

# **Quantitative Text Analysis**

## **Meeting 7**

**Petro Tolochko**

# Naïve Bayes Classifier

- Probabilistic classifier
- Simple
- Fast
- Good Accuracy

# Bayes Theorem

$$P(A \mid B) = \frac{P(B \mid A) \times P(A)}{P(B)}$$

# Bayes Theorem

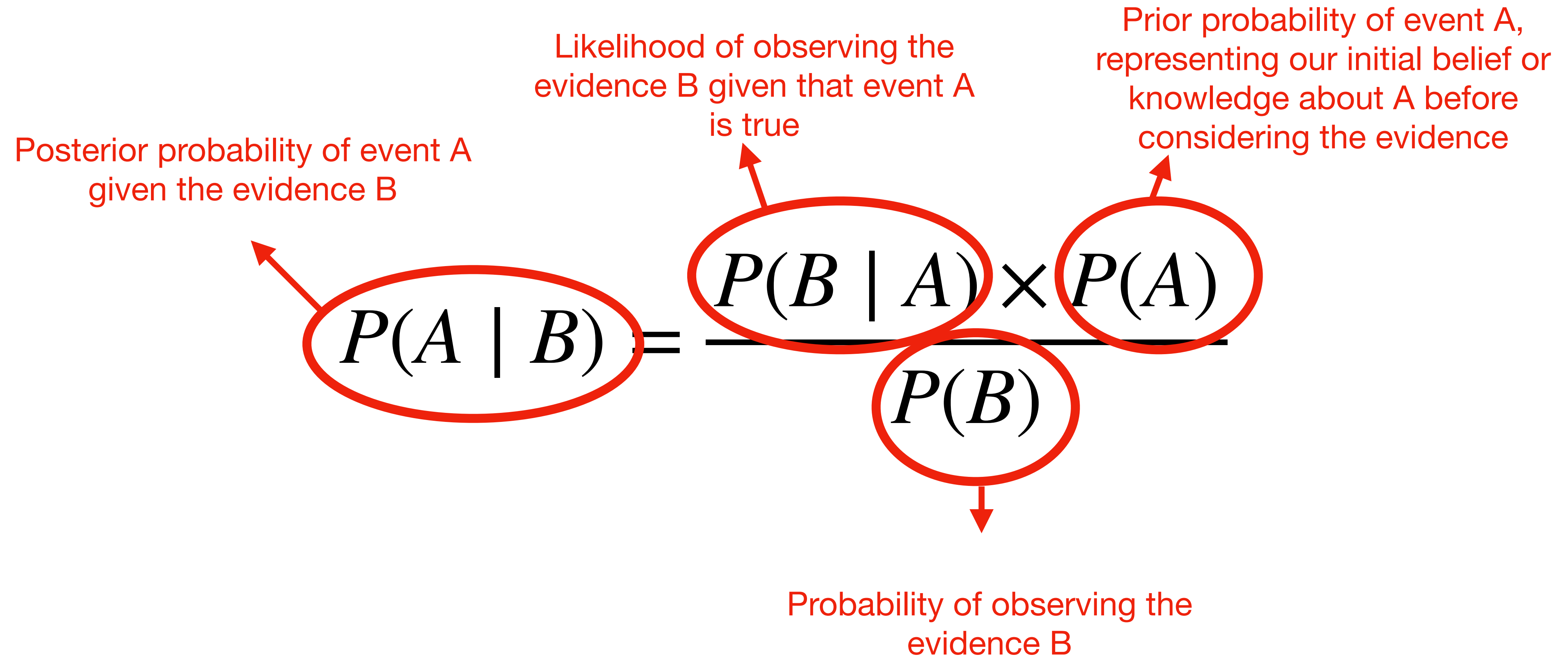


Diagram illustrating Bayes Theorem with annotations:

Posterior probability of event A given the evidence B

Likelihood of observing the evidence B given that event A is true

Prior probability of event A, representing our initial belief or knowledge about A before considering the evidence

Probability of observing the evidence B

$$P(A | B) = \frac{P(B | A) \times P(A)}{P(B)}$$

The diagram shows the Bayes Theorem formula with red ovals highlighting each term. Red arrows point from descriptive text labels to these terms: 'Posterior probability of event A given the evidence B' points to  $P(A | B)$ ; 'Likelihood of observing the evidence B given that event A is true' points to  $P(B | A)$ ; 'Prior probability of event A, representing our initial belief or knowledge about A before considering the evidence' points to  $P(A)$ ; and 'Probability of observing the evidence B' points to  $P(B)$ .

