Machine Learning

Module 3.4 - Models: Support Vector Machine

Marc-Olivier Boldi

Master in Management, Business Analytics, HEC UNIL

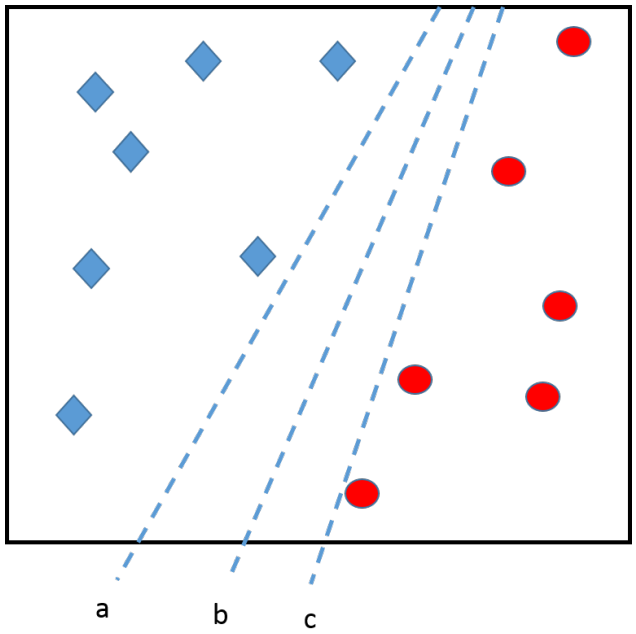
Spring 2025

Table of Contents

Table of Contents

Concept

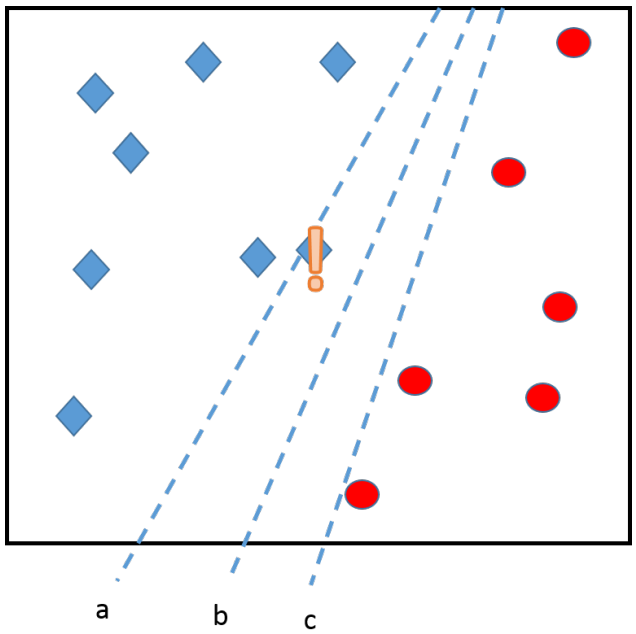
Consider a binary classification with two features (axes)



The 3 dashed lines represents 3 different classifiers, equally good in terms of predictions. Intuitively, line b may be preferable. Why?

Concept

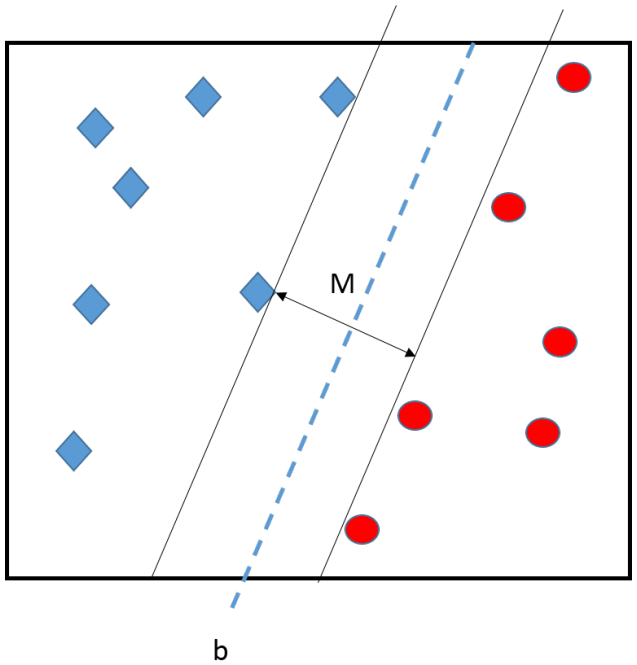
"Line b" model has more capacity to generalize. Indeed, suppose we observe a new diamond, with line a, the model would make an incorrect prediction, even if this diamond is not especially unexpected. The symmetric situation with circles would exclude line c also => line b is the most robust choice.



