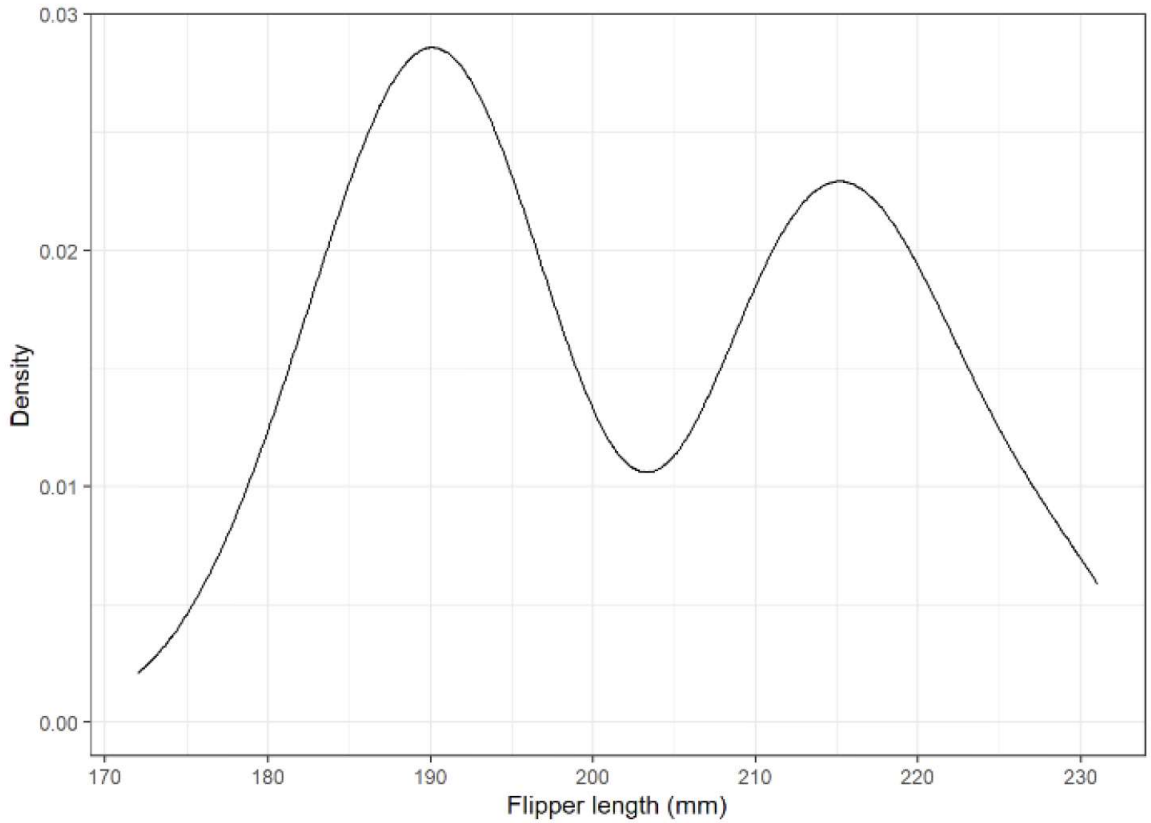
We first assume some probabilistic model (with unknown parameters) for our data (for each of the classes, the feature vectors are generated by a multivariate Gaussian distribution).

Then, under such an assumption, the optimal classifier (Bayes classifier) is a linear classifier. We can derive this classifier based on the probabilistic model.

Finally, the parameters of the probabilistic model can be determined by the maximum likelihood approach.

1. Probabilistic model for the data

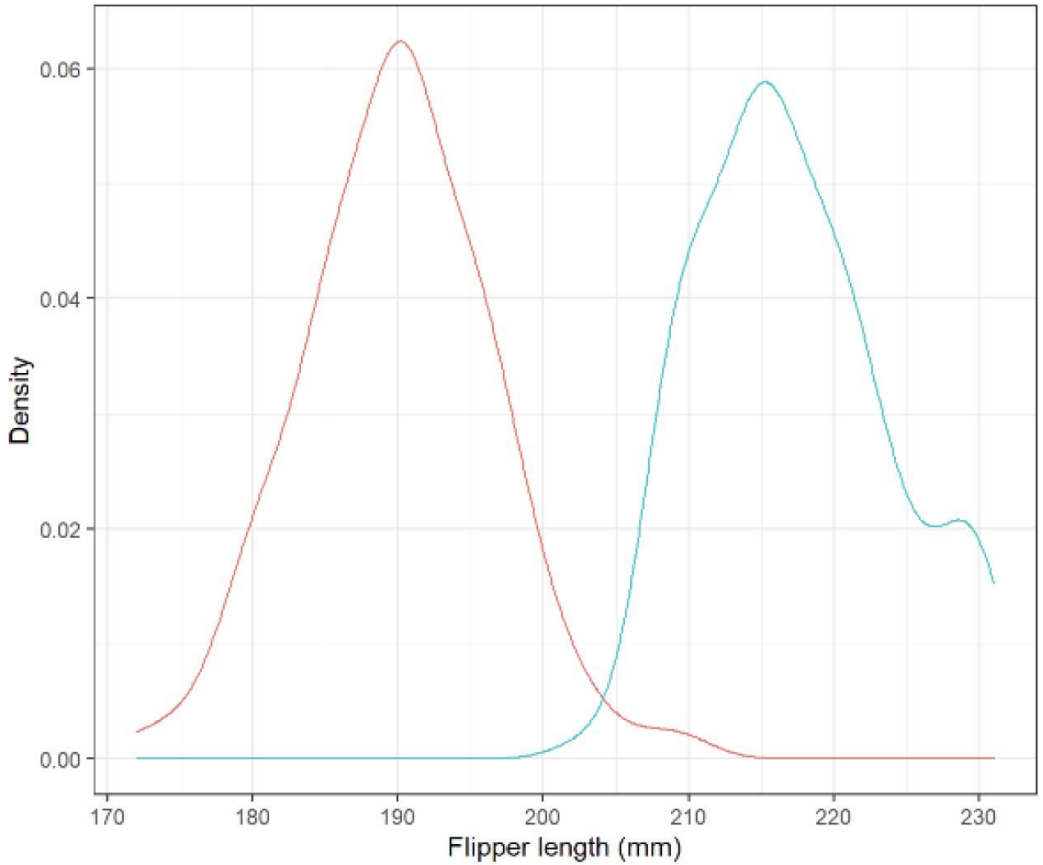
Suppose we want to learn a classifier to distinguish between Gentoo and Adelie penguins (given features like flipper lengths).



Density of flipper lengths (for all penguins in the sample)

Probabilistic model for the data

Suppose we want to learn a classifier to distinguish between Gentoo and Adelie penguins (given features like flipper lengths).



Density of flipper lengths for each of the two classes

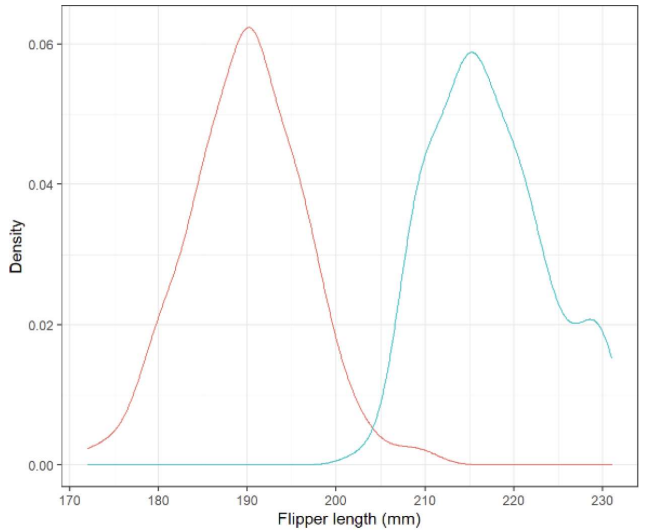
The data associated with each of the two individual classes looks quite Gaussian.