# Bayes Theorem

$$P(A|B) \propto P(B|A) \times P(A)$$

# Naïve Bayes Classifier

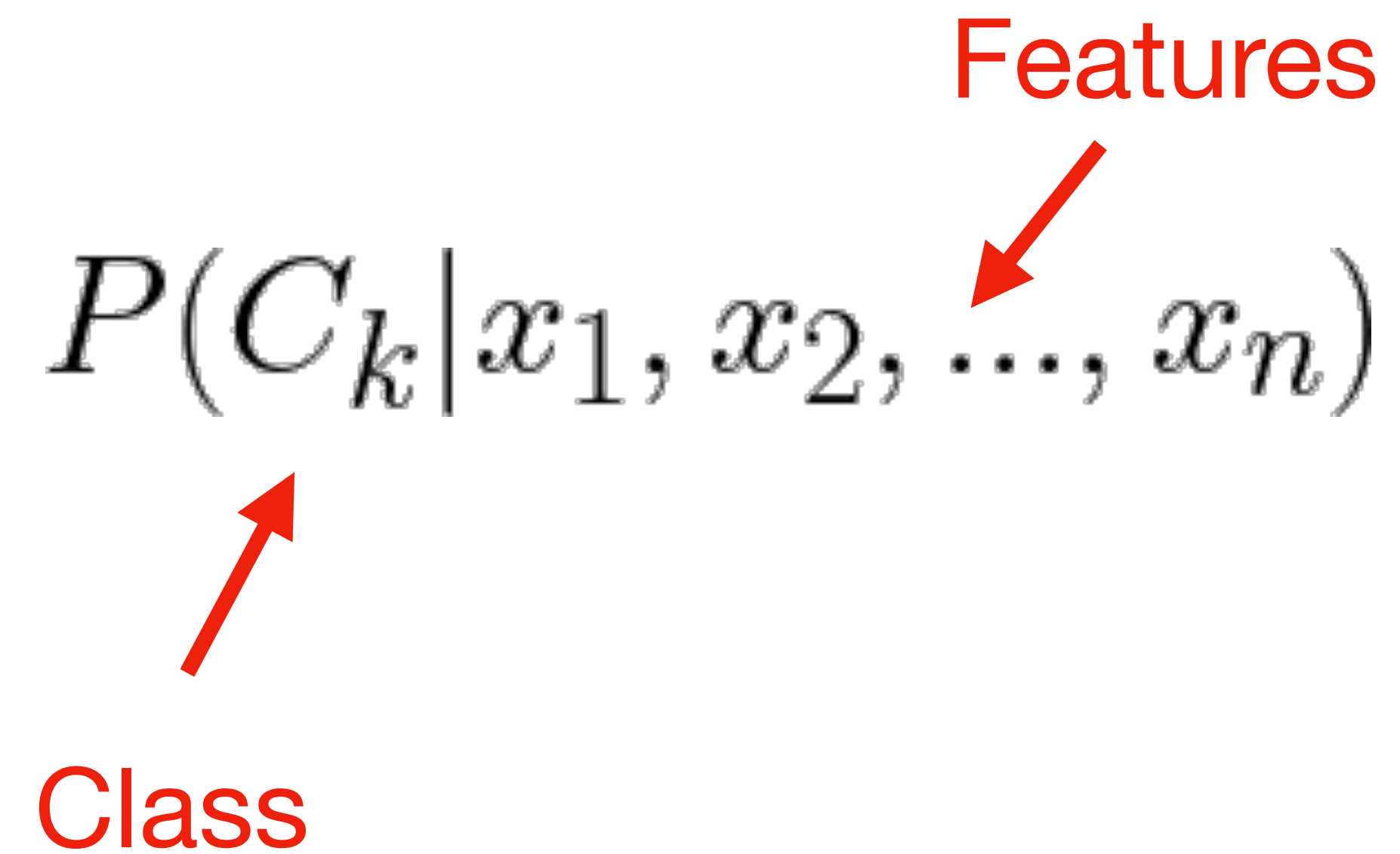
$$P(C_k | x_1, x_2, \dots, x_n)$$

# Naïve Bayes Classifier

$$P(C_k | x_1, x_2, \dots, x_n)$$

Features

Class



The diagram shows the equation  $P(C_k | x_1, x_2, \dots, x_n)$  in a black serif font. A red arrow points from the word 'Features' to the ellipsis '...', and another red arrow points from the word 'Class' to the term  $C_k$ .

