Data Management

Supervised Machine Learning: Classifications

Malka Guillot

HEC Liège | ECON2306

Reference:

- JWHT: chap 2.2.3, 4

Classification Framework

- Response/target variable y is **qualitative** (or **categorical**):
  - 2 categories $\rightarrow$ binary classification
  - More than 2 categories $\rightarrow$ multi-class classification
- Features X:
  - can be high-dimensional
- We want to assign a class to a **quantitative response**

  $\rightarrow$ probability to belong to the class
- **Classifier**: An algorithm that maps the input data to a specific category.
- Performance measures specific to classification

Application examples

- In business:
  - Loan default prediction
  - Type of costumer
- In public economics:
  - Tax evasion prediction
- In political sciences:
  - political affiliation of author of texts
- In medical sciences:
  - Diagnostic diseases, drug choice
- Other:
  - email filtering, speech recognition...

Why not fitting a linear regression?

- **Technically possible** to fit a linear model using a categorical response variable but it implies
  - an **ordering** on the outcome
  - a **scale** in the class difference

$\rightarrow$ If the response variable was coded differently, the results could be completely different

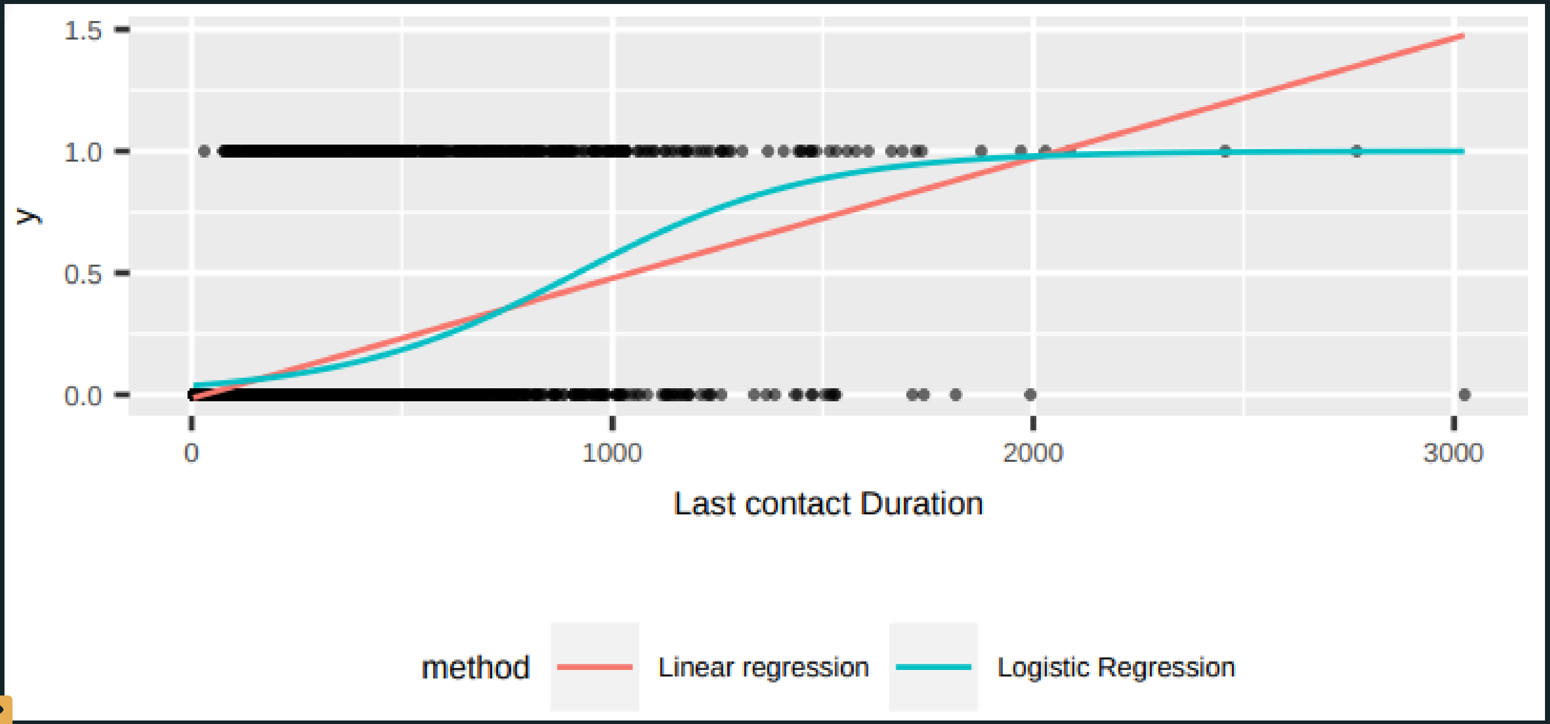
- Less problematic if the response variable is **binary**
  - The result of the model would be stable
  - But prediction may lie outside of $[0, 1]$: hard to interpret them in terms of probabilities