Linear discriminant analysis

The idea within linear discriminant analysis is:

Let's fit a Gaussian distribution to the data from each of the two classes.

Then we obtain a probabilistic model for the data. We can then use this probabilistic model to generate a rule based on the probability of the two classes.



Next, we will explore this idea. For simplicity, we assume two classes (i.e., binary classification)

The linear discriminant analysis model

Aim: Build a probabilistic model for the data generation process for $(X, Y) \in \mathbb{R}^d \times \{0, 1\}$

1. The binary labels $Y \in \{0, 1\}$ are modelled as Bernoulli random variables.

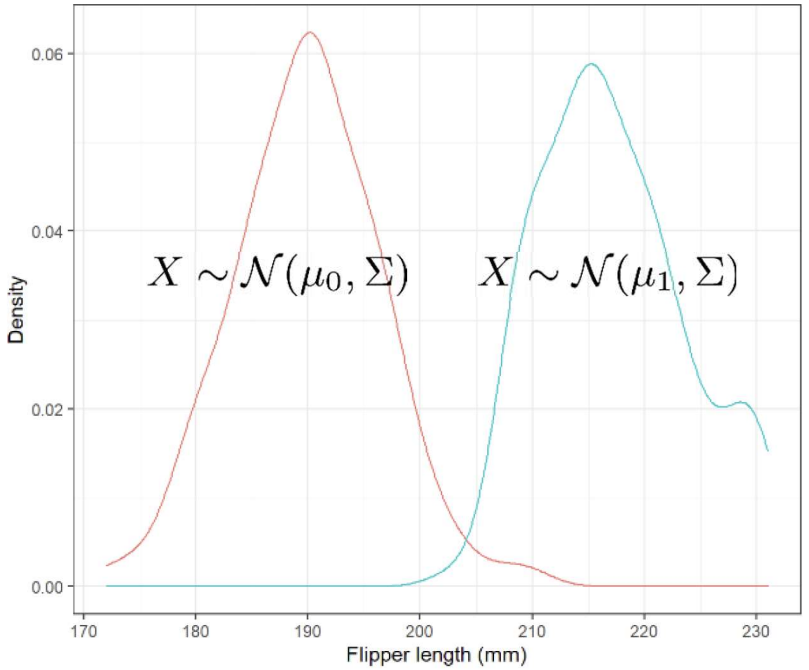
$$Y \sim \mathcal{B}(q) \quad \text{for some fixed} \quad q \in [0, 1].$$

2. The feature vectors $X \in \mathbb{R}^d$ are modelled as class conditional Gaussians,

$$X \sim \mathcal{N}(\mu_0, \Sigma) \quad \text{if} \quad Y = 0$$

$$X \sim \mathcal{N}(\mu_1, \Sigma) \quad \text{if} \quad Y = 1$$

where $\mu_0, \mu_1 \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d \times d}$



Recall: Bernoulli random variables

Let $X \sim \mathcal{B}(q)$ be a Bernoulli random variable.

The probability mass function is given by

$$p_X(x) = \begin{cases} 1-q & \text{if } x = 0, \\ q & \text{if } x = 1, \\ 0 & \text{otherwise.} \end{cases}$$

The expectation

$$\mathbb{E}(X) := \sum_{x \in \mathbb{R}} x \cdot p_X(x) = (1-q) \times 0 + q \times 1 + \underbrace{\sum_{x \in \mathbb{R} \setminus \{0,1\}} x \cdot p_Z(x)}_{} = q.$$

The variance

$$\mathrm{Var}(X) := \mathbb{E}\left[(X - \mathbb{E}(X))^2\right] = \sum_{x \in R} p_X(x) \cdot (x-q)^2 = (1-q) \cdot q^2 + q \cdot (1-q)^2 = q(1-q)$$

Recall: Multivariate Gaussians

A classical example of continuous random vector is a multivariate Gaussian $X = (X_1, \cdots, X_d)$.

Its parameters are

1) A mean vector: $\mu = \mathbb{E}(X) \in \mathbb{R}^d$
2) A covariance matrix: $\Sigma = \mathbb{E}\big[(X - \mathbb{E}(X))(X - \mathbb{E}(X))^T\big] \in \mathbb{R}^{d \times d}$.

The probability density function $f_{\mu,\Sigma} : \mathbb{R}^d \to (0, \infty)$ is given by

$$f_{\mu,\Sigma}(x) := \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

where $x \in \mathbb{R}^d$, $|\Sigma|$ is the determinant of Σ, Σ^{-1} is the inverse of Σ, and $(x-\mu)^T$ is the transpose of $x - \mu$.

For comparison: $f_{\mu,\sigma}(x) := \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$.

Note: if $d = 1$, then the multivariate Gaussian defined above reduces to a univariate Gaussian that we discussed previously.

The linear discriminant analysis model

Aim: Build a probabilistic model for the data generation process for $(X, Y) \in \mathbb{R}^d \times \{0, 1\}$

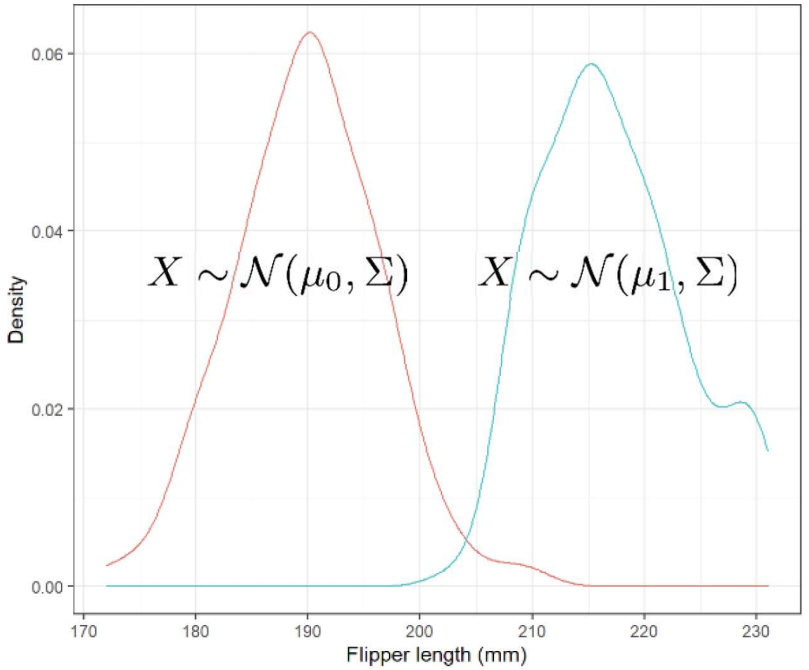
1. The binary labels $Y \in \{0, 1\}$ are modelled as Bernoulli random variables.

$$\mathbb{P}(Y = y) = \begin{cases} q & \text{if} \quad y = 1, \\ 1 - q & \text{if} \quad y = 0. \end{cases}$$

2. The feature vectors $X \in \mathbb{R}^d$ are modelled as class conditional Gaussians,

$$f(X = x \mid Y = y)$$
$$= \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(\frac{1}{2}(x - \mu_y)\Sigma^{-1}(x - \mu_y)^T\right)$$

for $y \in \{0, 1\}$.



The linear discriminant analysis model

With the probabilistic model, we will discuss in the next steps:

1. Assuming that the parameters of the probabilistic model are known, what is the best classifier?

2. How to estimate the parameters of the probabilistic model from the data?

2. The optimal classifier for the LDA model

Note: With the probabilistic model, we can model the optimal classifier

Recall our discussion in the last lecture:

For a classifier ϕ, the Expected test error is defined as: $\mathcal{R}(\phi) := \mathbb{P}(\phi(X) \neq Y)$.