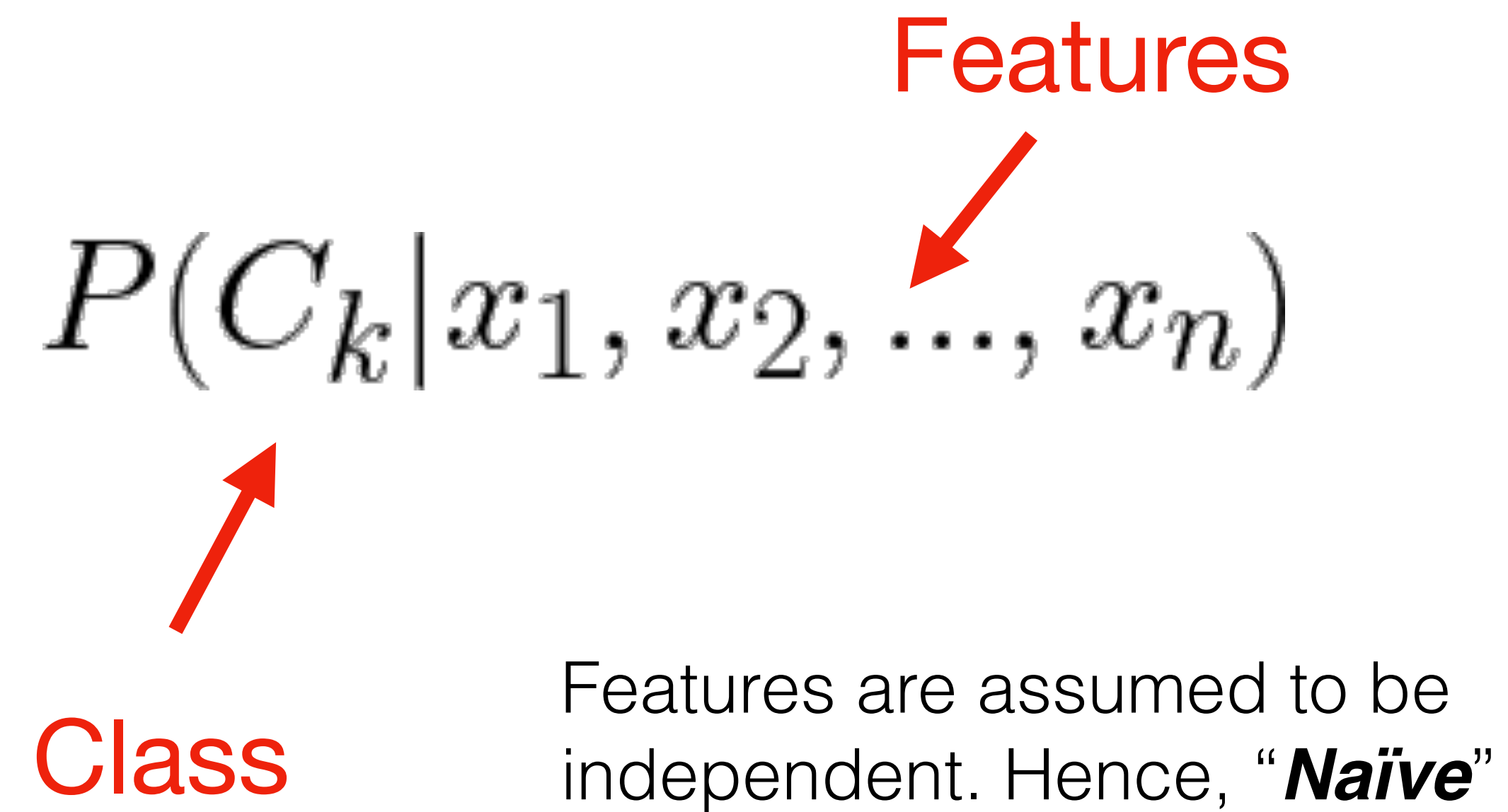
# Naïve Bayes Classifier

$$P(C_k | x_1, x_2, \dots, x_n)$$

Class

Features

Features are assumed to be independent. Hence, “**Naïve**”

The diagram shows the probability equation for a Naïve Bayes Classifier. The equation is  $P(C_k | x_1, x_2, \dots, x_n)$ . A red arrow points from the label 'Class' to the term  $C_k$  in the numerator. Another red arrow points from the label 'Features' to the ellipsis  $\dots$  in the denominator. Below the equation, a text block explains that features are assumed to be independent, which is why the classifier is called 'Naïve'.



# Naïve Bayes Classifier

$$P(C_k|\mathbf{x}) = \frac{P(C_k) \times P(\mathbf{x}|C_k)}{P(\mathbf{x})}$$

# Naïve Bayes Classifier

$$P(C_k|\mathbf{x}) \propto P(C_k) \times P(\mathbf{x}|C_k)$$

# Naïve Bayes Classifier

$$\begin{aligned} p(C_k \mid x_1, \dots, x_n) &\propto p(C_k, x_1, \dots, x_n) \\ &\propto p(C_k) p(x_1 \mid C_k) p(x_2 \mid C_k) p(x_3 \mid C_k) \cdots \\ &\propto p(C_k) \prod_{i=1}^n p(x_i \mid C_k), \end{aligned}$$

# Decision Rule

$$\hat{y} = \operatorname{argmax} p(C_k) \prod_{i=1}^n p(x_i|C_k)$$

# Naïve Bayes Classifier

- Implemented in many stats/ML packages

# Support Vector Machine

- Comes from computer science
  - Very good
  - Rather difficult math
- 
- Considered one of the best of-the-shelf classification algorithms

# Hyperplane

- $n-1$  dimensional plane that separates the  $n$ -dimensional space

# Hyperplane

- n-1 dimensional plane that separates the n-dimensional space
- 2-dimensional hyperplane:

$$\beta_0 + \beta_1 X_1 = 0$$



# Hyperplane

- n-1 dimensional plane that separates the n-dimensional space
- 2-dimensional hyperplane:
- line equation

$$\beta_0 + \beta_1 X_1 = 0$$

# Hyperplane

- n-1 dimensional plane that separates the n-dimensional space
- 2-dimensional hyperplane:
- line equation

