Quantitative Text Analysis

Meeting 6

Petro Tolochko

Machine Learning

- Supervised
    - An outcome variable is defined
    - Focus is on prediction
- Unsupervised
    - No outcome variable has been defined
    - Focus is on patterns

Machine Learning

- **Supervised**
    - **An outcome variable is defined**
    - **Focus is on prediction**
- Unsupervised
    - No outcome variable has been defined
    - Focus is on patterns

Supervised

- Objective:
    - Classification of documents into pre existing categories

Supervised

- Create a labeled data set
- Classify documents with supervised learning algorithm
- Check performance

Labeled Dataset

- How:
  - Human coders annotate parts of the corpus (what we did together)
  - Found data (e.g., self-reported profession in users' profile)

- Considerations:
  - Sampling should be representative for the corpus (e.g., Random, Stratified sample e.g., across time and source)
  - Quality of human coding matters (Assess the intercoder reliability)
  - Number of documents

Labeled Dataset

- Number of documents
    - the higher the number of categories and the lower the reliability of the coders, the higher the number of documents (Barberá et al., 2021)
- increase the sizes of manually coded validation dataset as large as possible (e.g., more than 1% of all data to be examined), assuming acceptable reliability (equal to or higher than .7) (Song et al., 2021)

Splitting the Data

- Split labeled data in training data and test data (validation data)
- Training data
    - The subset that is used to learn the model parameters
- Test data
    - Another subset used to evaluate the model’s predictive quality
    - Not used for learning!
- Validation data

Document Classification

- Classifier learns the mapping between features and the labels in the training set
- define a model Y=g(X)
- And apply a learning algorithm to establish which features in X (features extracted from the training documents) matter to recover Y (i.e, the labels of the training documents)
- We fit the model

Machine Learning 101

Machine Learning 101

- Model: $$\boldsymbol{Y} = f(\boldsymbol{X})$$

- Objective function (e.g.,): $$MSE = \frac{1}{n}\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

- Optimisation: $$argmin_{\hat{Y}}\frac{1}{n}\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Machine Learning 101

- Model: $$\boxed{\boldsymbol{Y} = f(\boldsymbol{X})}$$

- Objective function (e.g.,): $$MSE = \frac{1}{n}\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

- Optimisation: $$argmin_{\hat{Y}}\frac{1}{n}\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Machine Learning 101

- Model: $$\boxed{Y = f(X)}$$ **Machine**

- Objective function (e.g.,): $$MSE = \frac{1}{n}\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

- Optimisation: $$argmin_{\hat{Y}}\frac{1}{n}\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Learning

Classify documents with supervised learning

- Considerations:
  - Feature representation (Bag of words representation or embeddings)
  - Feature selection (remove irrelevant features)
  - Classifier selection
    - E.g., Naive Bayes, SVM, KNN, or ensemble methods

Checking Performance

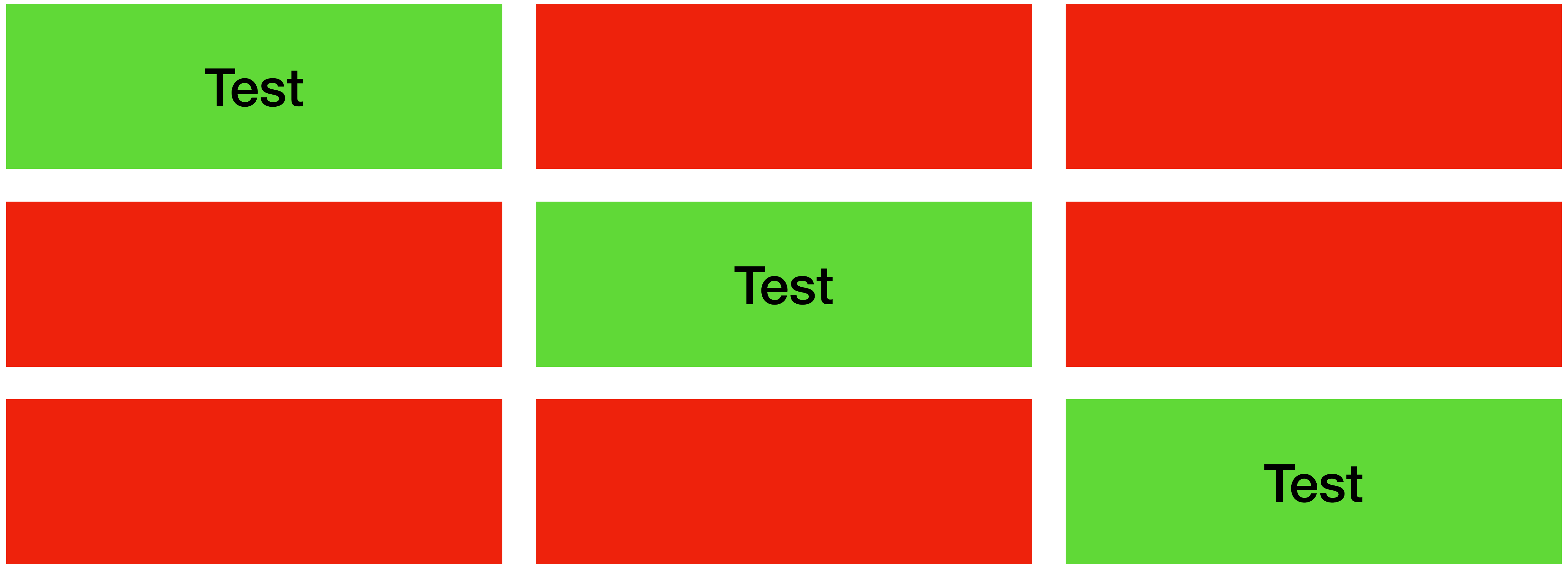
- The fitted model (the trained classifier) is applied to a held-out test set (which is a part of the labeled set but was not used for training the model).

- Considerations:
    - Danger of overfitting (focus on features that work well with training set but do not generalize)
    - Solutions: cross-validation
    - Performance metric (i.e., recall, precision)

Checking performance

- k-fold cross-validation
    - We randomly split the data into k sets (“folds”) of roughly equal size
    - Each set is hold out once as test set, while training on the remaining sets
    - The problem of a lucky split is reduced

K-Fold Cross-Validation



Confusion Matrix

	Actual label	
	Negative	Positive
Negative	True negative	False positive
Positive	False negative	True positive

Precision/Recall

$$Accuracy = \frac{True\ Negative + True\ Positive}{True\ Negative + True\ Positive + False\ Negative + True\ Positive}$$

$$Precision_{positive} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$Recall_{positive} = \frac{True\ Positive}{True\ Positive + False\ Negatives}$$

Dictionary vs. supervised machine learning

- Dictionaries can be applied directly to a new corpus (but validate!)