The margins

Line b is better because it has a larger margin M. SVM are built on that principle:

- Provide good predictions
- Be robust to new observations (large margin).



Geometry

The borders can be expressed by a vector $w = (w_1, \ldots, w_p)$ and a constant b. The central line is all the point $x = (x_1, \ldots, x_p)$ such that

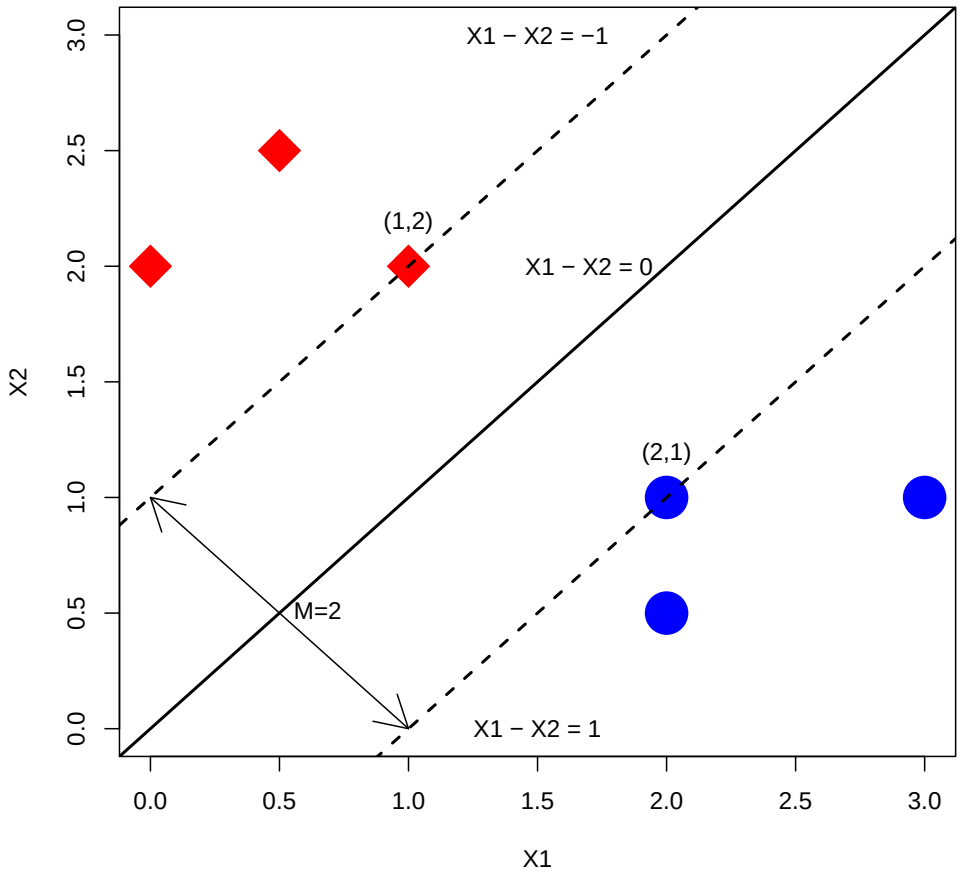
$$w_1 x_1 + w_2 x_2 + \cdots + w_p x_p + b = 0, \quad \text{i.e.,} \quad w^T x + b = 0.$$

The borders are the two lines limiting the two sets (diamonds only vs circles only).

They are the lines set such that $w^T x + b = -1$ and $w^T x + b = 1$.

Geometry

Below, the number of features is $p = 2$, $w = (1, -1)$, and $b = 0$.



