Inteligencia Artificial

UTN – FRVM

5º Año Ing. en Sistemas de Información

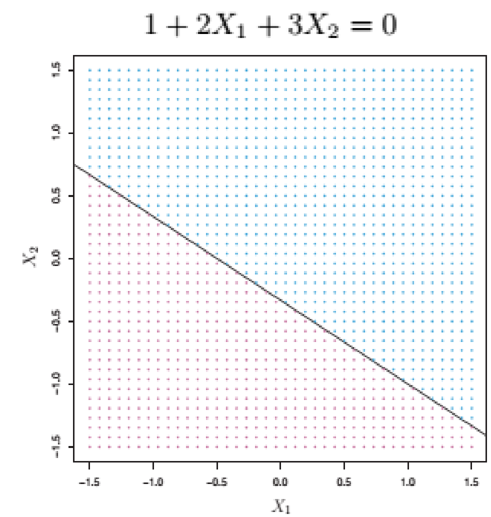
Agenda

- Support Vector Machines
  - *maximal margin classifier*
  - *support vector classifier*
  - *support vector machines*

Maximal Margin Classifier

- *Hyperplane.* In a *p*-dimensional space, a *hyperplane* is a flat affine subspace of hyperplane dimension $p - 1$.
- In two dimensions, a hyperplane is defined by the equation.

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$



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

p-dimensions

$$\boxed{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0}$$

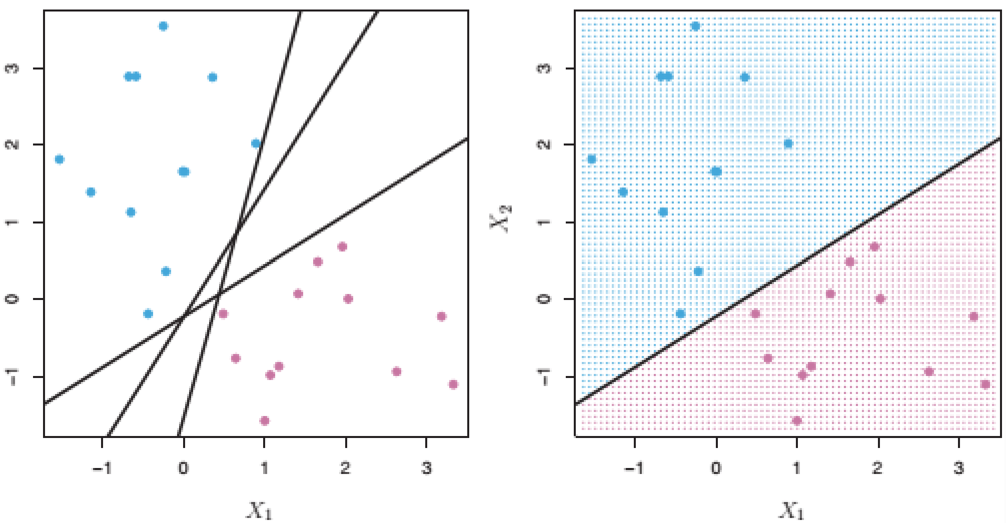
The example lies under the hyperplane

$$\boxed{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0.}$$

The example lies above the hyperplane

Classification Using a Separating Hyperplane

- *Separating hyperplane*. Suppose that it is possible to construct a hyperplane that separates the hyperplane training observations perfectly according to their class labels.



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

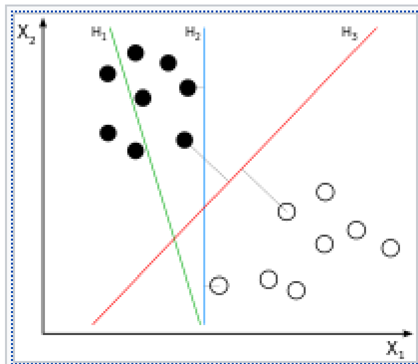
- If $f(x*)$ is positive, then we assign the test observation to class 1, and if $f(x*)$ is negative, then we assign it to class −1. We can also make use of the *magnitude* of $f(x*)$.
- If $f(x*)$ is far from zero, then this means that $x*$ lies far from the hyperplane,
and so we can be confident about our class assignment for $x*$

