Supervised Learning - SVM

Mark Asch - IMU/VLP/CSU

2023

Program

1. Data Analysis
   (a) Introduction: the 4 identifiers of "big data" and "data science"
   (b) Supervised learning methods: regression—advanced, k-NN, linear classification methods, SVM, NN, decision trees.
   (c) Unsupervised learning methods: k-means, principal component analysis, clustering.

Introduction

- the "support vector machine" is a classification algorithm (though it can be used as a regressor—see below)
- SVM provides excellent performance in a broad range of contexts
- SVM is considered to be the best "black box" classifier available
- SVM generalizes the boundaries between classes to non-linear curves, and operates in high dimensions too

Theory

- SVM is based on the concept of separating hyperplanes

Definition 1. A hyperplane in p-dimensional space is a subspace of dimension $p-1$.

- Example 1: in $\mathbb{R}^2$ a hyperplane is a straight line,

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0,$$

- Example 2: in $\mathbb{R}^3$ a hyperplane is a plane, and in general, if $\mathbf{x} = (x_1, x_2, \ldots, x_p)^{\mathrm{T}} \in \mathbb{R}^p$ is a point in the hyperplane, then $\mathbf{x}$ satisfies

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = 0.$$

If $\mathbf{x}$ does not satisfy the equation, then either

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p > 0,$$

or

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p < 0.$$

- Conclusion: the hyperplane divides the space of dimension p in two halves and thus separates the two classes, as shown in the Figure.

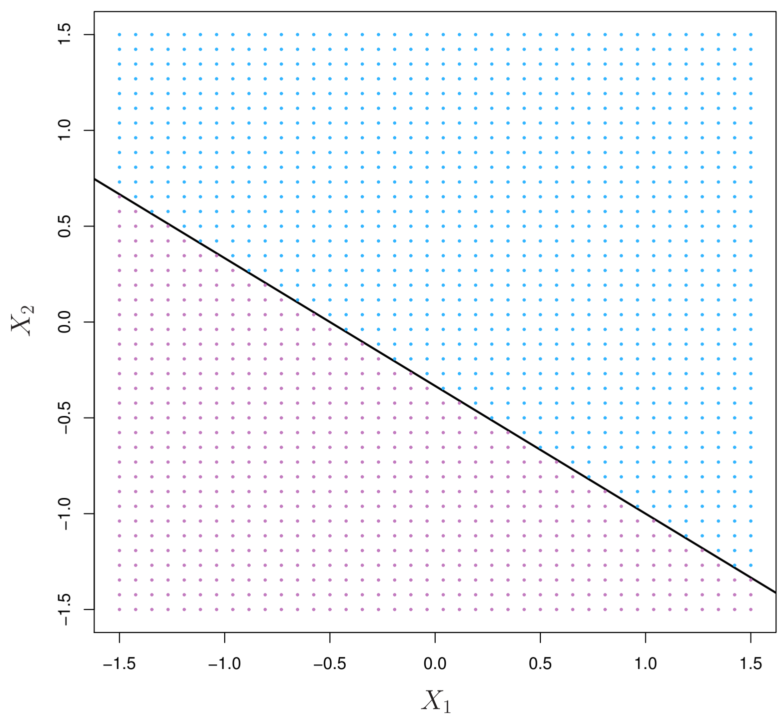

Figure 1: Hyperplane $1+2X_1+3X_2 = 0$ and the 2 classes $1+2X_1+3X_2 > 0$ (blue) and $1+2X_1+3X_2 < 0$ (violet).

Classification with a Hyperplane

- Suppose we have

  ⇒ a data matrix X of dimension $n \times p$ with n training observations in a p-dimensional space

$$x_1 = \begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix}, \ldots, x_n = \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix}$$

  ⇒ the observations fall into two classes denoted -1 and 1,

$$y_1, y_2, \ldots, y_n \in \{-1, 1\}$$

  ⇒ a vector of p test observations,

$$x^* = \left(x_1^* \ldots x_p^*\right)^T$$

- **Objective**: find a classifier, based on the training data, that will correctly class the test observations.