- Supervised machine learning requires (potentially larger amounts) labeled data

- If the training sample is large enough supervised learning will outperform dictionaries

Additional considerations

- Hyperparameter selection
    - Via systematic comparison of different hyperparameters per algorithm
- Random undersampling (Galar et al., 2011)
- Method to deal with unbalanced classes: use the max. number of positive instances per class and randomly sample the same number of instances of the negative class

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

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

Posterior probability of event A given the evidence B

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

Probability of observing the evidence B

Bayes Theorem

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

Naïve Bayes Classifier

$$P(C_k|x_1, x_2, ..., x_n)$$

Naïve Bayes Classifier

Features

$$P(C_k|x_1, x_2, ..., x_n)$$

Class

Naïve Bayes Classifier

Features

$$P(C_k|x_1, x_2, ..., x_n)$$

Class

Features are assumed to be independent. Hence, "***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

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

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

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

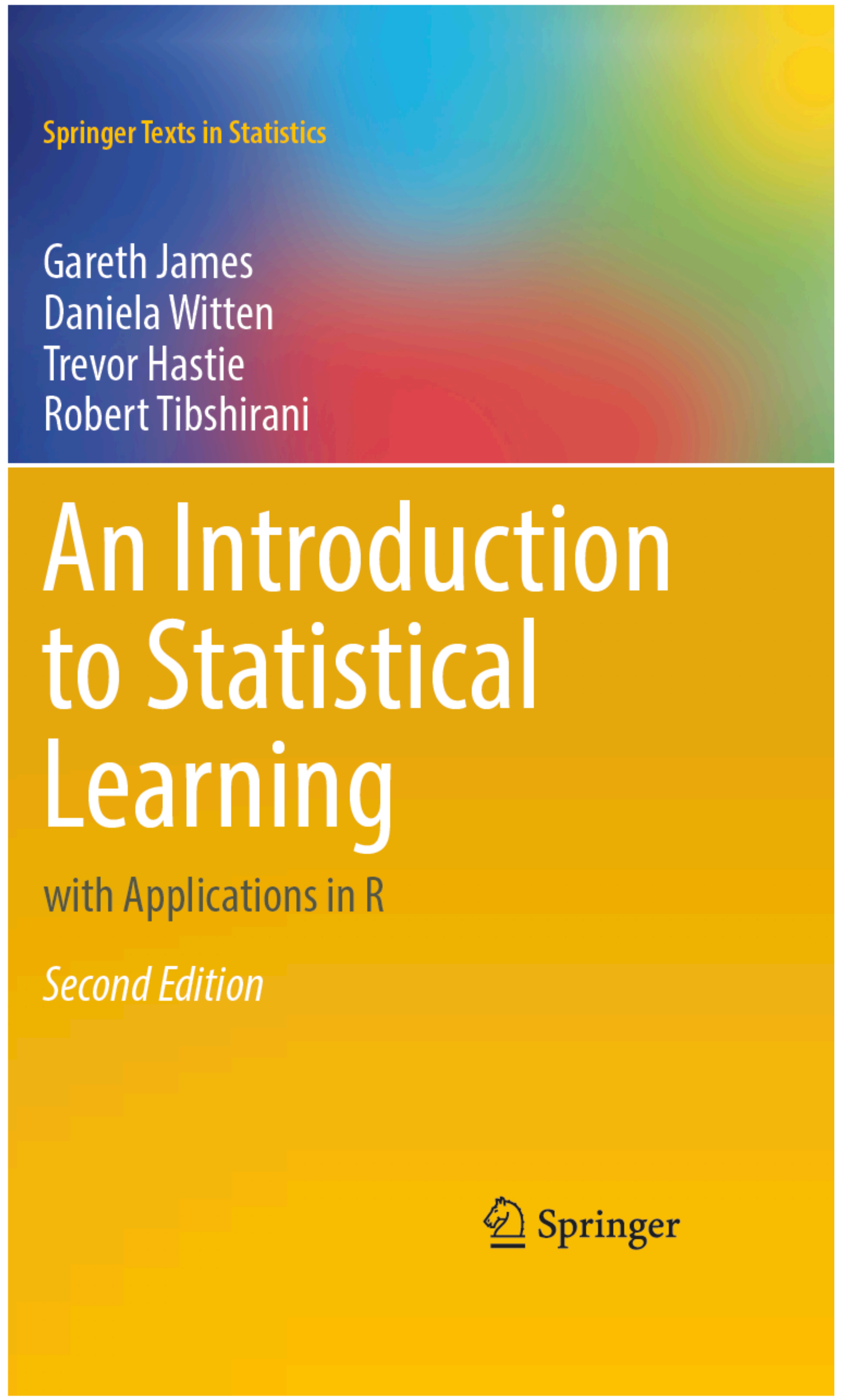
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