The Bayes classifier ϕ^* is a minimiser of the expected test error (over all possible classifiers).

$$\mathcal{R}(\phi^*) = \min \{\mathcal{R}(\phi) : \quad \phi : \mathcal{X} \to \mathcal{Y} \text{ is a classifier.}\}$$

In the binary case where $\mathcal{Y} := \{0, 1\}$, the Bayes classifier is given by:

$$\phi^*(x) := \begin{cases} 1 & \text{if} \quad \mathbb{P}(Y = 1 \mid X = x) \geq \mathbb{P}(Y = 0 \mid X = x) \\ 0 & \text{if} \quad \mathbb{P}(Y = 0 \mid X = x) > \mathbb{P}(Y = 1 \mid X = x) \end{cases}$$

Next, let's show that the Bayes classifier for LDA model is a linear classifier.

The optimal classifier for the LDA model

$$\phi^*(x) := \begin{cases} 1 & \text{if} \quad \mathbb{P}(Y=1 \mid X=x) \geq \mathbb{P}(Y=0 \mid X=x) \\ 0 & \text{if} \quad \mathbb{P}(Y=0 \mid X=x) > \mathbb{P}(Y=1 \mid X=x) \end{cases}$$

$\phi^*(x) = 1$ ⟺ means "if and only if"

⟺ $\mathbb{P}(Y=1 \mid X=x) \geq \mathbb{P}(Y=0|X=x)$

⟺ $$\frac{\mathbb{P}(X=x \mid Y=1) \cdot \mathbb{P}(Y=1)}{\mathbb{P}(X=x)} \geq \frac{\mathbb{P}(X=x \mid Y=0) \cdot \mathbb{P}(Y=0)}{\mathbb{P}(X=x)}$$

This is a consequence of the Bayes theorem:

$$\mathbb{P}(Y=y \mid X=x) = \frac{\mathbb{P}(X=x \mid Y=y) \cdot \mathbb{P}(Y=y)}{\mathbb{P}(X=x)}$$

The optimal classifier for the LDA model

$$\phi^*(x) := \begin{cases} 1 & \text{if} \quad \mathbb{P}(Y=1 \mid X=x) \geq \mathbb{P}(Y=0 \mid X=x) \\ 0 & \text{if} \quad \mathbb{P}(Y=0 \mid X=x) > \mathbb{P}(Y=1 \mid X=x) \end{cases}$$

$\phi^*(x) = 1$ $\Leftrightarrow$ means "if and only if"

$\Leftrightarrow \mathbb{P}(Y=1 \mid X=x) \geq \mathbb{P}(Y=0 | X=x)$

$$\Leftrightarrow \frac{\mathbb{P}(X=x \mid Y=1) \cdot \mathbb{P}(Y=1)}{\mathbb{P}(X=x)} \geq \frac{\mathbb{P}(X=x \mid Y=0) \cdot \mathbb{P}(Y=0)}{\mathbb{P}(X=x)}$$

$$\Leftrightarrow \mathbb{P}(X=x \mid Y=1) \cdot \mathbb{P}(Y=1) \geq \mathbb{P}(X=x \mid Y=0) \cdot \mathbb{P}(Y=0)$$

$$\Leftrightarrow \exp\left(-\frac{1}{2}(x-\mu_1)\Sigma^{-1}(x-\mu_1)^T\right) \cdot q \geq \exp\left(-\frac{1}{2}(x-\mu_0)\Sigma^{-1}(x-\mu_0)^T\right) \cdot (1-q)$$

This is because of our LDA model:

$$\mathbb{P}(Y=y) = \begin{cases} q & \text{if} \quad y=1, \\ 1-q & \text{if} \quad y=0. \end{cases}$$

$$\mathbb{P}(X=x \mid Y=y) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(\frac{1}{2}(x-\mu_y)\Sigma^{-1}(x-\mu_y)^T\right)$$

The optimal classifier for the LDA model

$$\phi^*(x) := \begin{cases} 1 & \text{if} \quad \mathbb{P}(Y=1 \mid X=x) \geq \mathbb{P}(Y=0 \mid X=x) \\ 0 & \text{if} \quad \mathbb{P}(Y=0 \mid X=x) > \mathbb{P}(Y=1 \mid X=x) \end{cases}$$

$\phi^*(x) = 1$ $\Leftrightarrow$ means "if and only if"

$\Leftrightarrow \mathbb{P}(Y=1 \mid X=x) \geq \mathbb{P}(Y=0|X=x)$

$$\Leftrightarrow \frac{\mathbb{P}(X=x \mid Y=1) \cdot \mathbb{P}(Y=1)}{\mathbb{P}(X=x)} \geq \frac{\mathbb{P}(X=x \mid Y=0) \cdot \mathbb{P}(Y=0)}{\mathbb{P}(X=x)}$$

$$\Leftrightarrow \mathbb{P}(X=x \mid Y=1) \cdot \mathbb{P}(Y=1) \geq \mathbb{P}(X=x \mid Y=0) \cdot \mathbb{P}(Y=0)$$

$$\Leftrightarrow \exp\left(-\frac{1}{2}(x-\mu_1)\Sigma^{-1}(x-\mu_1)^T\right) \cdot q \geq \exp\left(-\frac{1}{2}(x-\mu_0)\Sigma^{-1}(x-\mu_0)^T\right) \cdot (1-q)$$

$$\Leftrightarrow -\frac{1}{2}(x-\mu_1)\Sigma^{-1}(x-\mu_1)^T + \log q \geq -\frac{1}{2}(x-\mu_0)\Sigma^{-1}(x-\mu_0)^T + \log(1-q)$$

The optimal classifier for the LDA model

$$\phi^*(x) := \begin{cases} 1 & \text{if} \quad \mathbb{P}(Y=1 \mid X=x) \geq \mathbb{P}(Y=0 \mid X=x) \\ 0 & \text{if} \quad \mathbb{P}(Y=0 \mid X=x) > \mathbb{P}(Y=1 \mid X=x) \end{cases}$$

$$\phi^*(x) = 1$$

$$\Leftrightarrow \quad -\frac{1}{2}(x-\mu_1)\Sigma^{-1}(x-\mu_1)^T + \log q \geq -\frac{1}{2}(x-\mu_0)\Sigma^{-1}(x-\mu_0)^T + \log(1-q)$$

$$\Leftrightarrow \quad -\frac{1}{2}\left(\mu_1\Sigma^{-1}\mu_1^T - 2x\Sigma^{-1}\mu_1^T\right) + \log q \geq \left(\mu_0\Sigma^{-1}\mu_0^T - 2x\Sigma^{-1}\mu_0^T\right) + \log(1-q)$$

Observation: both sides of this inequality are linear functions of x.

$$\Leftrightarrow \quad x\Sigma^{-1}(\mu_1-\mu_0)^T + \left\{\log\left(\frac{q}{1-q}\right) - \frac{1}{2}\left(\mu_1\Sigma^{-1}\mu_1^T - \mu_0\Sigma^{-1}\mu_0^T\right)\right\} \geq 0$$

$$\Leftrightarrow \quad xw^T + w_0 \geq 0$$

where $w := \Sigma^{-1}(\mu_1-\mu_0)^T$, $w_0 := \log\left(\frac{q}{1-q}\right) - \frac{1}{2}\left(\mu_1\Sigma^{-1}\mu_1^T - \mu_0\Sigma^{-1}\mu_0^T\right)$

The optimal classifier for the LDA model

$$\phi^*(x) := \begin{cases} 1 & \text{if} \quad \mathbb{P}(Y=1 \mid X=x) \geq \mathbb{P}(Y=0 \mid X=x) \\ 0 & \text{if} \quad \mathbb{P}(Y=0 \mid X=x) > \mathbb{P}(Y=1 \mid X=x) \end{cases}$$

$$\phi^*(x) = 1$$

$$\Leftrightarrow \quad xw^T + w_0 \geq 0$$

where $w := (\mu_1 - \mu_0)\Sigma^{-1}$, $w_0 := \log\left(\frac{q}{1-q}\right) - \frac{1}{2}\left(\mu_1\Sigma^{-1}\mu_1^T - \mu_0\Sigma^{-1}\mu_0^T\right)$

Therefore, the Bayes classifier $\phi^*(x) = \mathbb{1}\{wx^T + w_0 \geq 0\}$

with $w := (\mu_1 - \mu_0)\Sigma^{-1}$,

$w_0 := \log\left(\frac{q}{1-q}\right) - \frac{1}{2}\left(\mu_1\Sigma^{-1}\mu_1^T - \mu_0\Sigma^{-1}\mu_0^T\right)$

The Bayes classifier in the linear discriminant model is linear!

