

STAT5003

Week 5: Classification technique

Jaslene Lin

The University of Sydney



THE UNIVERSITY OF
SYDNEY

Readings and R functions covered

! Important

- **Introduction to Statistical Learning**
- Classification covered in Chapter 4
- Support Vector Machines covered in Chapter 9
- **R** functions
- `e1071::svm` (Support Vector Machines)
- `stats::glm` (Logistic regression via Generalized Linear Model)
- `class::knn` (k-NN classifier)
- `MASS::LDA` (Linear Discriminant Analysis)

This presentation is based on the [SOLES reveal.js Quarto template](#) and is licensed under a [Creative Commons Attribution 4.0 International License](#).

Understanding Mean Squared Error (MSE)

- The **MSE** measures the **average squared difference** between the predicted values and the true values:

$$\text{MSE} = \mathbb{E}[(Y - \hat{f}(X))^2]$$

- It is a key measure of **prediction accuracy** in supervised learning.

MSE Can Be Decomposed Into Three Components:

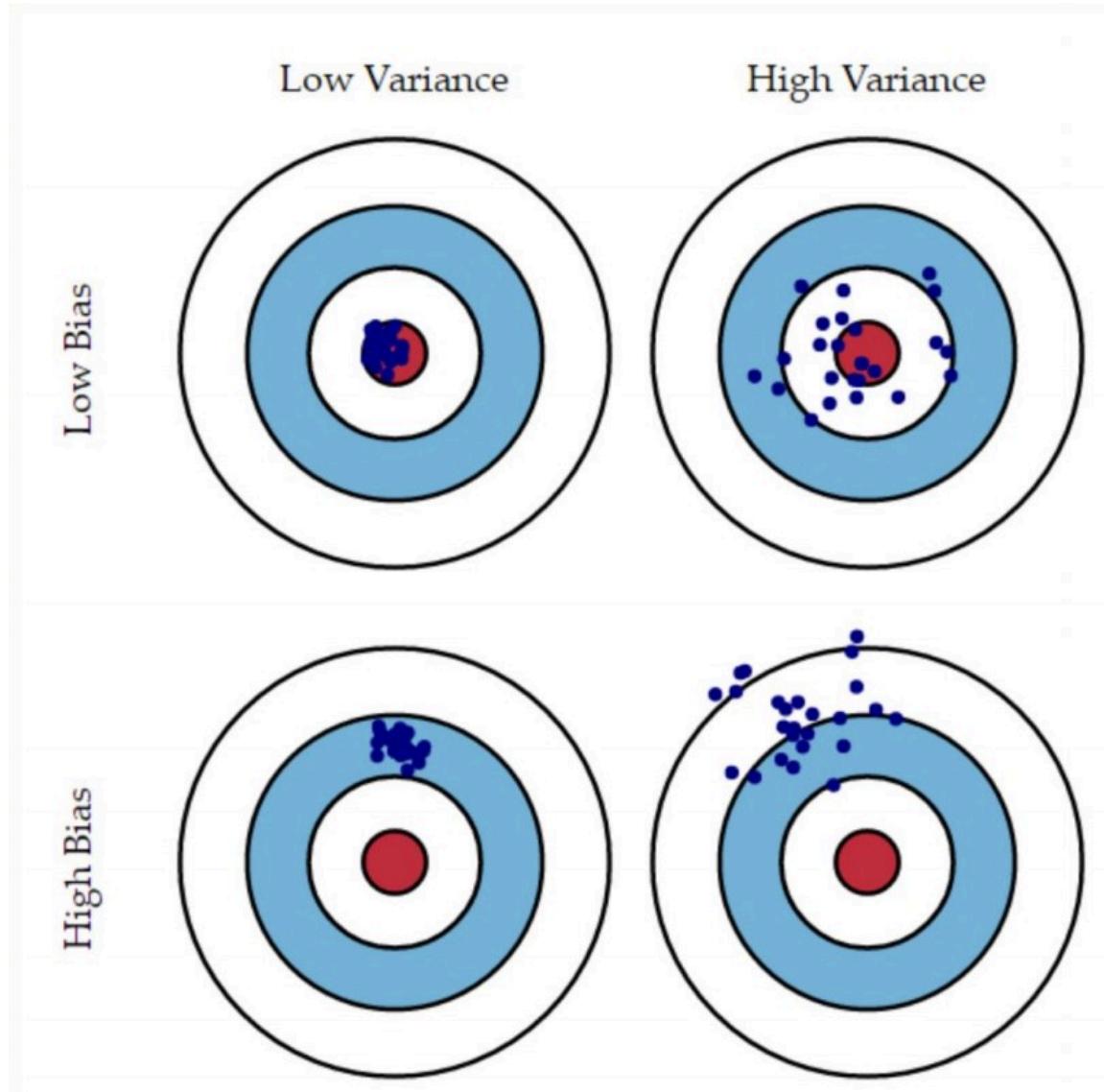
$$\text{MSE} = \mathbb{E}[(y_0 - \hat{f}(x_0))^2] = \text{Bias}^2 + \text{Var}(\hat{f}(x_0)) + \text{Var}(\varepsilon)$$

- $\hat{f}(x_0)$: model prediction at point x_0
- ε : noise in the data (irreducible error)

Bias-Variance Tradeoff

- A model with **high complexity/flexibility**:
 - ➡️  Low bias: fits the training data well, overfits
 - ➡️  High variance: sensitive to noise
- A model with **low complexity/flexibility**:
 - ➡️  High bias: too simple, underfits
 - ➡️  Low variance: more stable across different samples

Dart board interpretation of bias & variance



Classification

Basic Principles of Classification

Each Observation Consists of:

1. **Class Label** y – categorical variable
2. **Feature Vector** $\mathbf{x} = (x_1, x_2, \dots, x_p)$ – mix of categorical and continuous variables

Goal:

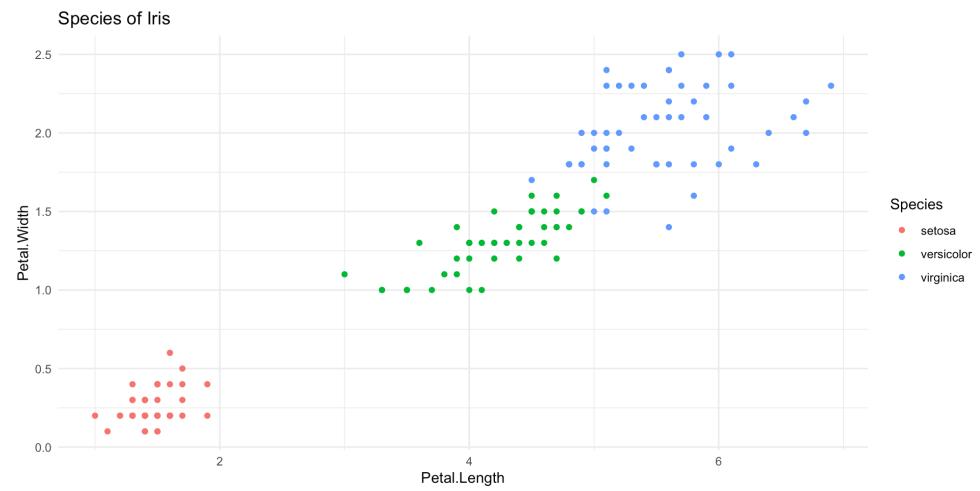
- To classify y using \mathbf{x}
 - Available data: $\{\mathbf{x}_i, y_i\}_{i=1}^n$
3. Classification models produce a continuous valued prediction, which is usually in the form of a probability (i.e., the predicted values of class membership for any individual sample are between 0 and 1 and sum to 1).
 4. Require a rule to assign observations to a category based on the predicted probability.

Classification vs Clustering

Clustering: classes are unknown, want to discover them from the data (unsupervised)



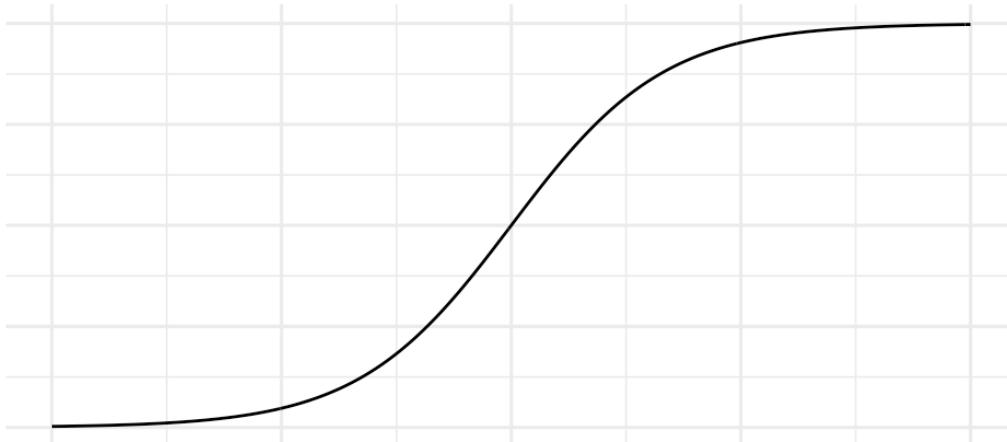
Classification: classes are predefined, want to use a training set of labeled objects to form a classifier for classification of future observations (supervised)



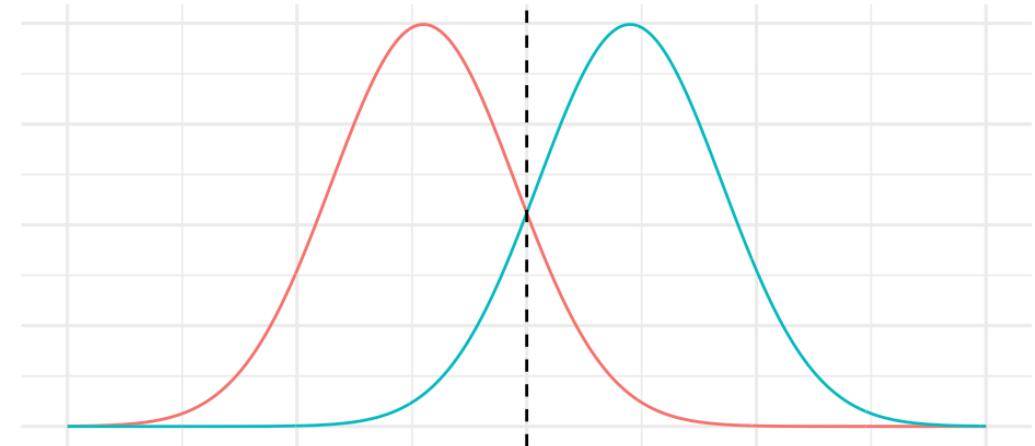
Classification algorithms to discuss

- Which of the four models are parametric classification models?

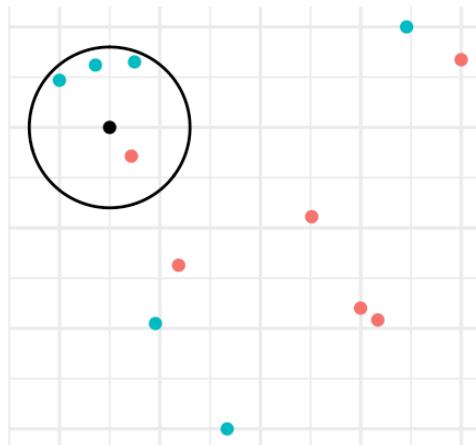
Logistic regression



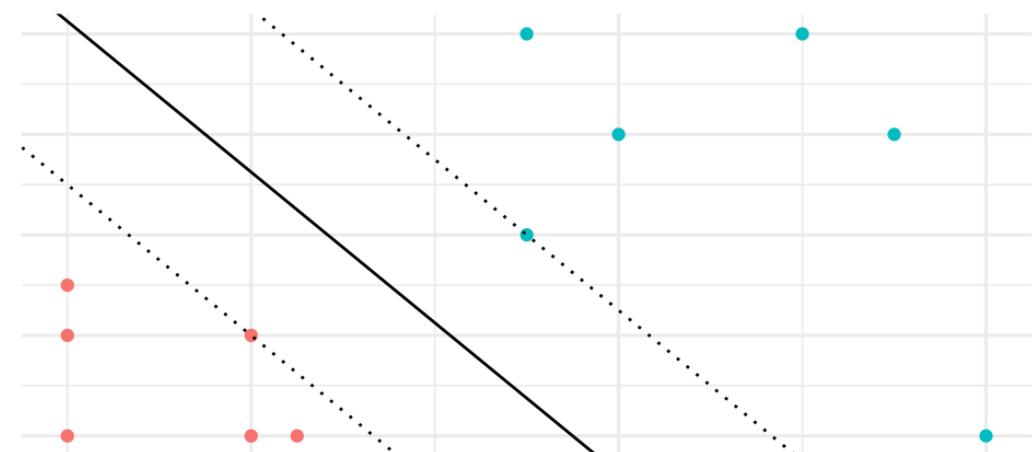
Linear Discriminant Analysis (LDA)



k-Nearest Neighbour (kNN)