Geometry

For a good separation, we generally look for a central line

$$w^T X + b = 0,$$

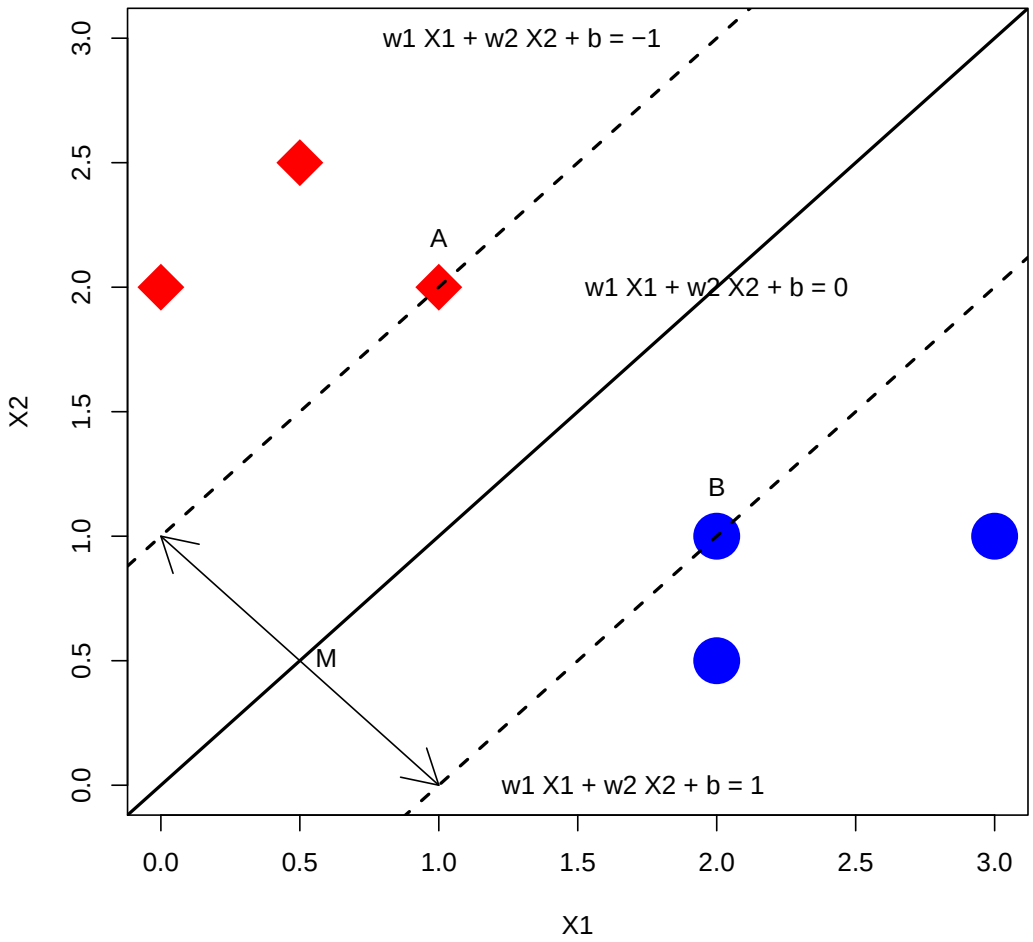
at the middle of the two closest points (A and B on the next slide) and such that

- $w^T X_A + b = -1$
- $w^T X_B + b = 1$

In addition, we want that all the red diamonds are above the upper border, and all the blue circles are below the lower border. This translates into

- $w^T X_i + b \leq -1$, for all i such that Y_i is "red diamond".
- $w^T X_i + b \geq 1$, for all i such that Y_i is "blue circle".

Geometry


w1 X1 + w2 X2 + b = −1
w1 X1 + w2 X2 + b = 0
w1 X1 + w2 X2 + b = 1
A
B
M
X1
X2
0.0
0.5
1.0
1.5
2.0
2.5
3.0

Geometry

It can be shown that the distance between the two borders is the margin M, and can be computed as

$$M = \frac{2}{\sum_{j=1}^{p} w_j^2} = \frac{2}{\sqrt{w^T w}}.$$

For a good model, we want

- good predictions: all the points are separated, i.e., previous inequalities are satisfied.
- robustness: the margin M is maximum or, equivalently, $w^T w$ is minimum.

Geometry

To further simplify the notation, note that, in this perfect configuration,

- Blue circles (below) are such that $w^T x_i + b \geq 1$
- Red diamonds (above) are such that $w^T x_i + b \leq -1$

Therefore, for a binary classification, we assign the outcome values $y_i = -1$ to red diamonds and $y_i = 1$ to blue circles.