Example

- We predict y, the **occupation of individuals**:

$$y = \begin{cases} 0 & \text{if blue-collar} \\ 1 & \text{if white-collar} \end{cases}$$

- based on their characteristics X (gender, wage, contract duration, experience, age...)



Classification process

1. Model probability
   - Probability of belonging to a category

$$P(y = 1 \mid X)$$

2. Predict probability
   - rely on this probability to assign a class to the observation.
     - For example, we can assign the class 1 for all observations where $P(y = 1|x) > 0.5$
     - But we can also select a different **threshold**.
3. We can make errors
   - False negative
   - False positive

Binary Classifier

- Logistic Regressions
- Support Vector Machine

Logistic Regression

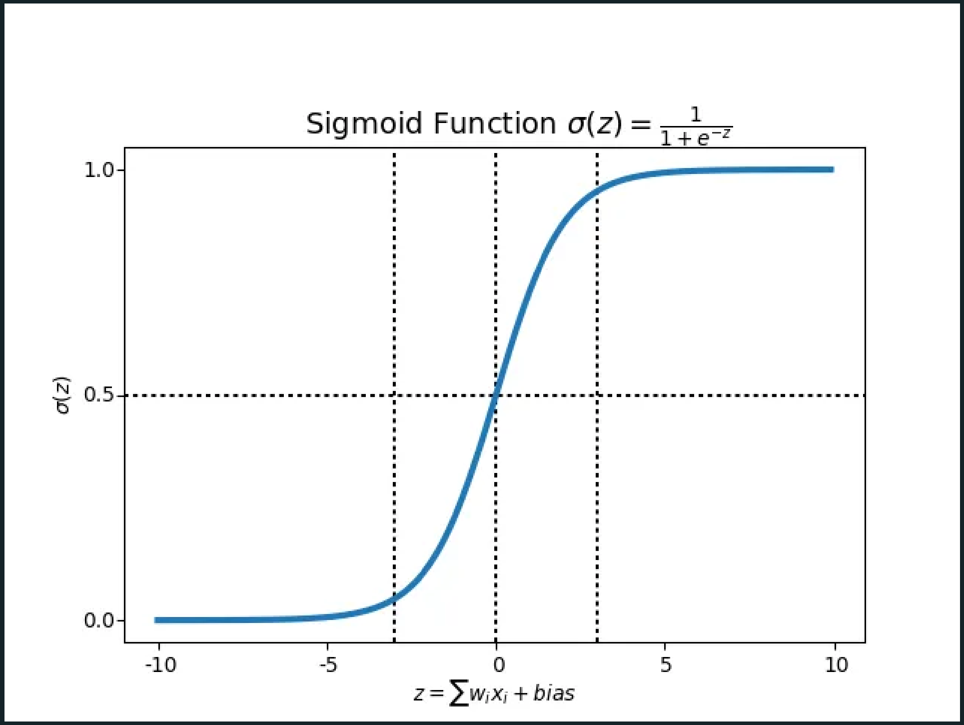
- Like OLS, logistic "regression" computes a weighted sum of the input features to predict the output.
  - But it transforms the sum using the **logistic function**.

