MACHINE LEARNING

Support Vector Machines, Kernels

Olivier LÉZORAY

olivier.lezoray@unicaen.fr

https://lezoray.users.greyc.fr/

LEARNING OBJECTIVES

- Risk minimization
- Support Vector Machines
- Soft-margin SVMs
- Kernels and Non-linear SVMs
- Multi-class SVMs

LEARNING OBJECTIVES

- **Risk minimization**
- Support Vector Machines
- Soft-margin SVMs
- Kernels and Non-linear SVMs
- Multi-class SVMs

RISK MINIMIZATION

- Consider the classical binary classification problem from data
  - We have a dataset $(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}_1, y_i), \cdots, (\mathbf{x}_N, y_N)\}$
  - with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{+1, -1\}$
  - The goal is to learn a function $y = f(\mathbf{x})$ that will correctly classify unseen examples
- **How do we find such a function ?**
  - By optimizing some measure of performance of the learned model
  - This is called **Risk Minimization**

EMPIRICAL RISK MINIMIZATION

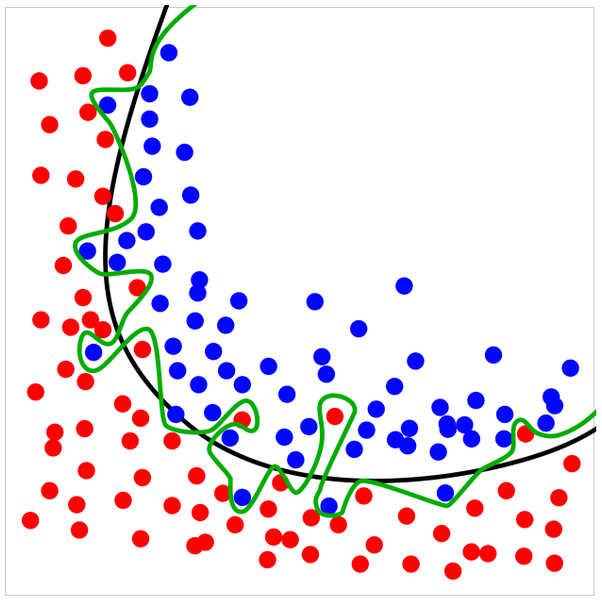
- A typical measure of performance consists in measuring the risk over the training set
- This is called the empirical risk and is given by

$$R_{emp} = \frac{1}{N} \sum_{i=1}^{N} L(f(\mathbf{x}_i), y_i)$$

- with a suitable loss function $L(f(\mathbf{x}_i), y_i) = (f(\mathbf{x}_i) - y_i)^2$
- Empirical risk minimization : find the function *f* that minimizes the average risk over the training set
- This works well only if sufficient training data is provided : the empirical risk converges to the true risk as $N \to \infty$

EMPIRICAL RISK MINIMIZATION : ISSUES

- For small datasets, one **cannot guarantee** that the ERM will also minimize the true risk
- The model will have poor generalization abilities in this case
- This is related to the concept of **overfitting**
- An overfitted model is a statistical model that contains more parameters than the number of data samples
- This is the famous **bias/variance dilemma**
  - biais : prediction error
  - variance : sensibility of the model to the data
  - Overfitting : low biais and high variance
- How to control overfitting ?
  - **prefer the simplest model that explains the data (Occam's razor)**



VC DIMENSION

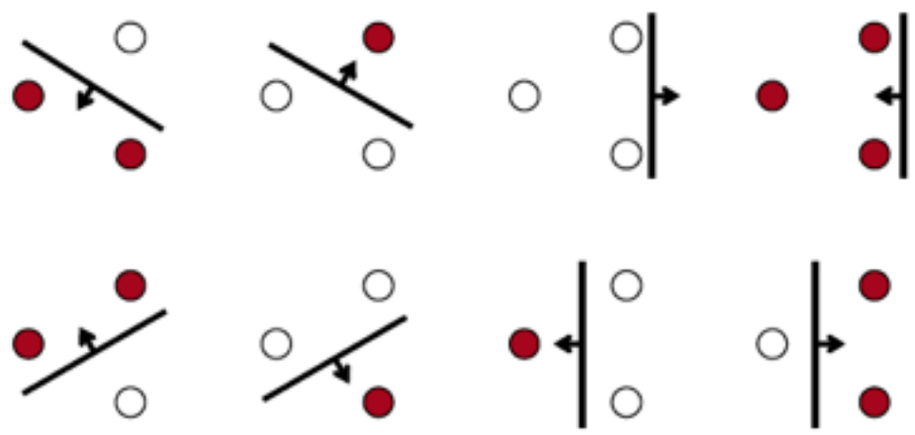
- **The Vapnik-Chervonenkis dimension is a measure of the complexity (or capacity) of a class of functions $f(\alpha)$**
  - The VC dimension measures the largest number of examples that can be explained by the family $f(\alpha)$
- **The basic argument is that high capacity and generalization properties are opposed**
  - If the family has enough capacity to explain every possible dataset, we should not expect these functions to generalize very well
  - If the functions have small capacity but they are able to explain our training dataset, we can believe they will also work well on unseen data (good generalization)

VC-DIMENSION

- Partitioning of a set of examples
  - Assume a binary classification problem with N examples in $\mathbb{R}^d$
  - There are 2^d possible dichotomies
    - For instance with $N=3$ examples, the set of all possible dichotomies is
      - {(000),(001),(010),(011),(100),(101),(110),(111)}
  - A class of functions $f(\alpha)$ is said as partitioning the dataset if for every possible dichotomy, there is a function in $f(\alpha)$ that models it
- The VC-dimension *VC(f)* is the largest dataset that can be partitioned by the set of functions $f(\alpha)$

VC-DIMENSION

- If the VC-dimension is *h*, then there exists at least one set of *h* points that can be partitioned by $f(\alpha)$
- Example : consider a binary classification problem in $\mathbb{R}^2$ and let $f(\alpha)$ be the family of oriented hyperplans (e.g., a perceptron)
- For *N=3*, one can perform a linear separation of all points for every possible class assignment
- The VC-dimension of the set of oriented hyperplans in $\mathbb{R}^2$ is 3



This does not mean that any set of 3 points can be partitioned by a linear classifier such as an hyperplan !

- It can be shown that the VC-dimension of the family of oriented hyperplanes in $\mathbb{R}^d$ is *d+1*

INTEREST OF THE VC-DIMENSION

- The VC-dimension is a more sophisticated measure of model complexity than the number of parameters
- The VC-dimension provides bounds on the expected risk as a function of the empirical risk and the number of available examples
- It can be shown that the following bound holds with probability $1 - \eta$

VC-confidence

$$R(f) \leq R_{emp}(f) + \sqrt{\frac{h\left(ln(\frac{2N}{h}) + 1\right) - ln(\frac{\eta}{4})}{N}}$$

- h: VC-dimension, N: number of examples

INTEREST OF VC-DIMENSION

$$R(f) \leq R_{emp}(f) + \sqrt{\frac{h\left(ln(\frac{2N}{h}) + 1\right) - ln(\frac{\eta}{4})}{N}}$$

- As the ratio *N/h* gets larger, the VC confidence becomes smaller and the actual risk becomes closer to the empirical risk
- This confirms that the Empirical Risk Minimization is suitable only when sufficient data is available
  - If you want to use ERM anyway : use cross-validation to better estimate it
- **This result is part of the statistical learning theory of Vapnik and Chervonenkis, from which Support Vector Machines originated**

STRUCTURAL RISK MINIMIZATION

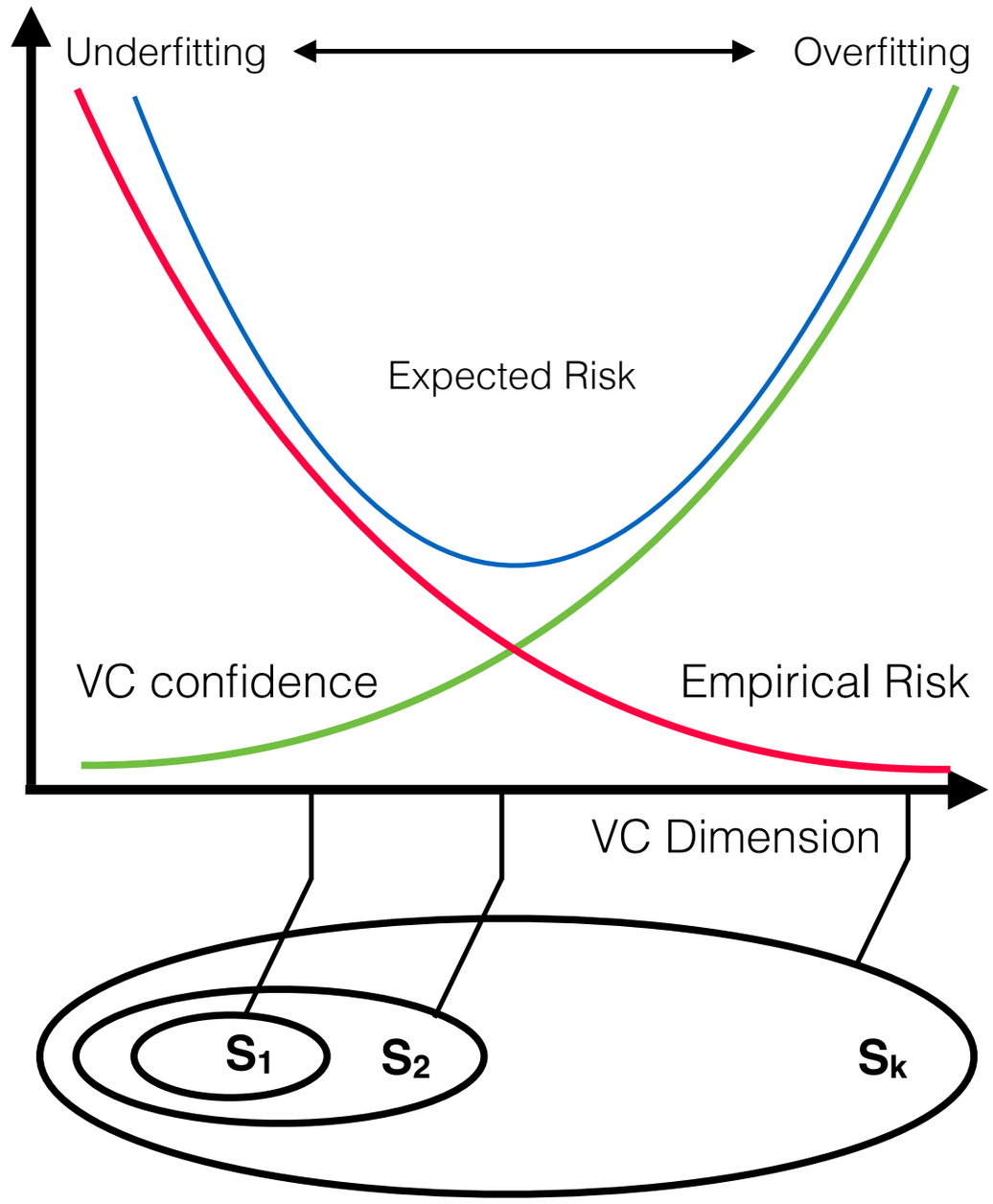
- **The optimal model is found by finding a balance between the empirical risk and the VC confidence**
- The SRM principle proceeds as follows:
  - Construct a nested structure for a family of functions classes of increasing VC-dimensions

$$F_1 \subset F_2 \subset \cdots F_k \quad \textit{and} \quad (h_1 \leq h_2 \leq \cdots h_k)$$