Support Vector Machine (SVM)



Binary or Two class classification

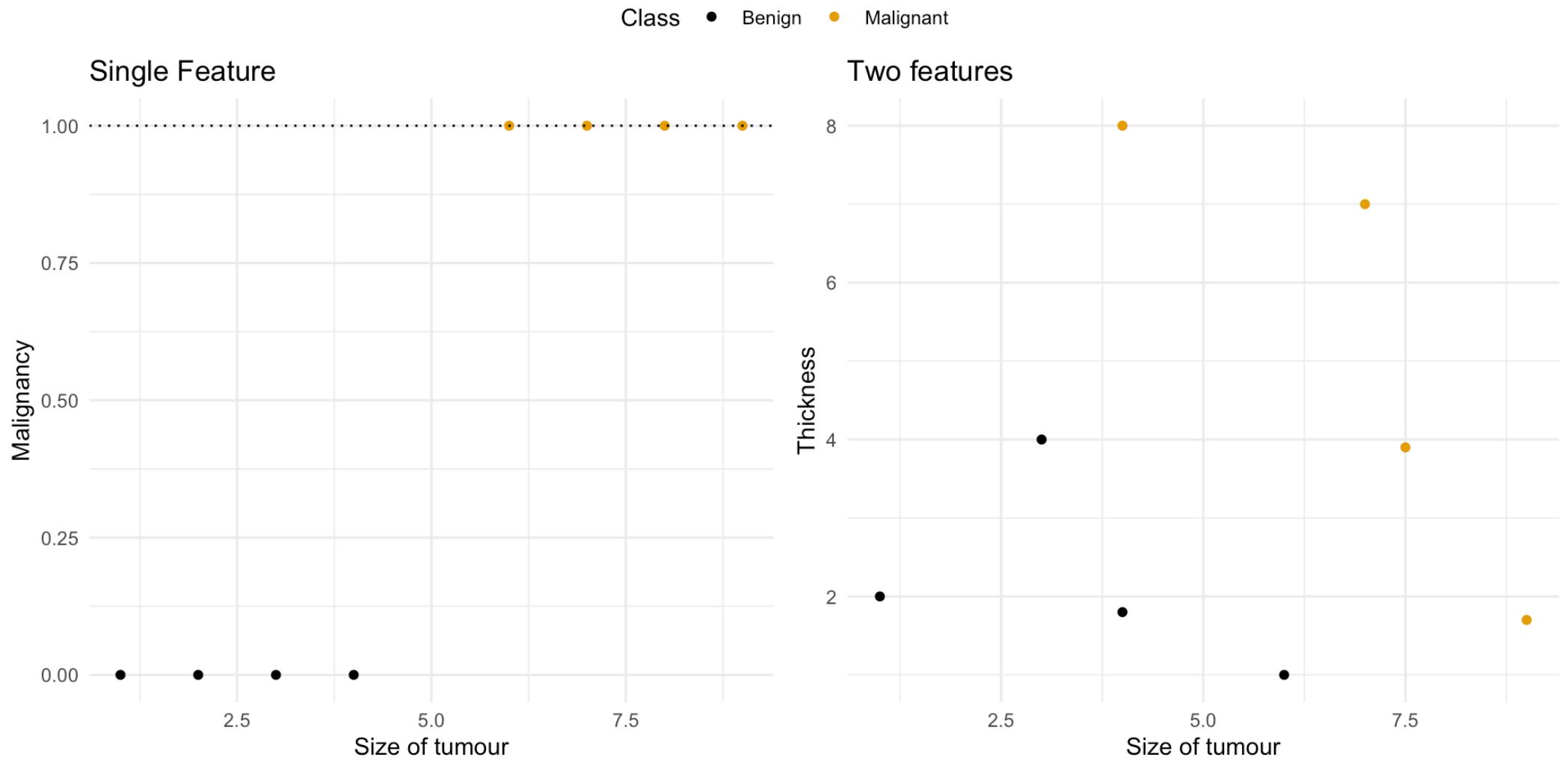
- Binary in there are two possible values (0 or 1, TRUE or FALSE)
- Examples of binary classification:
 - ➡ Email: Spam / Not Spam
 - ➡ Tumour: Malignant /Benign
- Labels are similarly described, $y \in \{0, 1\}$
 - ➡ 0: "negative class"
 - ➡ 1: "positive class"

Problem framing

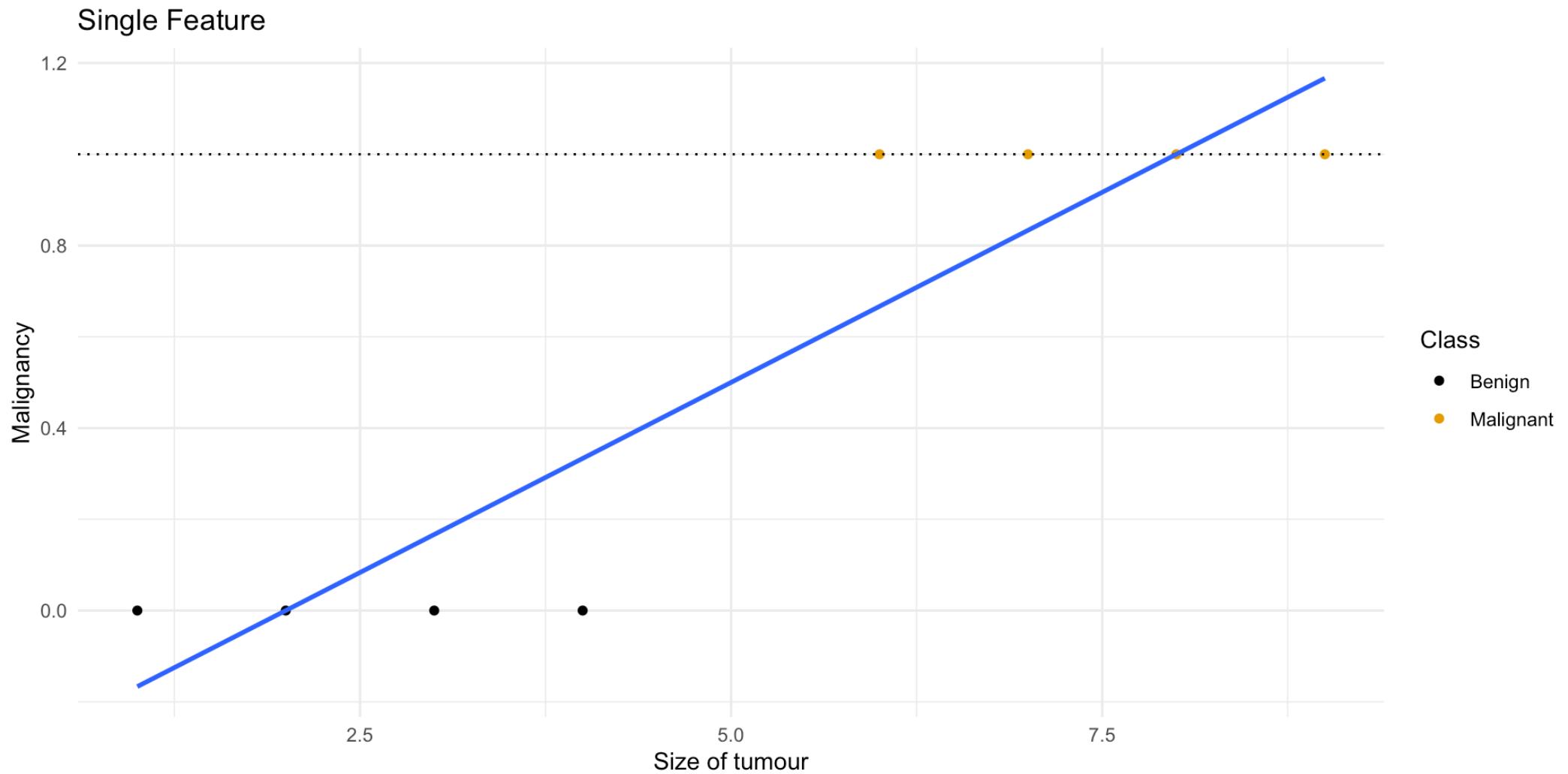
Aim is to predict the tumour type – whether it is benign (non-cancer) or malignant (cancer) – based on **tumour size**, **thickness**, and other clinical or imaging characteristics.

Need: a model outputs the probability of each tumour type. Using a decision rule, a tumour will be classified as **malignant** if $P(\text{Malignant} | \text{features}) > 0.5$, otherwise, it is assigned as **benign**.

Problem setup



Why not use simple linear regression?



Linear regression misspecifications

- The regression line $\beta_0 + \beta_1 x$ can span the entire real line
 - ➡ all values between $-\infty$ to ∞
- In the tumour diagnosis problem, the target variable y only takes two values: 0 or 1
- The linear regression model is not well specified for this purpose

Logistic regression

Logistic regression

- Previously we had the multiple regression

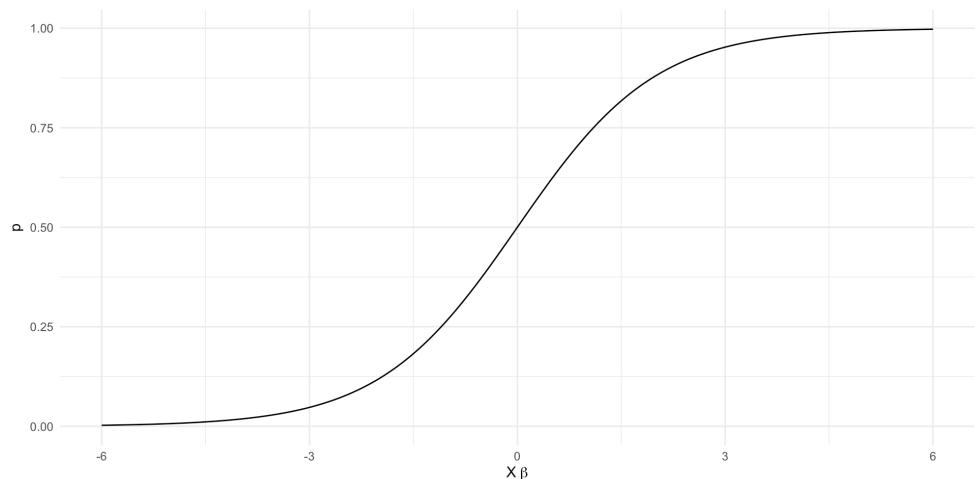
$$\begin{aligned}Y &= \mathbf{X}\boldsymbol{\beta} + \varepsilon \\&= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon\end{aligned}$$

Could write this as $\mathbb{E}[Y|\mathbf{X}] = \mathbf{X}\boldsymbol{\beta}$

- Can **generalise** this to $g(\mathbb{E}[Y|\mathbf{X}]) = \mathbf{X}\boldsymbol{\beta}$
- Logistic regression is a special case of one of these generalised linear models
- $\log\left(\frac{p}{1-p}\right) = \mathbf{X}\boldsymbol{\beta}, p = P(Y = 1|\mathbf{X})$

- Solve for p gives

$$p = g^{-1}(\mathbf{X}\boldsymbol{\beta}) = \frac{1}{1 + \exp(-\mathbf{X}\boldsymbol{\beta})}$$



Logistic regression terminology

- Logistic function $\frac{1}{1+e^{-\mathbf{X}\beta}}$
 - ➡ Responsible from mapping the features from $(-\infty, \infty) = \mathbb{R}$ to $(0, 1)$
- In logistic regression, we want the values in the logit space to be linear in \mathbf{X}
- We choose β that minimizes the negative log-likelihood (log-loss)

$$\ell(\beta) = \sum_{i=1}^n -y_i \log(p(\mathbf{x}_i)) - (1 - y_i) \log(1 - p(\mathbf{x}_i))$$

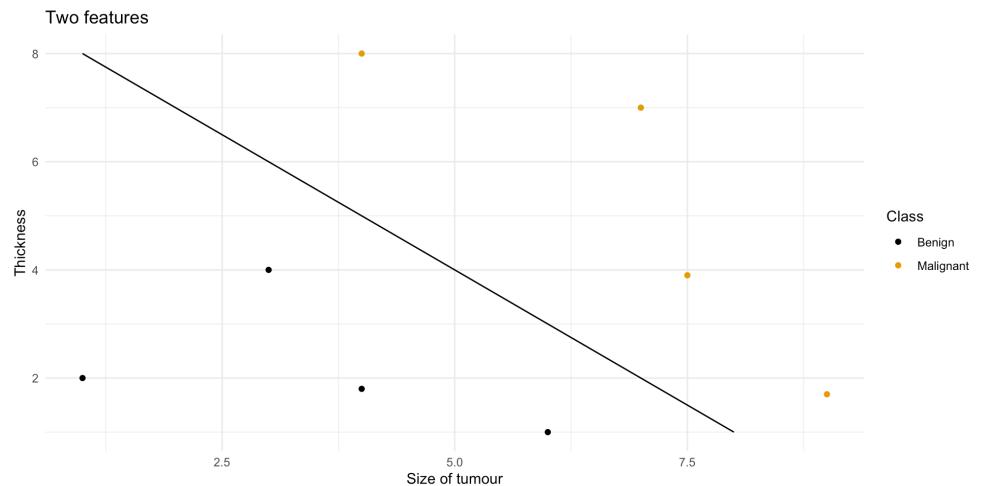
- ➡ $\{\mathbf{x}_i, y_i\}_{i=1}^n$: predictor and label pairs; $p(\mathbf{x}_i)$: predicted probability for i th data point
- ➡ Gradient-descent methods can be used for optimization