However, we don't know the parameters q, μ_0, μ_1, Σ. They need to be computed from the data.

3. Parameter estimation for LDA

How can we learn the parameters q, μ_0, μ_1, Σ for the linear discriminant analysis model?

We can fit the probabilistic model to the data, by using the maximum likelihood principle!

For simplicity let's look at the one-dimensional case $d = 1$.

$$\mathbb{P}(Y = y) = \begin{cases} q & \text{if} \quad y = 1, \\ 1 - q & \text{if} \quad y = 0. \end{cases} \qquad = q^y (1-q)^{1-y}$$

$$= q^y (1-q)^{1-y}$$

$$\mathbb{P}(X = x \mid Y = y) = \frac{1}{(2\pi)^d |\Sigma|} \exp\left(\frac{1}{2}(x - \mu_y)\Sigma^{-1}(x - \mu_y)^T\right)$$

$$= \frac{1}{\sqrt{(2\pi)\sigma^2}} \exp\left(\frac{1}{2}(x - \mu_y)^2/(2\sigma^2)\right)$$

Likelihood function

We want to compute the likelihood of the data $\mathcal{D} = ((X_1, Y_1), \cdots, (X_n, Y_n))$

$$\mathbb{P}(Y = y) = q^y (1-q)^{1-y}$$

$$\mathbb{P}(X = x \mid Y = y) = \frac{1}{\sqrt{(2\pi)\sigma^2}} \exp\left(\frac{1}{2}(x - \mu_y)^2/(2\sigma^2)\right)$$

$$\begin{aligned}\mathbb{P}(X = x, Y = y) &= \mathbb{P}(Y = y) \cdot \mathbb{P}(X = x \mid Y = y) \\ &= q^y (1-q)^{1-y} \cdot \left\{ \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x - \mu_y)^2\right) \right\}\end{aligned}$$

Assuming that (X_i, Y_i) are independent, the likelihood function is given by:

$$\begin{aligned}l(q, \mu_0, \mu_1, \sigma) &= \prod_{i=1}^{n} \mathbb{P}(X = X_i, Y = Y_i) \\ &= \prod_{i=0}^{n} \left\{ q^{Y_i} (1-q)^{1-Y_i} \cdot \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(X_i - \mu_{Y_i})^2\right) \right\}\end{aligned}$$

Log-likelihood function

We want to compute the likelihood of the data $\mathcal{D} = ((X_1, Y_1), \cdots, (X_n, Y_n))$

The likelihood function is given by:

$$l(q, \mu_0, \mu_1, \sigma) = \prod_{i=0}^{n} \left\{ q^{Y_i}(1-q)^{1-Y_i} \cdot \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(X_i - \mu_{Y_i})^2 \right) \right\}$$

To maximise the likelihood we will maximise the log-likelihood.

$$\log l(q, \mu_0, \mu_1, \sigma) = \sum_{i=1}^{n} \left\{ Y_i \log q + (1 - Y_i) \log(1-q) - \frac{1}{2} \log\left(\pi\sigma^2\right) - \frac{1}{2\sigma^2}(X_i - \mu_{Y_i})^2 \right\}$$

$$= \sum_{i=1}^{n} \{ Y_i \log q + (1 - Y_i) \log(1-q) \} - \frac{n}{2} \log(\pi\sigma^2)$$

$$- \sum_{\{i:Y_i=0\}} \frac{1}{2\sigma^2}(X_i - \mu_0)^2 - \sum_{\{i:Y_i=1\}} \frac{1}{2\sigma^2}(X_i - \mu_1)^2$$

Maximum likelihood

To maximise the likelihood we will maximise the log-likelihood.

$$\log l(q, \mu_0, \mu_1, \sigma) = \sum_{i=1}^{n} \{Y_i \log q + (1 - Y_i) \log(1 - q)\} - \frac{n}{2} \log(\pi\sigma^2)$$
$$- \sum_{\{i:Y_i=0\}} \frac{1}{2\sigma^2}(X_i - \mu_0)^2 - \sum_{\{i:Y_i=1\}} \frac{1}{2\sigma^2}(X_i - \mu_1)^2$$

To maximise the log-likelihood we find the point where the derivatives are equal to zero.

1. derivative with respect to q:

$$\frac{\partial}{\partial q} \log l(q, \mu_0, \mu_1, \sigma) = (\sum_{i=1}^{n} Y_i) \cdot \left\{ \frac{1}{q} + \frac{1}{1-q} \right\} - \frac{n}{1-q}$$

Taking $\frac{\partial}{\partial q} \log l(q, \mu_0, \mu_1, \sigma) = 0$, give an MLE of $\hat{q} = \frac{1}{n} \sum Y_i$.

Maximum likelihood

To maximise the likelihood we will maximise the log-likelihood.

$$\log l(q, \mu_0, \mu_1, \sigma) = \sum_{i=1}^{n} \{Y_i \log q + (1 - Y_i) \log(1 - q)\} - \frac{n}{2} \log(\pi\sigma^2)$$
$$- \sum_{\{i:Y_i=0\}} \frac{1}{2\sigma^2}(X_i - \mu_0)^2 - \sum_{\{i:Y_i=1\}} \frac{1}{2\sigma^2}(X_i - \mu_1)^2$$

To maximise the log-likelihood we find the point where the derivatives are equal to zero.

2. The derivative with respect to μ_0 is :

$$\frac{\partial}{\partial \mu_0} \log l(q, \mu_0, \mu_1, \sigma) = \frac{1}{\sigma^2} \sum_{\{i:Y_i=0\}} (X_i - \mu_0)$$

Taking $\frac{\partial}{\partial \mu_0} \log l = 0$ gives an MLE of $\hat{\mu}_0 = \frac{1}{n_0} \sum_{\{i:Y_i=0\}} X_i$ where $n_0 := |\{i : Y_i = 0\}|$.

Similarly, the MLE for μ_1 is $\hat{\mu}_1 = \frac{1}{n_1} \sum_{\{i:Y_i=1\}} X_i$ where $n_1 := |\{i : Y_i = 0\}|$.

Maximum likelihood

To maximise the likelihood we will maximise the log-likelihood.

$$\log l(q, \mu_0, \mu_1, \sigma) = \sum_{i=1}^{n} \{Y_i \log q + (1 - Y_i) \log(1 - q)\} - \frac{n}{2} \log(\pi\sigma^2)$$
$$- \sum_{\{i:Y_i=0\}} \frac{1}{2\sigma^2}(X_i - \mu_0)^2 - \sum_{\{i:Y_i=1\}} \frac{1}{2\sigma^2}(X_i - \mu_1)^2$$

To maximise the log-likelihood we find the point where the derivatives are equal to zero.

3. The derivative with respect to σ is :

$$\frac{\partial}{\partial\sigma} \log l = -\frac{n}{\sigma} + \frac{1}{\sigma^3} \left\{ \sum_{\{i:Y_i=0\}} \{X_i - \mu_0\}^2 + \sum_{\{i:Y_i=1\}} \{X_i - \mu_1\}^2 \right\}$$

So the MLE is $\hat{\sigma}^2 = \frac{1}{n} \left\{ \sum_{\{i:Y_i=0\}} \{X_i - \hat{\mu}_0\}^2 + \sum_{\{i:Y_i=1\}} \{X_i - \hat{\mu}_1\}^2 \right\}$.