  - For each class F_i find the solution f_i that minimizes the empirical risk
  - Choose the function class F_i and the corresponding solution f_i that minimizes the risk bound

STRUCTURAL RISK MINIMIZATION

- Choose a set of machines, one for each subset
- For a given subset, train to minimize the empirical risk
- Choose the machine that has the lowest sum of empirical risk and VC-confidence



THE VC-DIMENSION IN PRACTICE

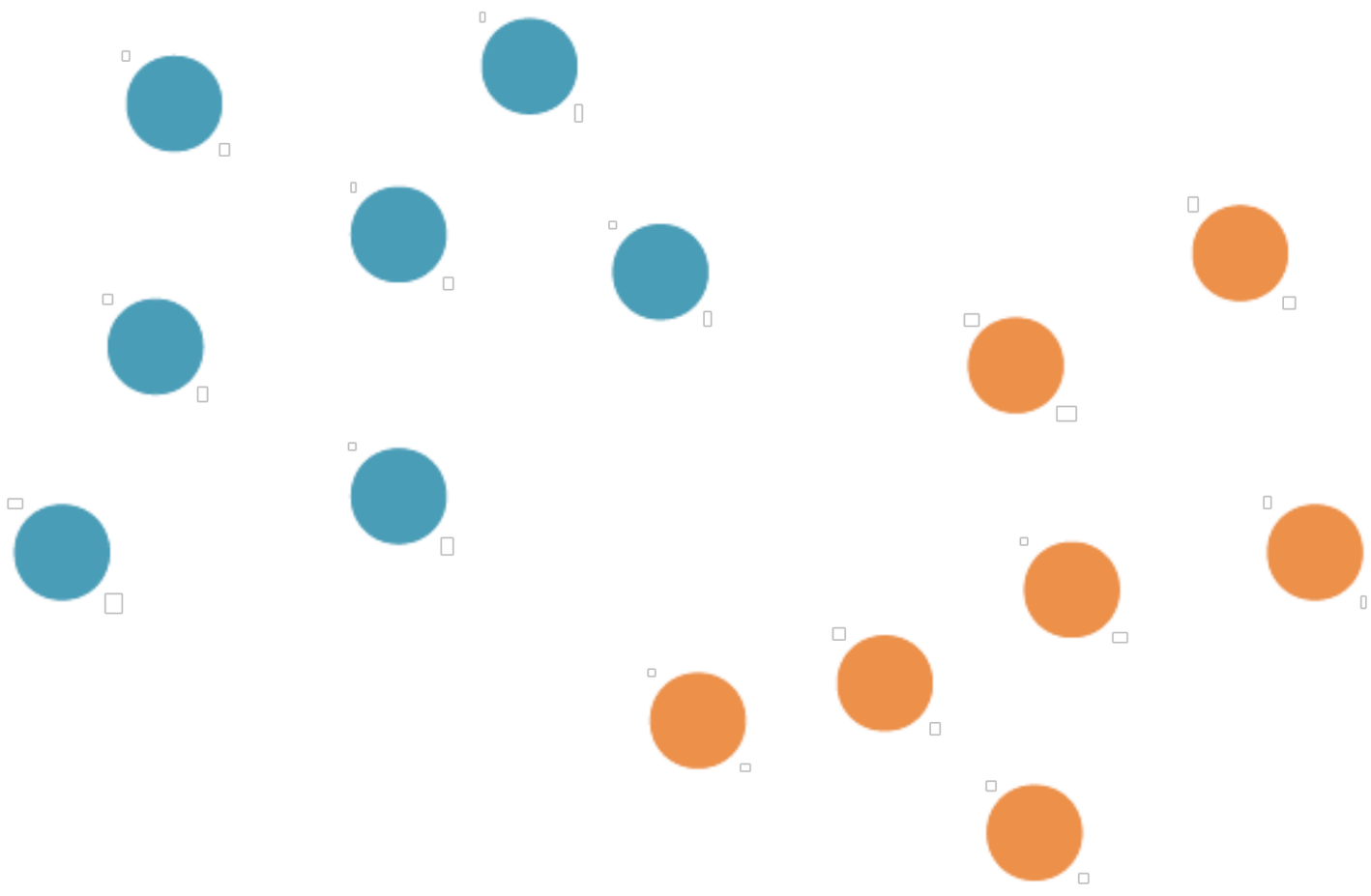
- **Having an upper bound of the VC-dimension is not always useful**
  - The supplied bound may be too large
  - VC-dimension cannot be easily estimated for non-linear models such as neural networks
  - Seeking to directly minimize the structural risk leads to a difficult optimization problem
  - The VC-dimension can be infinite (e.g., for k=1 nearest neighbors) or require a very large number of examples
- **Fortunately this is useful for linear separation models with the help of statistical learning theory**

LEARNING OBJECTIVES

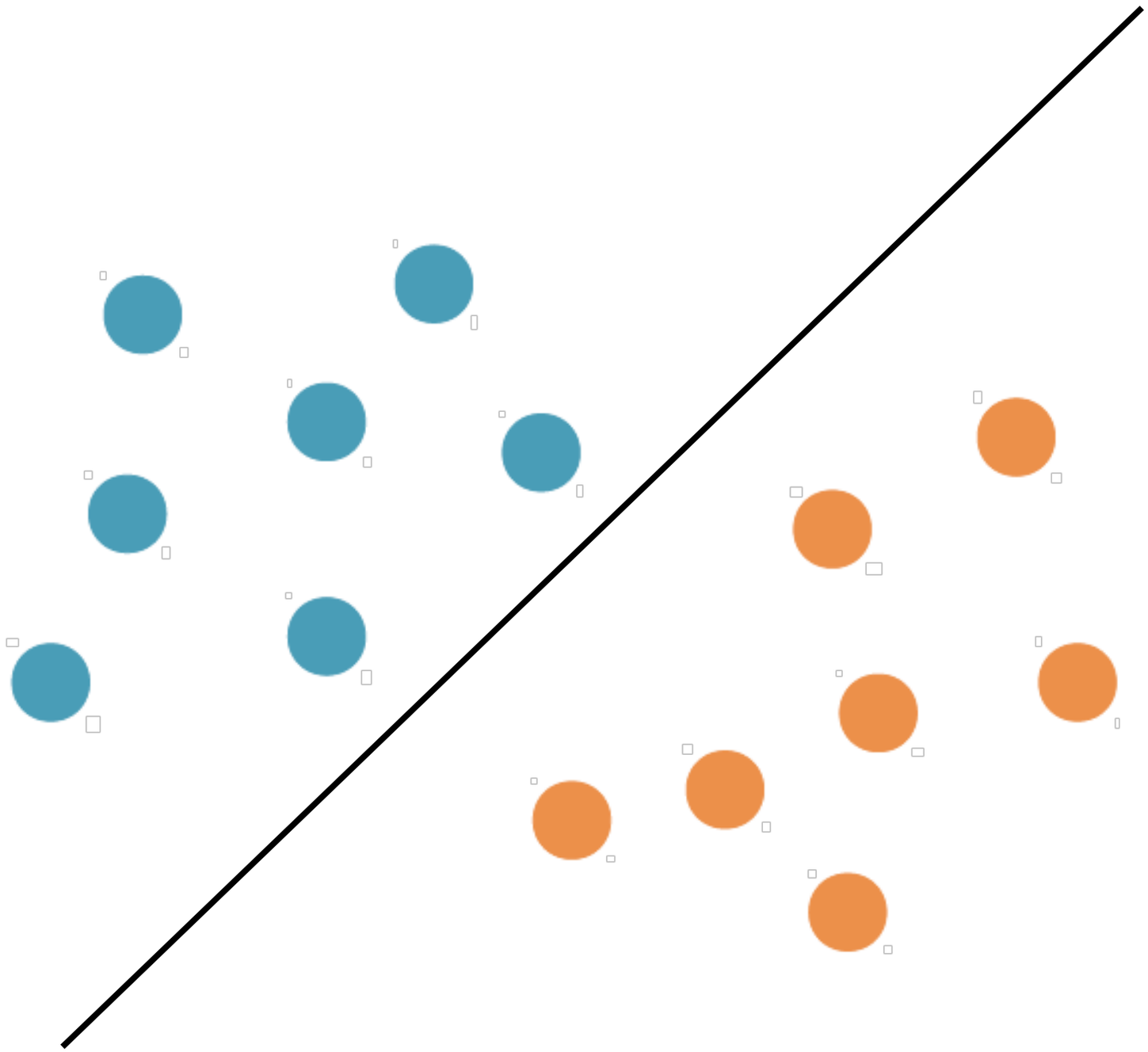
- Risk minimization
- **Support Vector Machines**
- Soft-margin SVMs
- Kernels
- Non-linear SVMs
- Multi-class SVMs

LINEAR CLASSIFIER

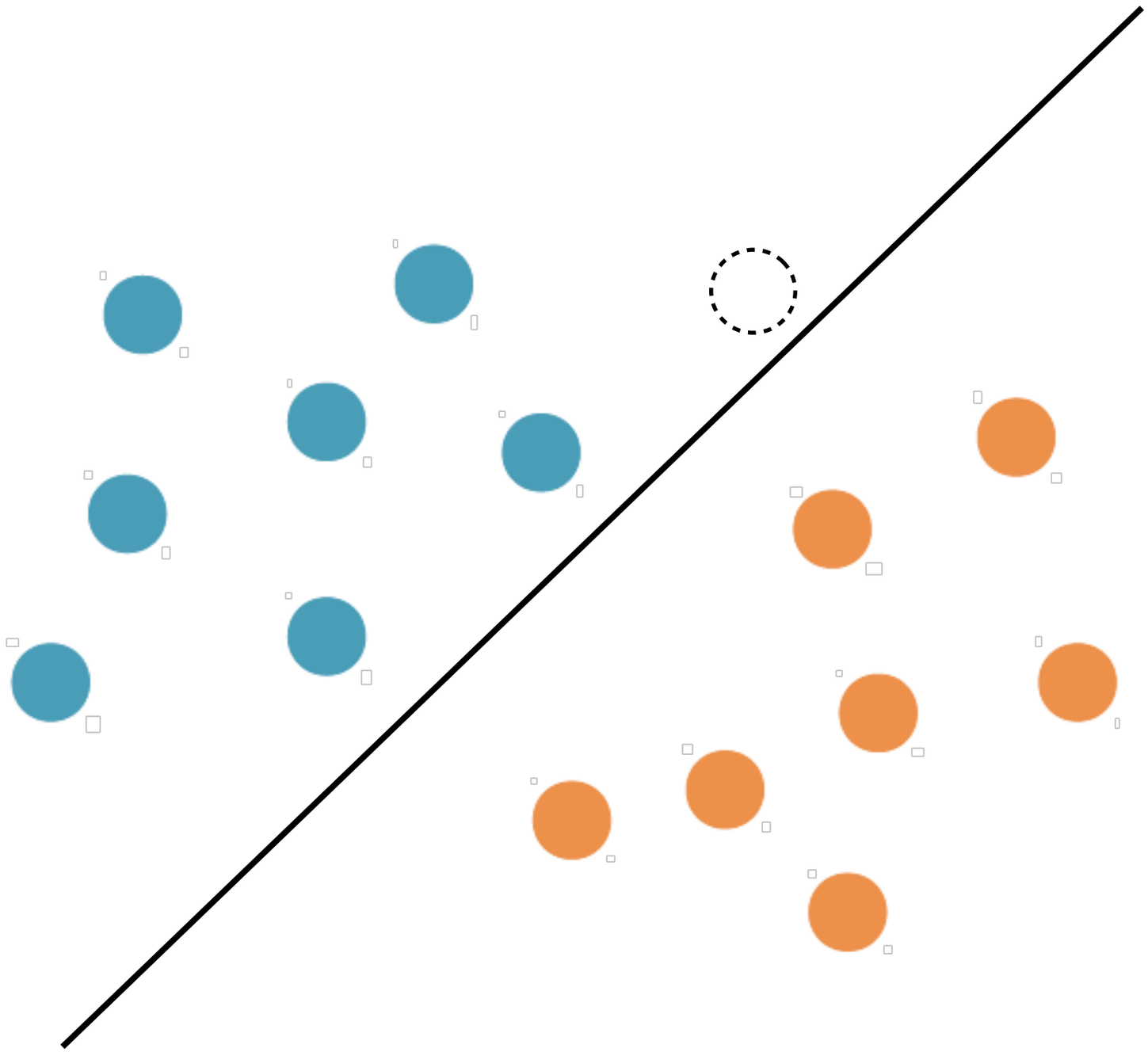
- We assume that the data is **linearly separable** : the exists an oriented hyperplane (a line in 2D) that separates the two classes