- If it is possible to construct a separating hyperplane that separates perfectly the training observations according to their class labels—we will say that the two classes are perfectly separable—then a test observation is affected to a class as a function of which side of the hyperplane it is in.

$\Rightarrow$ the sign of

$$f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \cdots + \beta_p x_p^*$$

defines the class, and

$\Rightarrow$ the magnitude of $f(x^*)$ gives us a measure of the confidence in our classification of x^*.

- But there are many possible such hyperplanes—see Figure below—so we need to optimize.

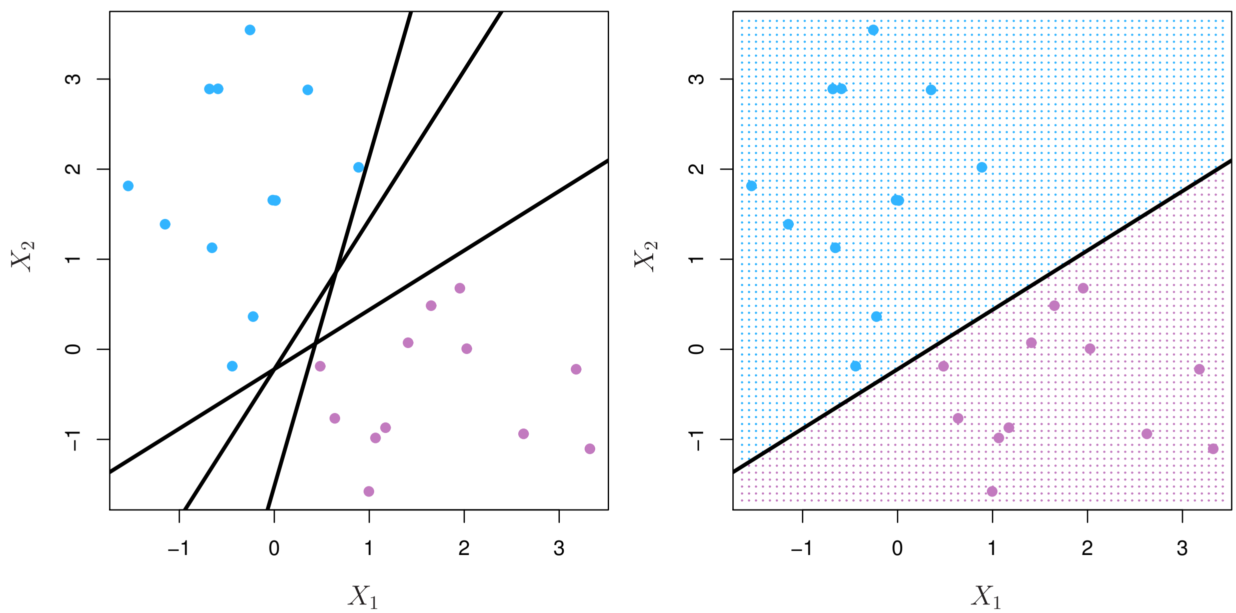

Figure 2: Two classes, blue and violet, with observations of 2 variables. Left: 3 possible separating planes.Right: an optimal separating plane.

- Optimal Separating Hyperplane: we seek the plane for which the minimal (orthogonal) distance to the observations is maximized (“maximal margin hyperplane”)

$\Rightarrow$ this plane depends exclusively on support vectors that are the orthogonal projections onto the hyperplane from the equidistant, closest points on either side of the hyperplane---see next Figure.