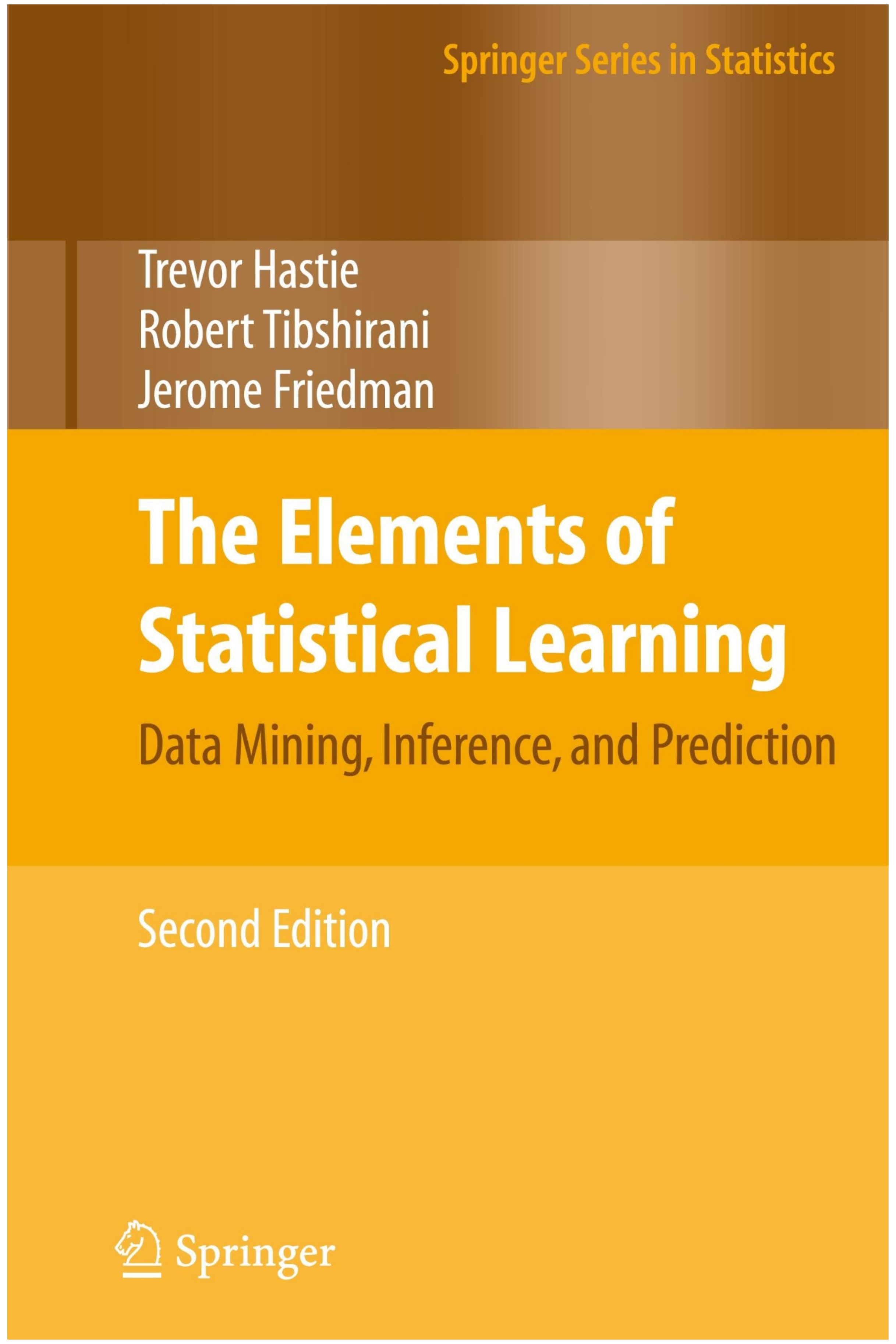
Classification

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

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

Following images from:





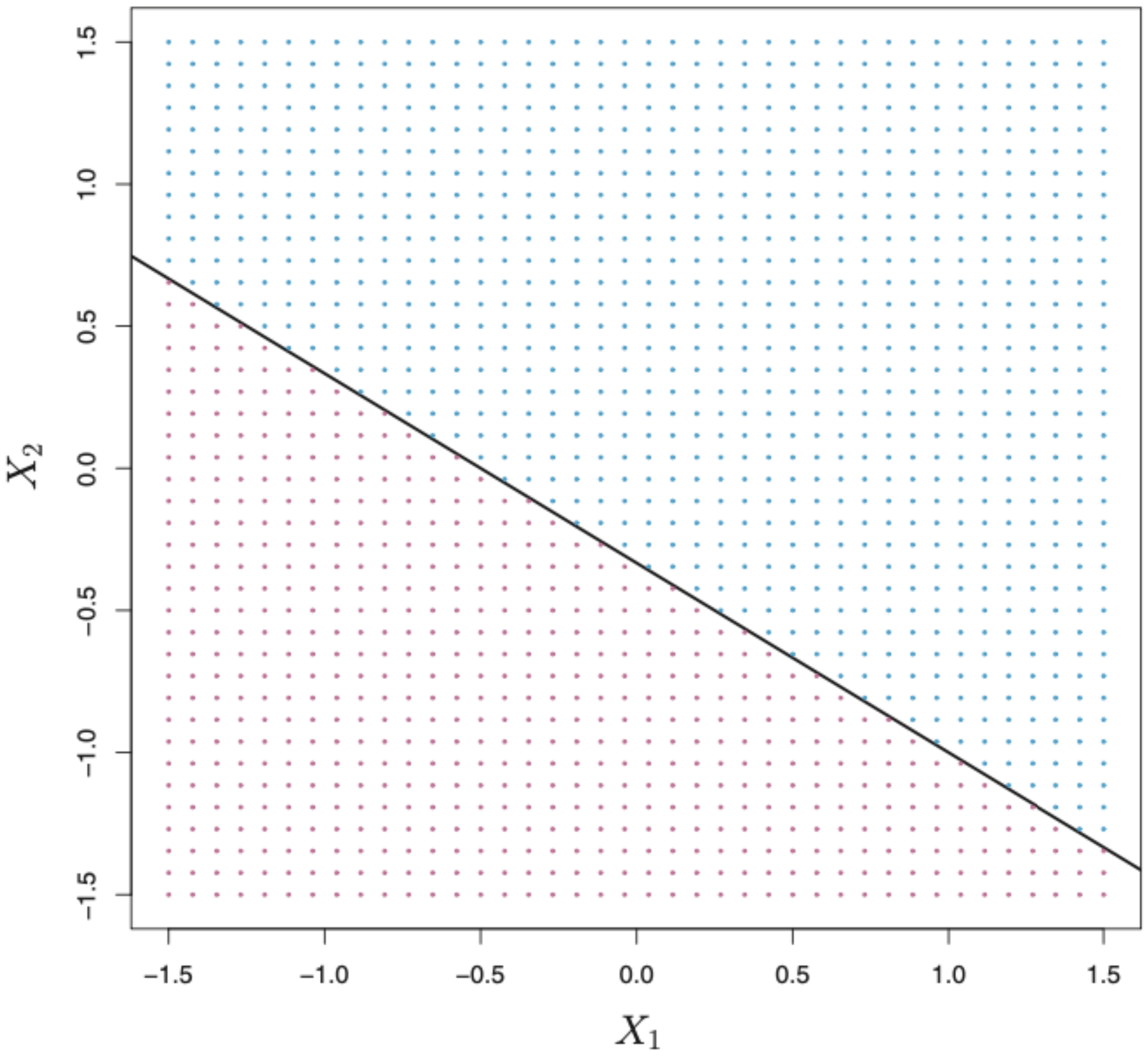
Springer Texts in Statistics

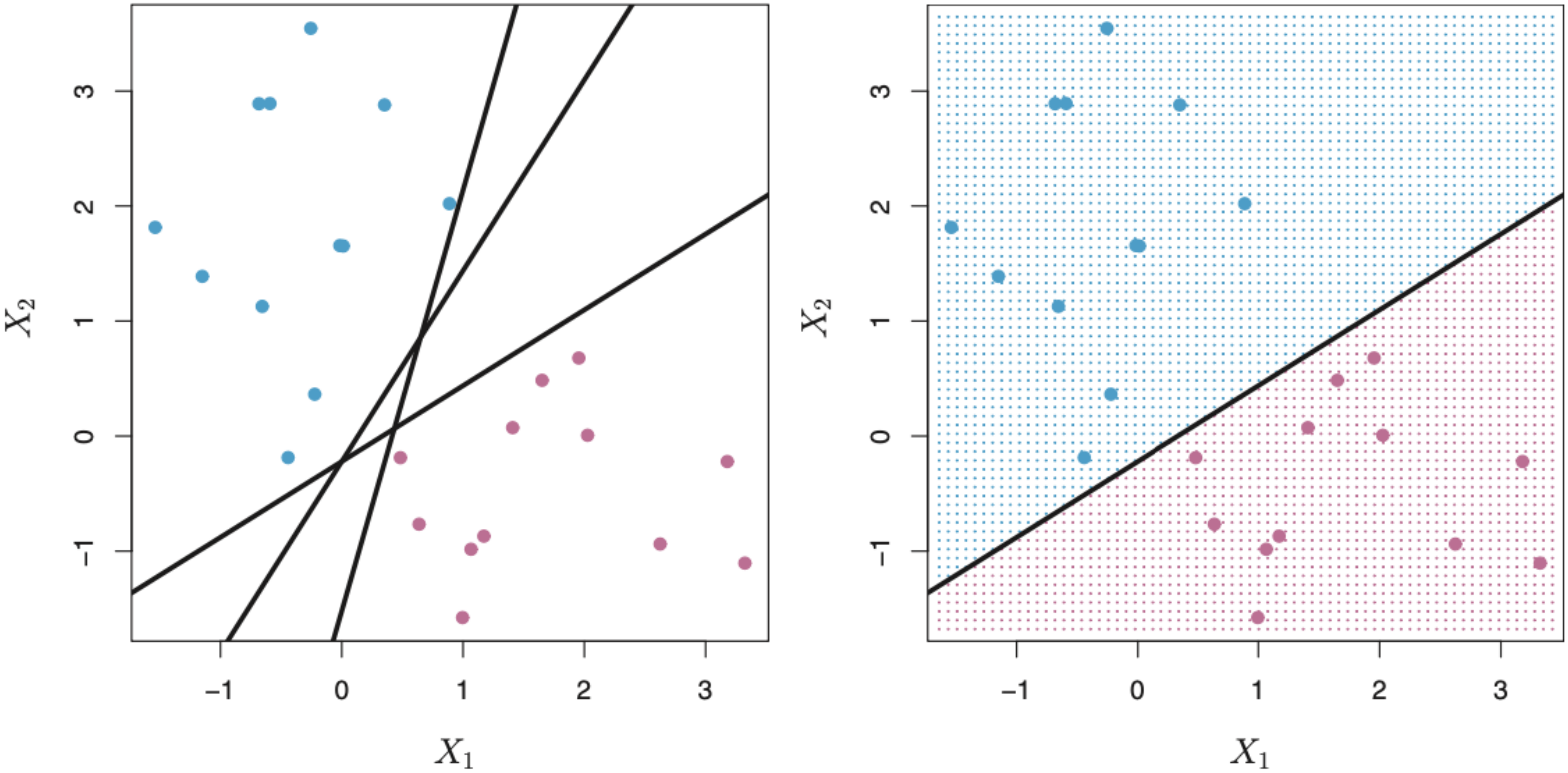
Gareth James · Daniela Witten · Trevor Hastie · Robert Tibshirani · Jonathan Taylor