Logistic regression: decision boundary

- Decision boundary

$$P(Y = 1 | \mathbf{X}) = \frac{1}{1 + \exp(-\mathbf{X}\boldsymbol{\beta})}$$

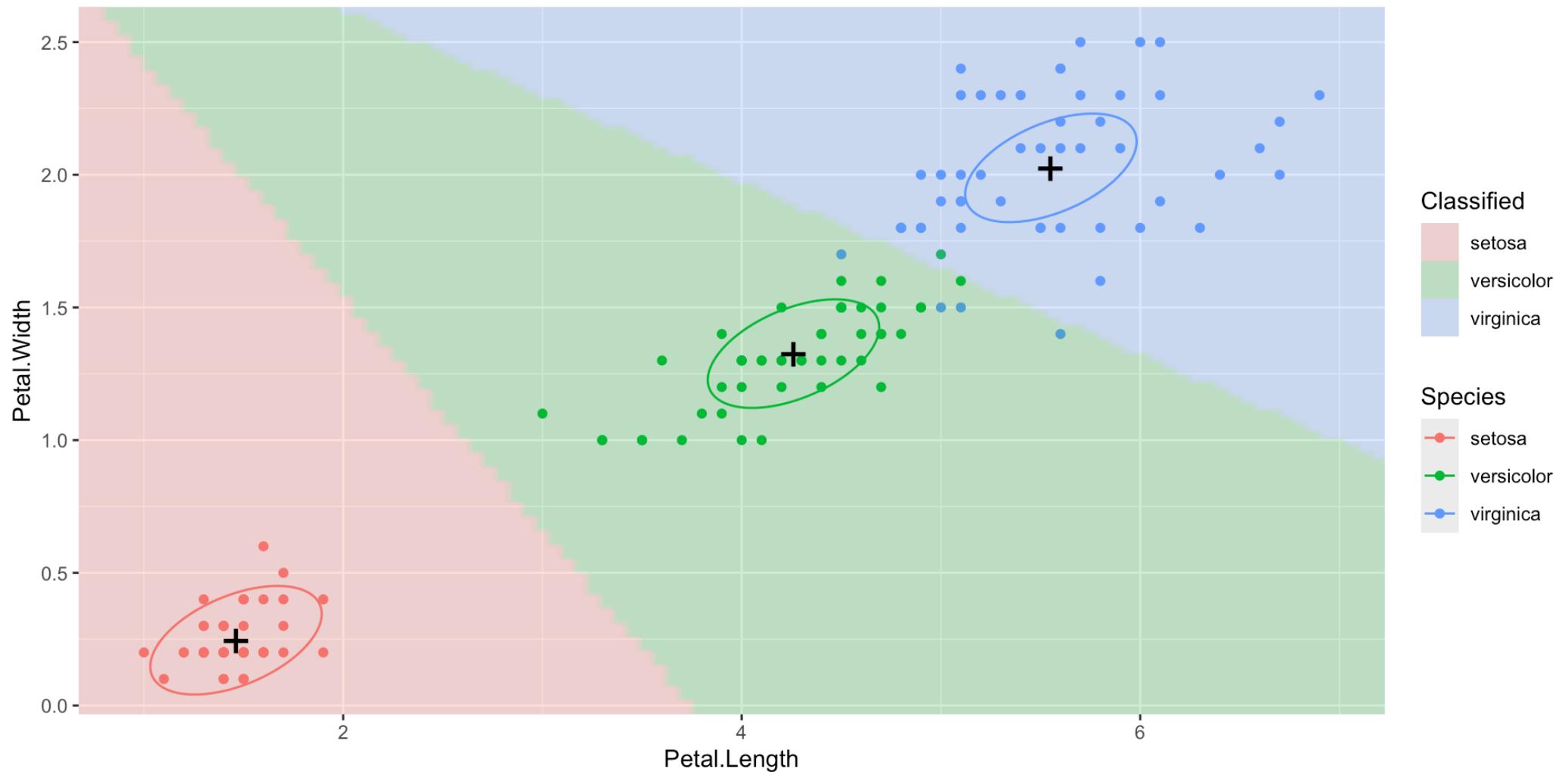
- Predict $Y = 1$ if $\mathbf{X}\boldsymbol{\beta} \geq 0$



Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA)

LDA classifies data by finding the decision boundary that best separates different classes, assuming the data follows Gaussian distributions with a class specific mean (μ_k) and a common variance (σ^2).



Bayes' Theorem in the classification context

$$p_k(x) = P(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{\ell=1}^K \pi_\ell f_\ell(x)}$$

Posterior: The probability of classifying observation to group k given it has features x

Prior: The prior probability of an observation in general belonging to group k

$f_k(x)$ is the density function for feature x given it's in group k

LDA setup

Using the Bayes' Theorem, we model

$$p_k(x) = P(Y = k | X = x) = \frac{\pi_k f_k(x)}{\sum_{\ell=1}^K \pi_\ell f_\ell(x)}$$

π_k : Probability of coming from class k (prior probability)

$f_k(x)$: Density function for X given that X is an observation from class k

- **A generative approach** that models the distribution of each class (assuming Gaussian distributions) and then uses Bayes' theorem to find the decision boundary.

LDA estimates of π_k and $f_k(x)$

- We can estimate π_k and $f_k(x)$ to compute $p_k(x)$
- The most common model for $f_k(x)$ is the Normal Density

$$f_k(x) \sim N(\mu_k, \sigma_k^2)$$

- Using the above density, we only need to estimate three quantities to compute $p_k(x)$
 - ➡ That is, μ_k , σ_k^2 , and π_k
- For simplicity, assume common variance, i.e., $\sigma = \sigma_k$ for all $k = 1, \dots, K$
 - ➡ We can allow the observations in the k th class to have a class-specific variance σ_k^2 , leading to the quadratic discriminant analysis

Use training data set for estimation

- The mean μ_k could be estimated by the average of all training observations from the k^{th} class
- The variance σ^2 could be estimated as the weighted average of variances of all K classes
- The proportion π_k is estimated as the proportion of the training observations that belong to the k^{th} class
 - ➡ n : sample size
 - ➡ n_k : number of samples in the k^{th} class

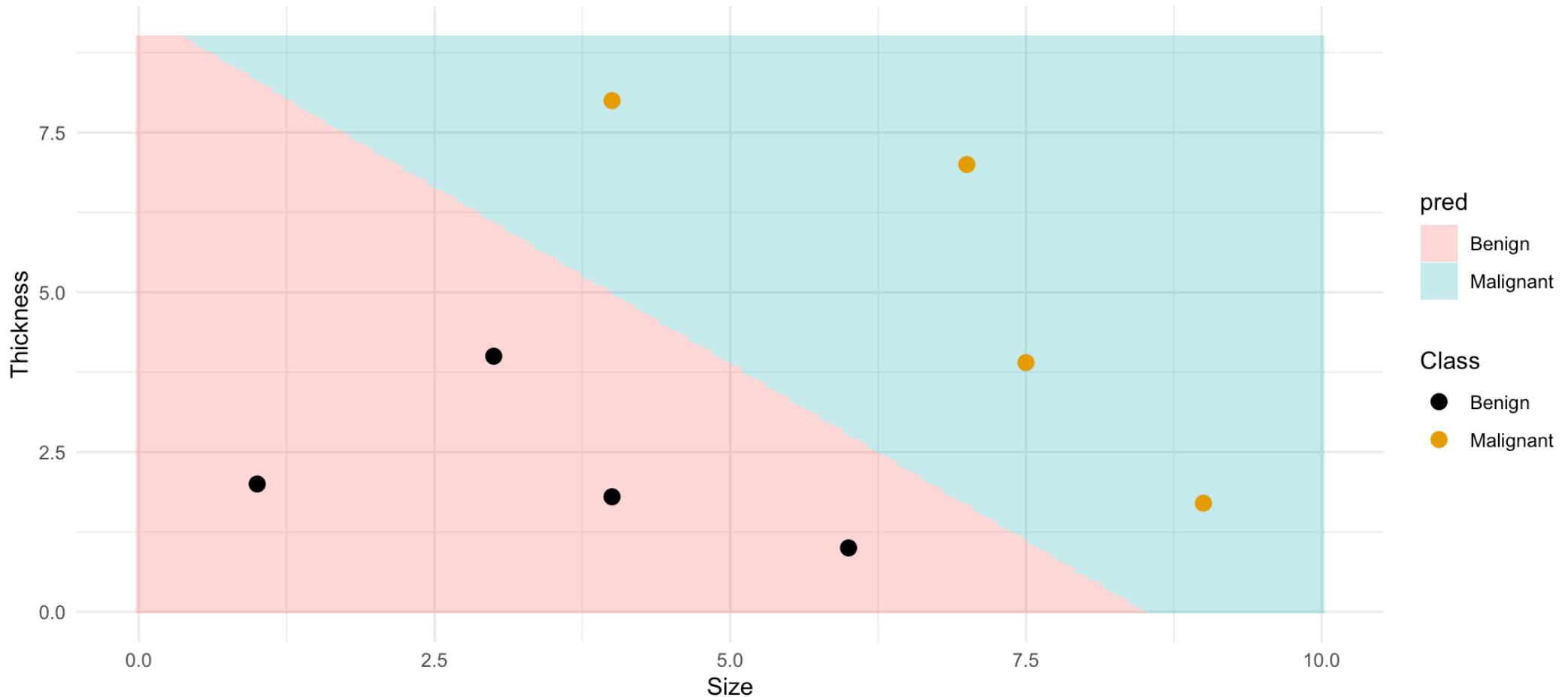
$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n - K} \sum_{k=1}^K \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2$$

$$\hat{\pi}_k = n_k/n$$

Tumour example

LDA Classification of Tumour Type
Benign vs Malignant with Decision Boundary



Why LDA?

- In the case where n is small, and the distribution of predictors \mathbf{X} is approximately normal, then LDA is **more stable** than Logistic Regression.
- LDA is more popular when we have **more than two response classes**. More intuitive to predict class assignment. But logistic regression CAN be generalised for classification problems with more than two classes.
- When classes are **perfectly separated**, the parameter estimates in logistic regression become infinite. This can lead to instability in software (e.g., R) when estimating parameters. In contrast, LDA remains stable and does not face this issue.
- LDA may perform poorly for *categorical predictors* due to the normality assumption.