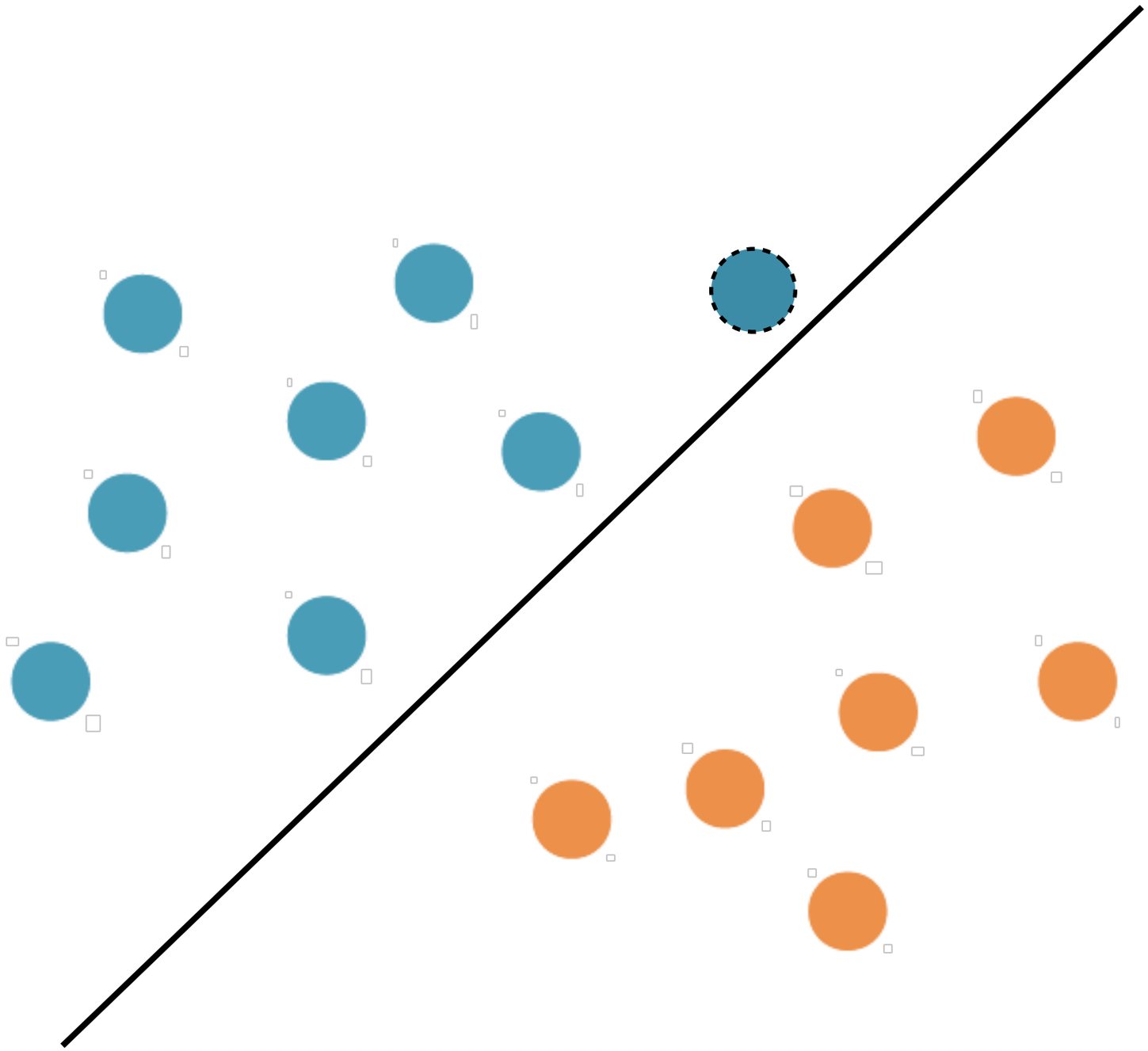
LINEAR CLASSIFIER



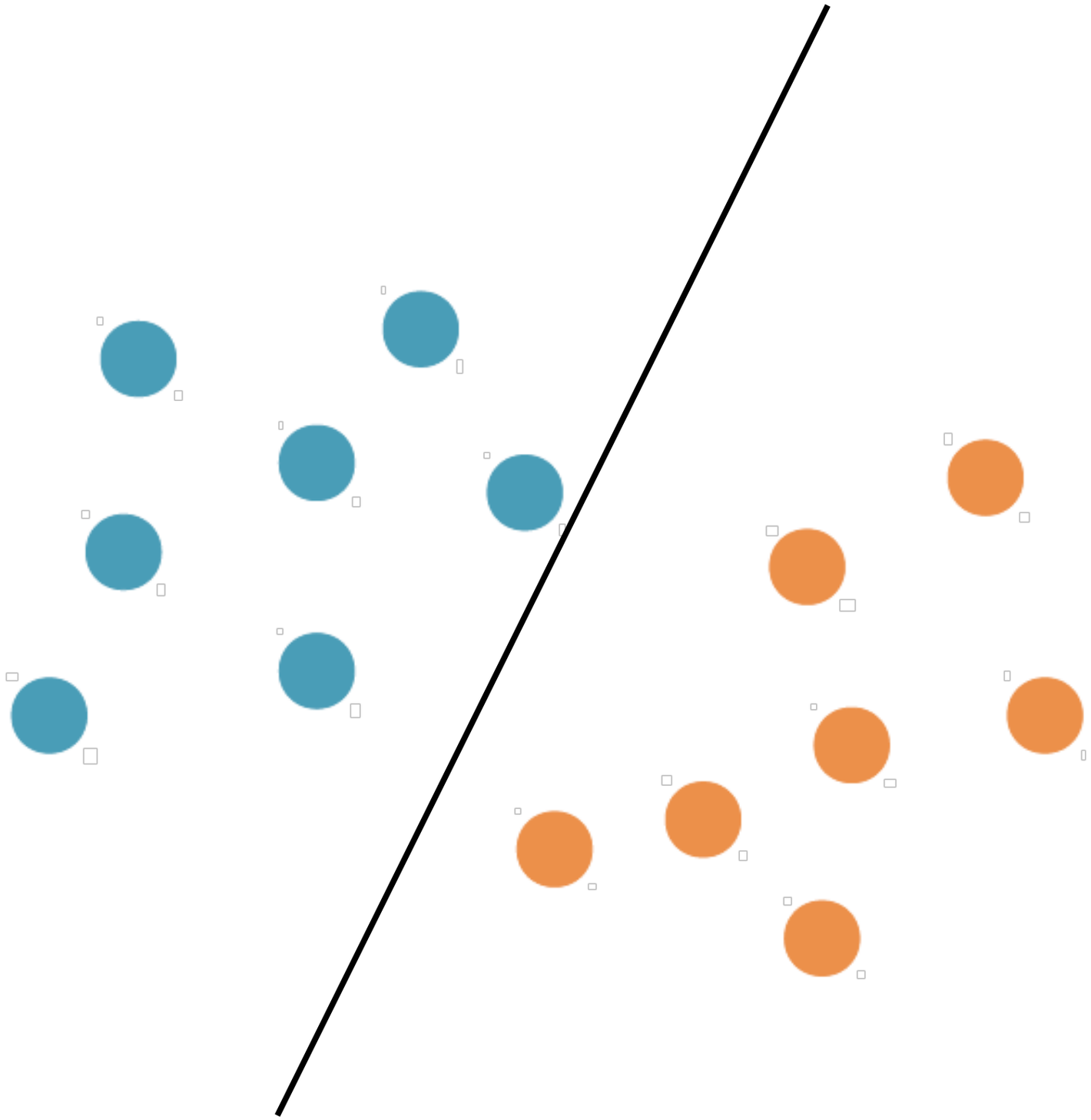
LINEAR CLASSIFIER



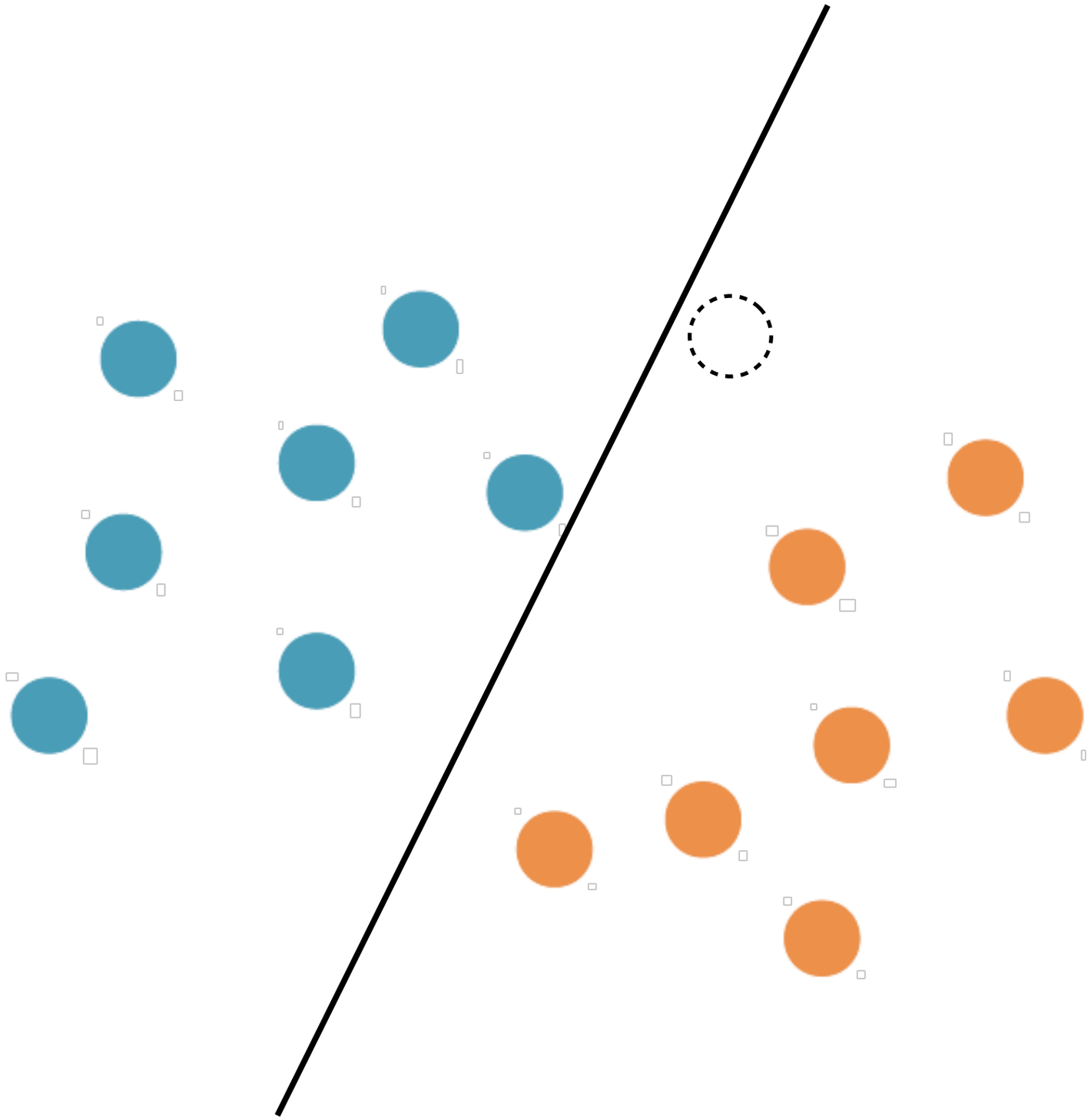
LINEAR CLASSIFIER



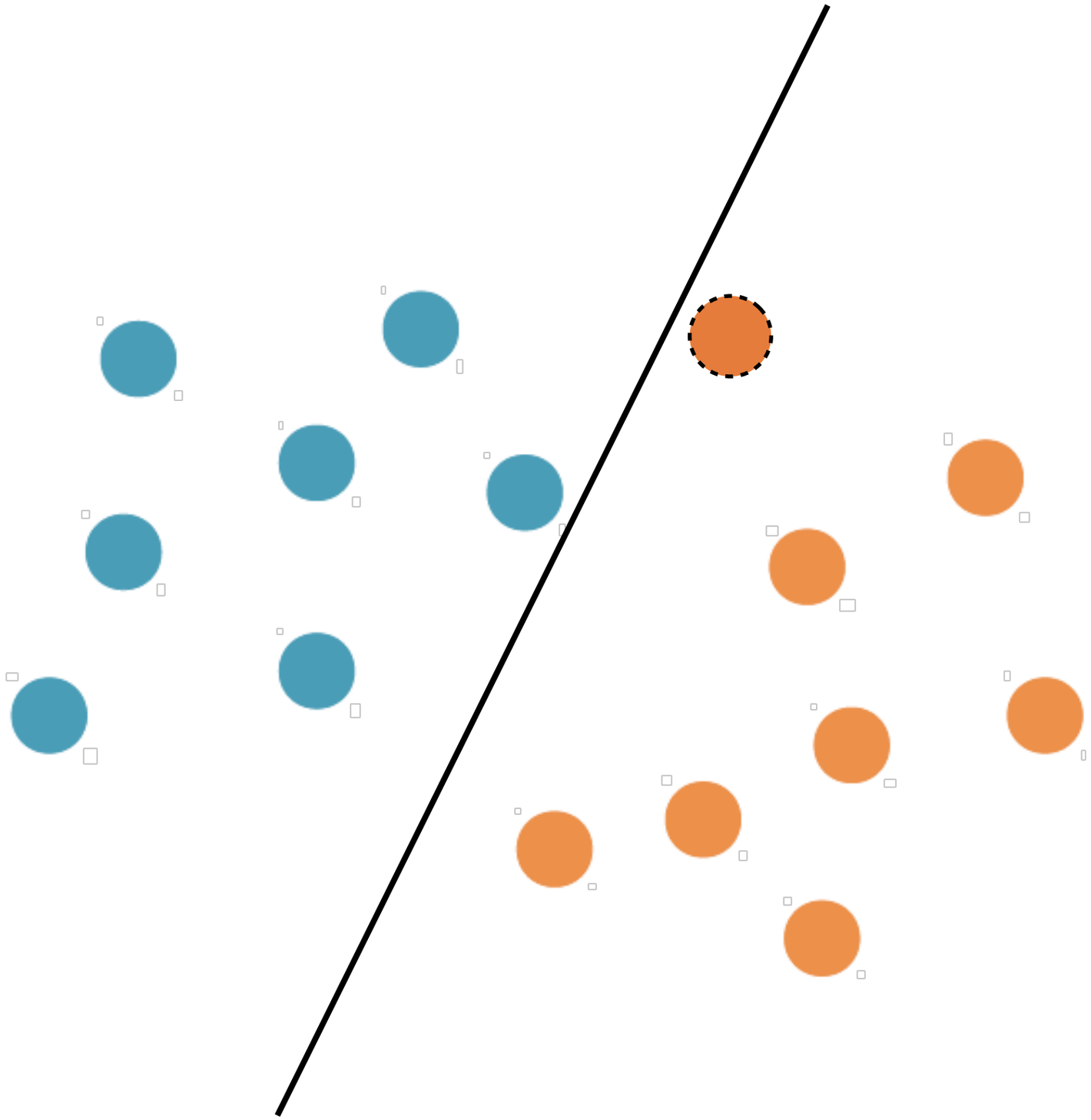