An Introduction to Statistical Learning

with Applications in Python

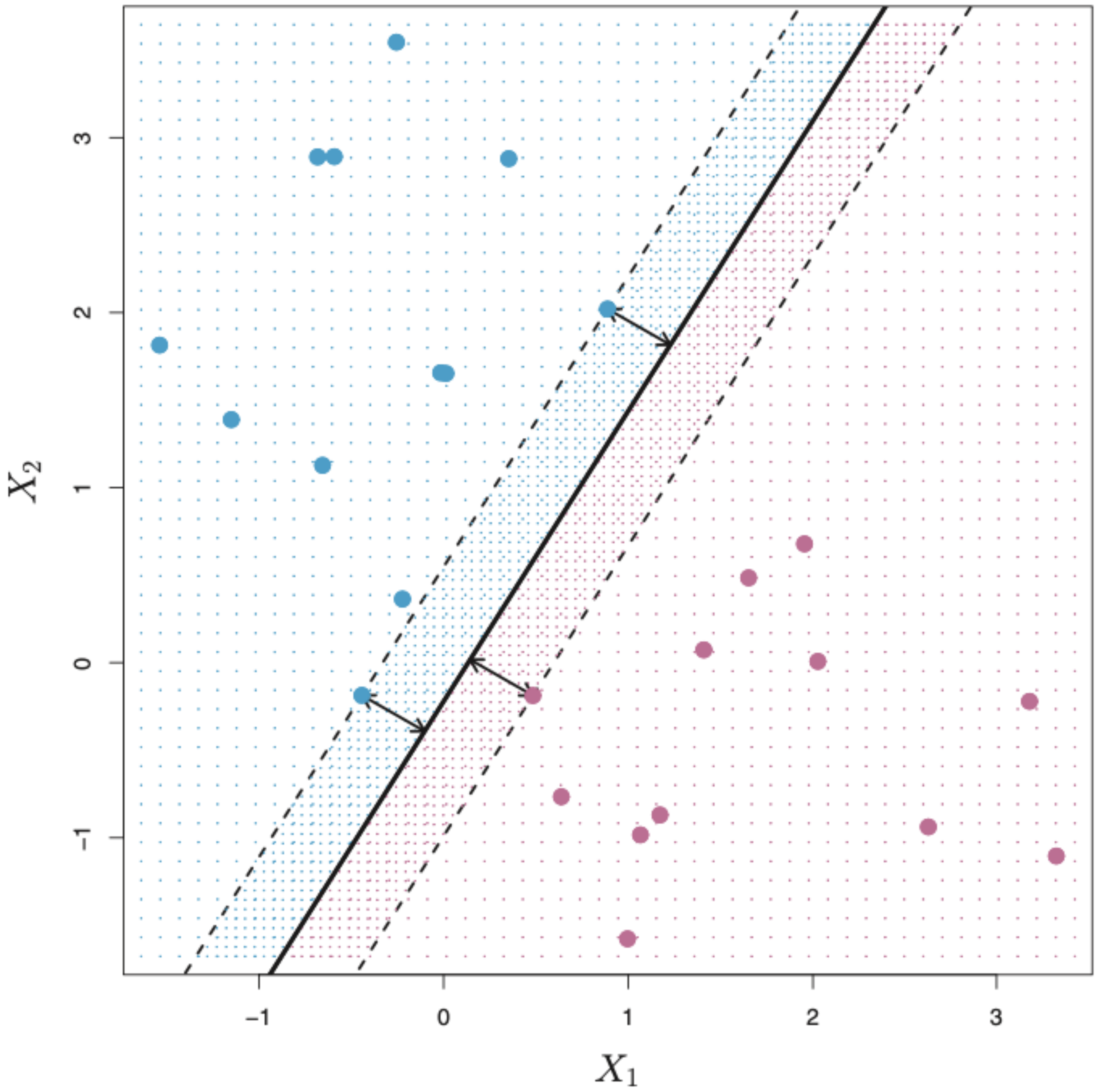
Springer


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
X_2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
X_1



X_2
−1
0
1
2
3
X_1
X_2
−1
0
1
2
3
X_1

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

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

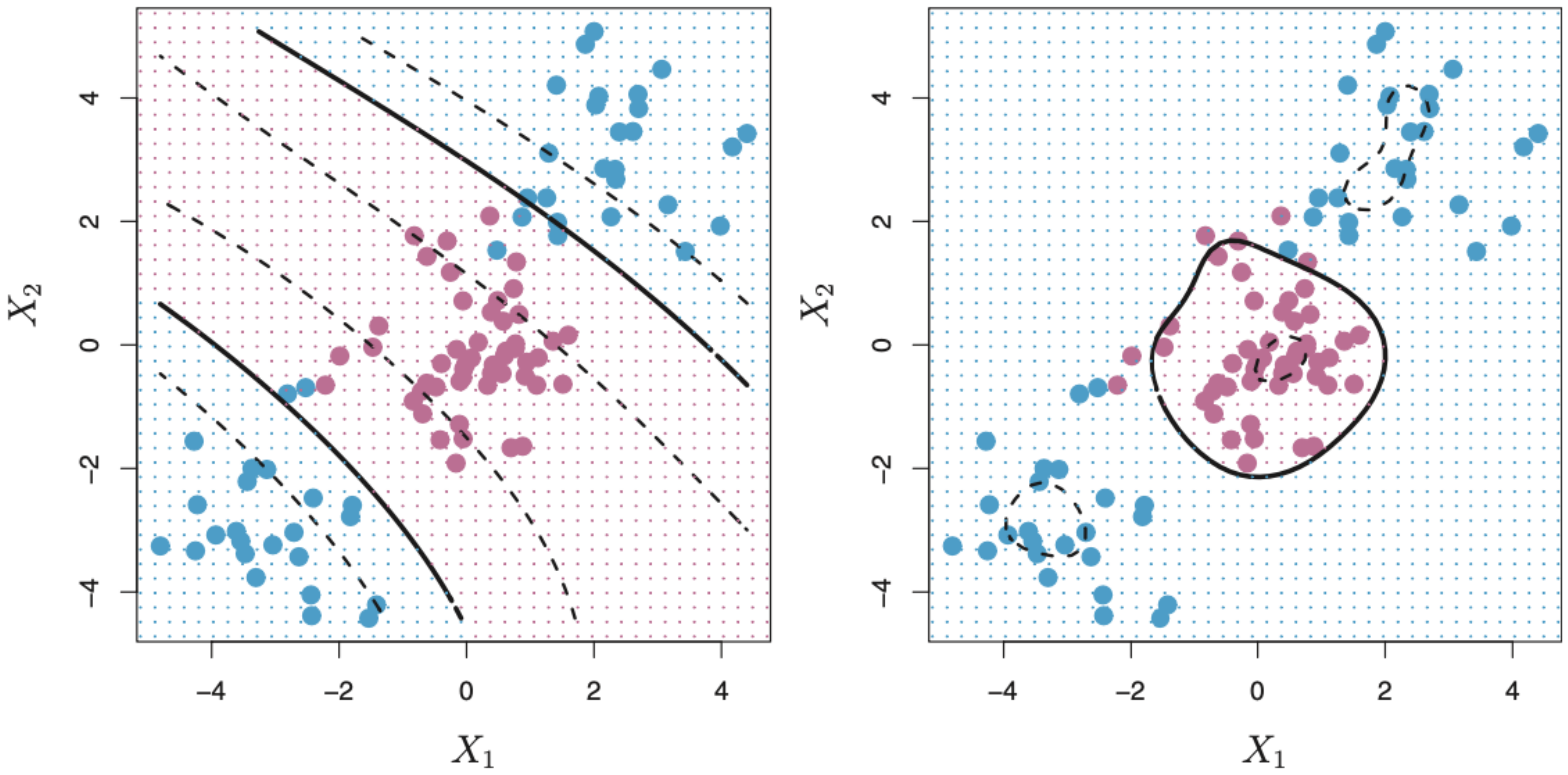
$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i)$$