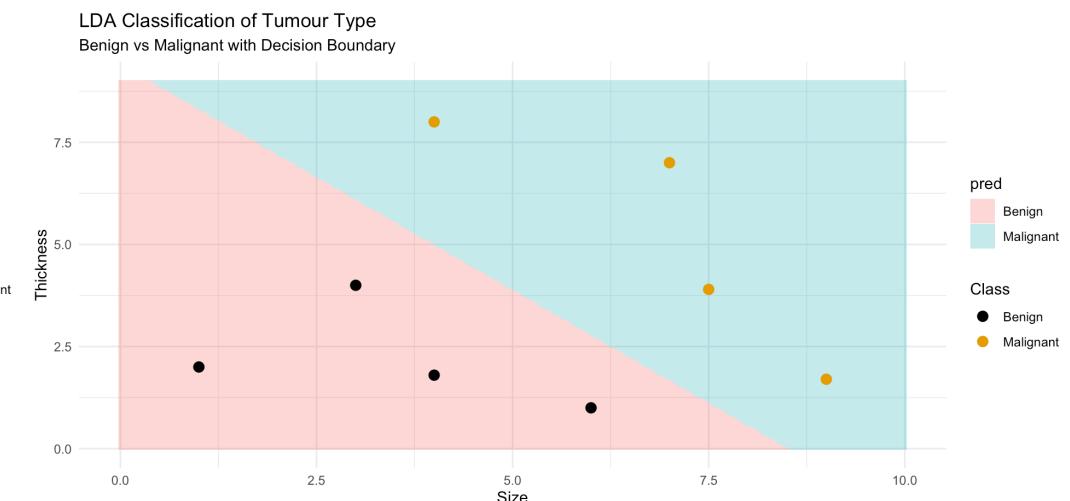
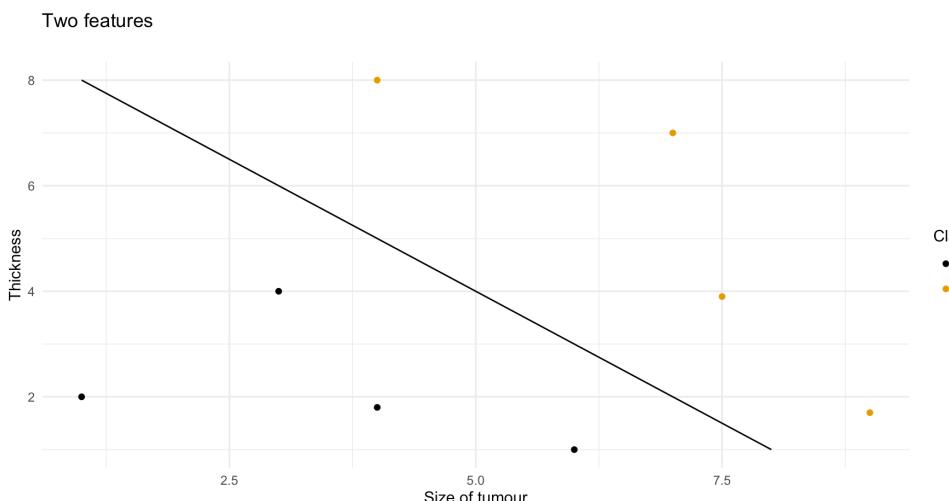
Logistic Regression vs LDA

Similarity:

- Both Logistic Regression and LDA produce **linear boundaries**

Differences:

- LDA assumes that the observations are drawn from the normal distribution with common variance in each class, while logistic regression does not have this assumption
- LDA would do better than Logistic Regression if the assumption of normality hold, otherwise logistic regression may outperform LDA



k-Nearest Neighbours (kNN)

k -Nearest Neighbours

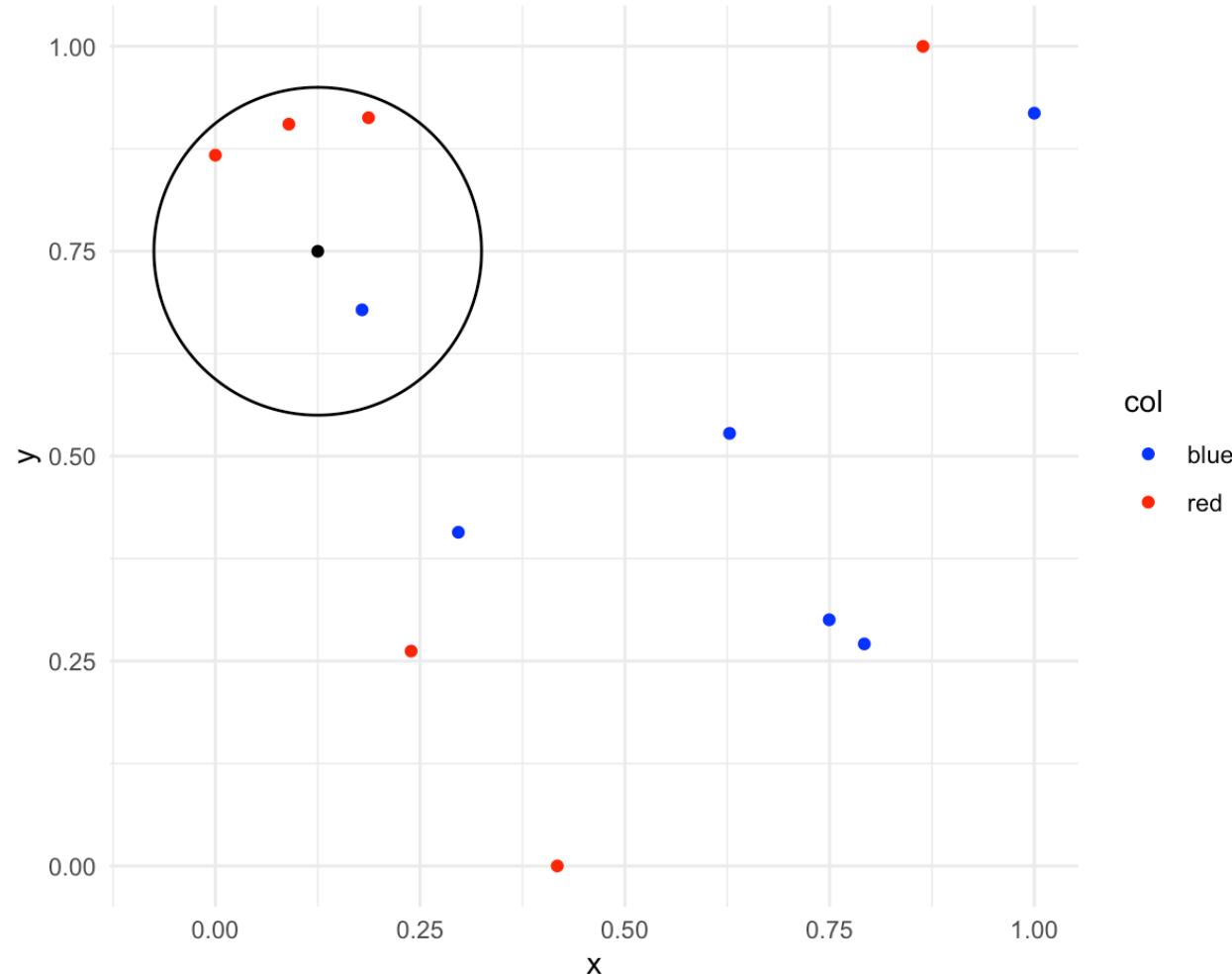
- The kNN model computes the probability of an observation with features \mathbf{x} belonging to group ℓ based on the membership of the nearest points (denoted by $N_{\mathbf{x}}^k$) to \mathbf{x}

$$P(Y = \ell | \mathbf{x}) = \frac{1}{k} \sum_{i \in N_{\mathbf{x}}^k} \mathbf{1}_{\{y_i = \ell\}}$$

⇒ $\sum_{i \in N_{\mathbf{x}}^k} \mathbf{1}_{\{y_i = \ell\}}$ counts of the closest k points that belong to group ℓ

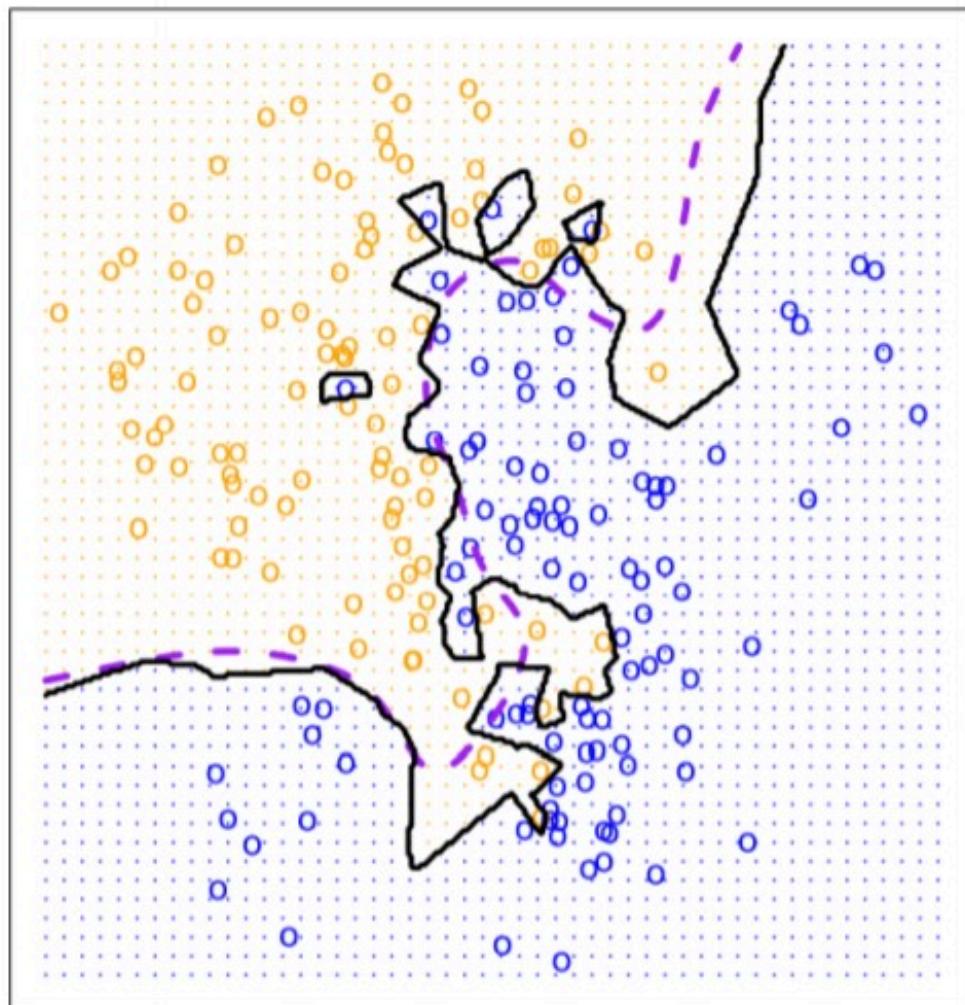
k -Nearest Neighbours

- Suppose $k = 4$ is chosen. At the candidate black point. The four nearest neighbours are inspected. There is probability 3/4 of being in the red group and 1/4 for being in the blue group.

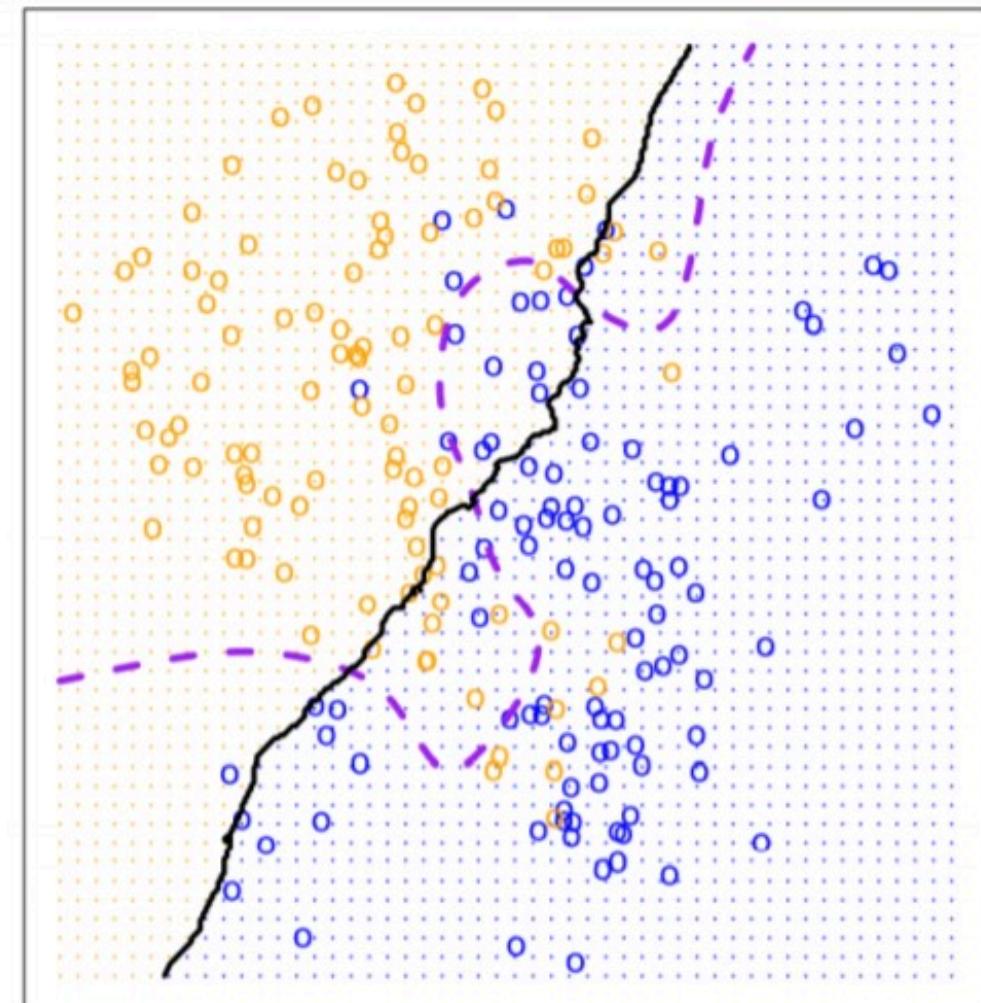


k -Nearest Neighbours

KNN: K=1



KNN: K=100



kNN vs LDA and Logistic Regression

kNN is **completely non-parametric**: No assumptions are made about the shape of the decision boundary

Advantage

We can expect kNN to dominate both LDA and Logistic Regression when the decision boundary is **highly non-linear**

Disadvantage

kNN does not tell us which predictors are important (no table of coefficients)

- The choice of the number of K matters.