The Maximal Margin Classifier

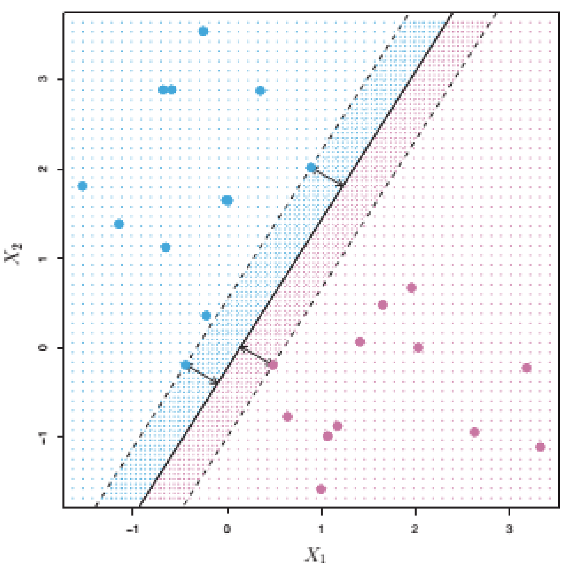
- In general, if our data can be perfectly separated using a hyperplane, then there will in fact exist an infinite number of such *hyperplanes*.
- A natural choice is the ***maximal margin hyperplane*** (also known as the maximal margin hyperplane *optimal separating hyperplane*), which is the separating hyperplane that is ***farthest*** from the training observations.
- We can compute the (perpendicular) distance from each training observation to a given separating hyperplane; the smallest such distance is the minimal distance from the observations to the hyperplane, and is known as the ***margin.***
- The **maximal margin hyperplane** is the separating hyperplane for which the margin is ***largest***.

Support Vectors

- the maximal margin hyperplane represents the mid-line of the widest “slab” that we can insert between the two classes.



Support Vectors: they “support” the maximal margin hyperplane in the sense that if these points were moved slightly, then the maximal margin hyperplane would move as well. Interestingly, the maximal margin hyperplane depends directly on the support vectors



The maximal margin hyperplane depends directly on only a **small subset of the observations**

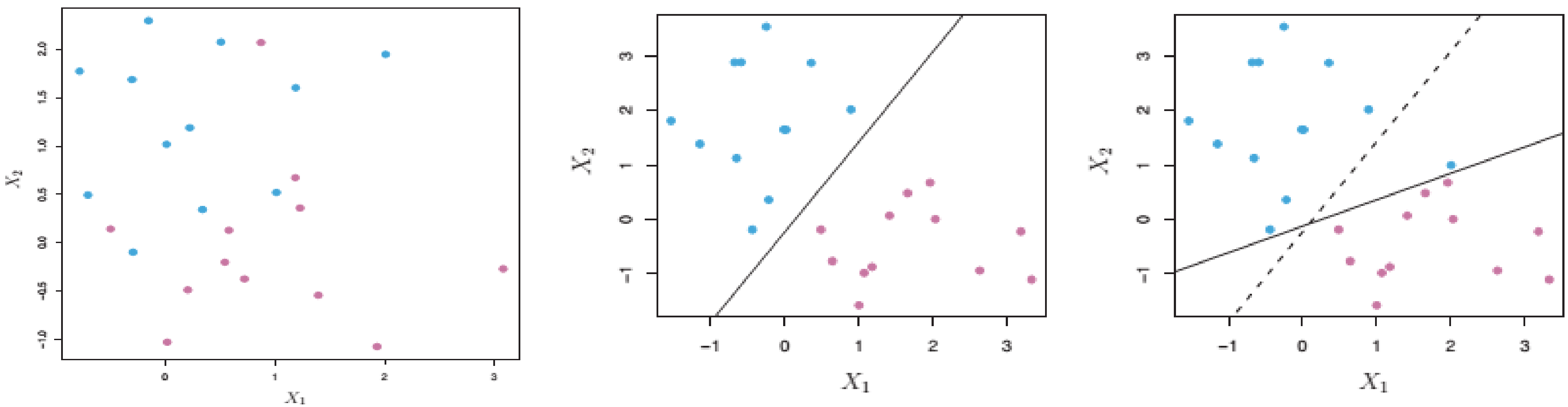
$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\; M$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \quad \forall\, i = 1, \ldots, n.$$