The maximum likelihood estimates

$$\mathbb{P}(Y = y) = q^y (1-q)^{1-y}$$

$$\mathbb{P}(X = x \mid Y = y) = \frac{1}{(2\pi)^d |\Sigma|} \exp\left(\frac{1}{2}(x-\mu_y)\Sigma^{-1}(x-\mu_y)^T\right)$$

Given data $\mathcal{D} = ((X_1, Y_1), \cdots, (X_n, Y_n))$, the MLE of the parameters is:

$$\hat{q} = \frac{1}{n}\sum Y_i$$

$$\hat{\mu}_0 = \frac{1}{n_0}\sum_{\{i:Y_i=0\}} X_i \text{ where } n_0 := |\{i : Y_i = 0\}|$$

$$\hat{\mu}_1 = \frac{1}{n_1}\sum_{\{i:Y_i=1\}} X_i \text{ where } n_1 := |\{i : Y_i = 1\}|$$

$$\hat{\Sigma}_{\text{MLE}} = \frac{1}{n}\left\{\sum_{\{i:Y_i=0\}} \{X_i - \hat{\mu}_0\}\{X_i - \hat{\mu}_0\}^T + \sum_{\{i:Y_i=1\}} \{X_i - \hat{\mu}_1\}\{X_i - \hat{\mu}_1\}^T\right\}.$$

The linear discriminant analysis classifier

Recall that the Bayes classifier $\phi^*(x) = \mathbb{1}\{wx^T + w_0 \geq 0\}$

with $w := (\mu_1 - \mu_0)\Sigma^{-1}$,

$w_0 := \log\left(\frac{q}{1-q}\right) - \frac{1}{2}\left(\mu_1\Sigma^{-1}\mu_1^T - \mu_0\Sigma^{-1}\mu_0^T\right)$

The linear discriminant analysis classifier is given by $\hat{\phi}(x) = \mathbb{1}\{\hat{w}x^T + \hat{w}_0 \geq 0\}$

with $\hat{w} := (\hat{\mu}_1 - \hat{\mu}_0)\Sigma_{\text{MLE}}^{-1}$, $\hat{w}_0 := \log\left(\frac{\hat{q}}{1-\hat{q}}\right) - \frac{1}{2}\left(\hat{\mu}_1\hat{\Sigma}_{\text{MLE}}^{-1}\hat{\mu}_1^T - \hat{\mu}_0\hat{\Sigma}_{\text{MLE}}^{-1}\hat{\mu}_0^T\right)$

where $\hat{q} = \frac{1}{n}\sum Y_i$

$\hat{\mu}_0 = \frac{1}{n_0}\sum_{\{i:Y_i=0\}} X_i$ where $n_0 := |\{i : Y_i = 0\}|$

$\hat{\mu}_1 = \frac{1}{n_1}\sum_{\{i:Y_i=1\}} X_i$ where $n_1 := |\{i : Y_i = 1\}|$

$$\hat{\Sigma}_{\text{MLE}} = \frac{1}{n}\left\{\sum_{\{i:Y_i=0\}} \{X_i - \hat{\mu}_0\}\{X_i - \hat{\mu}_0\}^T + \sum_{\{i:Y_i=1\}} \{X_i - \hat{\mu}_1\}\{X_i - \hat{\mu}_1\}^T\right\}.$$

Note: $\hat{\Sigma}_{\text{MLE}}$ is biased!

The linear discriminant analysis classifier

Recall that the Bayes classifier $\phi^*(x) = \mathbb{1}\{wx^T + w_0 \geq 0\}$

with $w := (\mu_1 - \mu_0)\Sigma^{-1}$,

$w_0 := \log\left(\frac{q}{1-q}\right) - \frac{1}{2}\left(\mu_1\Sigma^{-1}\mu_1^T - \mu_0\Sigma^{-1}\mu_0^T\right)$

The linear discriminant analysis classifier is given by $\hat{\phi}(x) = \mathbb{1}\{\hat{w}x^T + \hat{w}_0 \geq 0\}$

with $\hat{w} := (\hat{\mu}_1 - \hat{\mu}_0)\Sigma_{\text{MLE}}^{-1}$, $\hat{w}_0 := \log\left(\frac{\hat{q}}{1-\hat{q}}\right) - \frac{1}{2}\left(\hat{\mu}_1\hat{\Sigma}_{\text{U}}^{-1}\hat{\mu}_1^T - \hat{\mu}_0\hat{\Sigma}_{\text{U}}^{-1}\hat{\mu}_0^T\right)$

where $\hat{q} = \frac{1}{n}\sum Y_i$

$\hat{\mu}_0 = \frac{1}{n_0}\sum_{\{i:Y_i=0\}} X_i$ where $n_0 := |\{i : Y_i = 0\}|$

$\hat{\mu}_1 = \frac{1}{n_1}\sum_{\{i:Y_i=1\}} X_i$ where $n_1 := |\{i : Y_i = 1\}|$

$$\hat{\Sigma}_{\text{U}} = \frac{1}{n-2}\left\{\sum_{\{i:Y_i=0\}} \{X_i - \hat{\mu}_0\}\{X_i - \hat{\mu}_0\}^T + \sum_{\{i:Y_i=1\}} \{X_i - \hat{\mu}_1\}\{X_i - \hat{\mu}_1\}^T\right\}.$$

Note: here we considered an alternative of the MLE: the unbiased estimate $\hat{\Sigma}_U$.

4. The linear discriminant analysis classifier in R

Example: Suppose we want to learn a classifier $\phi : \mathcal{X} \to \mathcal{Y}$ which takes a feature vector of morphological features and predicts whether a penguin belongs to either the Adelie species or the Gentoo species.

The data is given by:

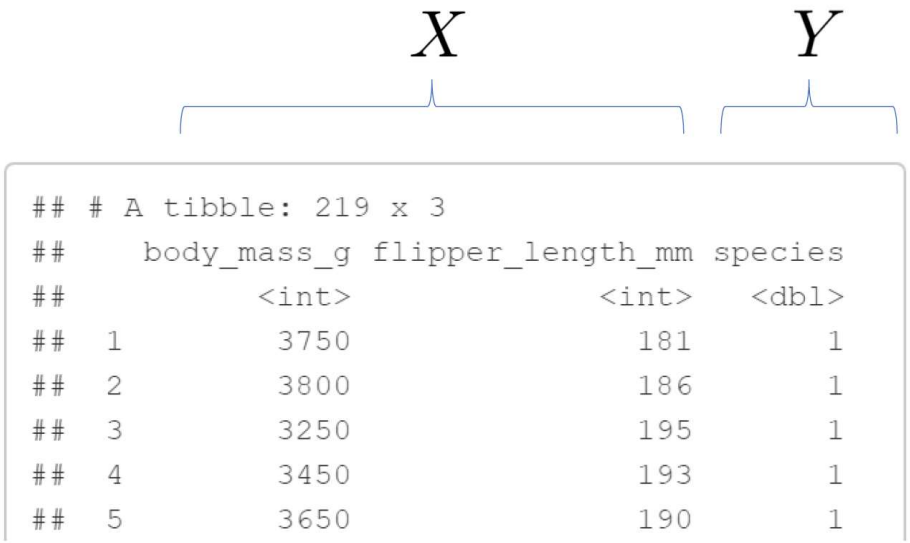
```
library(tidyverse)
library(palmerpenguins)

peng_total<-penguins%>% # prepare our data
  select(body_mass_g,flipper_length_mm,species)%>%
  filter(species!="Chinstrap")%>%
  drop_na()%>%
  mutate(species=as.numeric(species=="Adelie"))
```

```
peng_total
```

```
## # A tibble: 219 x 3
##    body_mass_g flipper_length_mm species
##          <int>             <int>   <dbl>
##  1        3750               181       1
##  2        3800               186       1
##  3        3250               195       1
##  4        3450               193       1
##  5        3650               190       1
##  6        3625               181       1
##  7        4675               195       1
##  8        3475               193       1
##  9        4250               190       1
## 10        3300               186       1
## # ... with 209 more rows
```