In such case, the inequalities can be written as

$$y_i(w^T x_i + b) \geq 1, \quad \text{for all } i.$$

Optimization problem

Overall, to solve the SVM, one looks for w for which $w^T w$ is minimum and

$$y_i(w^T x_i + b) \geq 1, \quad \text{for all } i.$$

This can be written as the optimization problem

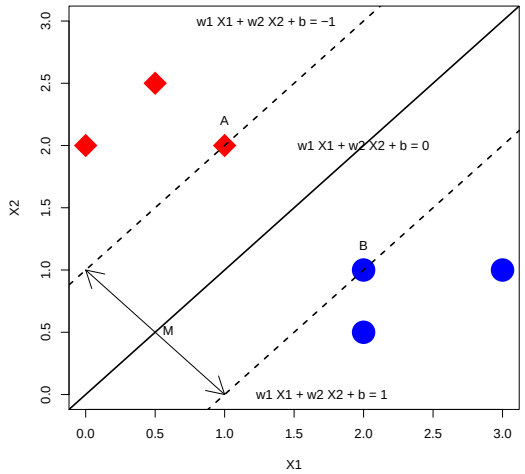
$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} w^T w \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1, \quad i = 1, \ldots, n \end{aligned}$$

This problem can be solved using quadratic program under linear constraints (see your favorite optimization course).

The solution of this problem will provide a perfect separation between the two categories with the largest margin.

Support vectors

The points that are far inside their margins are not playing any role. E.g., moving a red diamond that is not A will not change the border line.



The margins depends on the points **on** the margins, i.e., (x_i, y_i) such that

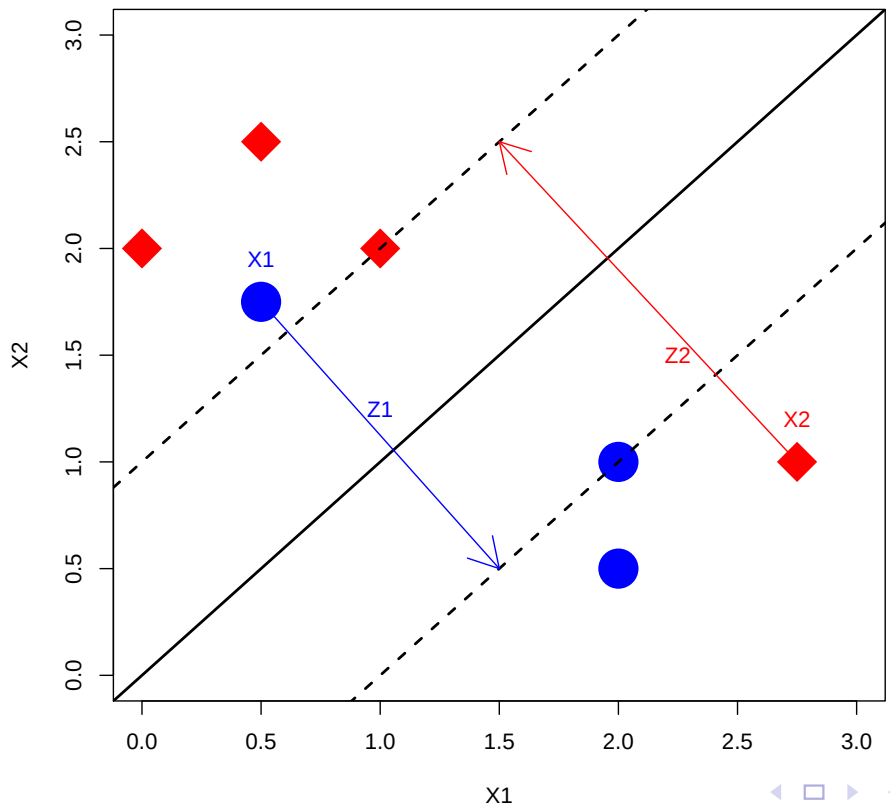
$$y_i(w^T x_i + b) = 1.$$

These points are called **support vectors**.

Non linearly separable case

However, this approach is valid only if the categories are **linearly perfectly separable**: there exists a line that perfectly separates the classes.

Most cases are non-separable: no line can reach the perfect separation.



Soft margins

Under the perfect separation case, $y_i(w^T x_i + b) \geq 1$, for all i.

In non-separable case, some i are not well classified, i.e., there is a tolerance $z_i \geq 0$ such that

$$y_i(w^T x_i + b) \geq 1 - z_i.$$

- If $z_i = 0$ then the instance i is correctly classified,
- If $z_i > 0$ then the instance i is misclassified.
- The larger z_i, the more the instance i is misclassified.