Kernel function

Support Vector Machines

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

Polynomial Kernel

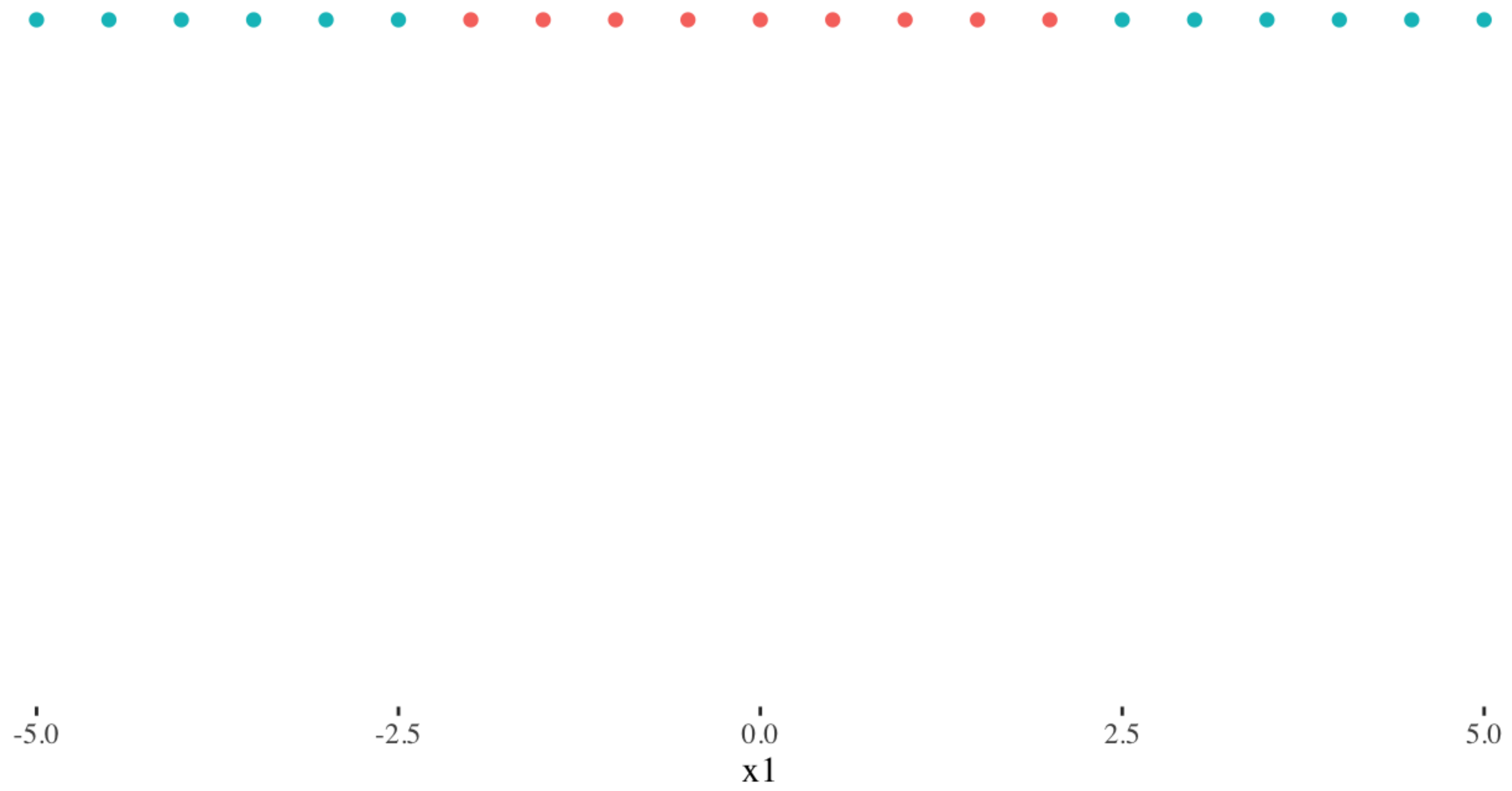


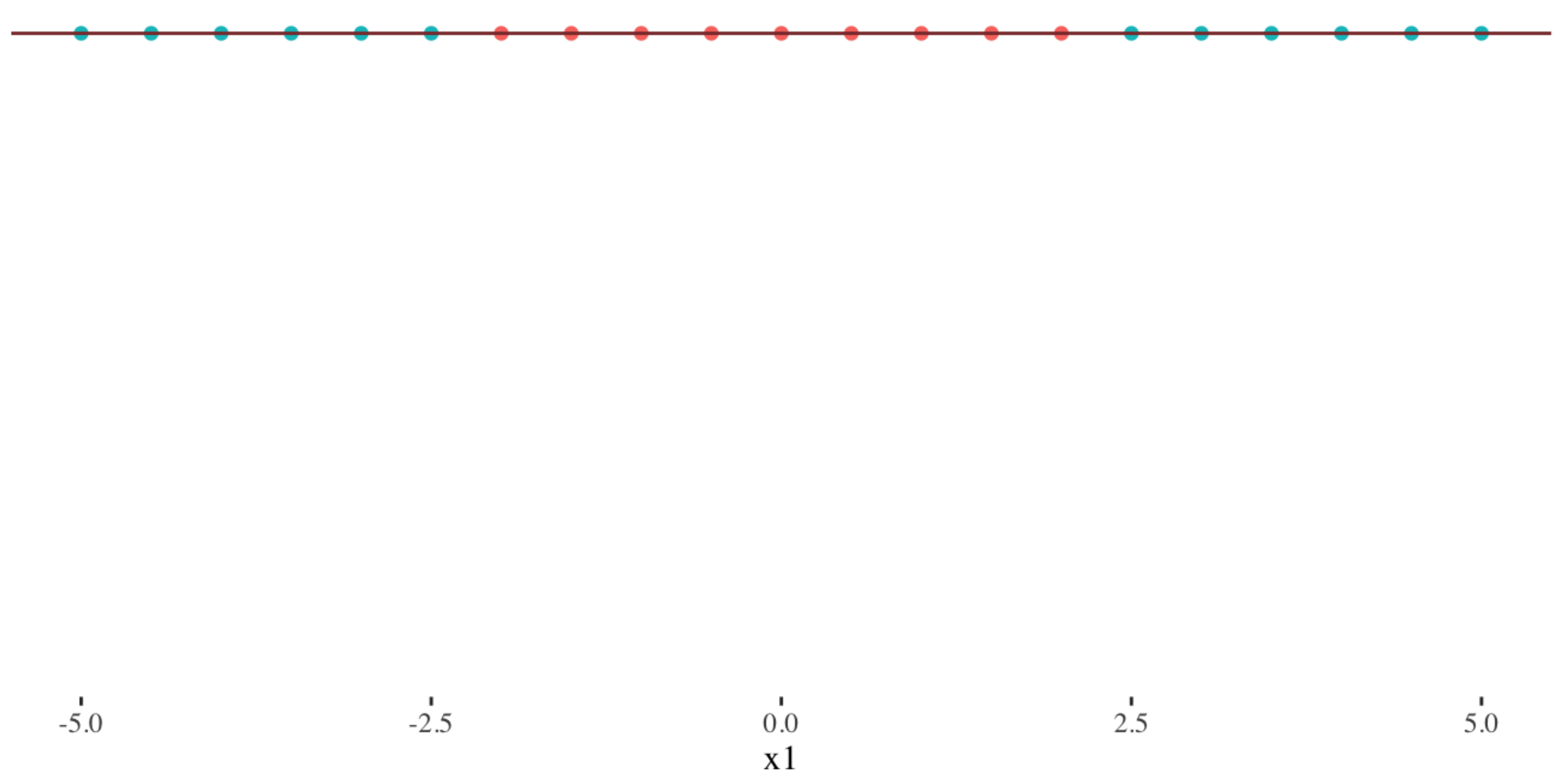
X_2
4
2
0
−2
−4
−4
−2
0
2
4
X_1
X_2
4
2
0
−2
−4
−4
−2
0
2
4
X_1