$$\hat{p} = \Pr(Y_i = 1) = \sigma(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_p x_p)$$

where $\sigma(\cdot)$ is the sigmoid function

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

Logistic Regression



- Prediction:

$$\hat{y} = \begin{cases} 0 & \text{if } \hat{p} < .5 \\ 1 & \text{if } \hat{p} \geq .5 \end{cases}$$

Logistic Regression Cost Function

- The cost function to minimize is

$$J(\theta) = \underbrace{-\frac{1}{m}}_{\text{negative}} \sum_{i=1}^{m} [\underbrace{y_i}_{y_i=1} \underbrace{\log(\hat{p}_i)}_{\text{log prob} y_i=1} + \underbrace{(1-y_i)}_{y_i=0} \underbrace{\log(1-\hat{p}_i)}_{\text{log prob} y_i=0}]$$

  - this does not have a closed form solution
  - but it is convex, so gradient descent will find the global minimum.
- Just like linear models, logistic can be regulared with L1 or L2 penalties, e.g.:

$$J_2(\theta) = J(\theta) + \alpha_2 \frac{1}{2} \sum_{i=1}^{n} \theta_i^2$$

Support Vector Machine

- Context: developed in the mid-1990s
- A generalization of the early logistic regression (1930s)
- One of the best “out of the box” classifiers
- Core idea: hyperplane that separates the data as well as possible, while allowing some violations to this separation

Support Vector Machine: context and concepts

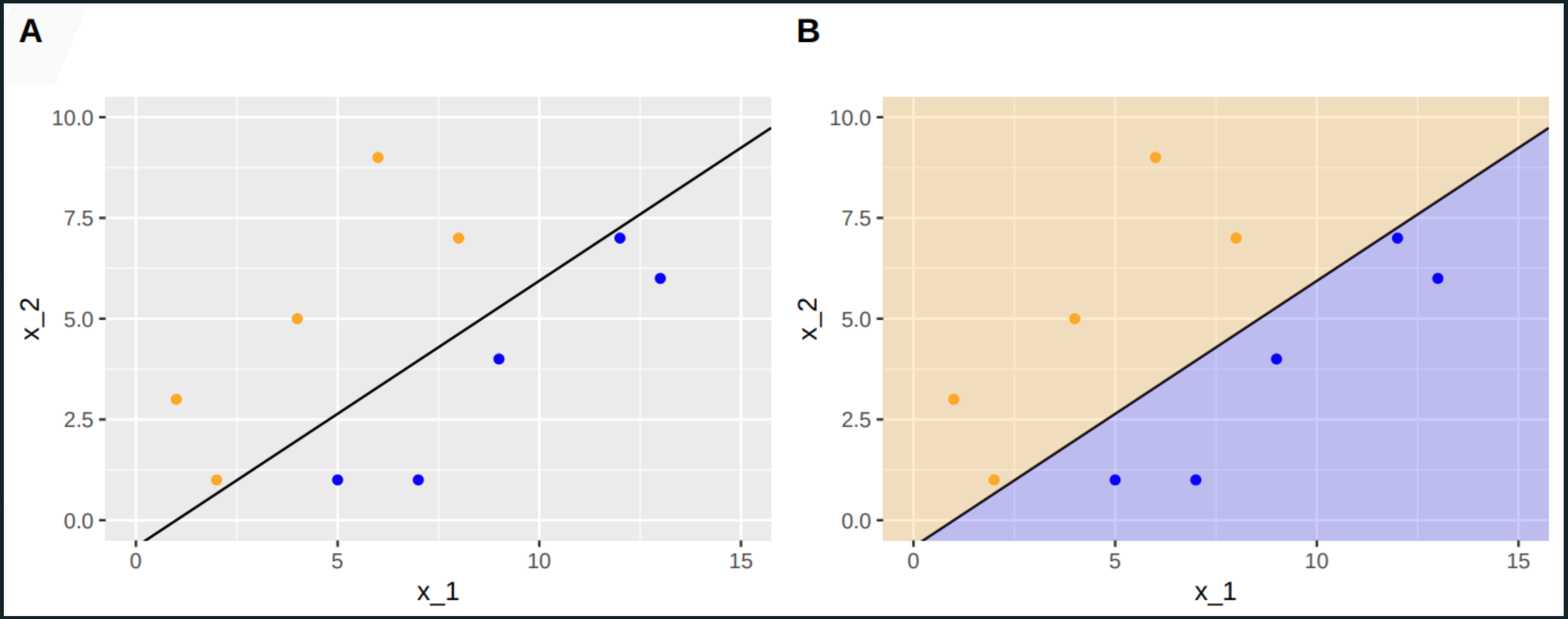
- Pieces of the puzzle:

1. A **maximal margin classifier**: requires that classes be separable by a linear boundary.
2. A **support vector classifier**: extension of the maximal margin classifier.
3. **Support vector machine**: further extension to accommodate non-linear class boundaries.

- For binary classification, can be extended to multiple classes

Classification and Hyperplane

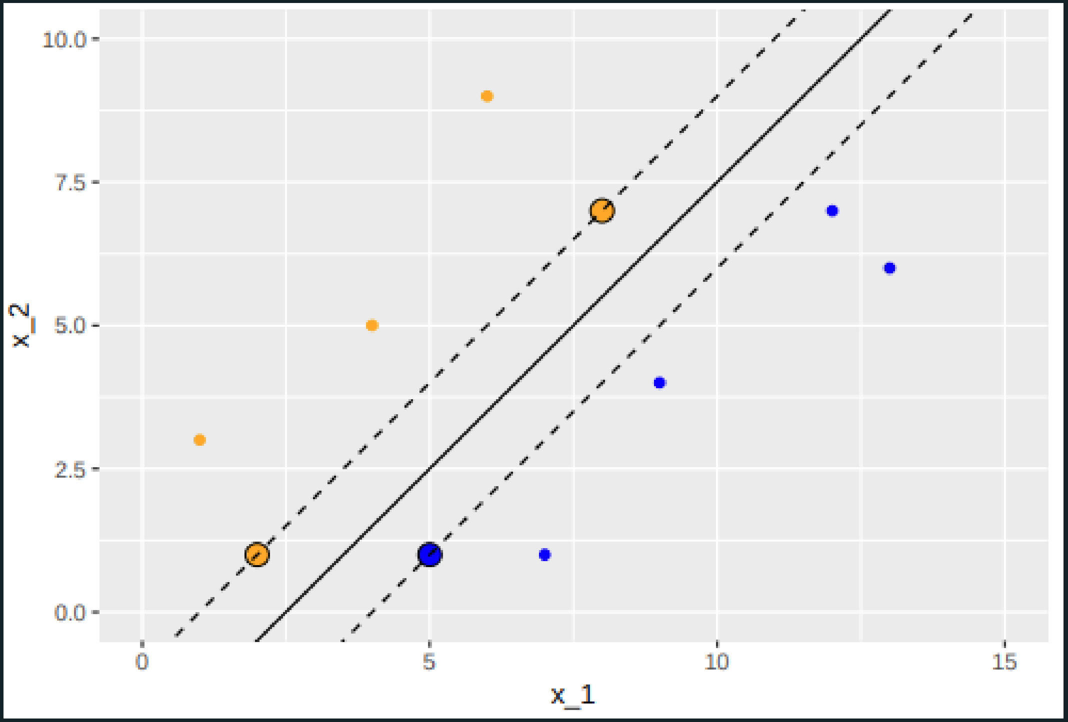
A perfectly separating linear hyperplan for a binary outcome



There are an infinity of such separating hyperplan
→ we need to choose one

Maximum Margin

Maximum margin classifier for a perfectly separable binary outcome variable



Criterium for optimal choice: the separating hyperplane for which the margin is the farthest from the observations
i.e., to select the **maximal margin hyperplane**

Support Vector

Support vector = the 3 observations from the training set that are equidistant from the maximal margin hyperplane

→ they "support" the maximal margin hyperplane (if they move, the the maximal margin hyperplane also moves)