$$\beta_0 + \beta_1 X_1 = 0$$

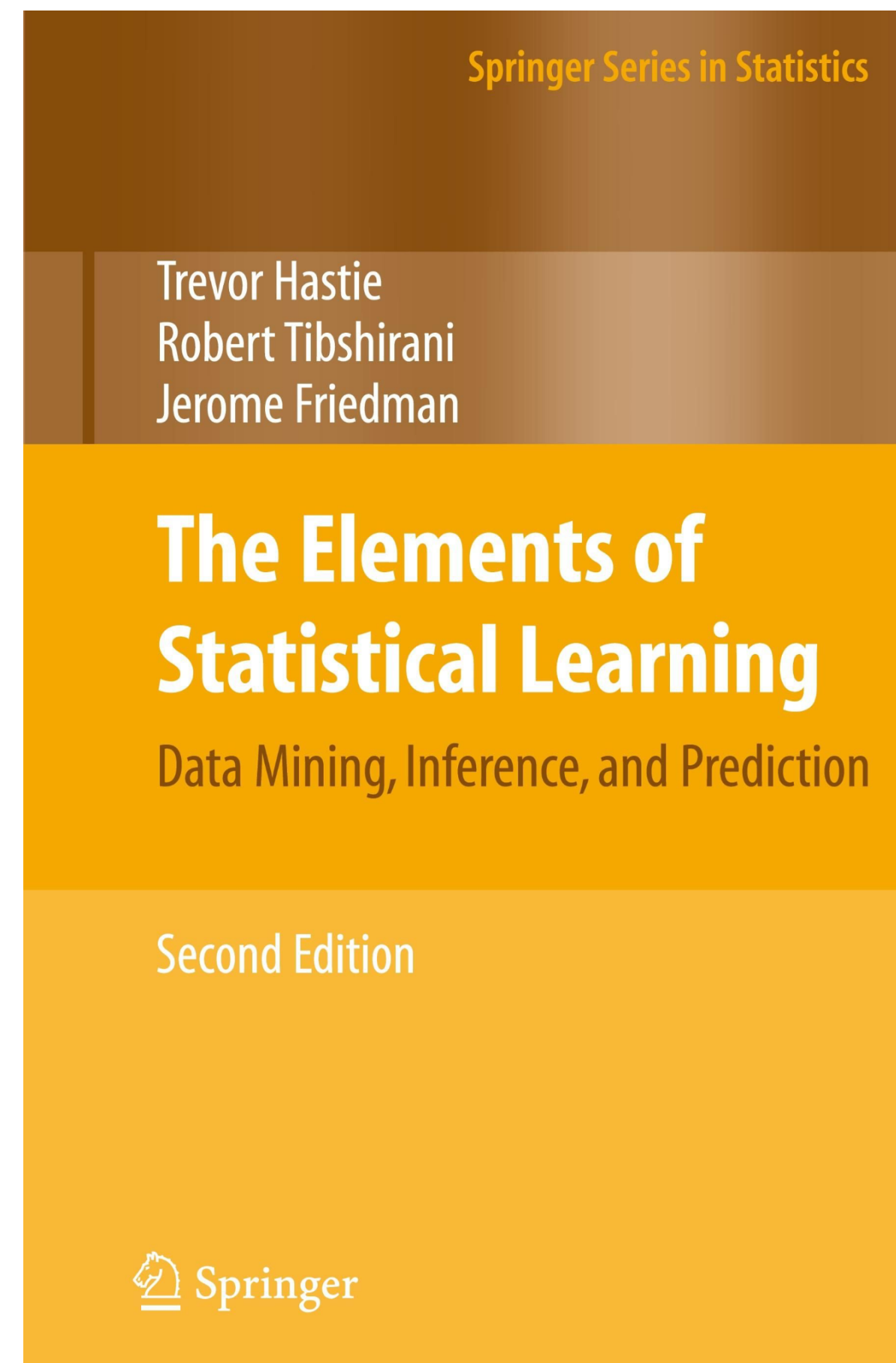
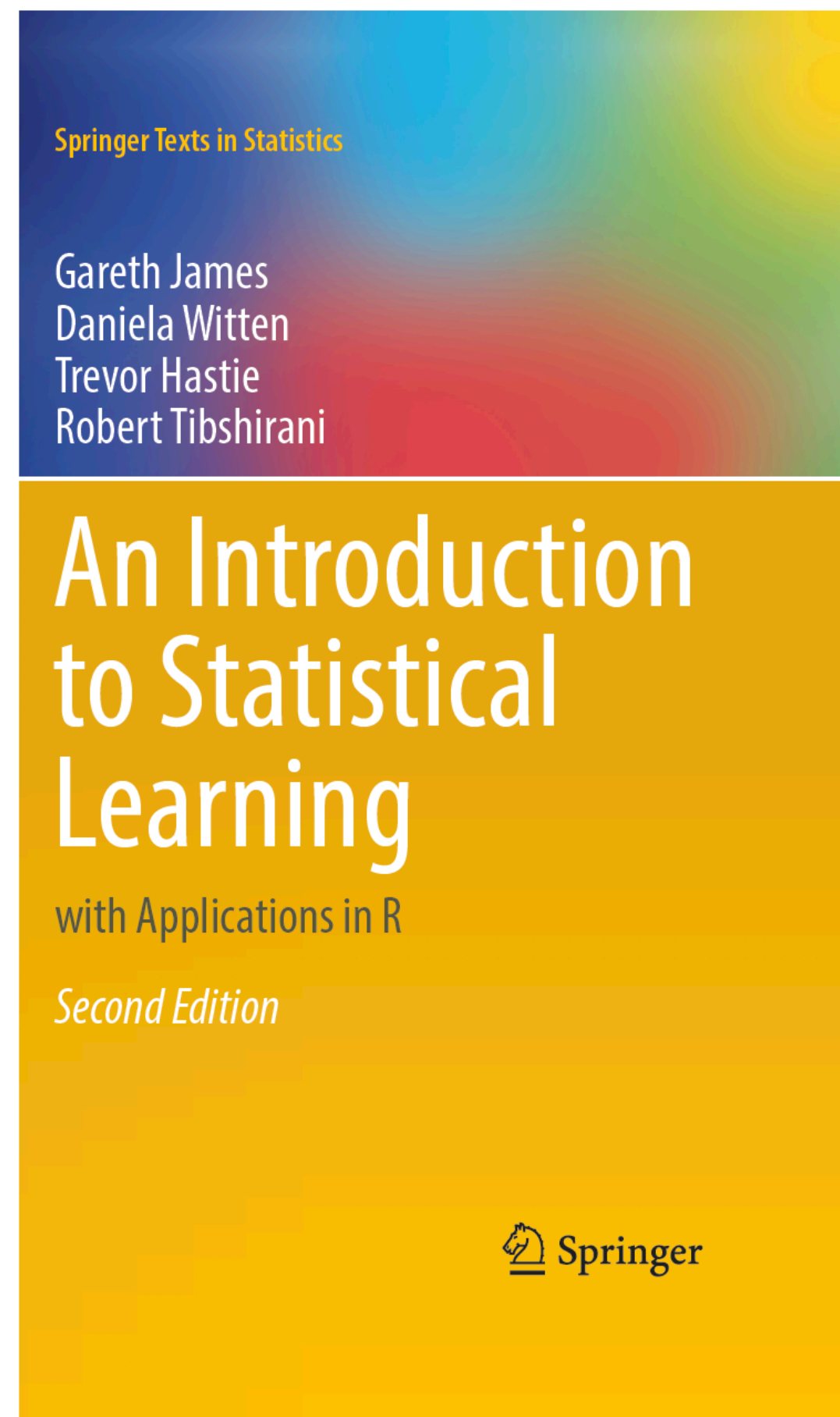
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = 0$$