Kernel Trick

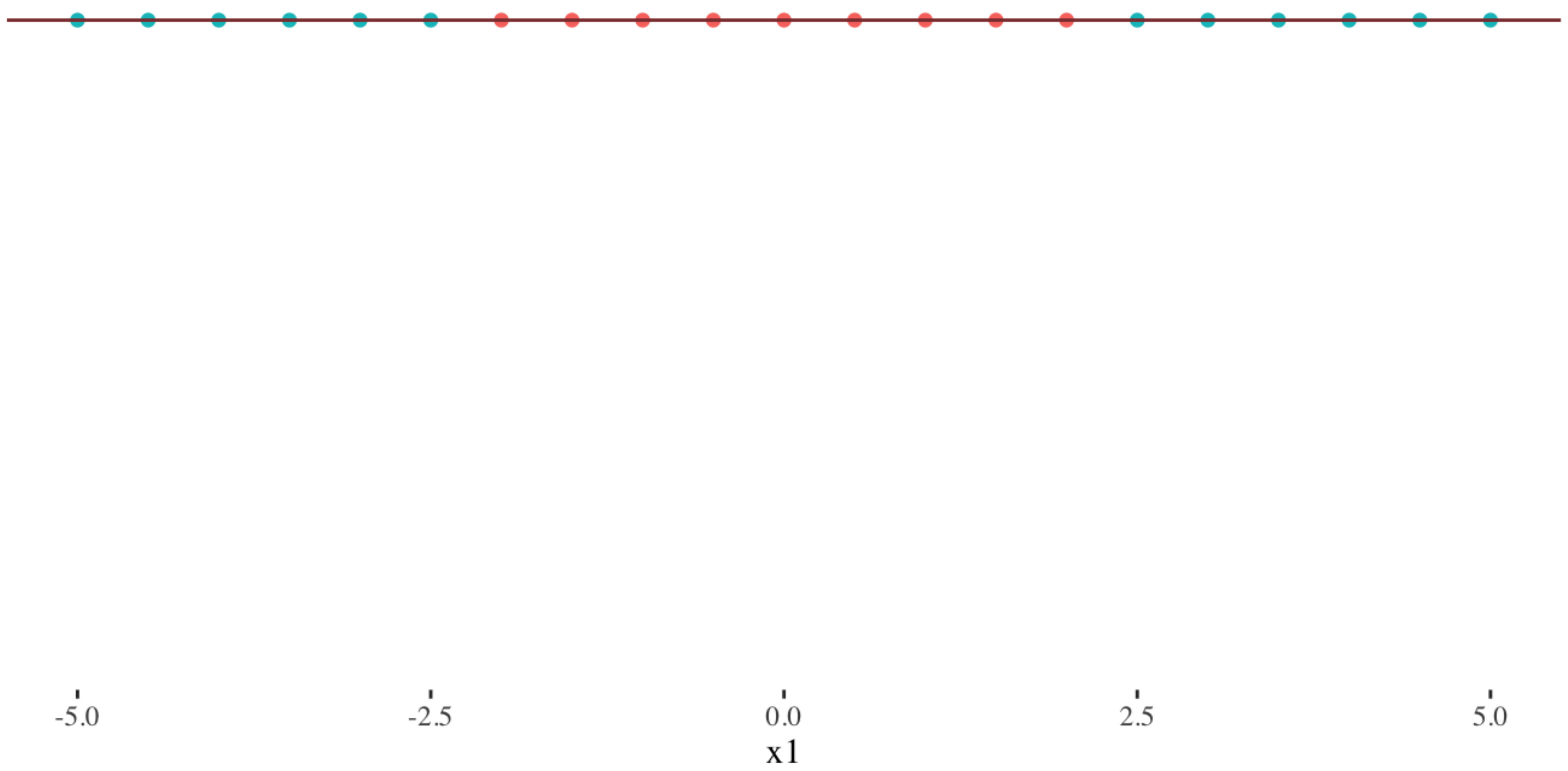
Kernel Trick

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

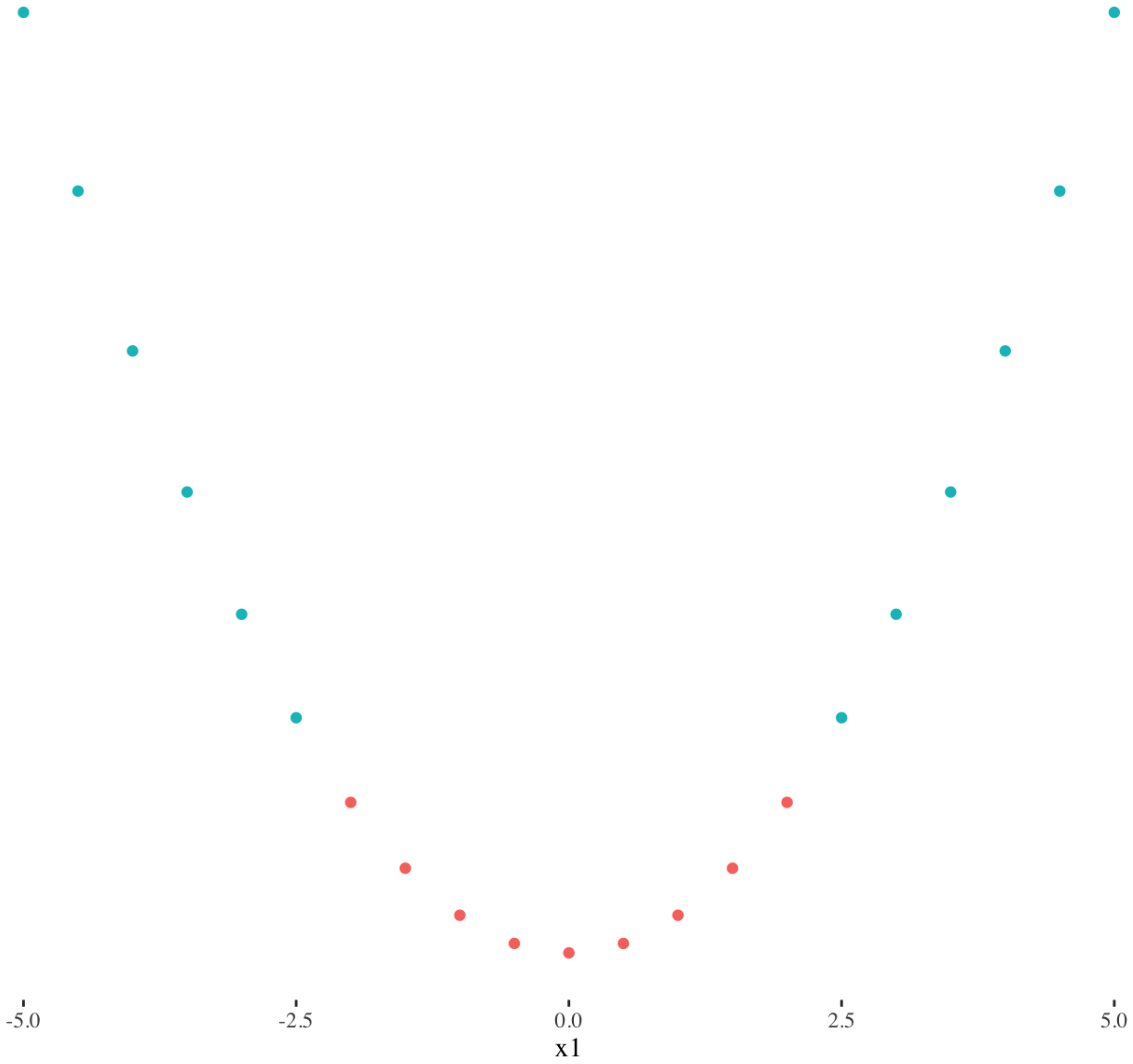

-5.0
-2.5
0.0
2.5