Example: Penguin classification

Example: Suppose we want to learn a classifier $\phi : \mathcal{X} \to \mathcal{Y}$ which takes a feature vector of morphological features and predicts whether a penguin belongs to either the Adelie species or the Chinstrap species.




```
## # A tibble: 219 x 3
##    body_mass_g flipper_length_mm species
##          <int>             <int>   <dbl>
##  1        3750               181       1
##  2        3800               186       1
##  3        3250               195       1
##  4        3450               193       1
##  5        3650               190       1
```

Features:

$$X = (X^1, X^2) \in \mathcal{X} := \mathbb{R}^2$$

X^1 = the weight of the penguin (grams)

X^2 = the flipper length of the penguin (mm)

Labels:

$$Y \in \mathcal{Y} := \{0, 1\}$$

$$Y = \begin{cases} 1 & \text{if the penguin is an Adelie} \\ 0 & \text{if the penguin is a Chinstrap} \end{cases}$$

Example: Penguin classification

Now let's carry out a train test split.

```
num_total<-peng_total%>%nrow() # number of penguin data
num_train<-floor(num_total*0.75) # number of train examples
num_test<-num_total-num_train # number of test samples

set.seed(1) # set random seed for reproducibility
test_inds<-sample(seq(num_total),num_test) # random sample of test indicies
train_inds<-setdiff(seq(num_total),test_inds) # training data indicies

peng_train<-peng_total%>%filter(row_number() %in% train_inds) # train data
peng_test<-peng_total%>%filter(row_number() %in% test_inds) # test data
```

We can also separate out the feature vectors and labels (i.e., separate out X and Y into two different data frames).

```
peng_train_x<-peng_train%>%select(-species) # train feature vectors
peng_train_y<-peng_train%>%pull(species) # train labels

peng_test_x<-peng_test%>%select(-species) # test feature vectors
peng_test_y<-peng_test%>%pull(species) # test labels
```

Example: Penguin classification

We can now build a linear discriminant analysis model with R as follows:

```
lda_model <- MASS::lda(species ~ ., data=peng_train) # fit LDA model
```

We can make predictions and check the training error as follows:

```
lda_train_predicted_y<-predict(lda_model,peng_train_x)$class%>%
  as.character()%>%as.numeric() # get vector of predicted ys

lda_train_error<-mean(abs(lda_train_predicted_y-peng_train_y)) # compute train error

lda_train_error
```

```
## [1] 0.01463415
```

Example: Penguin classification

We can now build a linear discriminant analysis model with R as follows:

```
lda_model <- MASS::lda(species ~ ., data=peng_train) # fit LDA model
```

We can make predictions and check the test error as follows:

```
lda_test_predicted_y<-predict(lda_model,peng_test_x)$class%>%
  as.character()%>%as.numeric() # get vector of predicted ys

lda_test_error<-mean(abs(lda_test_predicted_y-peng_test_y)) # compute test error

lda_test_error
```

```
## [1] 0.01449275
```

Logistic regression

Logistic regression

To find a linear classifier, we can use the idea of logistic regression.

The first thing to remember about logistic regression is that its not a regression algorithm!

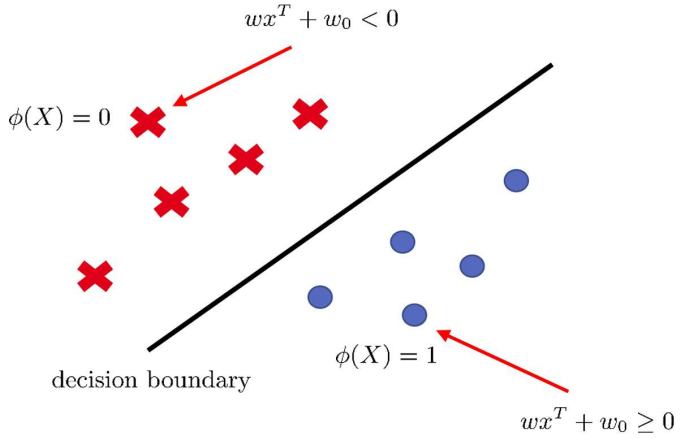
The logistic regression approach assumes a different probabilistic model (from the LDA model).

Then, under such an assumption, the optimal classifier (Bayes classifier) is a linear classifier. We can derive this classifier based on the probabilistic model.

Finally, the parameters of the probabilistic model can be determined by the maximum likelihood approach.

1. Logistic regression probabilistic model

Logistic regression is a method for learning a linear classifier $\phi(x) = \mathbb{1}\left(\omega x^T + \omega_0 \geq 0\right)$



Observation: The Bayes classifier is given by

$$\phi^*(x) := \begin{cases} 1 & \text{if} \quad \mathbb{P}(Y = 1 \mid X = x) \geq \mathbb{P}(Y = 0 \mid X = x) \\ 0 & \text{if} \quad \mathbb{P}(Y = 0 \mid X = x) > \mathbb{P}(Y = 1 \mid X = x) \end{cases}$$

Hence, we only need to model $\mathbb{P}(Y = y \mid X = x)$.

This is more straightforward and easier than modelling $\mathbb{P}(X, Y)$.

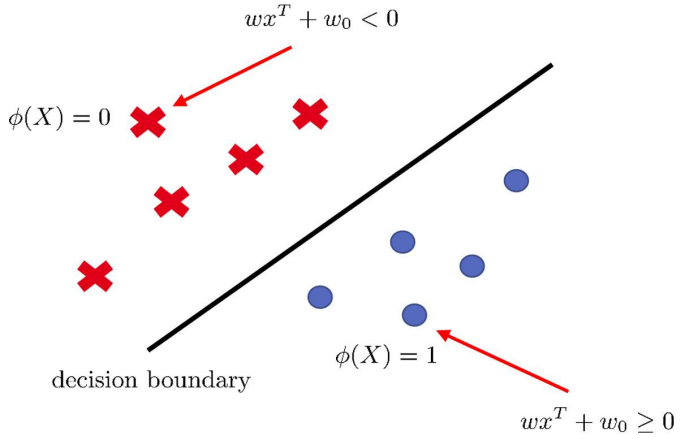
We can consider constructing the conditional probability by using $wx^T + w_0$.

Idea 1: Can we try $\mathbb{P}(Y = 1 \mid X = x) = wx^T + w_0$?

This is a bad idea as it would lead to probabilities outside of $[0, 1]$.

Logistic regression probabilistic model

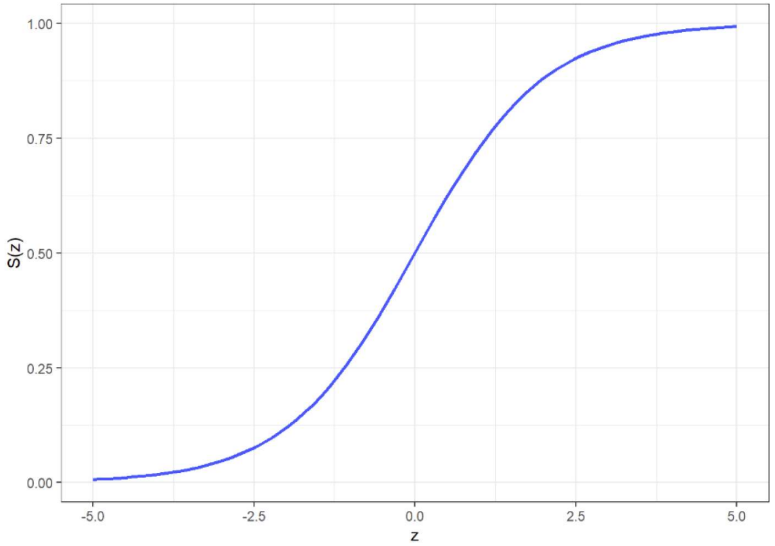
Logistic regression is a method for learning a linear classifier $\phi(x) = \mathbb{1}\left(\omega x^T + \omega_0 \geq 0\right)$



Idea 1: Can we try $\mathbb{P}(Y = 1 \mid X = x) = wx^T + w_0$?