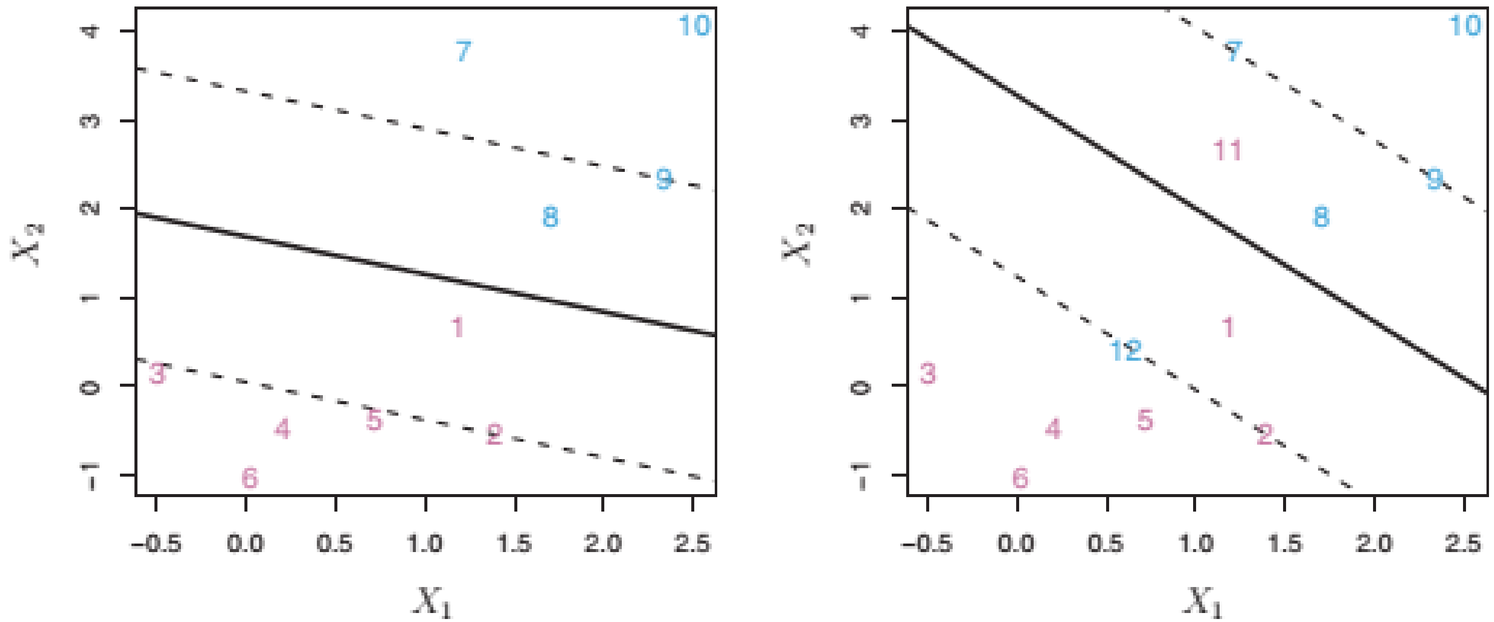
The Non-separable Case

- As we have hinted, in many cases no separating hyperplane exists, and so there is no maximal margin classifier. In this case, the optimization problem showed before has no solution with $M > 0$.



- That is, it could be worthwhile to misclassify a few training observations inorder to do a better job in classifying the remaining observations.
- The *support vector classifier*, sometimes called a *soft margin classifier*, does exactly this. Rather than seeking the largest possible margin so that every observation is not only on the correct side of the hyperplane but also on the correct side of the margin, we instead allow some observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane. (The margin is *soft* because it can be violated by some of the training observations.)

Support Vector Classifier



- ***Left***: *Observations* 3, *4,* *5,* *and* 6 *are on the correct side of the margin, observation* 2 *is on the margin, and observation 1 is on the wrong side of the margin.* Blue observations: *Observations* 7 *and* 10 *are on the correct side of the margin, observation* 9 *is on the margin, and observation* 8 *is on the wrong side of the margin. No observations are on the wrong side of the hyperplane.* ***Right***: *Same as left panel with two additional points,* 11 *and* 12. *These two observations are on the wrong side of the hyperplane and the wrong side of the margin.*

Support Vector Classifier

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

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

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

The slack variable ϵ_i tells us where the *i*th observation is located, relative to the hyperplane and relative to the margin. If $\epsilon_i = 0$ then the *i*th observation is on the correct side of the margin.