5.0
x1



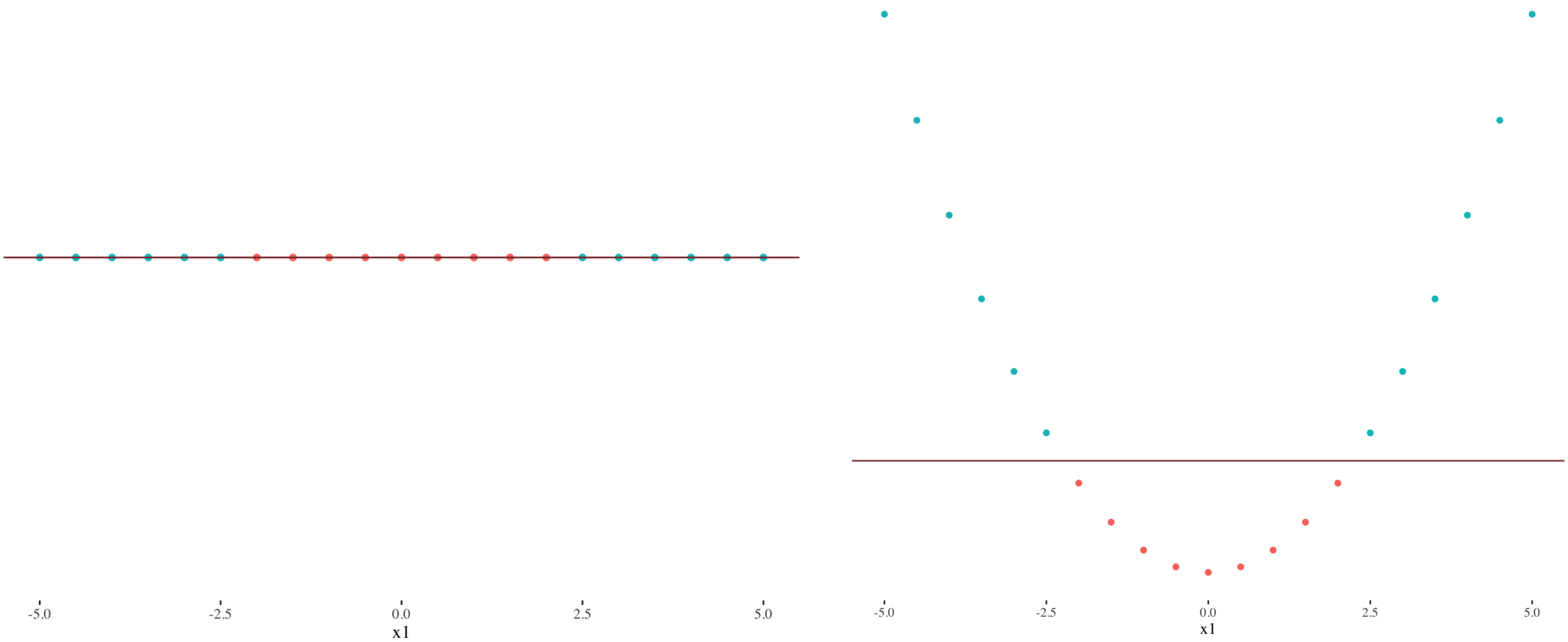
-5.0
-2.5
0.0
2.5
5.0
x1



-5.0
-2.5
0.0
2.5
5.0
x1



-5.0
-2.5
0.0
2.5
5.0
x1


-5.0
-2.5
0.0
2.5
5.0
x1
-5.0
-2.5
0.0
2.5
5.0
x1