LINEAR CLASSIFIER



LINEAR CLASSIFIER

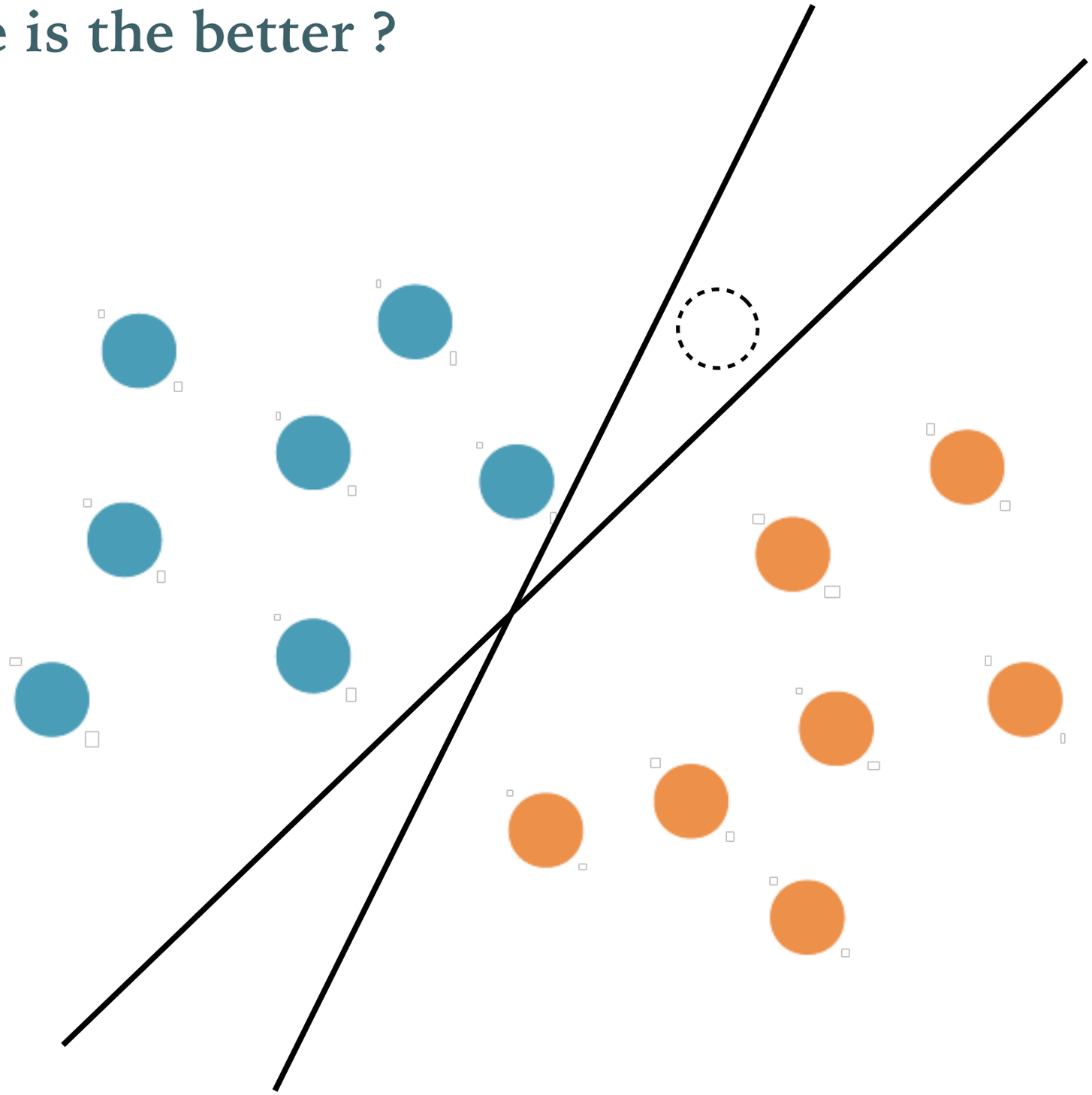


LINEAR CLASSIFIER



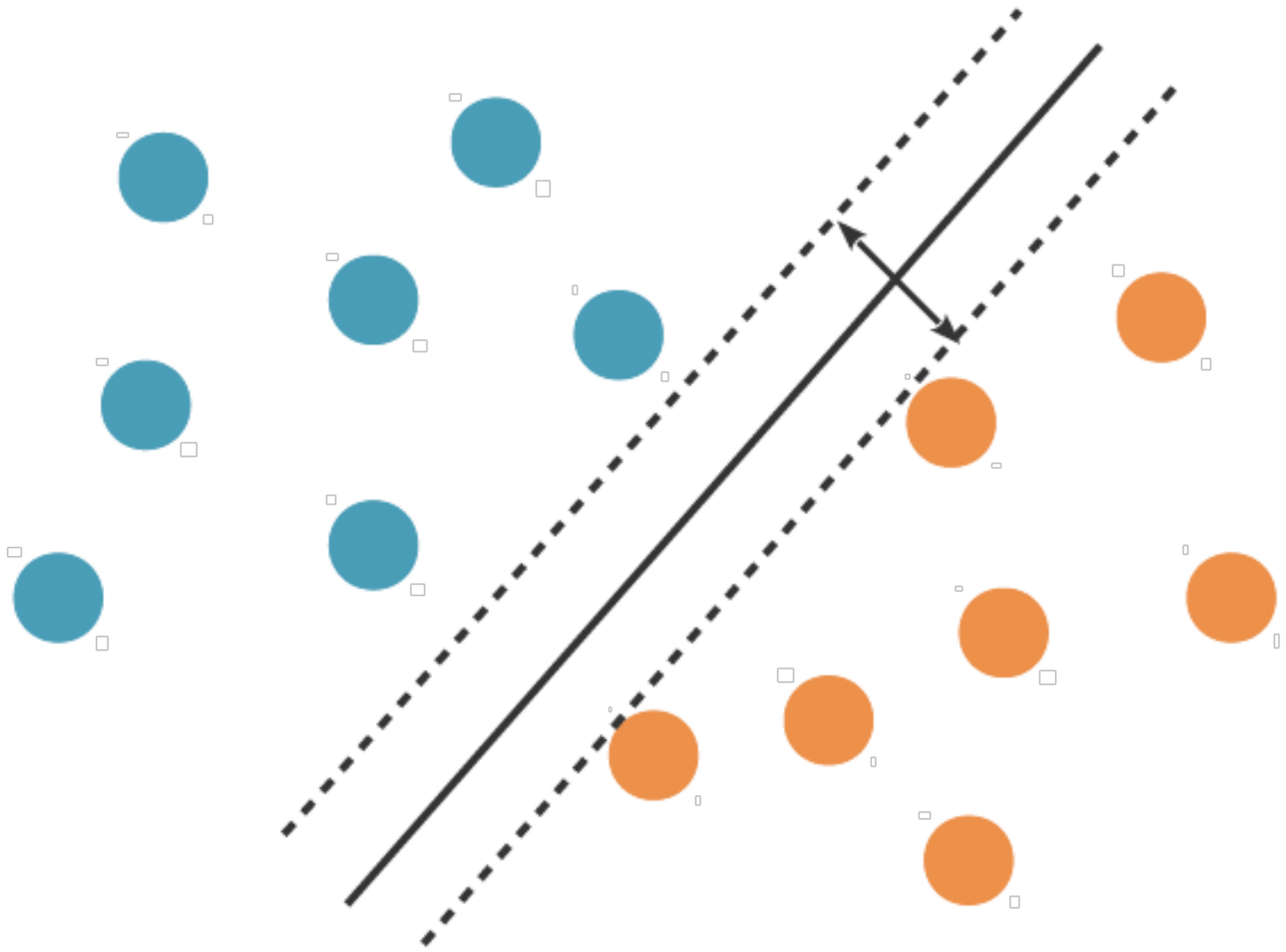
LINEAR CLASSIFIER

- Which one is the better ?