This is a bad idea as it would lead to probabilities outside of $[0, 1]$.

Idea 2: Use the sigmoid function $S : \mathbb{R} \to (0, 1)$ to map real numbers to probabilities: $S(z) := \frac{1}{1+e^{-z}}$.



Facts.

✓ $1 - S(z) = S(-z)$, i.e., symmetric about $(0, \frac{1}{2})$.

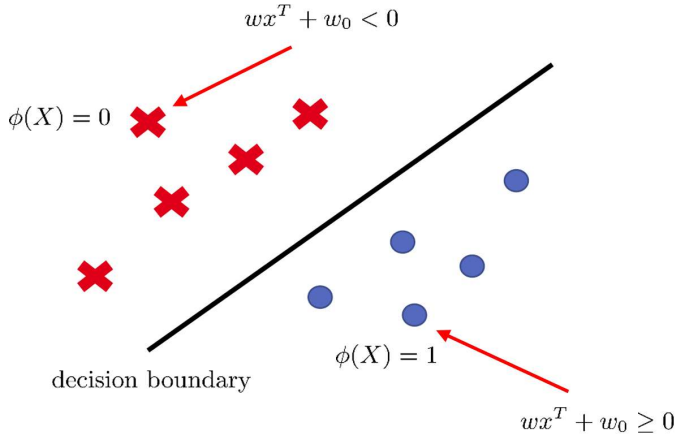
✓ $\frac{\partial \log S(z)}{\partial z} = S(-z)$

✓ $S(z) \geq \frac{1}{2}$ if and only if $z \geq 0$.

The sigmoid function S is also known as the logistic function.

Logistic regression probabilistic model

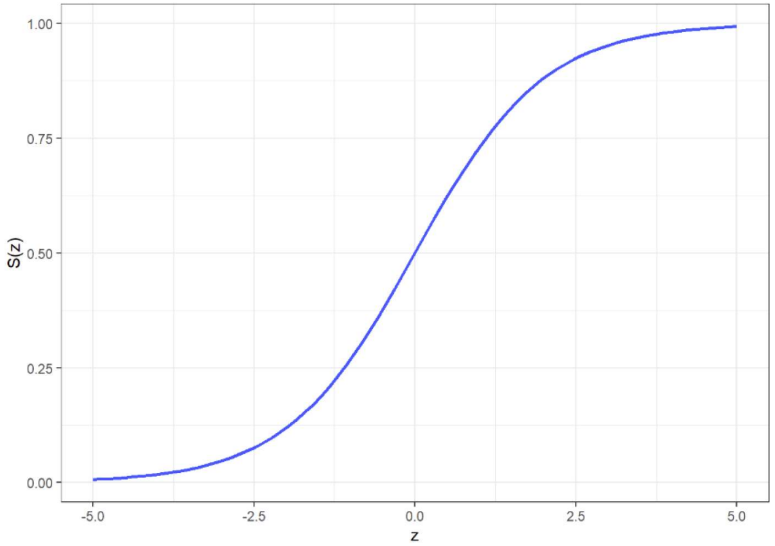
Logistic regression is a method for learning a linear classifier $\phi(x) = \mathbb{1}\left(\omega x^T + \omega_0 \geq 0\right)$



Idea 1: Can we try $\mathbb{P}(Y = 1 \mid X = x) = wx^T + w_0$?

This is a bad idea as it would lead to probabilities outside of $[0, 1]$.

Idea 2: Use the sigmoid function $S : \mathbb{R} \to (0, 1)$ to map real numbers to probabilities: $S(z) := \frac{1}{1+e^{-z}}$.



We use the logistic sigmoid model

$$\mathbb{P}(Y = 1 \mid X = x) = S(wx^T + w^0) = \frac{1}{1 + e^{-wx^T - w^0}}$$

2. The optimal classifier for the logistic model

Note: With the probabilistic model, we can model the optimal classifier

Recall our discussion in the last lecture:

For a classifier ϕ, the Expected test error is defined as: $\mathcal{R}(\phi) := \mathbb{P}(\phi(X) \neq Y)$.

The Bayes classifier ϕ^* is a minimiser of the expected test error (over all possible classifiers).

$$\mathcal{R}(\phi^*) = \min \{\mathcal{R}(\phi) : \quad \phi : \mathcal{X} \to \mathcal{Y} \text{ is a classifier.}\}$$

In the binary case where $\mathcal{Y} := \{0, 1\}$, the Bayes classifier is given by:

$$\phi^*(x) := \begin{cases} 1 & \text{if} \quad \mathbb{P}(Y = 1 \mid X = x) \geq \mathbb{P}(Y = 0 \mid X = x) \\ 0 & \text{if} \quad \mathbb{P}(Y = 0 \mid X = x) > \mathbb{P}(Y = 1 \mid X = x) \end{cases}$$

Next, we will justify that under the logistic sigmoid model, the Bayes classifier is a linear classifier.

The optimal classifier for the logistic model

We use the logistic sigmoid model

$$\mathbb{P}(Y = 1 \mid X = x) = S(wx^T + w^0) = \frac{1}{1 + e^{-wx^T - w^0}}$$

$\phi^*(x) = 1$ $\Leftrightarrow$ means "if and only if"

$\Leftrightarrow \mathbb{P}(Y = 1 \mid X = x) \geq \mathbb{P}(Y = 0 | X = x)$

$\Leftrightarrow \mathbb{P}(Y = 1 \mid X = x) \geq \frac{1}{2}$ (because $\mathbb{P}(Y = 1 \mid X = x) + \mathbb{P}(Y = 0 | X = x) = 1$)

$\Leftrightarrow S(wx^T + w^0) \geq \frac{1}{2}$

$\Leftrightarrow wx^T + w^0 \geq 0$

Therefore, $\phi^*(x) = \mathbb{1}\big(wx^T + w_0 \geq 0\big)$.

The optimal classifier for the logistic model

We use the logistic sigmoid model

$$\mathbb{P}(Y = 1 \mid X = x) = S(wx^T + w^0) = \frac{1}{1 + e^{-wx^T - w^0}}$$

Therefore, $\phi^*(x) = \mathbb{1}\big(wx^T + w_0 \geq 0\big)$.

So the Bayes classifier for the logistic model is a linear classifier.

The parameters ω, ω_0 are unknown.

We want to learn the parameters from the data. Then based on the learned parameters, we can construct the Bayes classifier.

3. Learning parameters for a logistic model

We use the logistic sigmoid model

$$\mathbb{P}(Y = 1 \mid X = x) = S(wx^T + w^0) = \frac{1}{1 + e^{-wx^T - w^0}}$$

We can learn the parameters ω, ω_0 of the model with the maximum likelihood principle.

First, we have

Recalling fact 1: $1 - S(z) = S(-z)$

$$\mathbb{P}(Y = y \mid X = x) = \begin{cases} S(wx^T + w^0) & \text{if } y = 1 \\ 1 - S(wx^T + w^0) = S(-wx^T - w^0) & \text{if } y = 0 \end{cases}$$

$$= S\Big\{(2y - 1) \cdot \Big(wx^T + w^0\Big)\Big\}$$

$2y - 1 = 1$ if $y = 1$ and =-1 if $y = 0$.