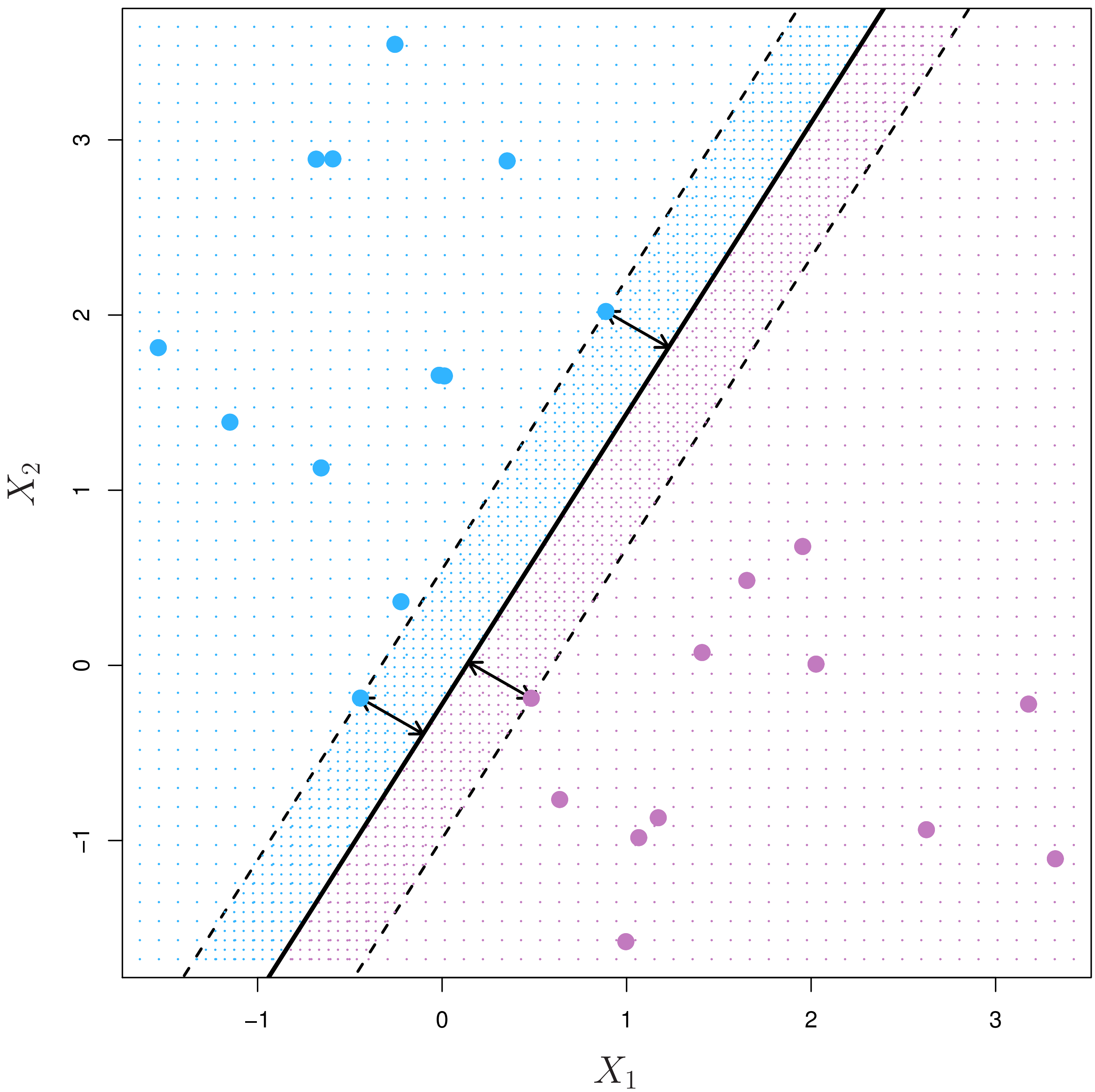

Figure 3: Optimal Hyperplane: three observations are equidistant and form the support vectors. Dashed lines form the margins.

Maximal Margin Classifier

Let $x_1, \ldots, x_n \in \mathbb{R}^p$ be a set of n training observations, with their associated class labels, $y_1, \ldots, y_n \in \{-1, 1\}$.

- The maximal margin hyperplane is the solution to the following optimization problem: Maximize, over $\beta_0, \ldots, \beta_p$, the distance (margin) M such that:

  $\Rightarrow$ $\sum_{j=1}^{p} \beta_j^2 = 1$,
  $\Rightarrow$ $y_i \left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}\right) \geq M \quad \forall i = 1, \ldots, n.$

- These two constraints guarantee that each observation is

  $\Rightarrow$ on the right side of the hyperplane, and
  $\Rightarrow$ at least a distance M from the hyperplane.