Support Vector Machines (SVM)

Support Vector Machines (SVM)

- SVM finds

a decision boundary (hyperplane) that best separates the two classes by maximising the margin between them.

What is a hyperplane?

- In p dimensions, it is a flat affine subspace of dimension $p - 1$
- General equation has the form

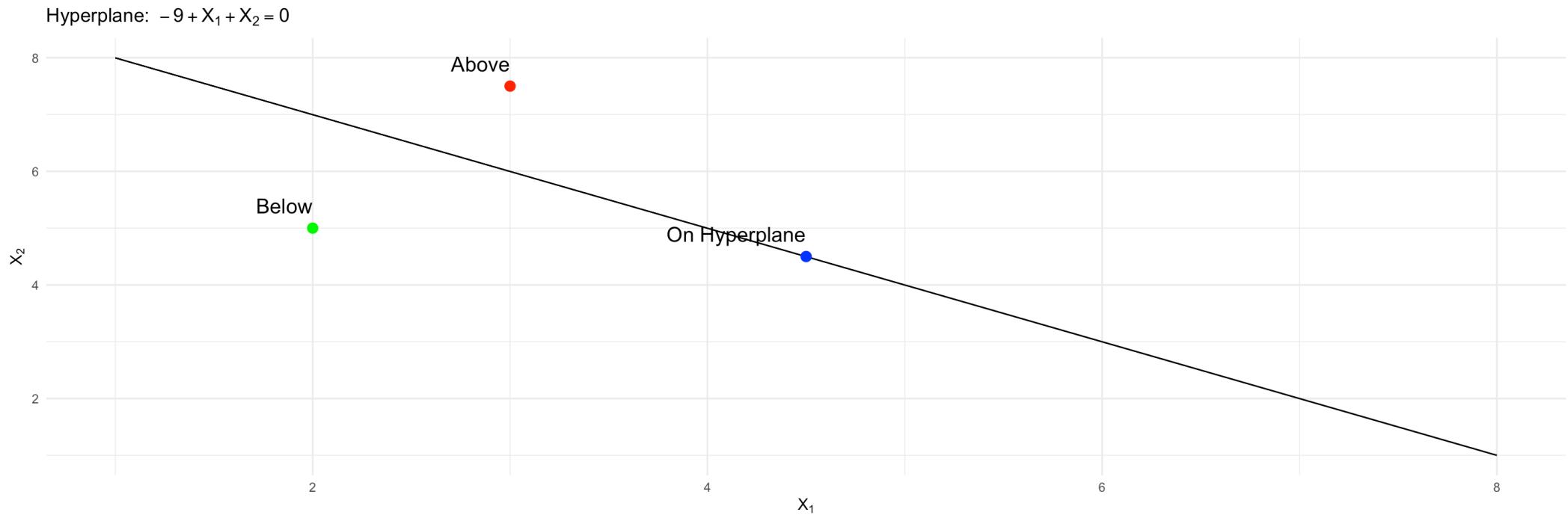
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = 0$$

If $\beta_0 = 0$, the hyperplane passes through the origin, otherwise it does not

- The vector $(\beta_1, \beta_2, \dots, \beta_p)$ is called the normal vector - It points in a direction orthogonal to the surface of the hyperplane

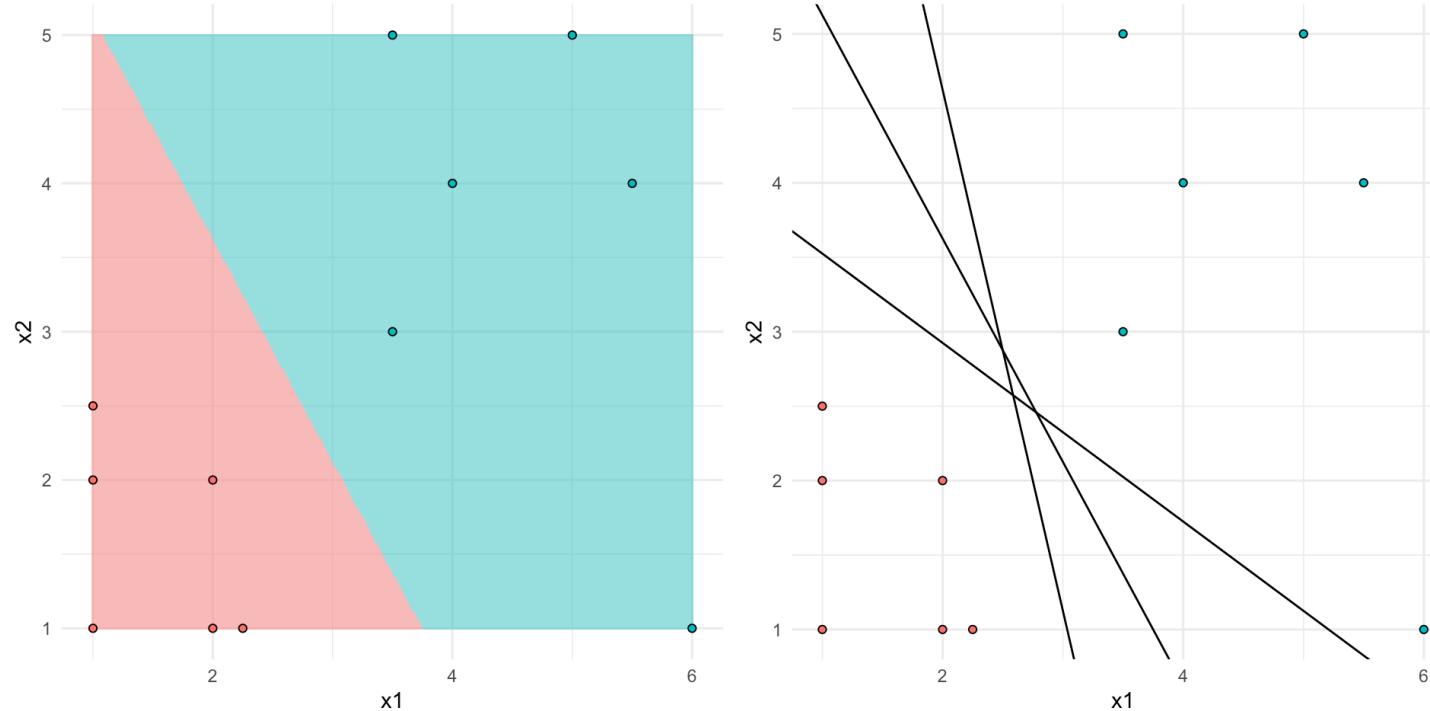
A Hyperplane

In $p = 2$ dimensions, the hyperplane is a line

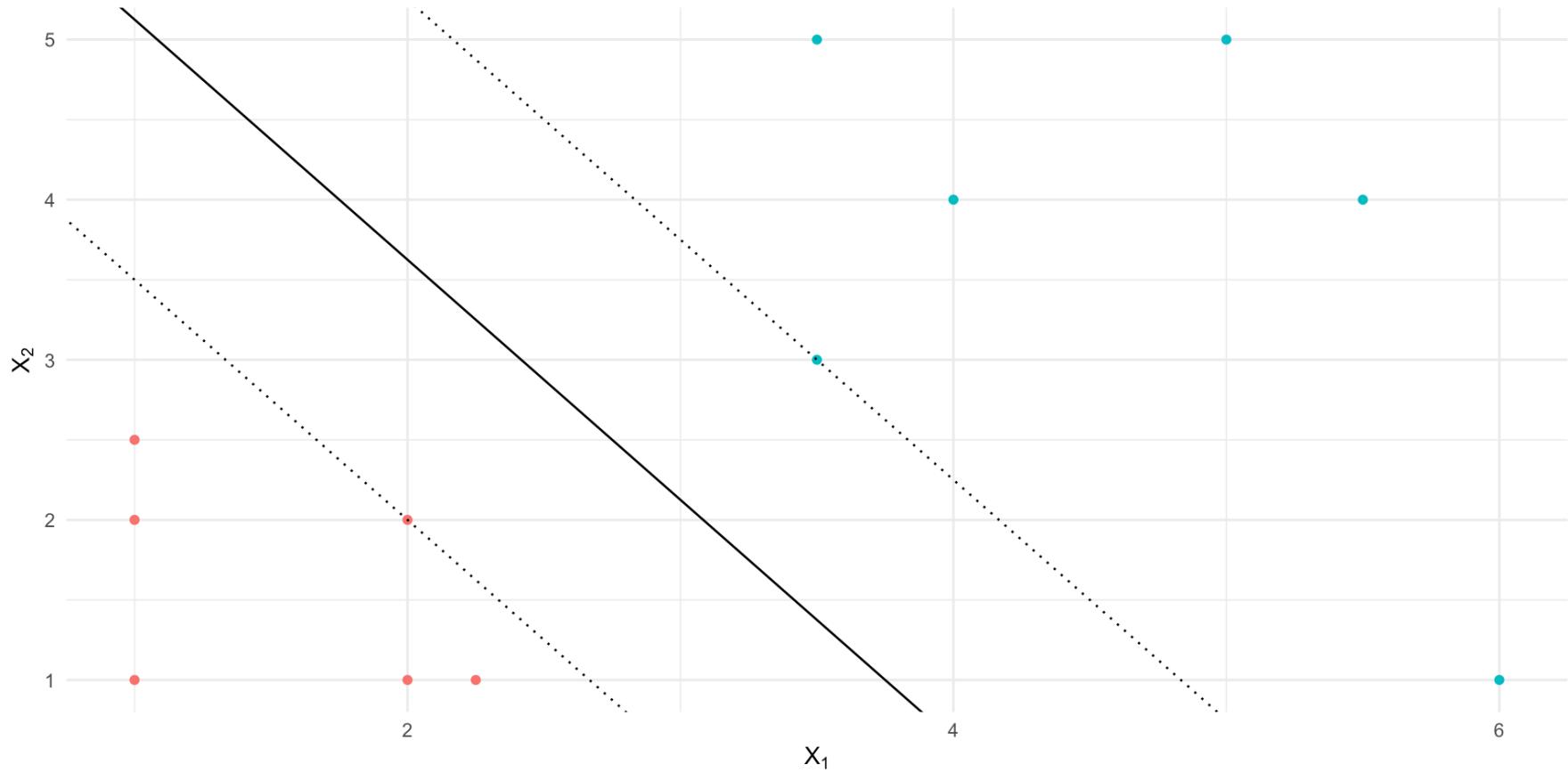


Separating hyperplanes

- Consider coding Benign (red) as $y_i = -1$ and malignant (blue) as $y_i = 1$
- Then $y_i f(x_i) > 0$ for all i , $f(x_i)$ defines a separating hyperplane.
- If $f(x_i) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$ defines a hyperplane
 - ⇒ $f(x) > 0$ defines a region on one side of the hyperplane
 - ⇒ $f(x) < 0$ defines a region on one other side of the hyperplane