Overcoming the perfectly separable hyperplan assumption

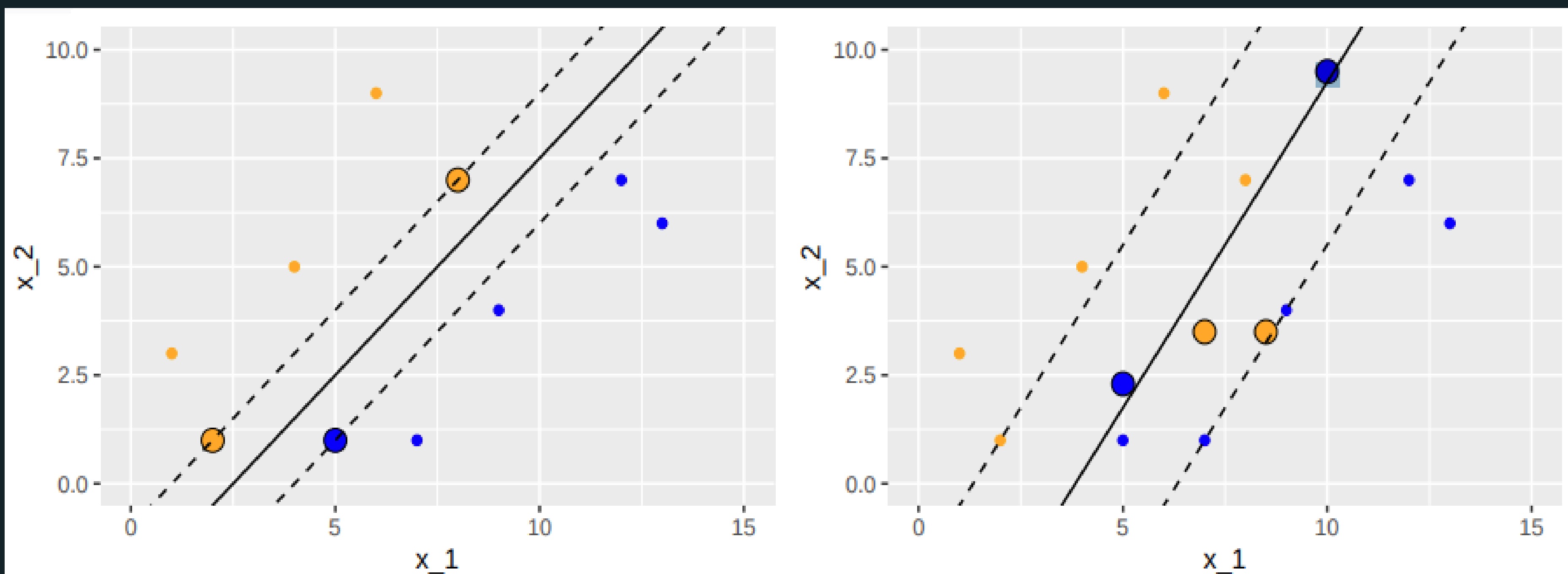
We allow some number of observations to violate the rules so that they can lie on the wrong side of the margin boundaries.

→ find a hyperplane that almost separates the classes

The **support vector classifier** generalizes the maximum margin classifier to the non-separable case.