General Case

✔ The maximal margin classifier is a very natural way to classify, but,

✘ in many cases a separating hyperplane simply does not exist, and hence there is no maximal margin classifier.

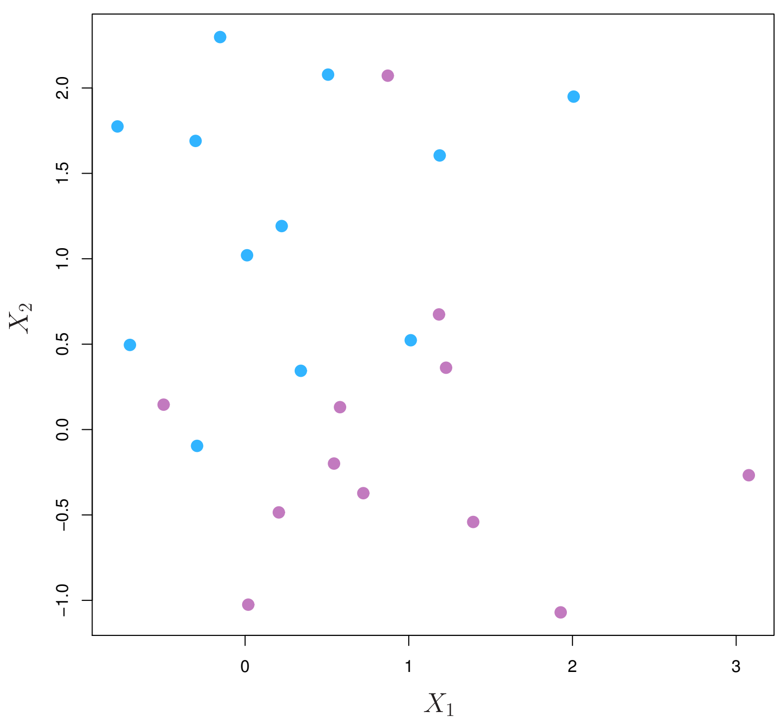

Figure 4: Example of two non-separable classes

- We can generalize the hyperplane approach by introducing a soft constraint to attain an approximate separation, the support vector classifier.

⇒ This hyperplane is chosen to separate most of the observations into the two classes, but it can misclassify some of them.

⇒ The optimization problem becomes

$$\max_{\beta_0,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n} M$$

$$\text{such that} \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i\left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}\right) \geq M(1-\epsilon_i),$$

$$\epsilon_1 \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

where $C > 0$ is an adjustment parameter, M is the width of the margin, and the ϵ_i allow individual observations to be on the wrong side of the margin or hyperplane.

⇒ **Note**: when C decreases, the tolerance for an observation to be on the bad side decreases too, and the margin itself becomes narrower.