# Classification

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p > 0.$$

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p < 0,$$

# Following images from:



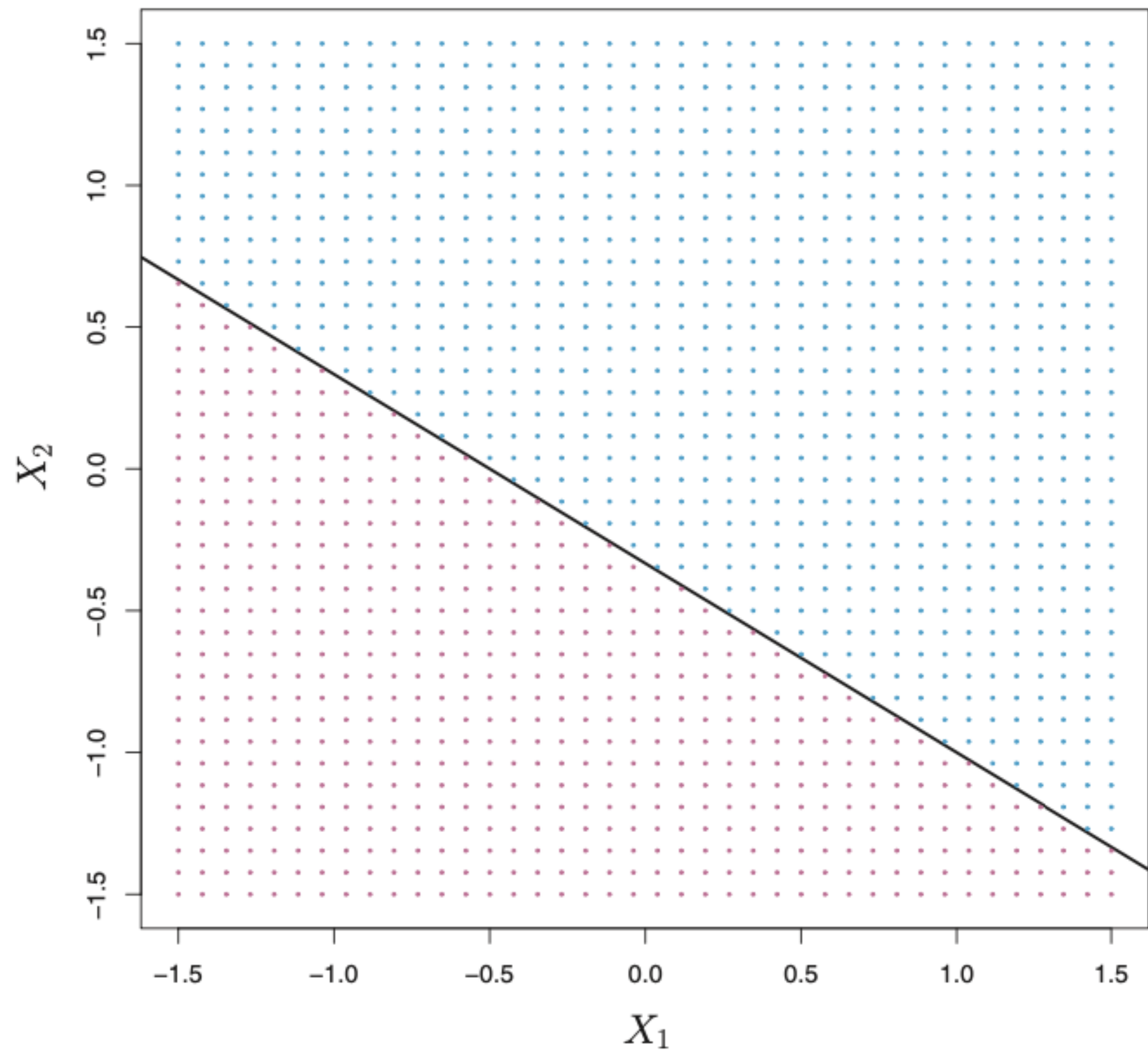
Springer Texts in Statistics

Gareth James · Daniela Witten · Trevor Hastie ·  
Robert Tibshirani · Jonathan Taylor