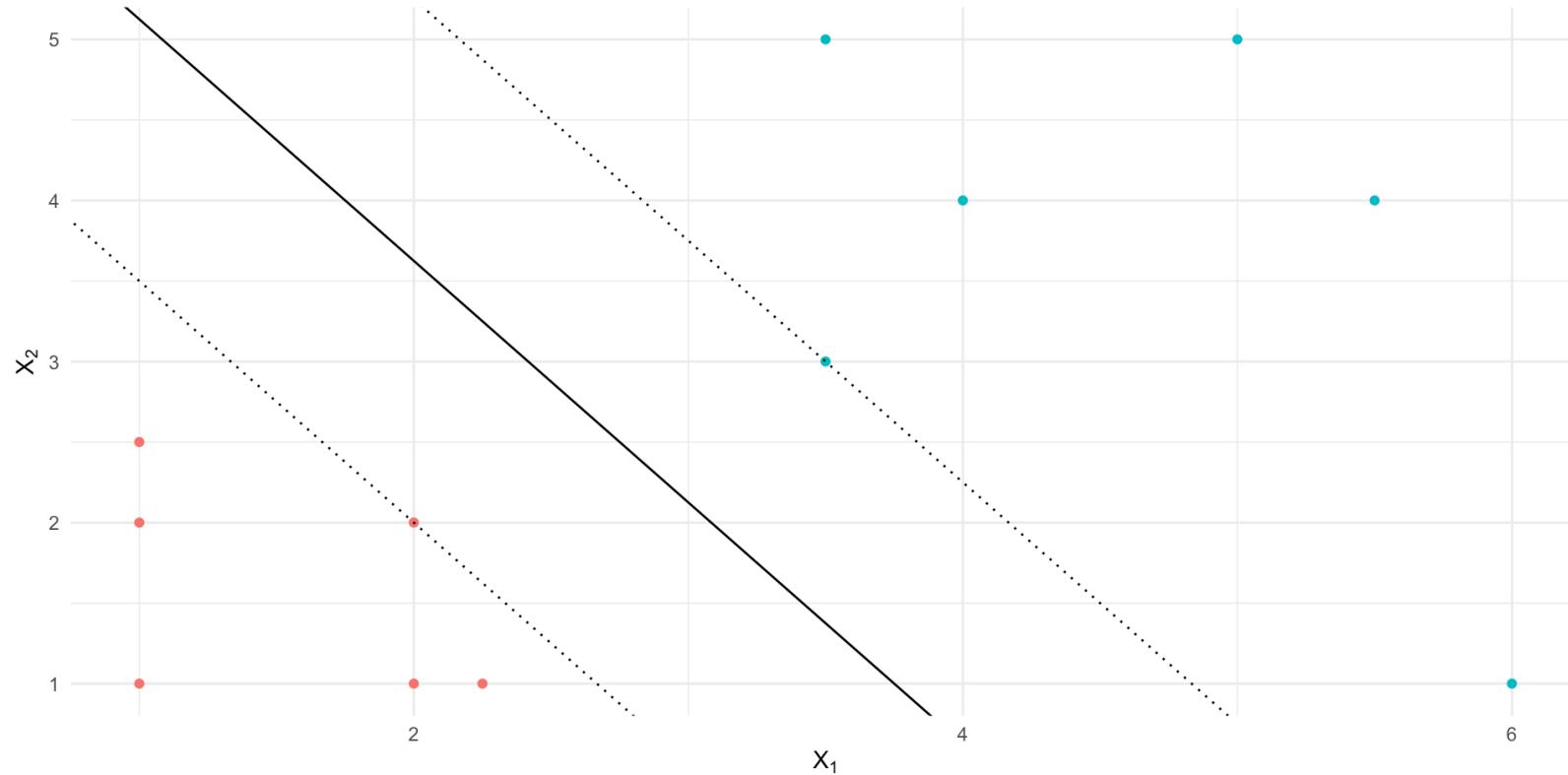


Maximal Margin Classifier



$$\max_{\beta_0, \beta_1, \beta_2, \dots, \beta_p} M \quad \text{such that} \quad \sum_{j=1}^p \beta_j^2 = 1$$
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip}) \geq M, \quad i = 1, \dots, n$$

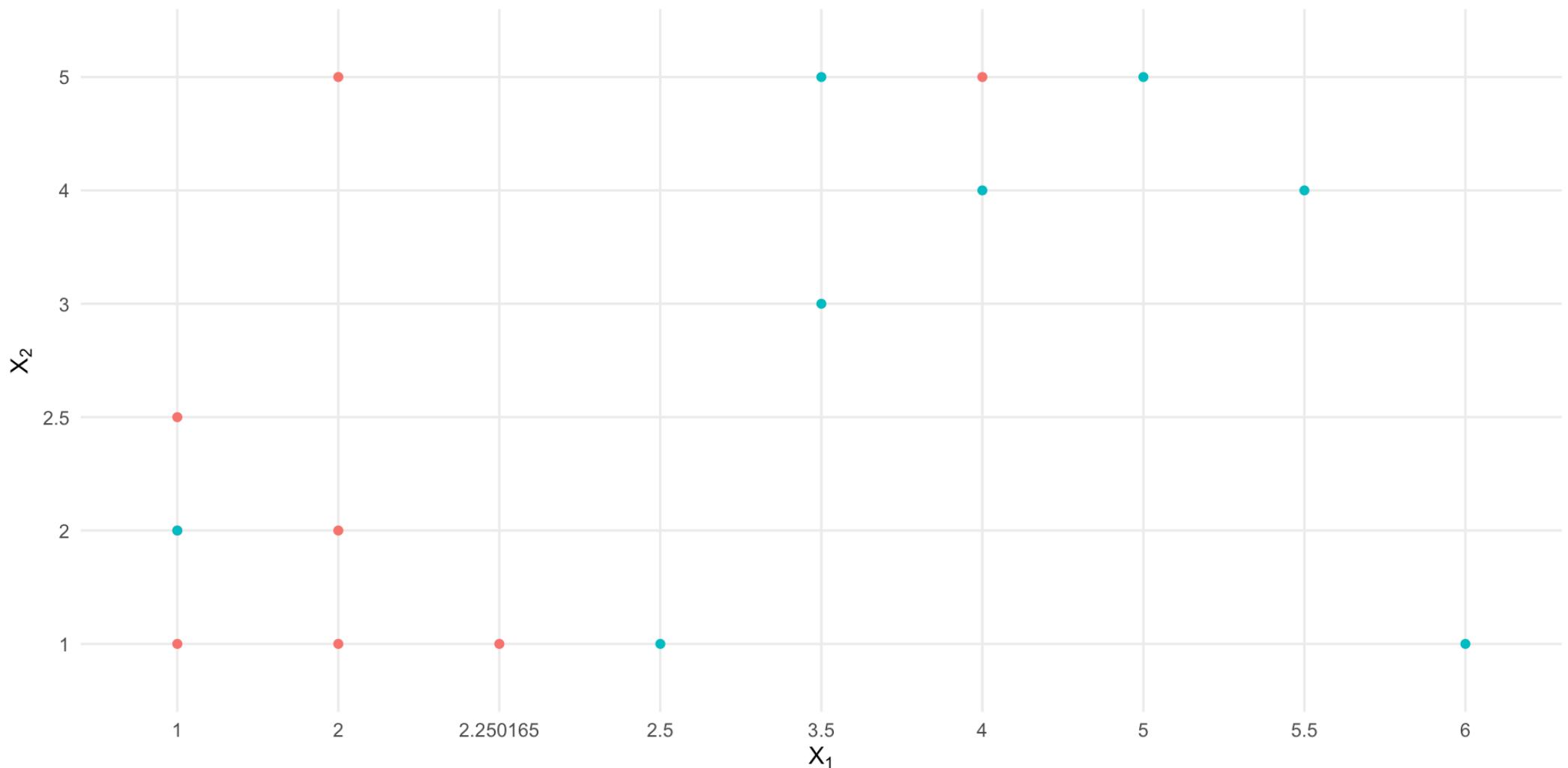
Maximal Margin Classifier



What are support vectors?

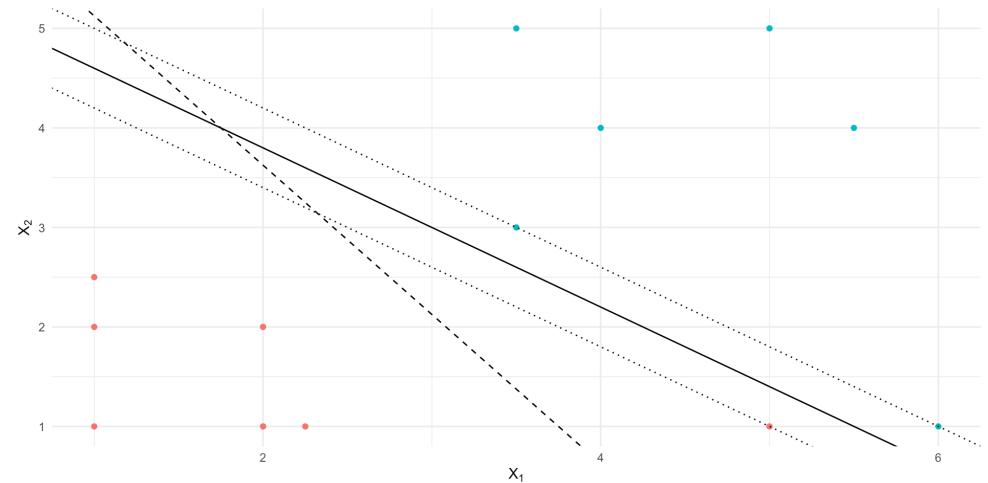
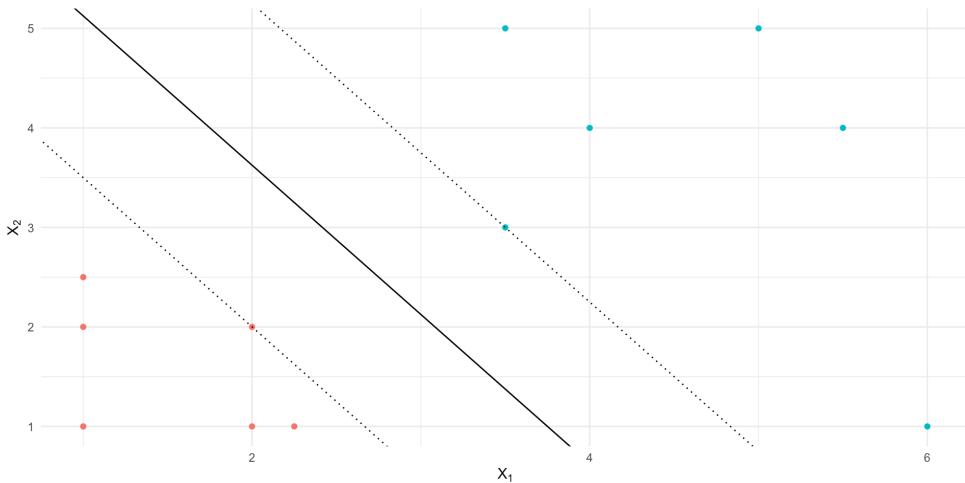
- A. all the points on the graph.
- B. the points that are closest to the decision boundary (dashed lines).
- C. the red and the blue point line on the dashed lines.

Difficult case 1: Non-perfect separation



- There is no linear boundary (hyperplane) that perfectly separates the classes

Difficult case 2: Effect of noisy data



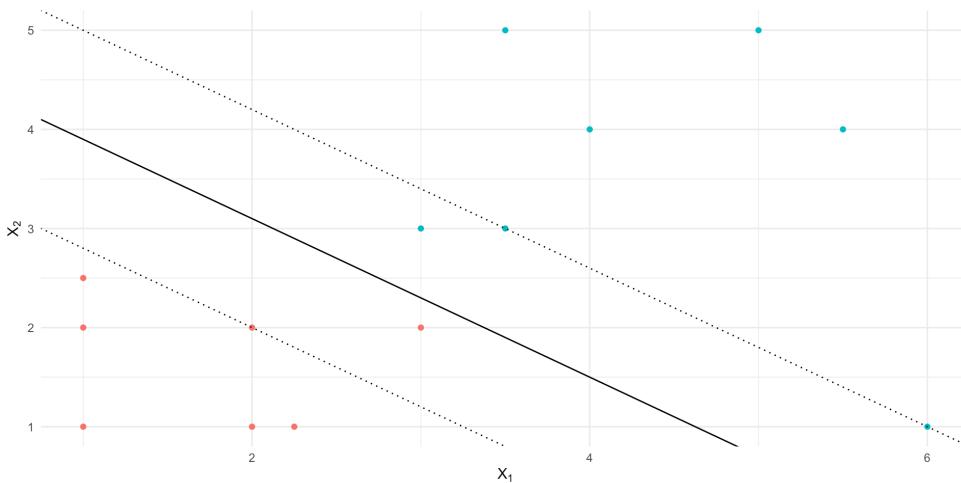
- Data could be separable, but noisy \rightsquigarrow unstable solution for the maximal margin classifier.

Support Vector Classifier

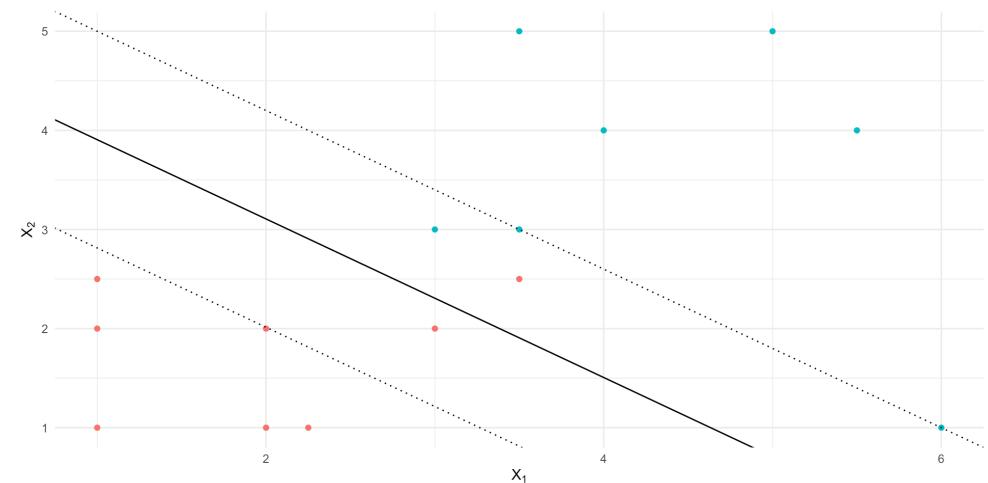
- If a basic mathematical plane is not possible due to non-perfect separation or outliers:
 - ➡ introduce a **soft margin** to relax the requirement for all observations to be on the correct side of the margin.