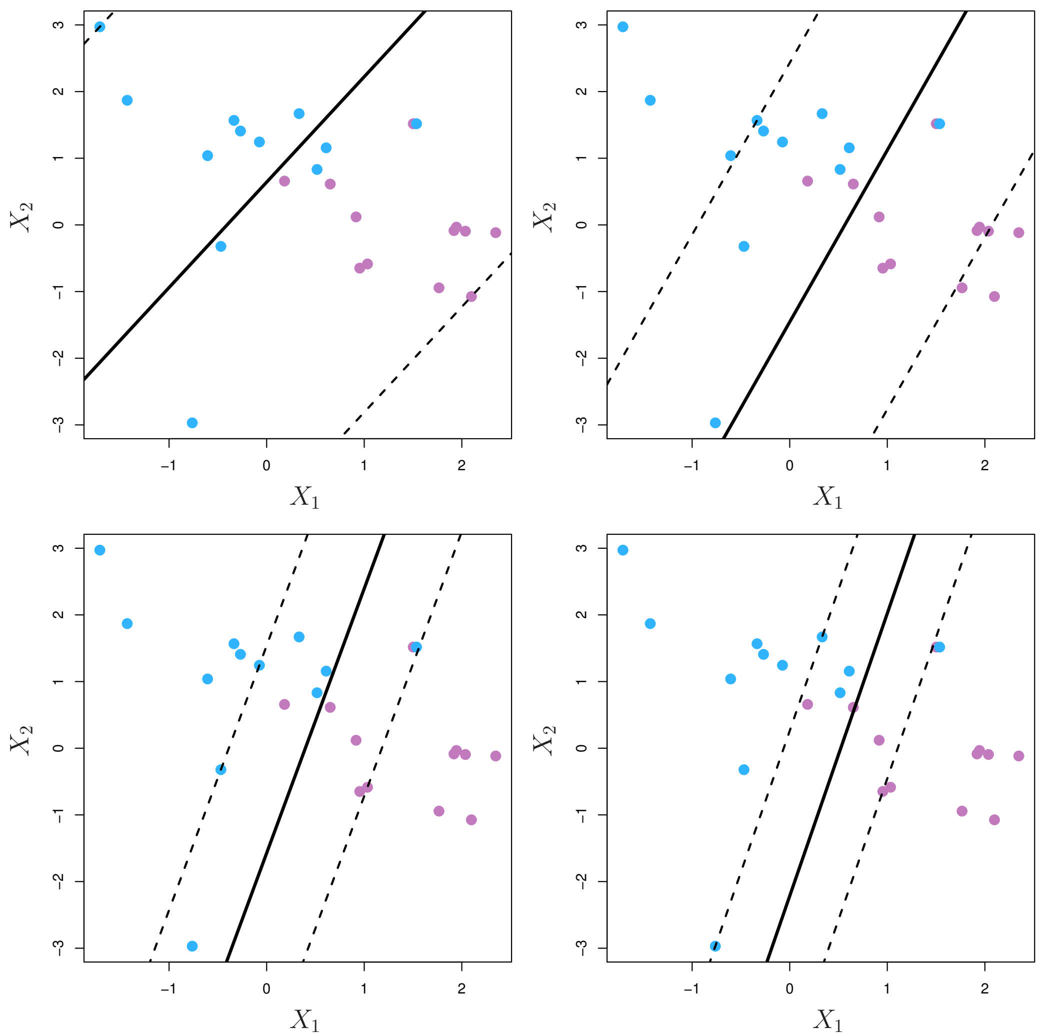

Figure 5: Support vector classifier with 4 values of C - biggest (top-left) to smallest (bottom-right)

Support Vector Machines (SVM)

- in practice, we will possibly encounter nonlinear boundaries between the classes...