The likelihood and log-likelihood

First, we have $$\mathbb{P}(Y = y \mid X = x) = S\Big\{(2y-1)\cdot\Big(wx^T + w^0\Big)\Big\}$$

Second, suppose we have training data $\mathcal{D} := ((X_1, Y_1), \cdots, (X_n, Y_n))$, the likelihood function is given by

$$l(\omega, \omega^0) = \prod_{i=1}^{n} S\Big\{(2Y_i - 1)\cdot\Big(wX_i^T + w^0\Big)\Big\}$$

The log-likelihood function is:

$$\log l(\omega, \omega^0) = \sum_{i=1}^{n} \log S\Big\{(2Y_i - 1)\cdot\Big(wX_i^T + w^0\Big)\Big\}$$

The likelihood and log-likelihood

First, we have $\mathbb{P}(Y = y \mid X = x) = S\Big\{(2y-1)\cdot\Big(wx^T + w^0\Big)\Big\}$

The log-likelihood function is:

$$\log l(\omega, \omega^0) = \sum_{i=1}^{n} \log S\Big\{(2Y_i - 1)\cdot\Big(wX_i^T + w^0\Big)\Big\}$$

To maximise the log-likelihood we compute the gradients.

$$\frac{\partial}{\partial \omega} \log l(\omega, \omega^0) = \sum_{i=1}^{n} (2Y_i - 1) S\Big\{(1 - 2Y_i)\cdot\Big(wX_i^T + w^0\Big)\Big\}\cdot X_i$$

$$\frac{\partial}{\partial \omega^0} \log l(\omega, \omega^0) = \sum_{i=1}^{n} (2Y_i - 1) S\Big\{(1 - 2Y_i)\cdot\Big(wX_i^T + w^0\Big)\Big\}$$

Unfortunately for logistic regression, there is no analytic solution for the MLE.

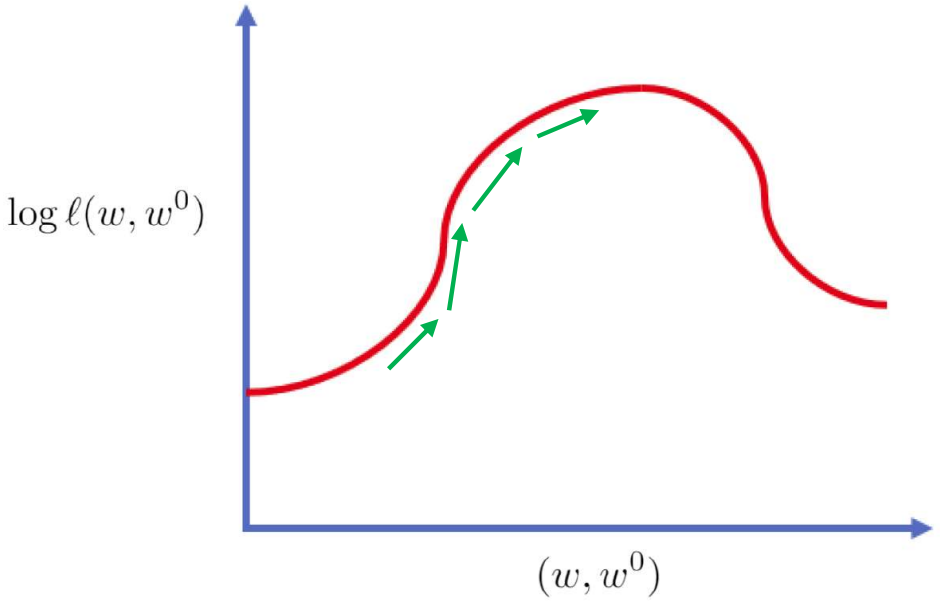
We can maximise the function numerically.

Maximise the log-likelihood

We maximise the log-likelihood $\log l(\omega, \omega_0)$ iteratively.

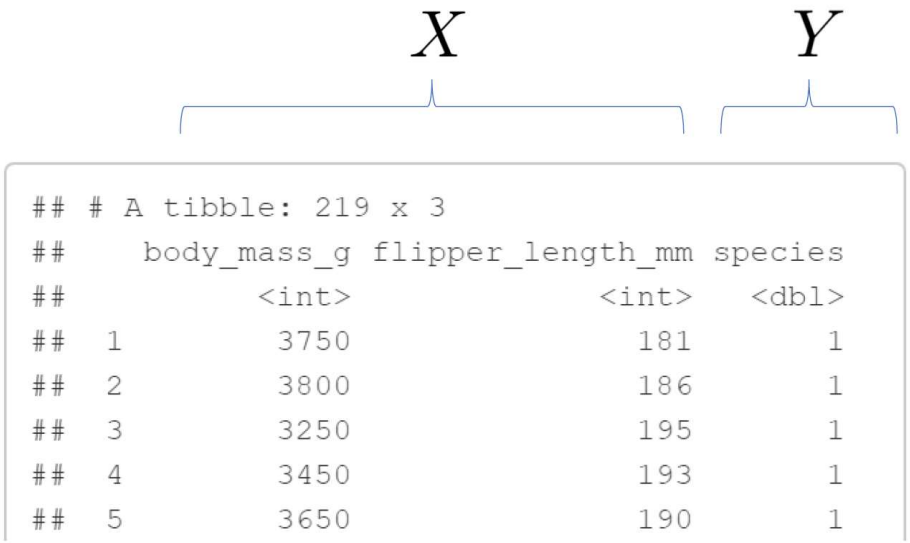
We can use a process of gradient ascent:

$$\omega \leftarrow \omega + \alpha \cdot \frac{\partial}{\partial \omega} \log l(\omega, \omega^0)$$
$$\omega^0 \leftarrow \omega^0 + \alpha \cdot \frac{\partial}{\partial \omega^0} \log l(\omega, \omega^0)$$



4. Logistic regression with R

Example: Suppose we want to learn a classifier $\phi : \mathcal{X} \to \mathcal{Y}$ which takes a feature vector of morphological features and predicts whether a penguin belongs to either the Adelie species or the Chinstrap species.



Features:

$$X = (X^1, X^2) \in \mathcal{X} := \mathbb{R}^2$$

X^1 = the weight of the penguin (grams)

X^2 = the flipper length of the penguin (mm)

Labels:

$$Y \in \mathcal{Y} := \{0, 1\}$$

$$Y = \begin{cases} 1 & \text{if the penguin is an Adelie} \\ 0 & \text{if the penguin is a Chinstrap} \end{cases}$$

Example: Penguin classification

Now let's carry out a train test split.

```
num_total<-peng_total%>%nrow() # number of penguin data
num_train<-floor(num_total*0.75) # number of train examples
num_test<-num_total-num_train # number of test samples

set.seed(1) # set random seed for reproducibility
test_inds<-sample(seq(num_total),num_test) # random sample of test indicies
train_inds<-setdiff(seq(num_total),test_inds) # training data indicies

peng_train<-peng_total%>%filter(row_number() %in% train_inds) # train data
peng_test<-peng_total%>%filter(row_number() %in% test_inds) # test data
```

We can also separate out the feature vectors and labels (i.e., separate out X and Y into two different data frames).

```
peng_train_x<-peng_train%>%select(-species) # train feature vectors
peng_train_y<-peng_train%>%pull(species) # train labels

peng_test_x<-peng_test%>%select(-species) # test feature vectors
peng_test_y<-peng_test%>%pull(species) # test labels
```

Example: Penguin classification

We can train a logistic model with the glmnet library.

```
library(glmnet) # load the glmnet library
logistic_model<-glmnet(x=peng_train_x%>%as.matrix(),y=peng_train_y,
                       family ="binomial",alpha=0,lambda=0) # train a logistic model
```

We compute the training error as follows.

```
logistic_train_predicted_y<-predict(logistic_model,peng_train_x%>%
                                    as.matrix(),type="class")%>%as.integer()

logistic_train_error<-mean(abs(logistic_train_predicted_y-peng_train_y)) # train error

logistic_train_error
```

```
## [1] 0.009756098
```

Test error:

```
logistic_test_predicted_y<-predict(logistic_model,peng_test_x%>%
                                   as.matrix(),type="class")%>%as.integer()

logistic_test_error<-mean(abs(logistic_test_predicted_y-peng_test_y)) # test error

logistic_test_error
```

```
## [1] 0.01449275
```

What have we covered?

We introduced the concept of a linear classifier.

We investigated the generative linear discriminant analysis (LDA) model:

- LDA models the joint probability distribution with Gaussians
- The maximum likelihood parameters can be estimated analytically

We investigated the discriminative logistic regression:

- Logistic regression models the class conditional model with a sigmoid linear model
- The maximum likelihood parameters must be estimated iteratively