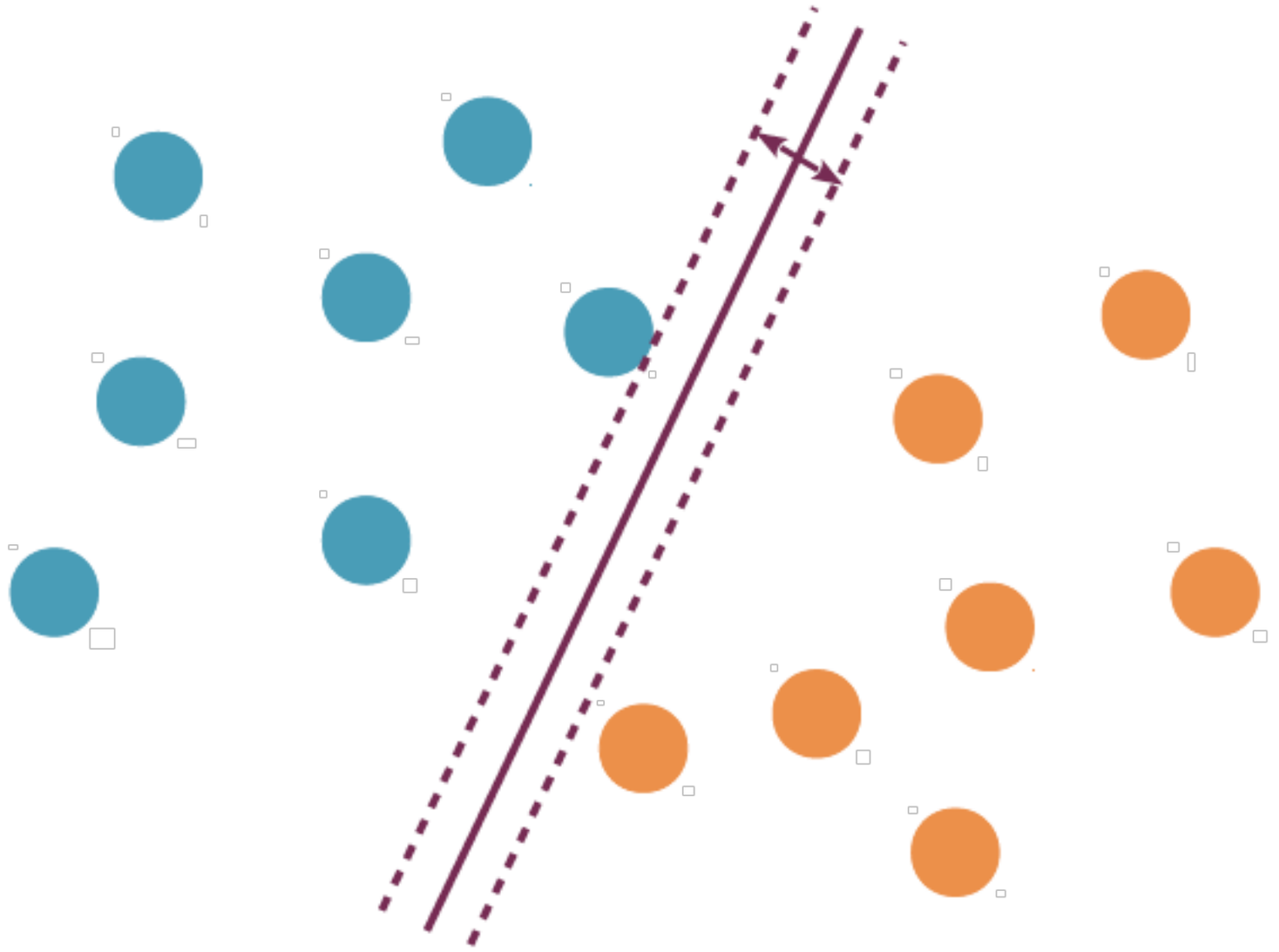


MARGIN OF A LINEAR CLASSIFIER

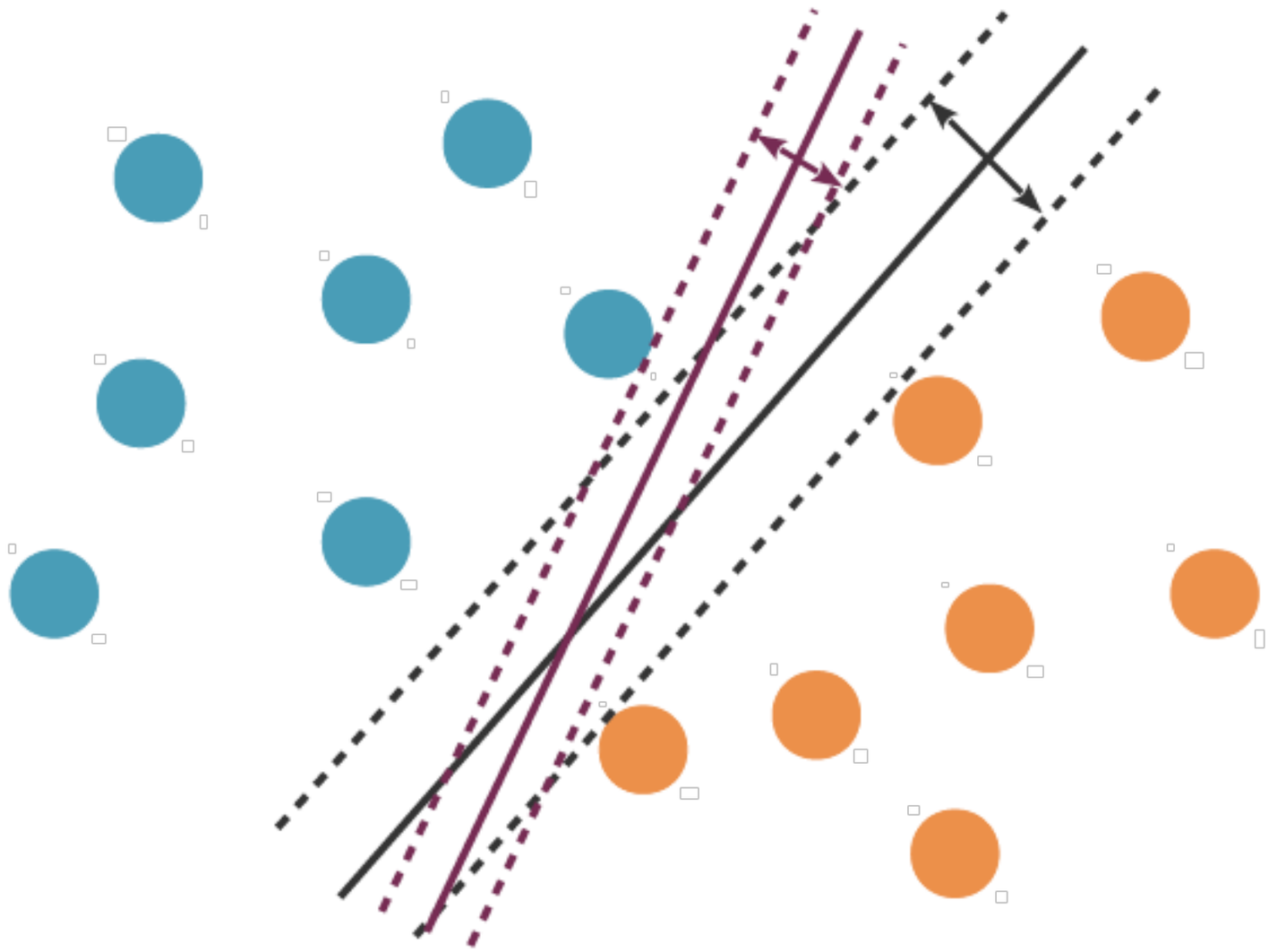
- Margin: twice the distance from the separating hyperplane to the closest training point