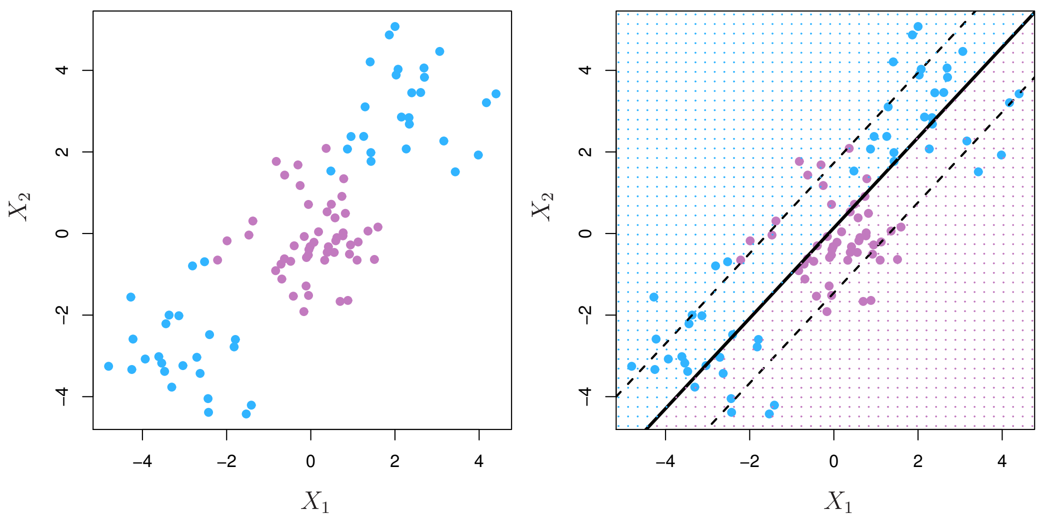

Figure 6: 2 classes with a nonlinear boundary between them (left). The support vector classifier cannot separate them (right).

- How to generalize? We observe that the computation of the linear SVM only depends on scalar products, of the form

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

where

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

- So we can replace these by general, kernel functions $K(x_i, x_{i'})$ that quantify the similarity between two observations.

  $\Rightarrow$ For example,

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

  which is a polynomial kernel of degree d.

  $\Rightarrow$ Another popular choice is the radial kernel,

  $$K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p} \left(x_{ij} - x_{i'j}\right)^2\right),$$

where the parameter γ is to be tuned to the class extent.

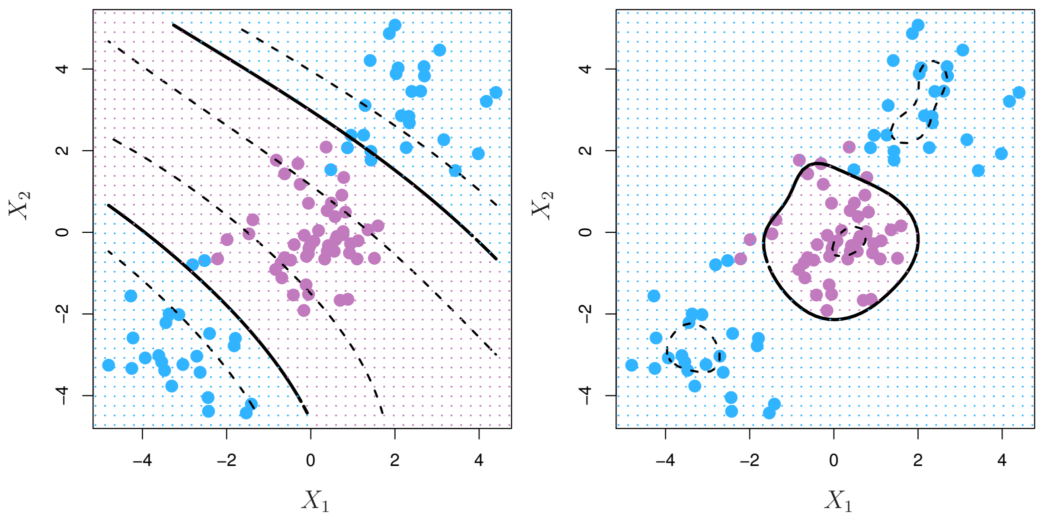

Figure 7: SVM with 2 kernels: polynomial of degree 3 (left) and radial (right)

Example : cardiac disease

- The `Heart` data are a database of 297 patients with 13 predictors including `Age`, `Sex`, `Chol` and a binary outcome `HD` for signs of chest pain. A `Yes` result signifies the presence of cardiac disease after an angiographic test, and `No` signifies the absence.

- Objective : use the predictors to predict whether an individual has a cardiac disease or not.