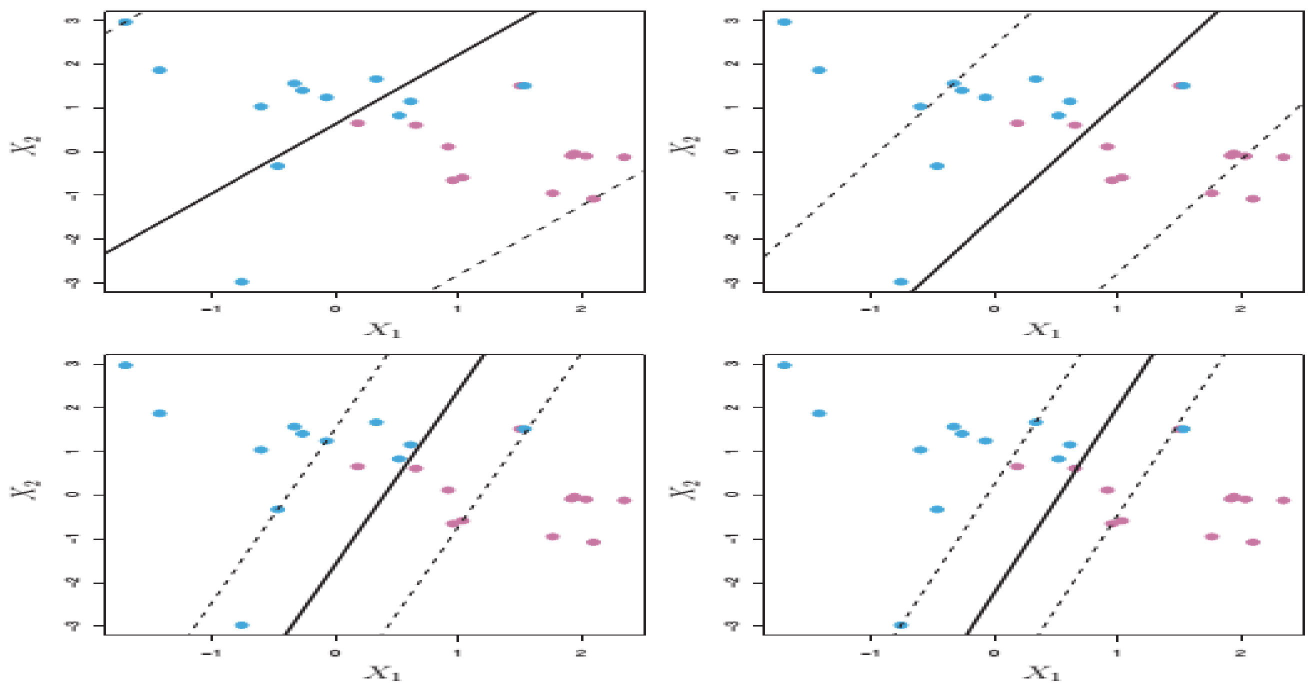
If $\epsilon_i > 0$ then the *i*th observation is on the wrong side of the margin, and we say that the *i*th observation has *violated* the margin.

If $\epsilon_i > 1$ then it is on the wrong side of the hyperplane.

C determines the number and severity of the violations to the margin (and to the hyperplane) that we will tolerate, so it is a *budget* for the amount that the margin can be violated by the *n* observations.
If $C = 0$ then there is no budget for violations to the margin. As the budget *C* increases, we become more tolerant of violations to the margin, and so the margin will widen. Conversely, as *C* decreases, we become less tolerant of violations to the margin and so the margin narrows.