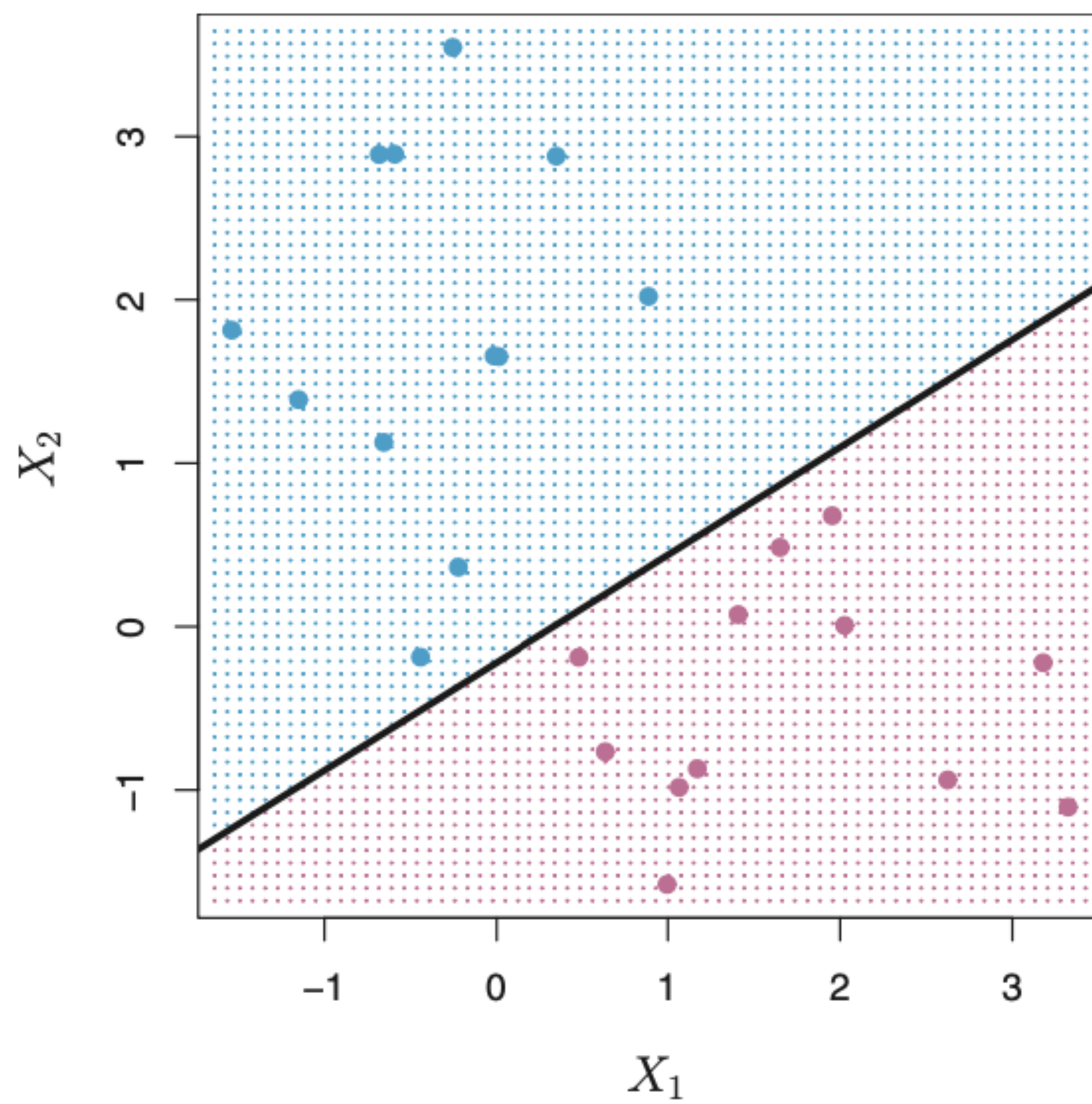
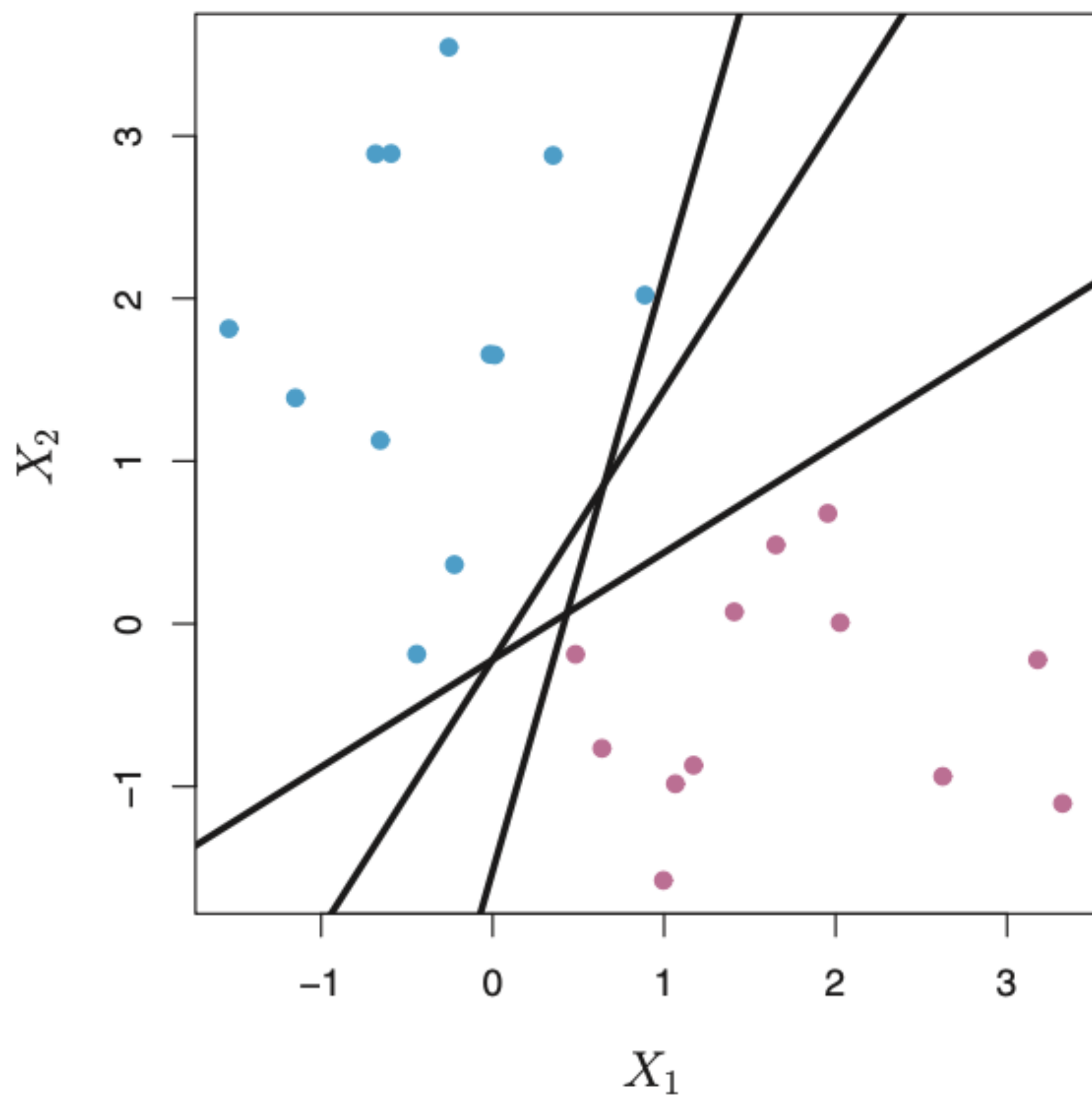
# An Introduction to Statistical Learning