  $\Rightarrow$ divide the data randomly into 207 training observations and 90 test observations

  $\Rightarrow$ statistical model 1: LDA and support vector classifier

  $\Rightarrow$ statistical model 2: SVM with a radial kernel

- the 2 classifiers calculate

  $\Rightarrow$ scores of the form

$$\hat{f}(X) = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \cdots + \hat{\beta}_p X_p$$

  for each observation

⇒ then, for a given threshold, t, classify the observations in the category *presence or absence of cardiac disease* according to

$$\hat{f}(X) < t \quad \text{or} \quad \hat{f}(X) \geq t$$

⇒ the ROC curve («receiver operating characteristic»)is obtained by varying t and calculating the false positive and false negative rates on the training data, then on the test data

→ the closer the curve is to the left and top, the better is the classification

→ objective is AUC = 1, where AUC is the Area Under the Curve

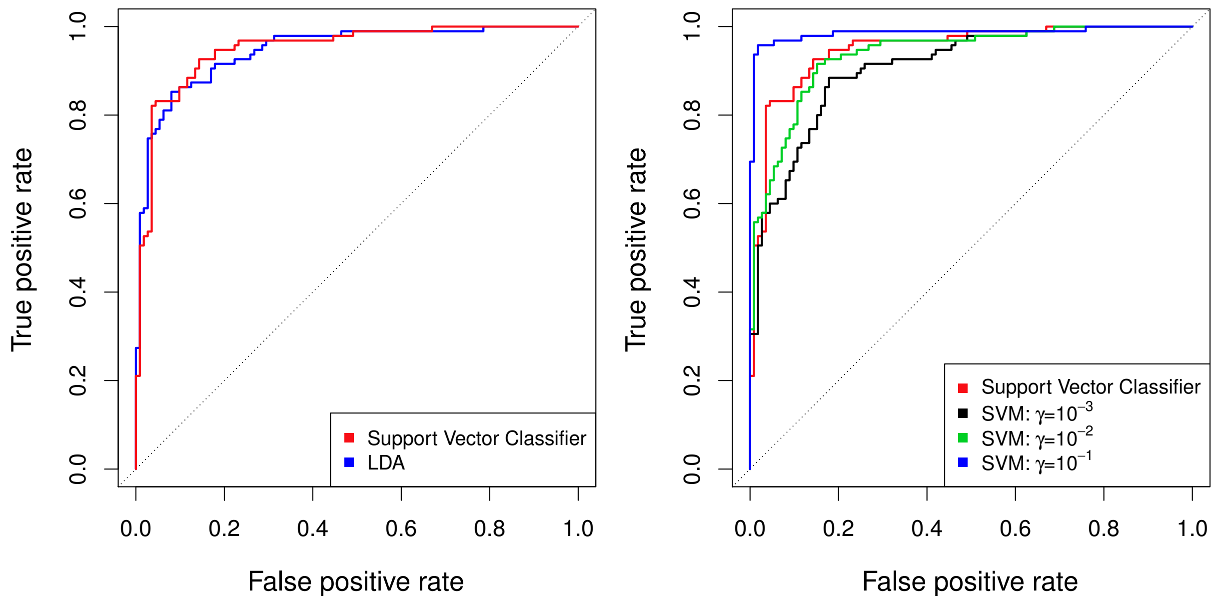

Figure 8: ROC curves for training data Heart

- Results for training data:

$\Rightarrow$ the linear SVM linear (red curve) is better than LDA...
$\Rightarrow$ for larger values of γ (more non linearity in the fit), the classification improves
$\Rightarrow$ the radial kernel with $\gamma = 10^{-1}$ is best, and better than the linear SVM