Soft margins

In order to reach a good classification, we try to reach a small sum of the z_i's

$$\sum_{i=1}^{n} z_i$$

The optimization problem is modified accordingly to

$$\begin{aligned} \min_{w,b,z} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{n} z_i \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 - z_i, \quad i = 1, \ldots, n \\ & z_i \geq 0, \quad i = 1, \ldots, n \end{aligned} \tag{1}$$

Soft margins

Parameter $C \geq 0$ is called the **cost** and is fixed by the user.

It is a way to control the tolerance to bad classification:

- If $C = 0$, then there is no penalty on the z_i's. These can be freely set. Thus, they can be large, and all the points can be misclassified.
- If C is large, then the optimization will try to reach that most z_i are be small. Thus, only little misclassification is allowed.

There is a trade-off between a large margin M (robustness) and a small misclassification (prediction quality).

Soft margin and support vector

With soft margins, the **support vectors** are those i such that

$$y_i(x_i^T w + b) = 1 - z_i.$$

- If $z_i = 0$, the support vectors lies exactly on the margin and thus are correctly classified.
- If $z_i > 0$, the support vector is away from the margin, on the wrong side, and is thus incorrectly classified.

Remember that, like in the perfectly separable case, any instance i is correctly classified if $z_i = 0$, that is,

$$y_i(x_i^T w + b) \geq 1.$$

Therefore, a support vector can be misclassified.

The dual problem

An equivalent way of writing the optimization problem (1) is in its dual form[1]

$$\begin{aligned} \max_{a} \quad & \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j y_i y_j x_i^T x_j \\ \text{s.t.} \quad & \sum_{i=1}^{n} a_i y_i = 0 \\ & 0 \leq a_i \leq C, \quad i = 1, \ldots, n \end{aligned} \tag{2}$$

Support vectors are i such that $a_i > 0$.

- If $a_i < C$ then they are on their margin (equiv. $z_i = 0$)
- If $a_i = C$ then they are inside the margin (equiv. $z_i > 0$).

[1] Again see your favorite Optimization course.

The prediction formula

If the dual is more difficult to interpret, it provides a nice prediction formula. The prediction of the class of a new instance x is based on a decision score (below θ contains all the parameters of the SVM)

$$d(x;\theta) = \sum_{i=1}^{n} a_i y_i x_i^T x.$$

The prediction then is

$$f(x;\theta) = \begin{cases} 1, & \text{if } d(x;\theta) > 0, \\ -1, & \text{if } d(x;\theta) < 0. \end{cases}$$

Kernel-based prediction

The decision has an interesting form:

$$d(x;\theta) = \sum_{i=1}^{n} a_i y_i x_i^T x.$$

It is a weighted sum of the y_i's in the training set. The weights are

- The support vectors, $0 < a_i \leq C$, enter into the sum with more or less weights. The other cases (non-support vectors), have $a_i = 0$ do not enter into the sum.
- The vectors such that $x_i^T x$ is large will participate more to the sum. This $x_i^T x$ is a measure of **proximity** (the larger, the more x_i and x are similar).

In summary, the prediction of y will be similar to the important support vectors y_i for which the features x_i are close to the new features x.

Kernel-based prediction

The proximity $x_i^T x$ is called a **kernel**. In general,

$$d(x;\theta) = \sum_{i=1}^{n} a_i y_i k(x_i, x).$$

The dual (2) is written

$$\begin{aligned} \max_{a} \quad & \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j y_i y_j k(x_i, x_j) \\ \text{s.t.} \quad & \sum_{i=1}^{n} a_i y_i = 0 \\ & 0 \leq a_i \leq C, \quad i = 1, \ldots, n \end{aligned}$$