with Applications in Python

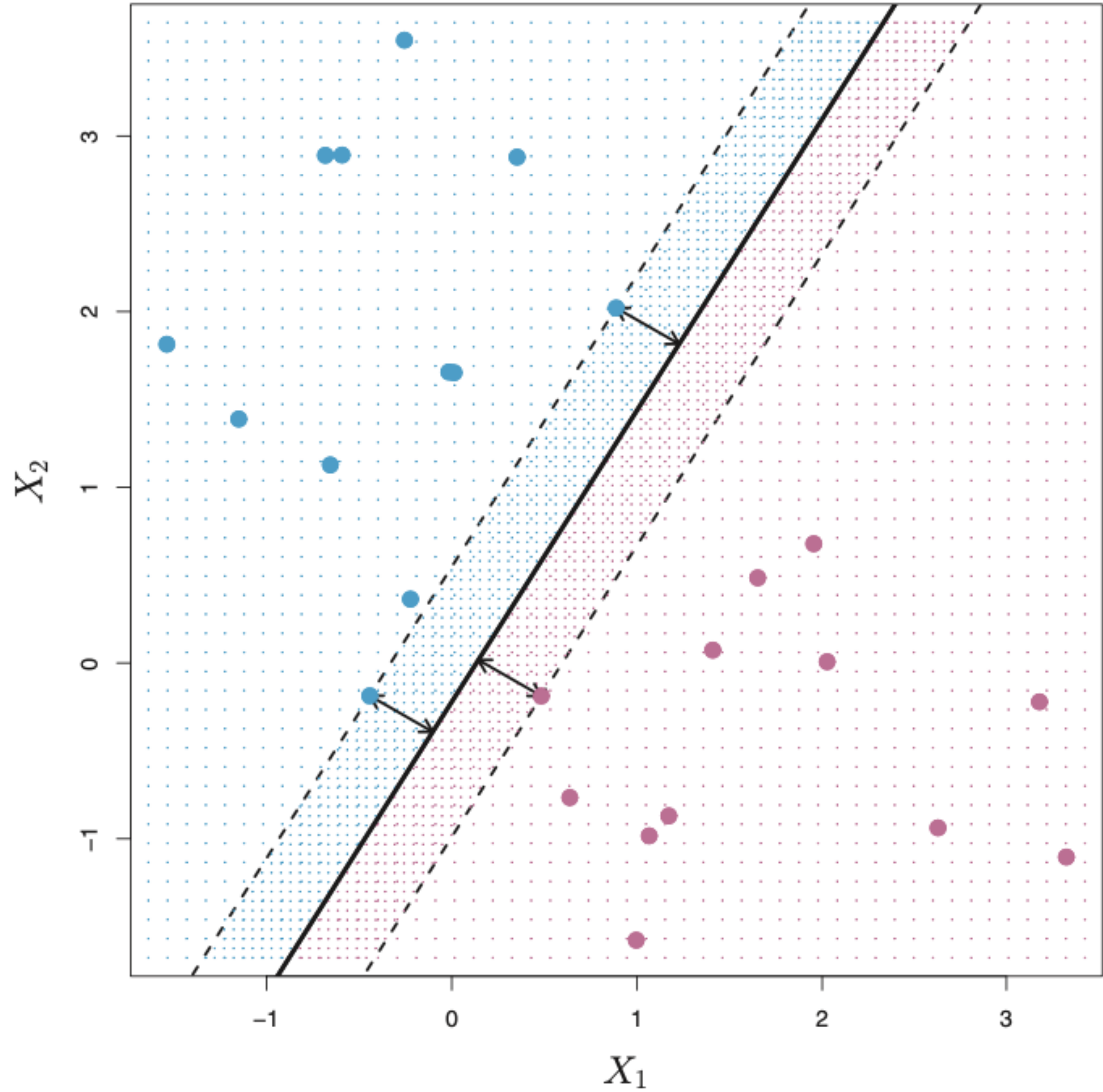
 Springer







# SV Classifier





# Support Vector Machine

- Non-linear version of the Support Vector Classifier
- Extension using Kernels

# Support Vector Machines

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_i \rangle$$

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i)$$

# Support Vector Machines

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_i \rangle$$