Support Vector Classifiers

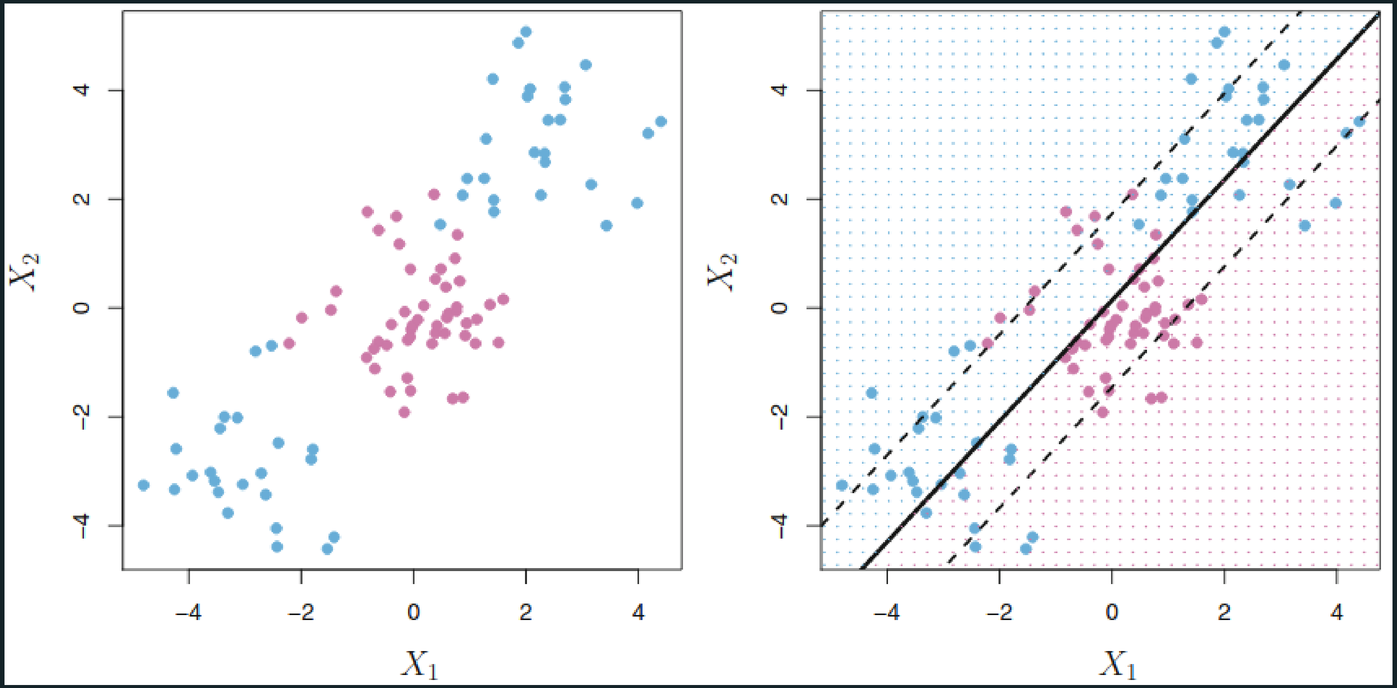
Maximal margin classifier (left) and support vector classifier (right)



Support Vector Classifiers: Details

- A **tuning parameter** C determines the severity of the violation ot the margin that the model tolerates
  - chosen by cross Validation
  - controls the bias-variance trade-off
- C small $\rightarrow$ narrow margins, rarely violated
- C large $\rightarrow$ wide margins, allow more violation
  - More bias classifier, but lower variance

Shortcomings of the linearity assumption:

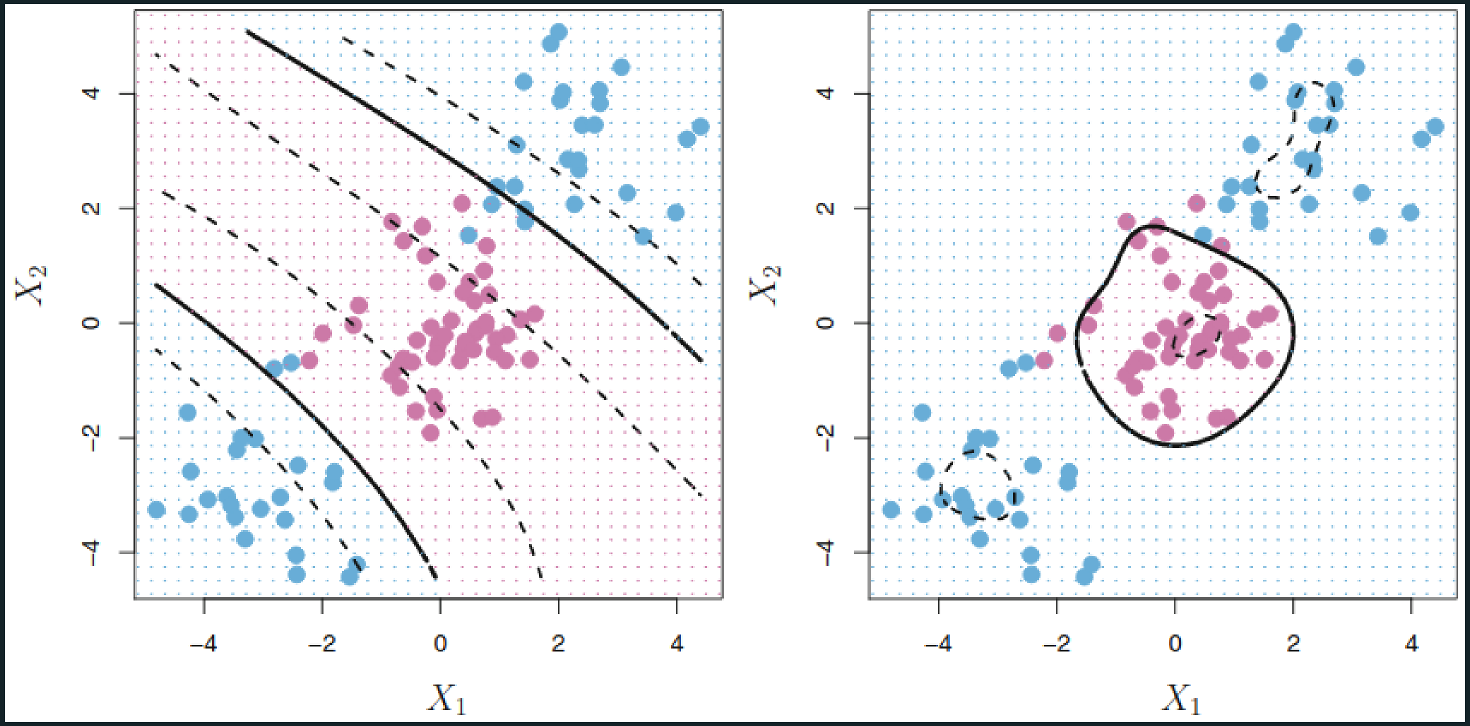


Overcoming the linearity assumption:

Support vector machines

- *Idea 1*: (polynomial) transformation of the features + StandardScaler + LinearSVC.
- *Idea 2*: convert a linear classifier into a classifier that produced **non-linear decision boundaries**. → using a **Kernel** such as:
  - Gaussian RBF kernel
  - Polynomial kernel
- **We do not open the kernel box**.
  - Just think as them as a way to construct non-linear hyperplans
  - Try out different kernel and distance specification

Support vector machines



- *Left*: polynomial kernel of degree 3;
- *Right*: radial kernel

Performance measures

Confusion Matrix

- For comparing the predictions of the fitted model to the actual classes.
- After applying a classifier to a data set with known labels *Yes* and *No*:

		Predicted class	
		no	yes
True class	no	TN	FP
	yes	FN	TP

Precision and Recall

- **Accuracy** : Proportion of rightly guessed observations
  - $\frac{\text{True Positives}+\text{True Negative}}{N}$
- **Precision**: accuracy of positive predictions
  - $\frac{\text{True Positives}}{\text{True Positives}+\text{False Positives}}$
  - decreases with false positives.
- **Recall**: proportion of true positives among all actual positives
  - $\frac{\text{True Positives}}{\text{True Positives}+\text{False Negatives}}$
  - decreases with false negatives.