MARGIN OF A LINEAR CLASSIFIER

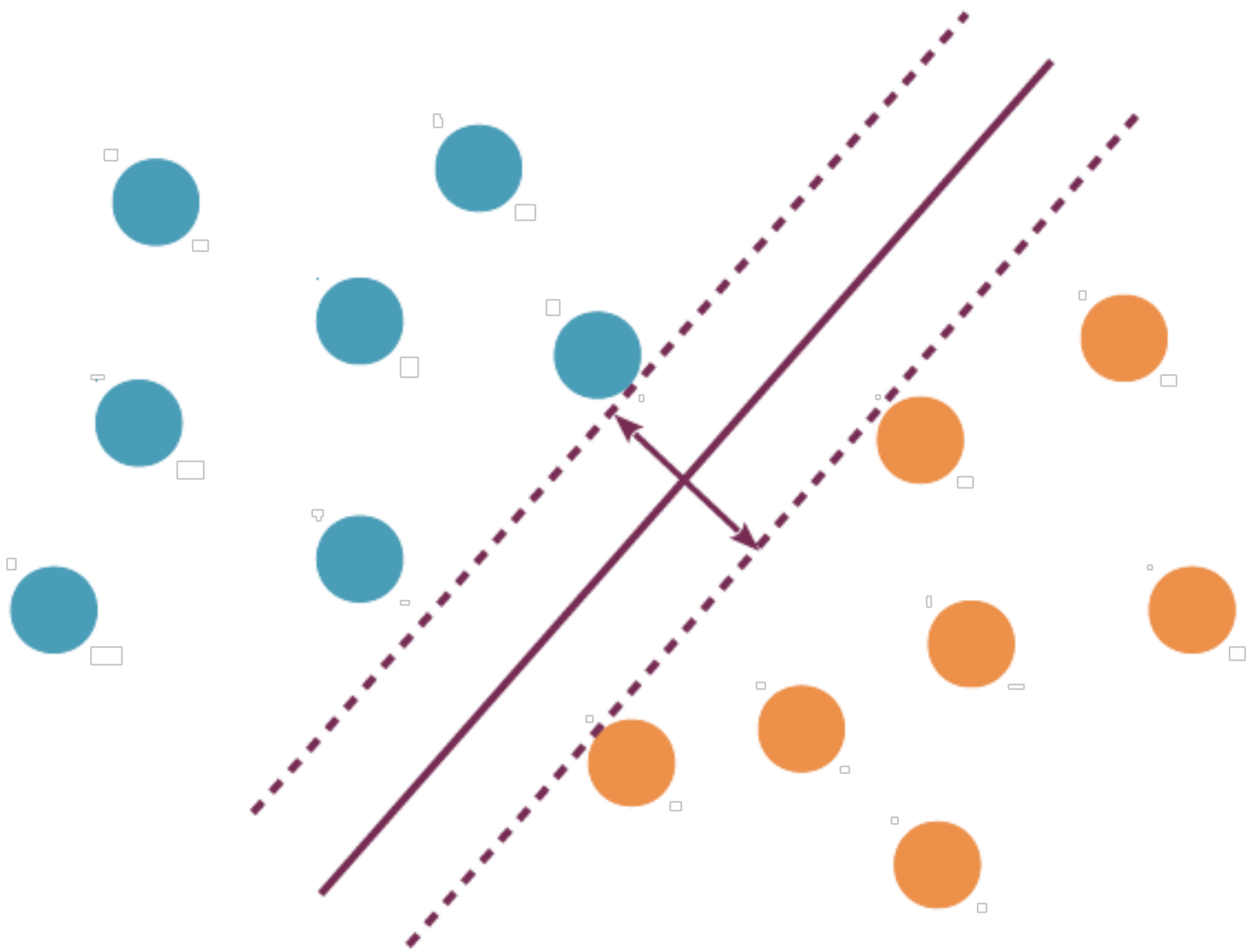


MARGIN OF A LINEAR CLASSIFIER

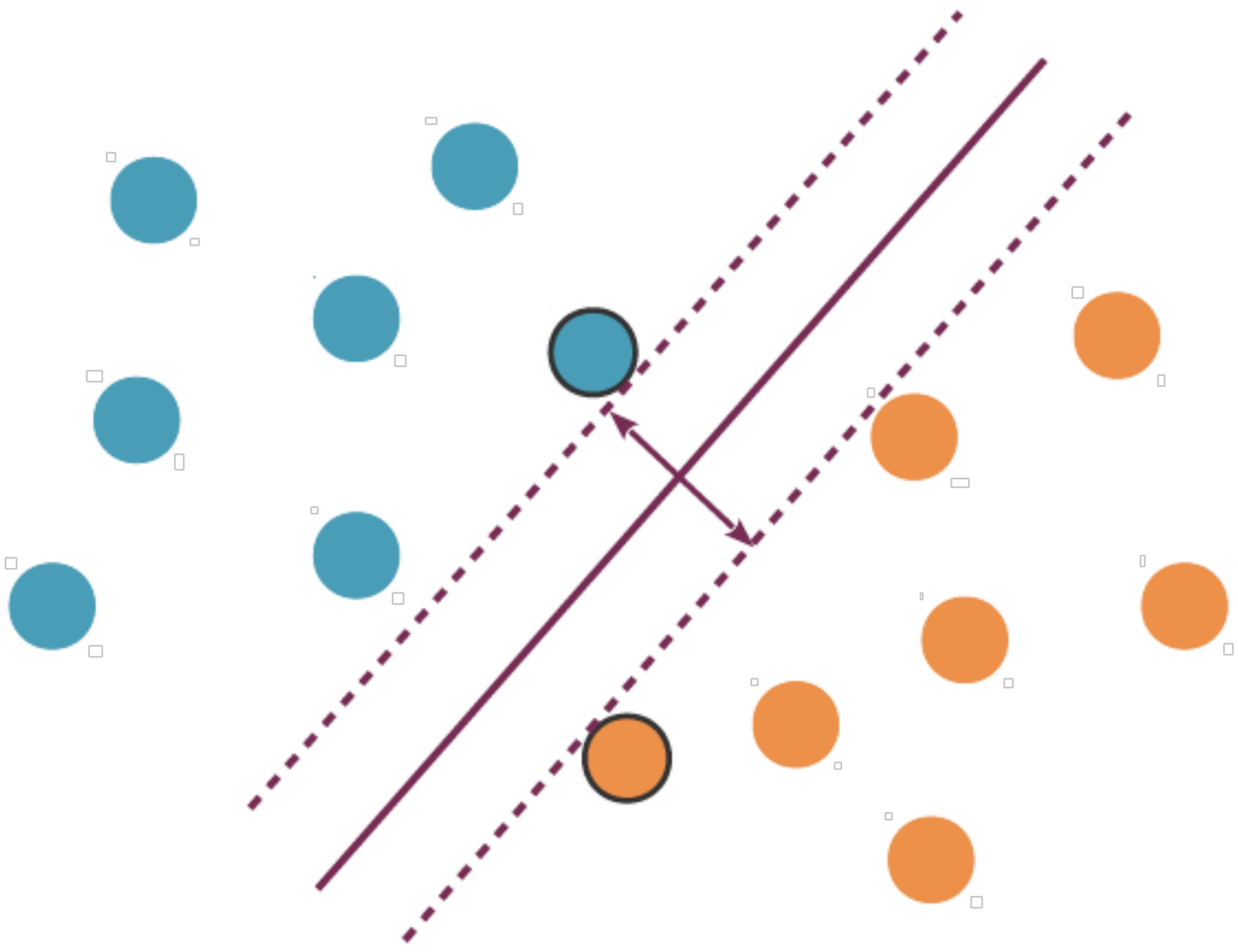


SUPPORT VECTOR MACHINES

- The largest margin classifier is called a **Support Vector Machine**

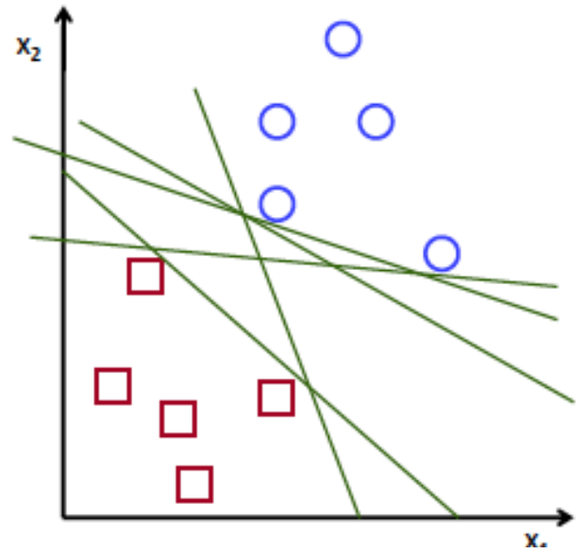


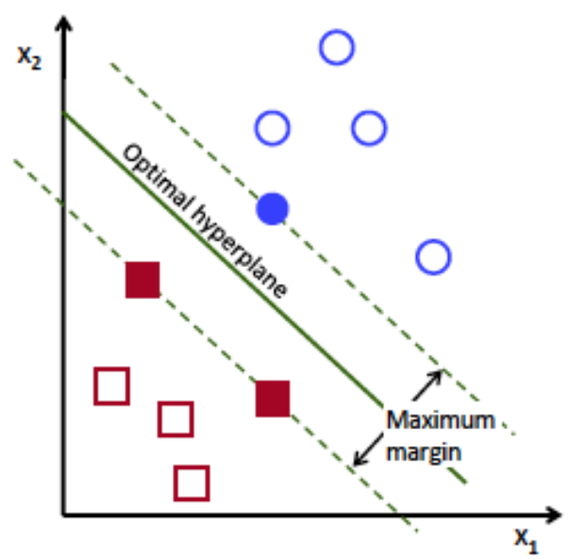
SUPPORT VECTORS



OPTIMAL SEPARATING HYPERPLANES

- Given a training dataset $(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}_1, y_i), \cdots, (\mathbf{x}_N, y_N)\}$
- With $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{+1, -1\}$
- **Find a separating hyperplane : which one should we choose ?**
  - An hyperplane passing too close to the training examples will be sensitive to noise and will not well generalize
  - An hyperplane that is far from the training examples will have better generalization capabilities
- **The optimal separating hyperplane is the one with the largest margin** (the minimum distance of an example to the hyperplane)





RELATION WITH THE VC-DIMENSION

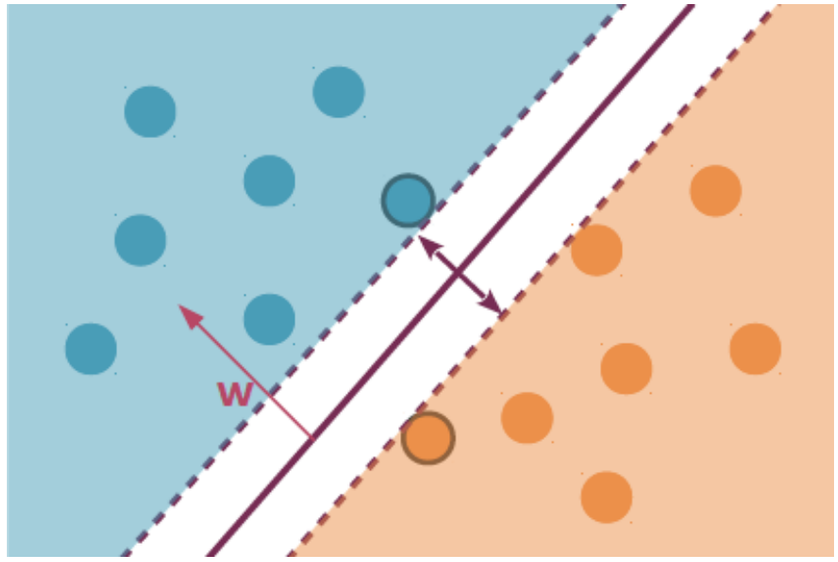
- One can show that the VC-dimension of a separating hyperplane with a margin m is bounded by

$$h \leq \min\left(\left\lceil \frac{R^2}{m^2} \right\rceil, d\right) + 1$$

- with R the radius of the smallest sphere containing all the examples
- **Maximizing the margin m minimizes the VC-dimension**
- Since the separating hyperplane has zero empirical risk (perfect classification), it also minimizes the bound on the expected risk
- **The separating hyperplane with maximum margin will also minimize the structural risk !**