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i)$$

Kernel function

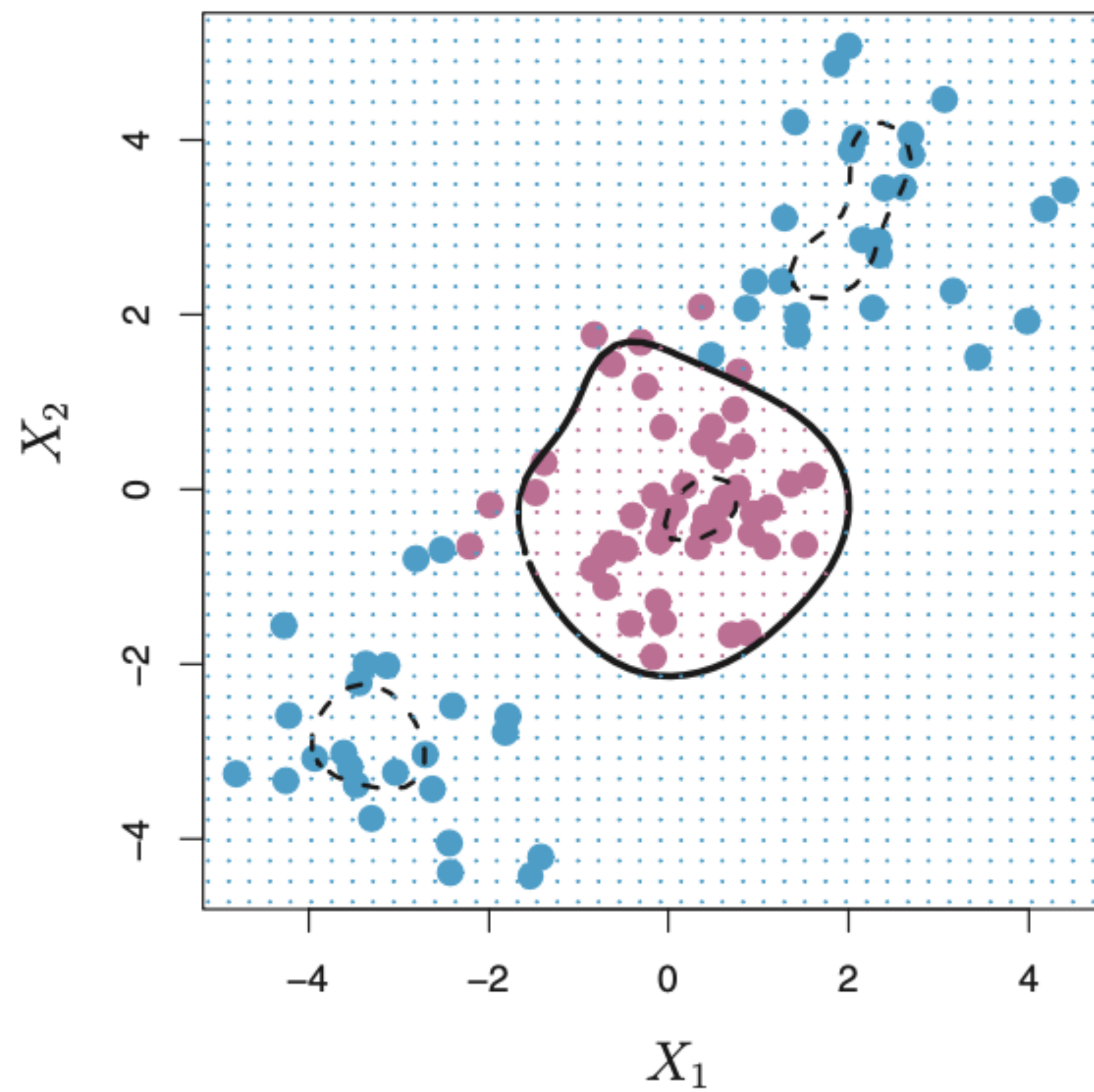
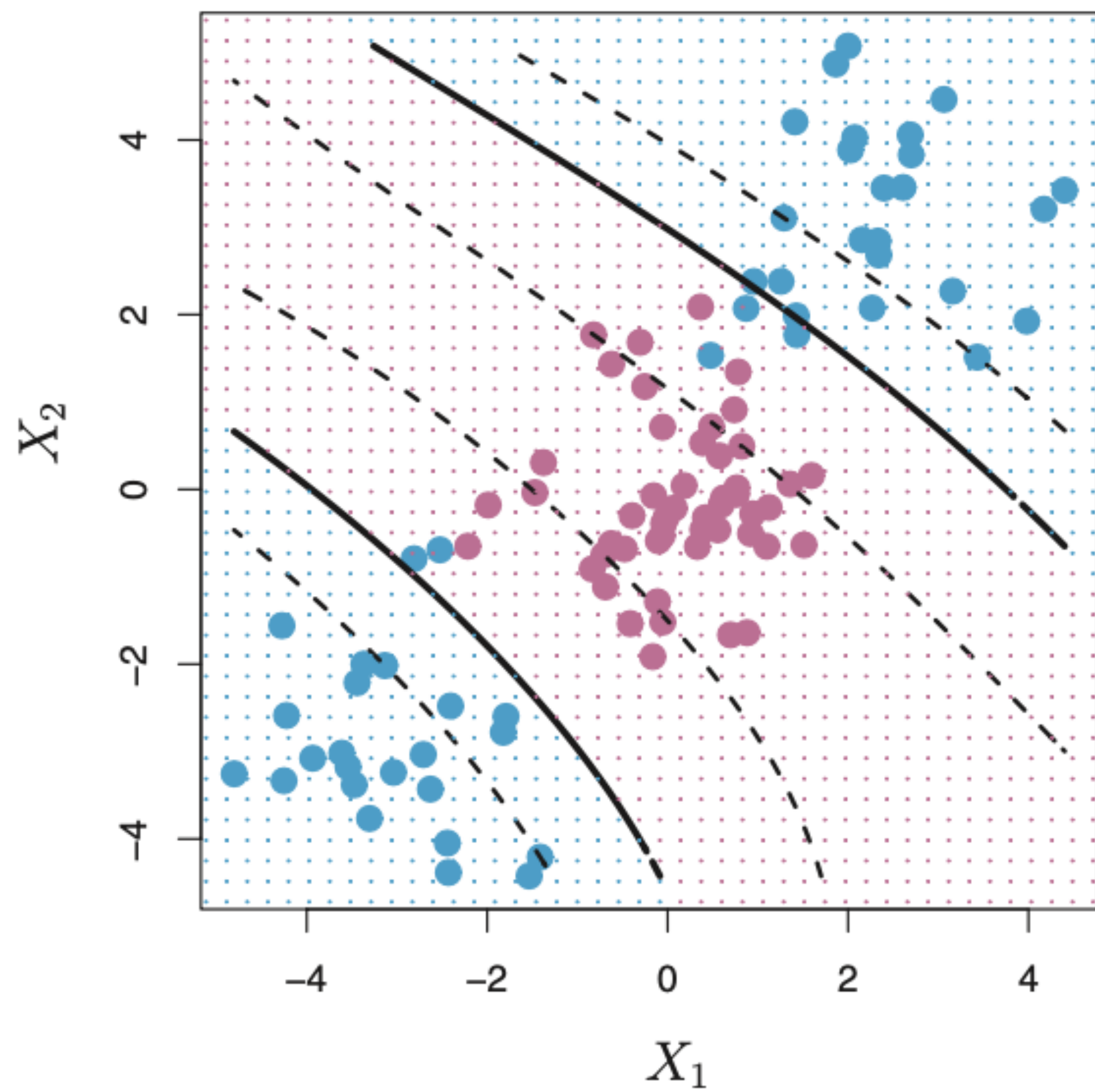


# Support Vector Machines

$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^p x_{ij} x_{i'j}\right)^d.$$

Polynomial Kernel





# Kernel Trick

# Kernel Trick

- Actual name
- Attempt to place  $n$ -dimensional data into  $n+1$  dimensional space



-5.0 -2.5 0.0 2.5 5.0  
 $x_1$





-5.0      -2.5      0.0      2.5      5.0  
x1