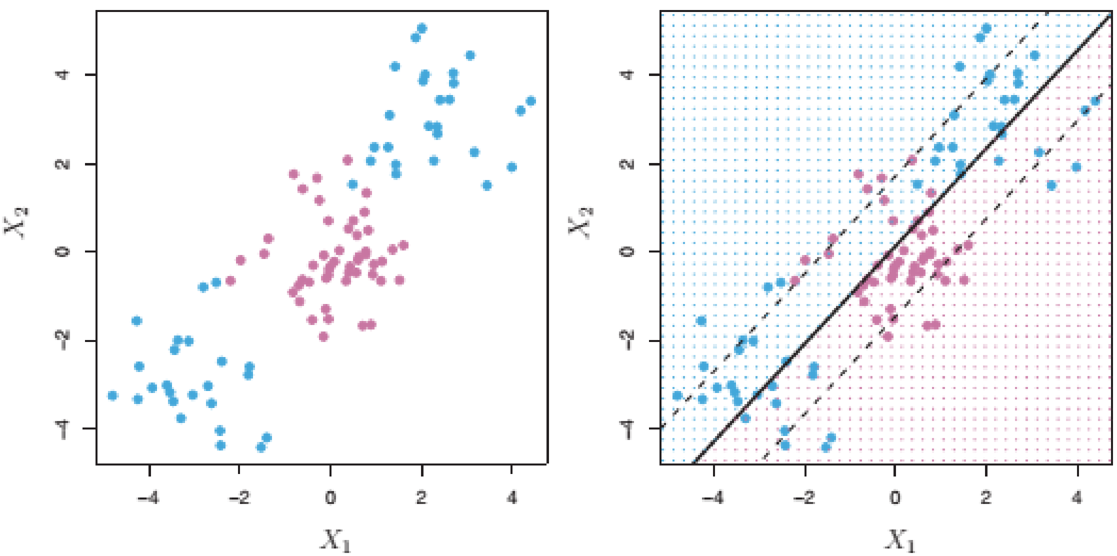
Support Vector Classifier

- In practice, C is treated as a tuning parameter that is generally chosen via cross-validation.
- C controls the bias-variance trade-off of the statistical learning technique. When C is small, we seek narrow margins that are rarely violated; this amounts to a classifier that is highly fit to the data, which may have low bias but high variance.
- On the other hand, when C is larger, the margin is wider and we allow more violations to it; this amounts to fitting the data less hard and obtaining a classifier that is potentially more biased but may have lower variance.



Support Vector Machines

- The support vector classifier is a natural approach for classification in this setting, if the boundary between the two classes is linear. However, in practice we are sometimes faced with non-linear class boundaries.
- In the case of the support vector classifier, we could address the problem of possibly non- linear boundaries between classes in a similar way, by enlarging the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors.



$$X_1, X_2, \ldots, X_p$$

$$X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2$$

$$\underset{\beta_0,\beta_{11},\beta_{12},\ldots,\beta_{p1},\beta_{p2},\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

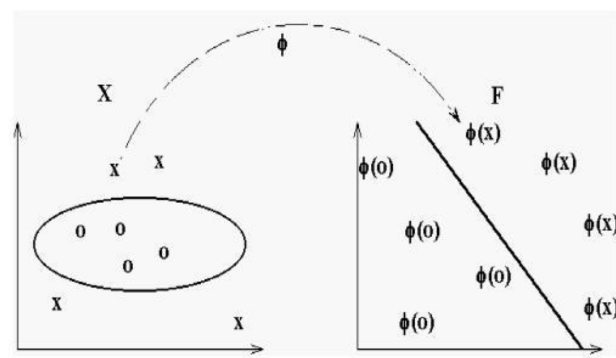
$$\text{subject to } y_i\left(\beta_0 + \sum_{j=1}^{p}\beta_{j1}x_{ij} + \sum_{j=1}^{p}\beta_{j2}x_{ij}^2\right) \geq M(1-\epsilon_i)$$

$$\sum_{i=1}^{n}\epsilon_i \leq C, \quad \epsilon_i \geq 0, \quad \sum_{j=1}^{p}\sum_{k=1}^{2}\beta_{jk}^2 = 1.$$