F1 Score

- The F_1 score provides a single combined metric it is the **harmonic mean** of precision and recall

$$
\begin{aligned}
F_1 &= \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \\
&= \frac{\text{Total Positives}}{\text{Total Positives} + \frac{1}{2}(\text{False Negatives} + \text{False Positives})}
\end{aligned}
$$

- The harmonic mean gives **more weight to low values.**
- The F1 score values precision and recall **symmetrically.**

The Precision/Recall Trade-off

- F_1 favors classifiers with similar precision and recall,
- but sometimes you want **asymmetry**:

1. **low recall + high precision is better**

    - e.g. **deciding “guilty” in court**, you might prefer a model that
    - lets many actual-guilty go free (high false negatives $\leftrightarrow$ low recall)...
    - ... but has very few actual-innocent put in jail (low false positives $\leftrightarrow$ high precision

2. **high recall + low precision is better**

The Precision/Recall Trade-off

- F_1 favors classifiers with similar precision and recall,
- but sometimes you want **asymmetry**:

1. **low recall + high precision is better**

2. **high recall + low precision is better**

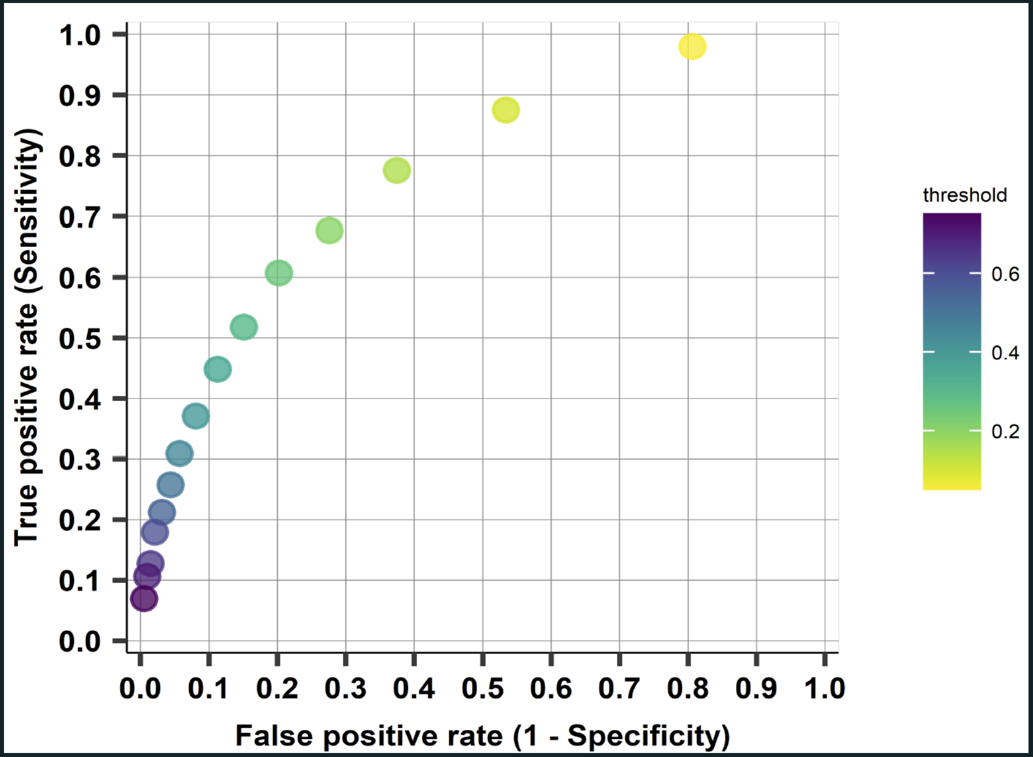
    - e.g classifier to **detect bombs during flight screening**, you might prefer a model that:
    - has many false alarms (low precision)...
    - ... to minimize the number of misses (high recall).

ROC Curve

- The ROC curve is a popular graphic for simultaneously displaying the types of errors for classification problems at various threshold settings
- For each threshold, we can compute confusion table –> calculate sensitivity and specificity
- The ROC curve of
  - a completely random probability prediction is the 45 degree line
  - a perfect probability prediction would jump from zero to one and stay at one

ROC Curve

- Plots *true positive rate* (recall) against the *false positive rate* ($\frac{FP}{FP+TN}$):



ROC Curve and AUC

- The Area Under (the ROC) Curve (AUC) is a popular metric ranging between:
  - 0.5
    - **random classification**
    - ROC curve = first diagonal
  - and 1
    - **perfect classification**
    - = area of the square
  - better classifier → ROC curve toward the top-left corner
- Good measure for model comparison