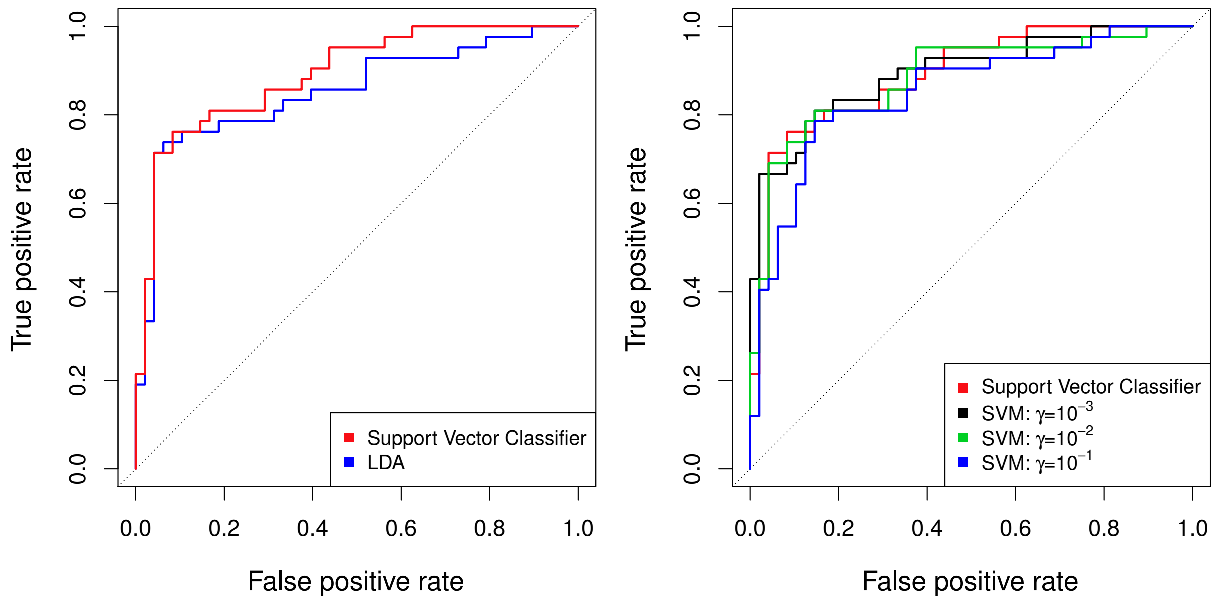

Figure 9: ROC curves for test data Heart

- Results for test data:

  $\Rightarrow$ linear SVM is slightly better than LDA

  $\Rightarrow$ the radial kernel with $\gamma = 10^{-1}$ is no longer the best

  $\Rightarrow$ a model with more flexibility (more complex) does not always produce the best test results (bias-variance tradeoff...)

SVM for more than 2 Classes

- we have only seen problems of binary classification ($K = 2$ classes)—though these are very common.

- how do we generalize to an arbitrary number of classes?

  $\Rightarrow$ classification one-to-one: construct $\binom{K}{2}$ binary classifiers, apply to test data, and classify according to highest frequency (default method)

  $\Rightarrow$ classification one-to-all : fit K SVMs where we compare one of the K classes to the other $K-1$ classes.

Function `svm` of R

- in the library `e1071`
- arguments:
  - ⇒ formula of the form `y ~ x` that describes the relation between the response variable and the explanatory variables (for a classification, `y` must be of type "factor")
  - ⇒ `data` = data matrix
  - ⇒ `scale` = vector of logical values describing which explanatory variables should be scaled (by default = `TRUE`)
  - ⇒ `kernel` = choice of kernel: linear, polynomial, radial, sigmoid
  - ⇒ `gamma` is the parameter in all the kernels except the linear one
  - ⇒ `cost` is the cost and equals $1/C$, where C is the regularization coefficient in the optimization (when `cost` is large, the margins are narrow and the number of support vectors will be fewer)

- tools:

  ⇒ `plot` : plot the classification

  ⇒ `table` : display the confusion table

  ⇒ `predict` : apply the model to the test data

  ⇒ `tune` : perform a grid-search for the hyper-parameters `cost` and `gamma`, by using cross validation (by default, 10-fold) or bootstrap, and return the best model in `best.tune()`.

Other examples

1. Random data: svm-1.html.

2. Genomic data: svm-genomic.html.

References

1. M. DeGroot, M. Schervish, *Probability and Statistics*, Addison Wesley, 2002.

2. Spiegel, Murray and Larry Stephens, *Schaum's Outline of Statistics,* 6th edition, McGraw Hill. 2017.

3. G. James, D. Witten, T. Hastie, R. Tibshirani. *An Introduction to Statistical Learning with Applications in R.* Springer. 2013.

4. T. Hastie, R. Tibshirani, J. Friedman. *The Elements of Statistical Learning*. Springer. 2009.

5. Rachel Schutt and Cathy O'Neil. *Doing Data Science.* O'Reilly. 2014.