The Support Vector Machine

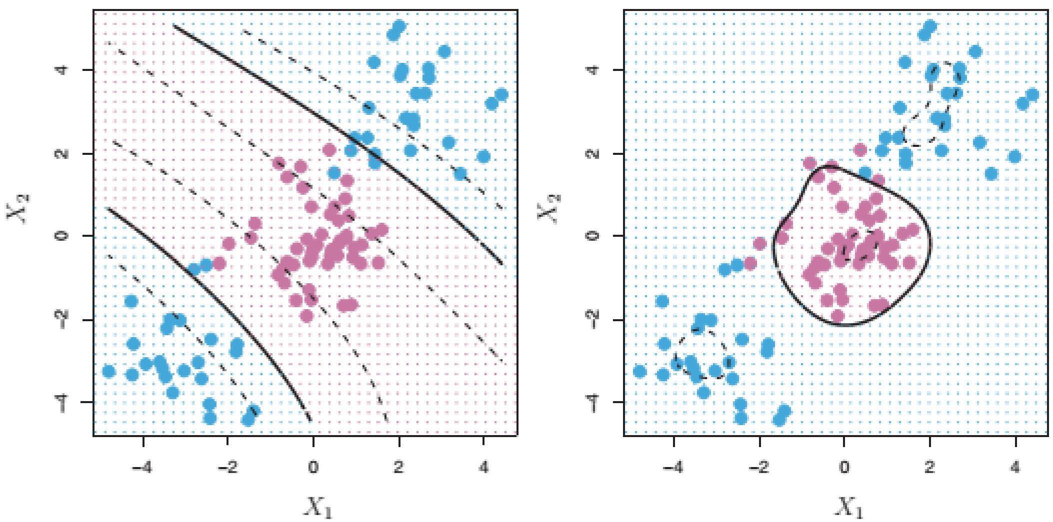
- The *support vector machine* (SVM) is an extension of the support vector classifier that results from enlarging the feature space in a specific way, using *kernels*.
- *Kernels* projects the original information to a space of more dimensions, so the input space X is mapped to a new space of a greater dimensionality (Hilbert):

Kernel trick



- $F = \{\varphi(x) | x \in X\}$
- $x = \{x_1, x_2, \cdot\cdot\cdot, x_n\} \rightarrow \varphi(x) = \{\varphi_1(x), \varphi_2(x), \cdot\cdot\cdot, \varphi_n(x)\}$
- When the support vector classifier is combined with a non-linear kernel, the resulting classifier is known as a ***support vector machine***

SVM



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

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

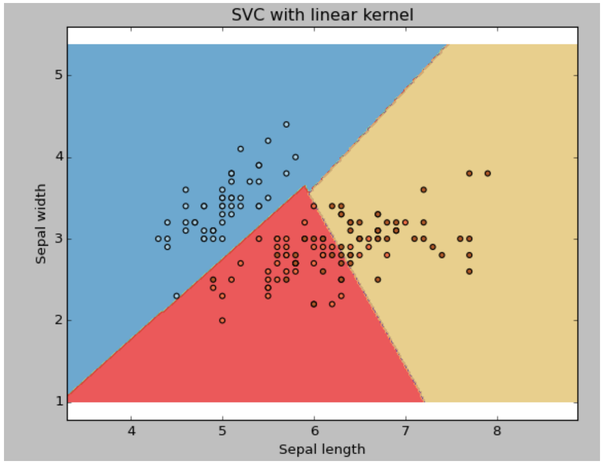
$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

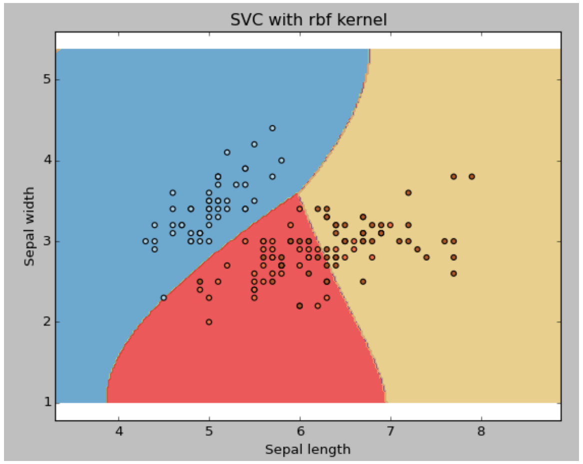
$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2).$$

- ***Left**: An SVM with a polynomial kernel of degree 3. **Right**: An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.*