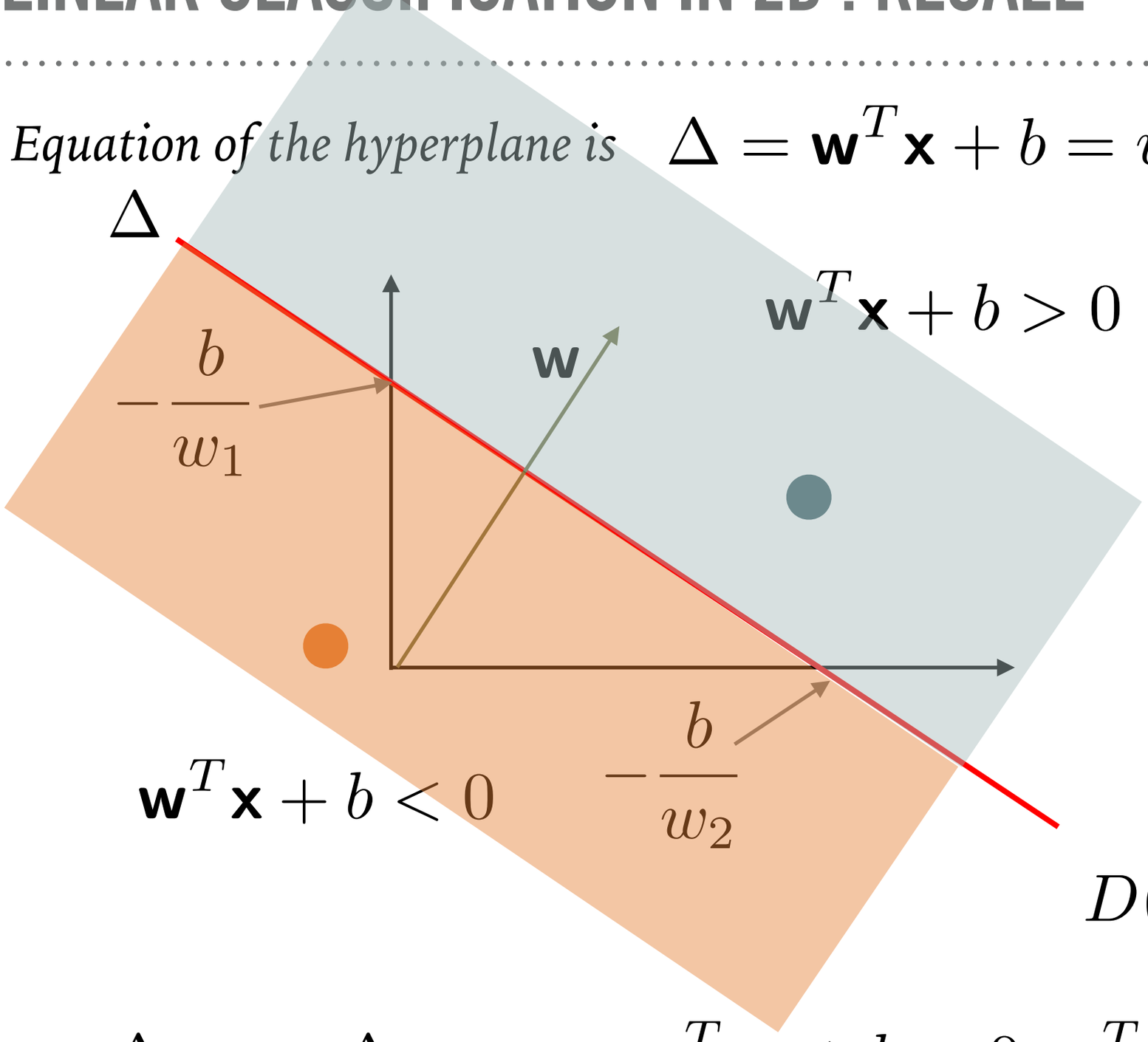
FORMALIZATION

- Given a training dataset $(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}_1, y_i), \cdots, (\mathbf{x}_N, y_N)\}$
- With $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{+1, -1\}$
- We seek $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = < \mathbf{w}, \mathbf{x} > + b$
- **What are the equations of the 3 parallel hyperplanes ?**
- **How are defined the two classification regions ?**
  - The sign of $f(\mathbf{x})$ will give the chosen class



LINEAR CLASSIFICATION IN 2D : RECALL

Equation of the hyperplane is $\Delta = \mathbf{w}^T\mathbf{x} + b = w_1x_1 + w_2x_2 + b = 0$



$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$D(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$$

$$\mathbf{x}_1 \in \Delta, \mathbf{x}_2 \in \Delta \quad \Leftrightarrow \quad \mathbf{w}^T\mathbf{x}_1 + b = 0, \mathbf{w}^T\mathbf{x}_2 + b = 0$$

$$\mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_2) = 0 \Leftrightarrow \quad \mathbf{w} \quad \textit{is orthogonal to} \quad \Delta$$

MAXIMIZE THE MARGIN

- Given the hyperplane decision $\Delta = \mathbf{w}^T\mathbf{x} + b$
- Maximize the margin:

$$\max_{\mathbf{w},b} \quad \min_{i=1}^{N} \quad dist(\mathbf{x}_i, \Delta)$$

The margin

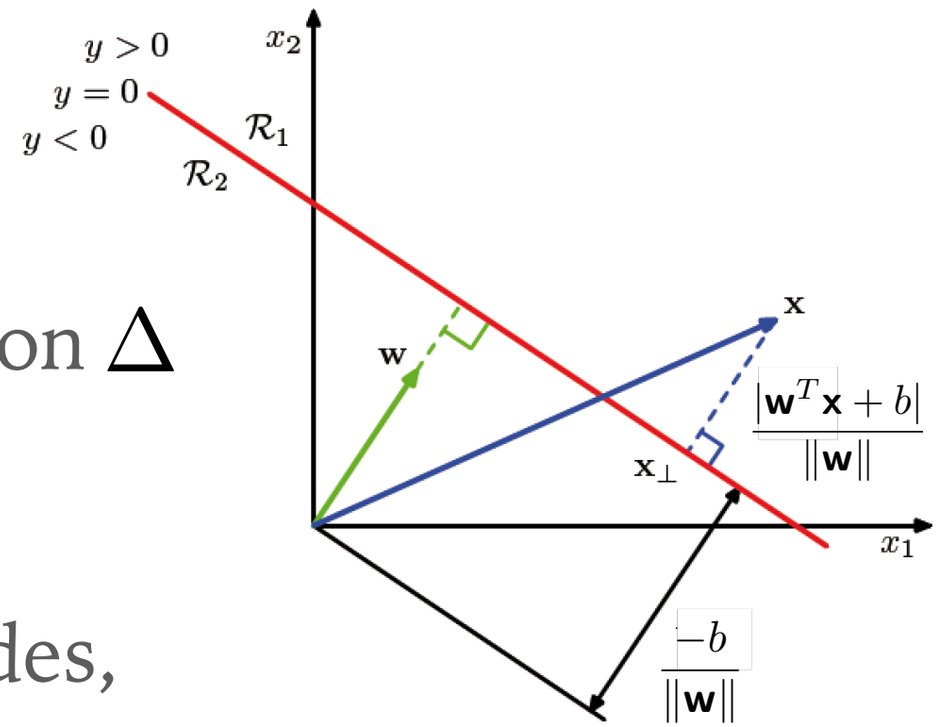
- One has $dist(\mathbf{x}_i, \Delta) = \arg\min_{p \in \Delta} \|\mathbf{x}_i - \mathbf{p}\|$
- One can show that $\arg\min_{p \in \Delta} \|\mathbf{x}_i - \mathbf{p}\| = \dfrac{|\mathbf{w}^T\mathbf{x}_i + b|}{\|\mathbf{w}\|}$

DISTANCE TO THE HYPERPLANE

- w is orthogonal to the decision hyperplane: $\mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_2) = 0$
- If a point p is on Δ then $f(\mathbf{p}) = \mathbf{w}^T\mathbf{p} + b = 0$

  and the normal distance from the origin to Δ is

$$\frac{\mathbf{w}^T\mathbf{p}}{\|\mathbf{w}\|} = \frac{-b}{\|\mathbf{w}\|}$$



- Let $\mathbf{x}_\perp$ be the orthogonal projection of $\mathbf{x}$ on Δ

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$$

- We multiply by $\mathbf{w}$ and add b on both sides, since $\mathbf{w}^T\mathbf{x}_\perp + b = 0$, we have

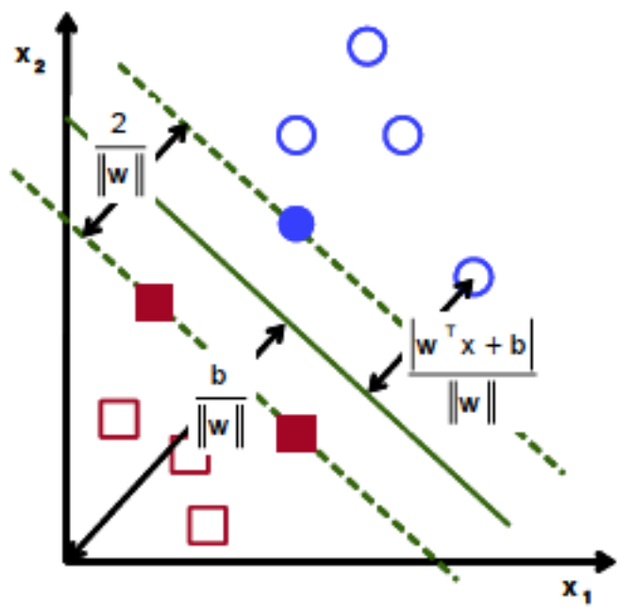
$$r = \frac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|}$$

THE MARGIN

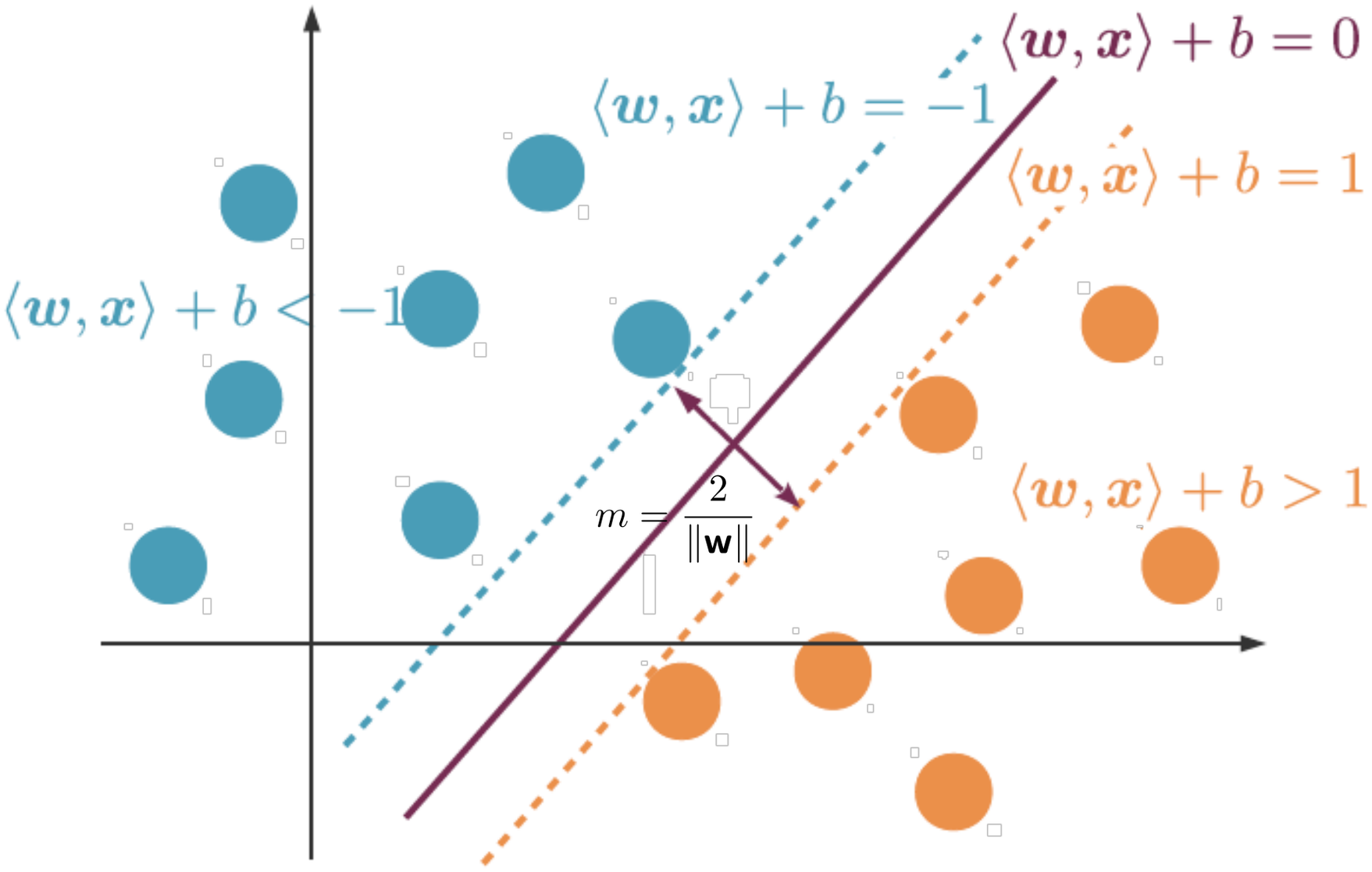
- Since we want to maximize the margin, use its expression as a function of the hyperplan (weight w and biais b)
- The distance between a point x and a plane parameterized by (w,b) is given by $\frac{|\mathbf{w}^T\mathbf{x} + b|}{\|\mathbf{w}\|}$
- Since there is an infinity of solutions (by scaling (w, b)), we choose the one that is 1 for examples close to the boundary