Other kernels

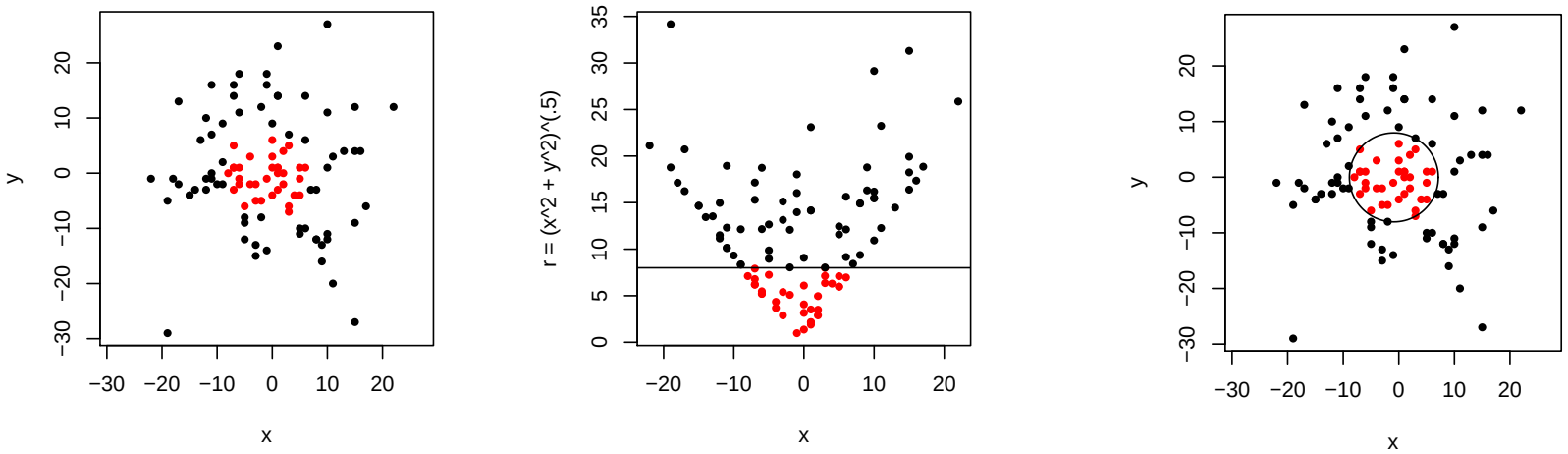
Usual kernels are

- Linear
$$k(x, x') = x^T x'$$
- Polynomial of degree q
$$k(x, x') = \left(k_0 + \gamma x^T x'\right)^q$$
- Radial basis
$$k(x, x') = \exp\left(-\gamma(x - x')^T(x - x')\right).$$
- Sigmoid
$$k(x, x') = \tanh\left(-\gamma x^T x' + k_0\right).$$

For each, the parameters k_0 and γ are set by the user.

Underlying features

Using a kernel other than $x^T x'$ can be shown to be equivalent to create new features $h(x)$. New features can help in separating the cases: e.g., below a non-separable case in (x, y) becomes separable by the new feature $(x^2 + y^2)$:



However, with the kernel, the exact form of these new features cannot be known.

Table of Contents

Interpretability

Support Vector Machine models are **not interpretable**.

Unlike regressions and trees, they do not provide any way to know the association between the features and the outcome.

Interpretability must therefore rely on generic methods (see later in the course).

Table of Contents

Model complexity

Like for regressions and trees, to apply the Occam's razor, we need to be able to control the model complexity. In the case of SVM, this is done by controlling the **cost** C. Reminder:

- If $C = 0$, then all the points can be misclassified. Therefore, **the model is simple**.
- If C is large, then only few misclassifications are allowed. Therefore, **the model is complex**.

The choice of C is made by the user and can be selected as a hyperparameter using data splitting strategies (see later in the course).

Table of Contents

SVM for multiclass

So far, we have seen only SVM for a **binary** classification. For the multiclass case ($L > 2$ classes), the principle remains the same except that several classifiers are built according to one of the following strategies:

- One versus the rest: for a new x, $d_\ell(x;\theta)$ is the decision function for class ℓ versus all the others. The final prediction is the class that has the highest decision value,

$$f(x;\theta) = \arg \max_{\ell=1,\dots,L} d_\ell(x;\theta).$$

- One versus one: for all the $\binom{L}{2}$ unique pair of classes (ℓ, ℓ'), build a SVM on the outcomes in these two classes. Then, the prediction is obtained by voting: the predicted class $f(x;\theta)$ is the one that has the highest number of decisions in its favor.

SVM for regression

This topic is outside the range of this course, but SVM can be adapted to regression with a similar property of robustness.

Similarly to the classification case, the final prediction is of the kernel form

$$f(x;\theta) = \sum_{i=1}^{n} a_i y_i k(x_i, x).$$