Soft margin examples

- Observations allowed in the margin
- Observations on the correct side of hyperplane



- Allow observations on incorrect side of hyperplane



Support Vector Classifier

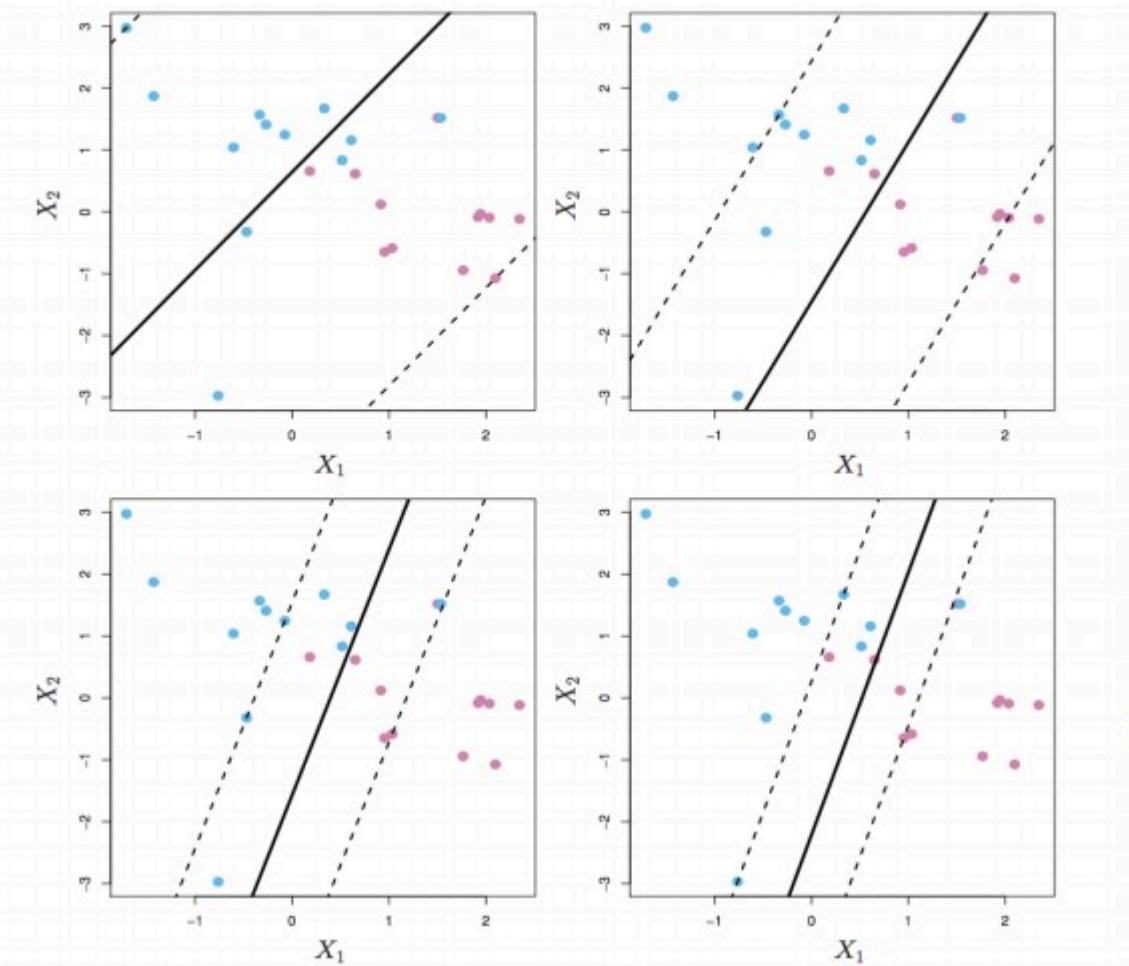
Support Vector Classifier solves the following optimization problem:

$$\begin{aligned} \max_{\beta_0, \beta_1, \beta_2, \dots, \beta_p, \epsilon_1, \dots, \epsilon_n} \quad & M \quad \text{such that} \quad \sum_{j=1}^p \beta_j^2 = 1 \\ & y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i), \quad i = 1, \dots, n \\ & \epsilon_i \geq 0, \quad i = 1, \dots, n, \quad \sum_{i=1}^n \epsilon_i \leq C \end{aligned}$$

- C is a non-negative tuning parameter
- M is the width of the margin
- ϵ_i are slack variables that allow observations to be on the wrong side of the margin
 - ⇒ if $\epsilon_i > 1$, then observation i is on the wrong side of the hyperplane
 - ⇒ if $0 < \epsilon_i \leq 1$, then observation i is on the correct side but inside margin
 - ⇒ if $\epsilon_i = 0$, then observation i is on the correct side and past the margin
- Support vectors: observations that either lie on or violate the margin

Impact of Tuning Parameter C

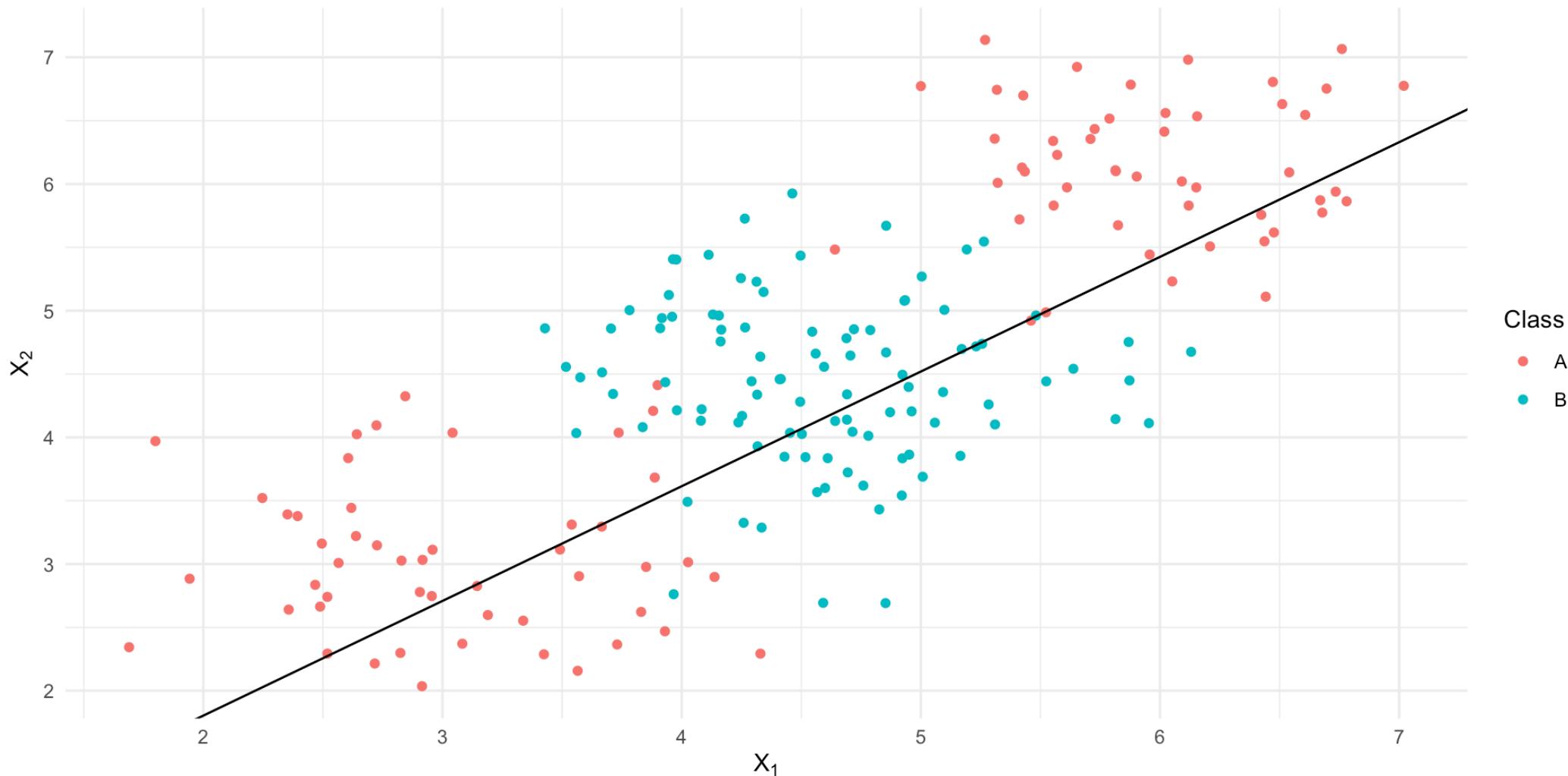
Largest C



Smallest C

Limitations of support vector classifier

- Single linear boundary can be insufficient

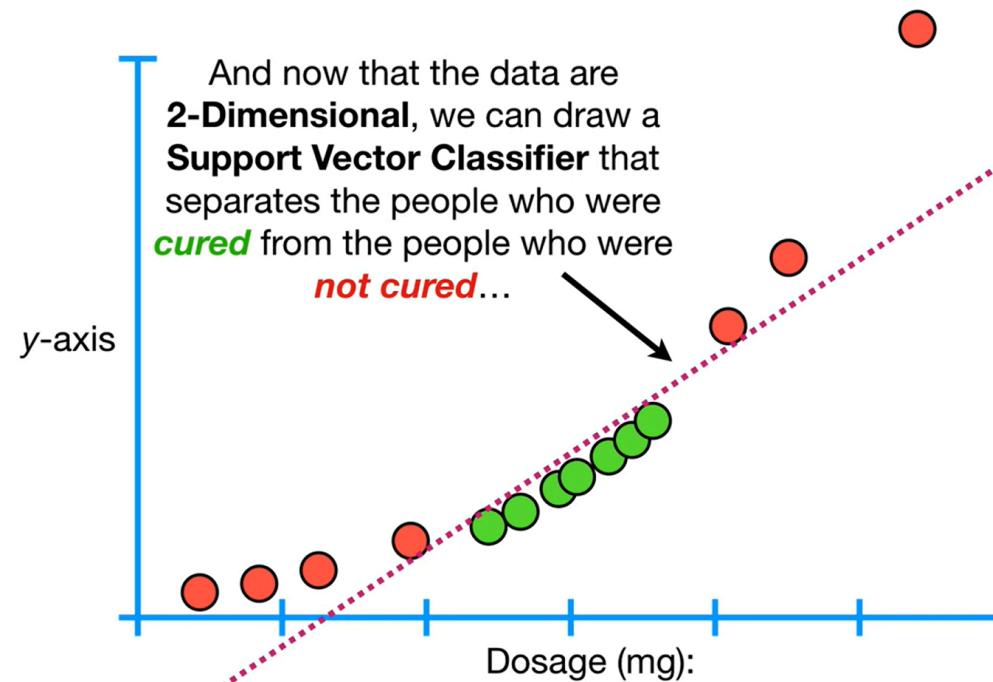


Graphical illustration of the idea

Original data:



Construct a new feature Dosage² using the original data:



Feature space expansion

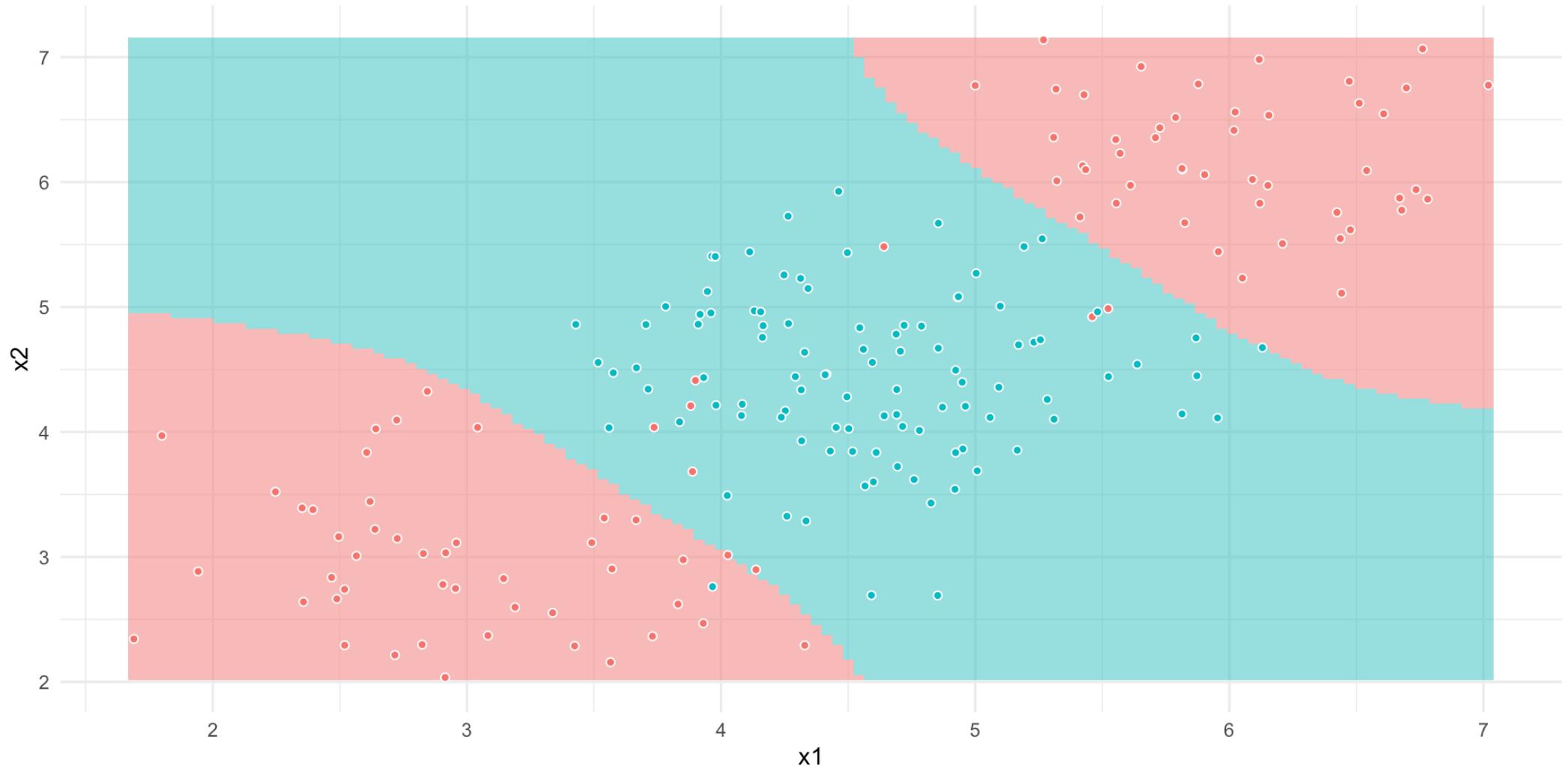
- Enlarge the space of features by including transformations:
 - ➡ e.g., new features that are powers and products X_1^2, X_1^3, X_1X_2
 - ➡ Hence go from p -dimensional space to $P > p$ dimensional
- Fit (linear) support vector classifier in the expanded feature space
 - ➡ Impact is a **non-linear** decision boundary in original feature space
- Example: Suppose we start off in 2-dimensional feature space (X_1, X_2)
 - ➡ Make new feature space $(X_1, X_2, X_1^2, X_2^2, X_1X_2)$
 - ➡ Then the decision boundary would be of the form:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 = 0$$

- This leads to non-linear decision boundary in the original space (quadratic conic sections)
- A nice illustration on [YouTube](#)

Sixth order polynomial

- Using degree six polynomial expansion



Non-linearity and kernels

- Polynomials add complexity and scale poorly in high dimensions.
- **Kernels** allow us to introduce non-linearity efficiently without explicitly adding features.
- The **inner product trick** enables the SVCs to classify non-linear data in an elegant and computationally efficient way.

Inner products and support vectors

- Recall $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ip}) \in \mathbb{R}^p$
- Inner product between vectors given by

$$\langle \mathbf{x}_i, \mathbf{x}_j \rangle = \sum_{k=1}^p x_{ik} x_{jk}$$

- Recall the hyperplane equation

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$$

- The linear support vector classifier can be represented as

$$f(\mathbf{x}) = \beta_0 + \sum_{i=1}^n \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle$$

Support set

- The parameters $\alpha_1, \alpha_2, \dots, \alpha_n$ and β_0 are estimated based on the inner products between pairs of training observations
- $\widehat{\alpha}_i$ is non-zero only for the support points \mathbf{x}_i (i.e., ones that lie on the margin), otherwise it equals to zero.

$$f(\mathbf{x}) = \beta_0 + \sum_{j \in S} \alpha_j \langle \mathbf{x}, \mathbf{x}_j \rangle$$

- Here, S is the support set of indices such that $\alpha_i > 0$

Kernel functions

Suppose now we replace the inner product with a generalized function of the form

$$K(\mathbf{x}_i, \mathbf{x}_j)$$

This function is called a kernel

In this context, the kernel quantifies the similarity of two observations

- Examples

⇒ Polynomial kernel

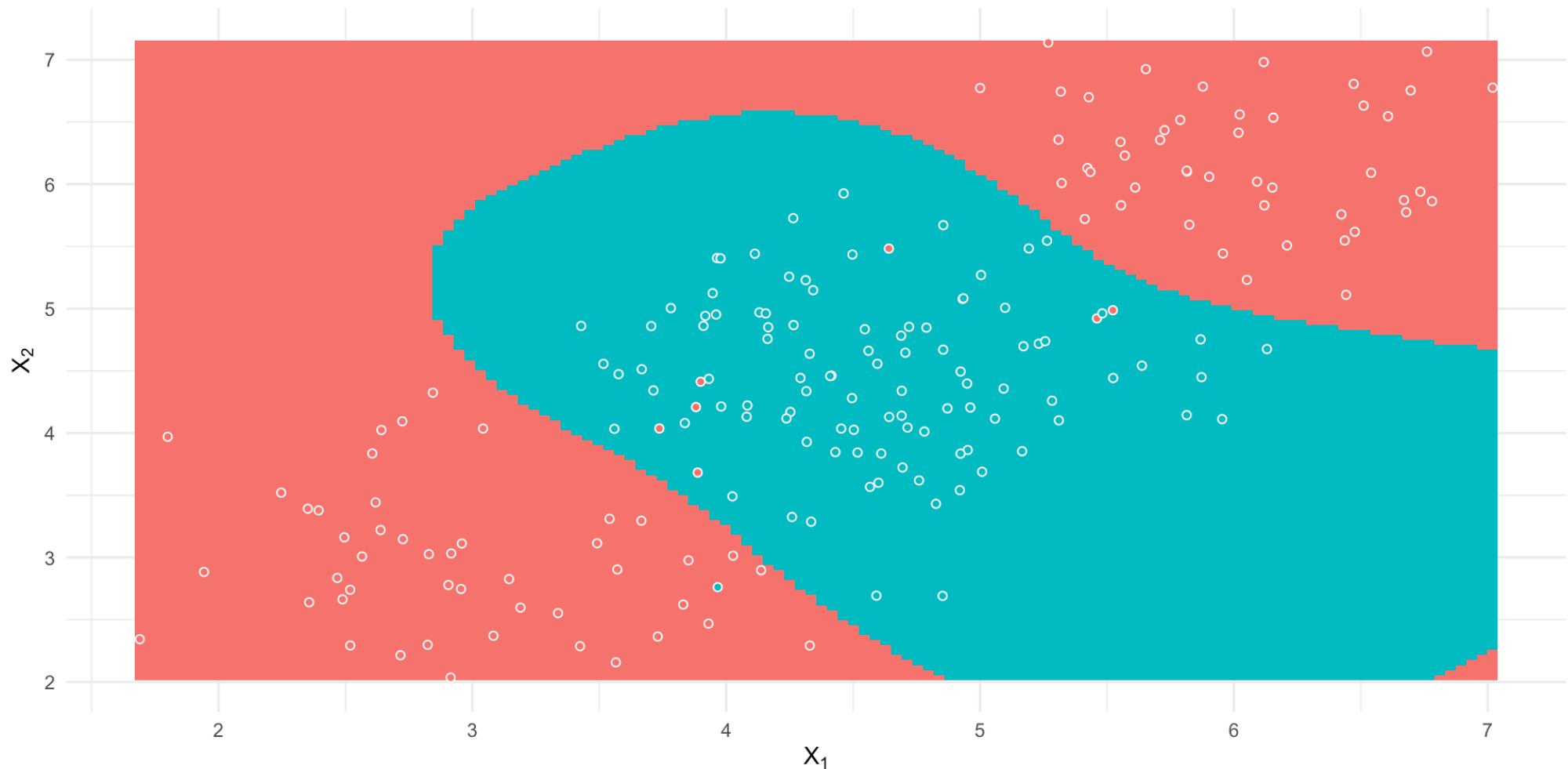
$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \langle \mathbf{x}_i, \mathbf{x}_j \rangle)^d$$

⇒ Gaussian radial kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp \left(-\gamma \sum_{k=1}^p (x_{ik} - x_{jk})^2 \right)$$

Support vector machines with the radial kernel

```
1 svm.model <- svm(x = data.mat[,-3], y = data.mat[[3]],  
2                     kernel = "radial", type = "C-classification",  
3                     degree = 4, cost = 64, fitted = FALSE)
```



SVM with more than two classes

- SVM covered previously are design for binary classification. To expand this to K classes there are two options.

1. One vs all:

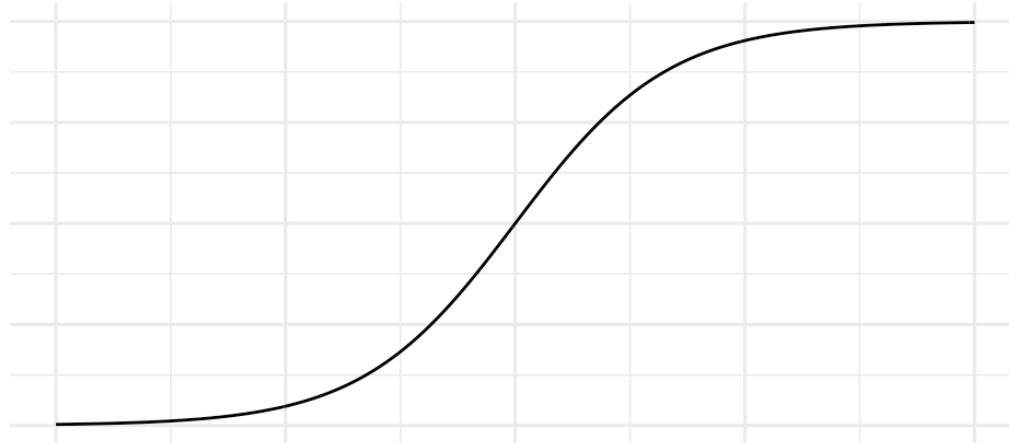
- Fit K different binary classifiers, $f_k(\mathbf{x})$ for $k = 1, 2, \dots, K$ where each boundary attempts to separate class k vs the rest.
- Then \mathbf{x}_i is classified to k^* where $f_{k^*}(\mathbf{x}_i) > f_j(\mathbf{x}_i)$ for all $j \neq k^*$ (i.e., the largest distance from the boundary).

2. One vs one:

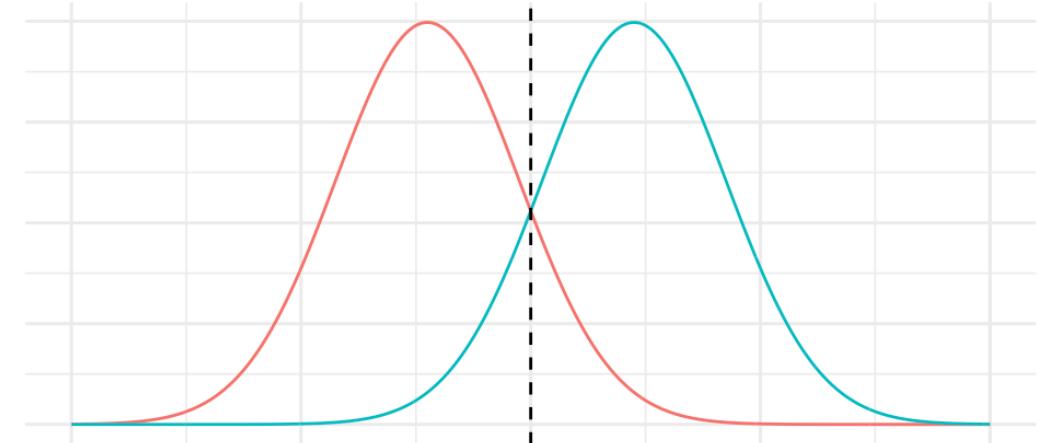
- Fit all $\binom{K}{2}$ pairwise classifiers.
- Fit \mathbf{x}_i to the class that wins the most pairwise comparisons.
- Which to use?
 - ➡ If K is small, do one vs one. Otherwise recommended one vs all.

Which of the four models are non-parametric classification models?

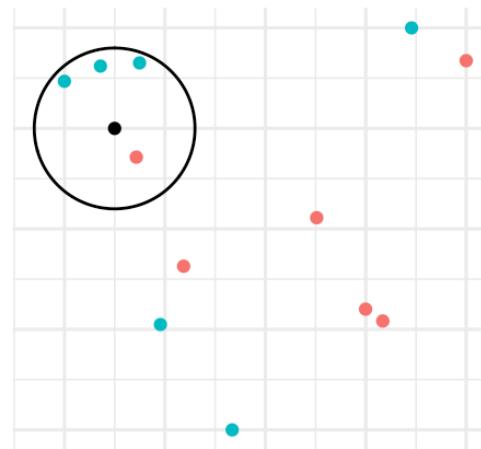
Logistic regression



Linear Discriminant Analysis (LDA)



k-Nearest Neighbour (kNN)



Support Vector Machine (SVM)