SVM's with More than Two Classes

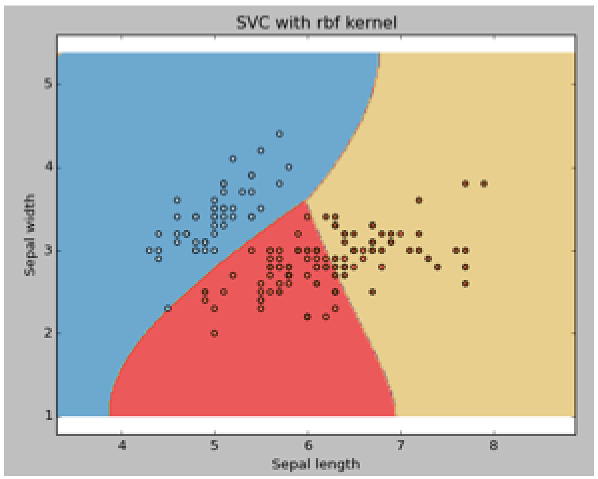
- *One - Versus - One Classification*
- *One – Versus – All*
- We fit ***K SVMs***, each time comparing one of all the ***K*** classes to the remaining $K - 1$ classes



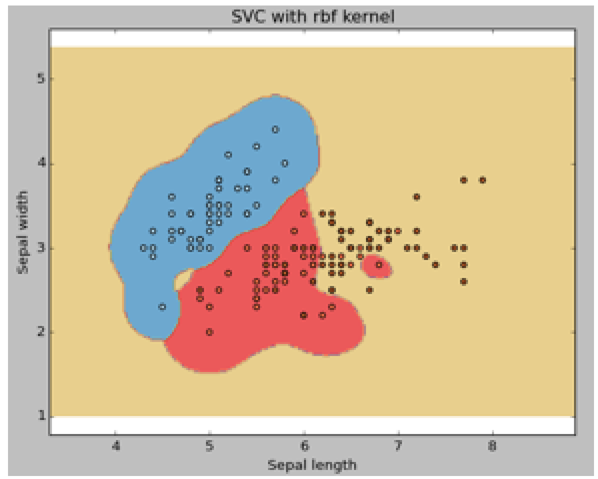


SVM

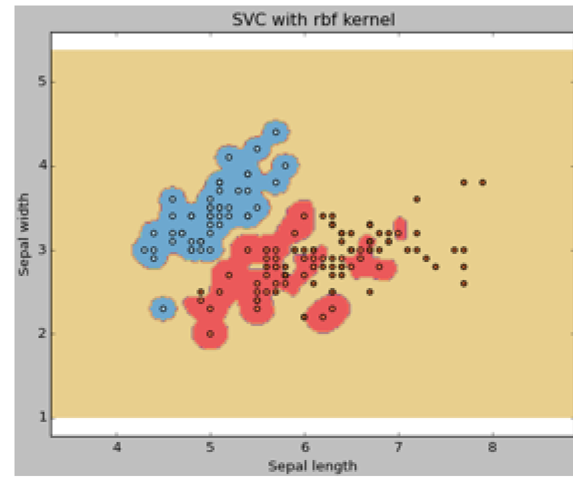
gamma =0



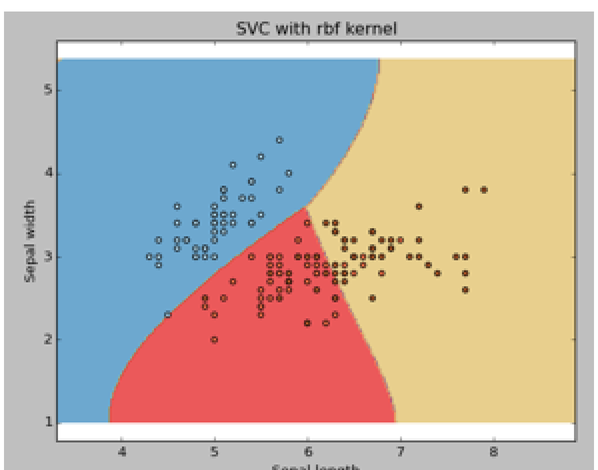
gamma =10



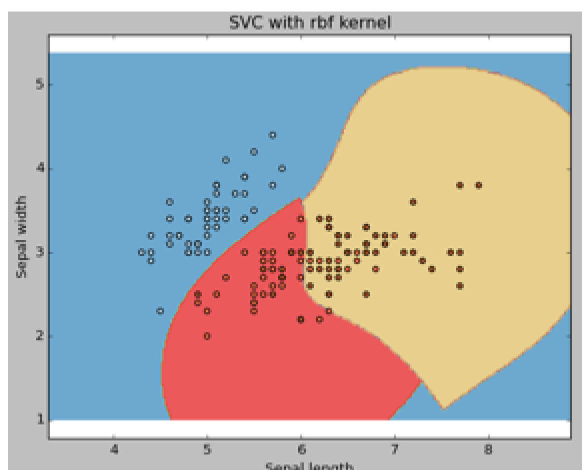
gamma =100



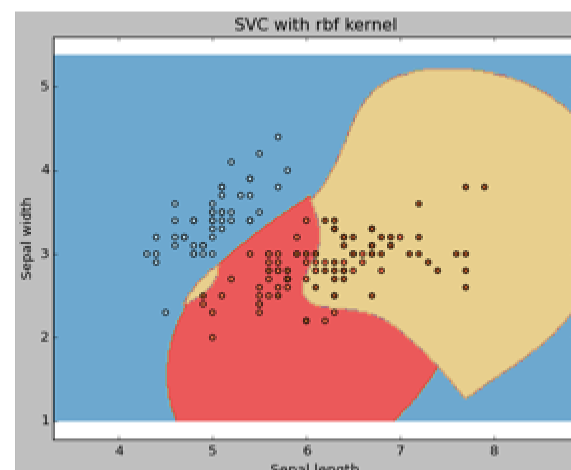
c =1



C =100

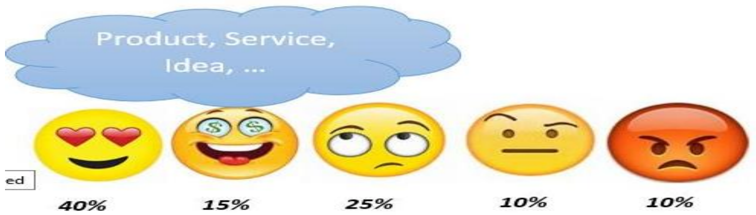


c =1000



Application Examples

- Spam filtering, a process which tries to discern E-mail spam messages from legitimate emails.
- Email routing, sending an email sent to a general address to a specific address or mailbox depending on topic
- Genre classification, automatically determining the genre of a text
- Readability assessment, automatically determining the degree of readability of a text, either to find suitable materials for different age groups or reader types or as part of a larger text simplification system
- Sentiment analysis, determining the attitude of a speaker or a writer with respect to some topic or the overall contextual polarity of a document.
- Health-related classification using social media in public health surveillance
- Article triage, selecting articles that are relevant for manual literature curation, for example as is being done as the first step to generate manually curated annotation databases in biology.



Application Examples

- Classification of images can also be performed using SVM.

Computers and Electronics in Agriculture 135 (2017) 81–95

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

ELSEVIER

Image classification for detection of winter grapevine buds in natural conditions using scale-invariant features transform, bag of features and support vector machines

Diego Sebastián Pérez [a,b,*], Facundo Bromberg [c], Carlos Ariel Diaz [a]

[a] *Universidad Tecnológica Nacional, Facultad Regional Mendoza, Laboratorio de Inteligencia Artificial DHARMa, Dpto. de Sistemas de la Información, Rodríguez 273, CP 5500 Mendoza, Argentina*

[b] *Universidad Nacional de Cuyo, Instituto universitario para las Tecnologías de la Información y las Comunicaciones, CONICET, Padre Jorge Contreras 1300, CP 5500 Mendoza, Argentina*

[c] *Universidad Tecnológica Nacional, Facultad Regional Mendoza, CONICET, Laboratorio de Inteligencia Artificial DHARMa, Dpto. de Sistemas de la Información, Rodríguez 273, CP 5500 Mendoza, Argentina*

ARTICLE INFO

Article history:
Received 30 April 2016
Received in revised form 6 January 2017
Accepted 23 January 2017
Available online 10 February 2017

ABSTRACT

In viticulture, there are several applications where bud detection in vineyard images is a necessary task, susceptible of being automated through the use of computer vision methods. A common and effective family of visual detection algorithms are the *scanning-window* type, that slide a (usually) fixed size window along the original image, classifying each resulting windowed-patch as containing or not containing the target object. The simplicity of these algorithms finds its most challenging aspect in the classification