$$|\mathbf{w}^T\mathbf{x}_i + b| = 1$$

- The distance of the closest point to the boundary is $\frac{|\mathbf{w}^T\mathbf{x} + b|}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|}$
- and the margin is $m = \frac{2}{\|\mathbf{w}\|}$



LARGEST MARGIN HYPERPLANE : RECAP



An example is well classified if $y_i(< \mathbf{w}, \mathbf{x}_i > +b) \geq 1$

and NOT $y_i(< \mathbf{w}, \mathbf{x}_i > +b) \geq 0$ *: no point allowed in the margin !*

OPTIMIZATION PROBLEM

- Given a training dataset $(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}_1, y_i), \cdots, (\mathbf{x}_N, y_N)\}$
- With $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{+1, -1\}$
- Assume the data is linearly separable

$$\exists(\mathbf{w}, b) \in \mathbb{R}^d \times \mathbb{R} \quad \textit{such that} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$

- Goal : find *(w*,b*)* that define the hyperplane of largest margin
- To do so, we add a constraint on the margin by incorporating the distance to the hyperplane : we want *(w*,b*)* that maximize this distance

$$\frac{y_i(\mathbf{w}^T\mathbf{x}_i + b)}{\|\mathbf{w}\|}$$

OPTIMIZATION PROBLEM

- General maximum margin solution

$$\arg\max_{\mathbf{w},b}\left\{\frac{1}{\|\mathbf{w}\|}\min_{n}\left[y_i(\mathbf{w}^T\mathbf{x}_i+b)\right]\right\}$$

- which is rewritten as (no points inside the margin)

$$\arg\max_{\mathbf{w},b}\left\{\frac{1}{\|\mathbf{w}\|}\right\} \quad s.t. \quad y_i(\mathbf{w}^T\mathbf{x}_i+b)\geq 1$$

- which is rewritten as

$$\arg\min_{\mathbf{w},b}\left\{\frac{1}{2}\|\mathbf{w}\|^2\right\} \quad s.t. \quad y_i(\mathbf{w}^T\mathbf{x}_i+b)\geq 1$$

OPTIMIZATION PROBLEM

$$\arg\min_{\mathbf{w},b}\left\{\frac{1}{2}\|\mathbf{w}\|^2\right\} \quad s.t. \quad y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$$

- This is a Quadratic Programming (QP) problem
- The objective is to minimize a quadratic function (that has a single global minimum) subject to linear constraints
- To solve this problem, we will use the Lagrangian optimization technique
- This yields an unconstrained optimization problem that introduces Lagrange multipliers $\alpha_i \geq 0$ for each example

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N}\alpha_i\left[y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\right]$$

LAGRANGIAN OPTIMIZATION

- Given an optimization problem

$$\begin{aligned} \text{Minimize} \quad & f(x) \\ \text{Subject to} \quad & g_i(x) \leq 0 \\ & h_i(x) = 0 \end{aligned}$$

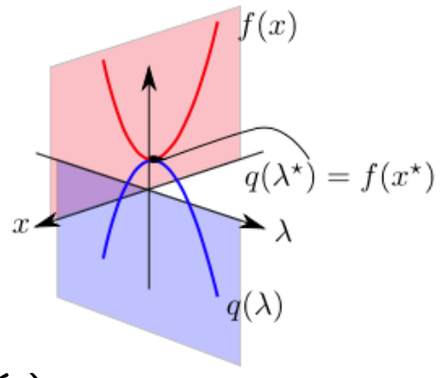
- We define the Lagrangian as

$$L(x, \alpha, \beta) = f(x) + \sum_{i=1}^{k} \alpha_i g_i(x) + \sum_{i=1}^{m} \beta_i h_i(x)$$

- that incorporates the inequality and equality constraints with the (α_i, β_i) called the Lagrange Multipliers

LAGRANGIAN OPTIMIZATION

- If (α^*, β^*) are solution of the dual Lagrangian problem $\max_{\alpha,\beta} L(x, \alpha, \beta)$
- Then the minimizer x^* of $L(x, \alpha^*, \beta^*)$ is the solution of the **primal Lagrange problem**
- The Slater condition from convex optimization guarantees that the solution from the dual problem is the same as the solution of the primal if f, g, h are convex and continuously differentiable
- In the primal we minimize, in the dual we maximize
- At the optimum x^*, (α^*, β^*) are such that



Stationarity

$$\frac{\partial L(x^*,\alpha^*,\beta^*)}{\partial x} = 0 \qquad \frac{\partial L(x^*,\alpha^*,\beta^*)}{\partial \beta} = 0$$

Primal admissibility

$$h_i(x^*) = 0 \qquad g_i(x^*) \leq 0$$

Dual admissibility

$$\alpha_i^* \geq 0$$

Complementarity

$$\alpha_i^* g_i(x^*) = 0$$

KARUSH-KUHN-TUCKER THEOREM

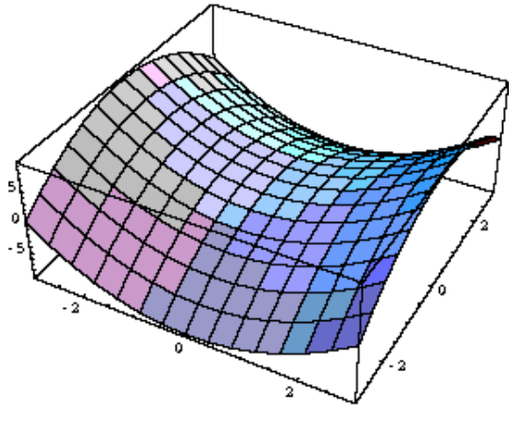
- The last condition is known as the KKT (Karush-Kuhn-Tucker) condition
  - It implies that
    - for active constraints $\alpha_i > 0$
    - for inactive constraints $\alpha_i = 0$
- The KKT condition will allow to identify the training examples that define the largest margin hyperplane
- α_i represents the influence of constraints, thus the influence of each training example
- These active examples will be called **Support Vectors**

OPTIMAL HYPERPLAN

- The Lagrangian to **minimize** is

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right]$$

- The solution $(\mathbf{w}^*, b^*, \boldsymbol{\alpha}^*)$ (unique thanks to convexity) is a saddle point of the Lagrangian:
  - $L(\mathbf{w}, b, \boldsymbol{\alpha}^*)$ is minimal with respect to the primal variables $(\mathbf{w}^*, b^*)$
  - $L(\mathbf{w}^*, b^*, \boldsymbol{\alpha})$ is maximal with respect to the dual variables $\boldsymbol{\alpha}^*$ (Lagrange Multipliers)



The Lagrangian saddle point

$$\min_{\mathbf{w},b} \max_{\boldsymbol{\alpha}} L(\mathbf{w}, b, \boldsymbol{\alpha}) \quad \Leftrightarrow \quad \max_{\boldsymbol{\alpha}} \min_{\mathbf{w},b} L(\mathbf{w}, b, \boldsymbol{\alpha})$$

FIRST KKT CONDITION

- The primal Lagragian is

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1\right]$$

- The first KKT condition amounts to compute the gradients on the primal variables *w, b:*

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\alpha})}{\partial b} = \sum_{i=1}^{N} \alpha_i y_i = 0$$

DUAL FORMULATION

- The Lagragian is

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right]$$

- The optimality conditions are: $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \quad \textit{and} \quad \sum_{i=1}^{N} \alpha_i y_i = 0$

- The Lagragian is rewritten as

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{w}^T \mathbf{x}_i - b \sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i$$

- which gives

$$L(\boldsymbol{\alpha}) = \underbrace{\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j}_{\frac{1}{2}\mathbf{w}^T\mathbf{w}} - \underbrace{\sum_{i=1}^{N} \alpha_i y_i \sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_i^T \mathbf{x}_j}_{\mathbf{w}^T\mathbf{w}} - \underbrace{b \sum_{i=1}^{N} \alpha_i y_i}_{0} + \sum_{i=1}^{N} \alpha_i$$

DUAL FORMULATION

- The dual Lagrangian problem to **maximize** is then

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

- under the constraints

$$\alpha_i \geq 0 \qquad \sum_{i=1}^{N} \alpha_i y_i = 0$$

- The primal problem scales with **dimensionality** (w has one coefficient per dimension) whereas the dual problem scales with the **number of examples** (one Lagrange multiplier per example)

- The classification of a new example is given by

$$f(\mathbf{x}) = sgn\left(\sum_{i=1}^{N} y_i \alpha_i \mathbf{x}_i^T \mathbf{x} + b \right)$$

SUPPORT VECTORS

- The KKT condition states that

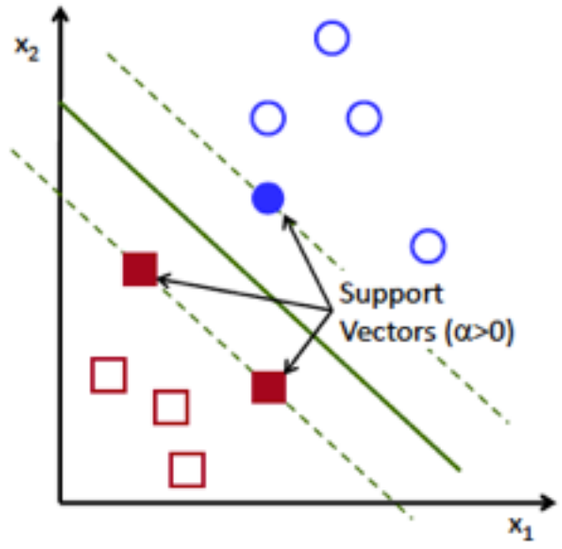
$$\alpha_i \left[y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right] = 0 \quad \forall i = 1 \dots N$$

- So we have either
  - $\alpha_i = 0$ and $|\mathbf{w}^T \mathbf{x}_i + b| > 1$

    These examples are useless
  - $\alpha_i > 0$ and $|\mathbf{w}^T \mathbf{x}_i + b| = 1$

    These examples are on the hyperplane margin as $y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 = 0$

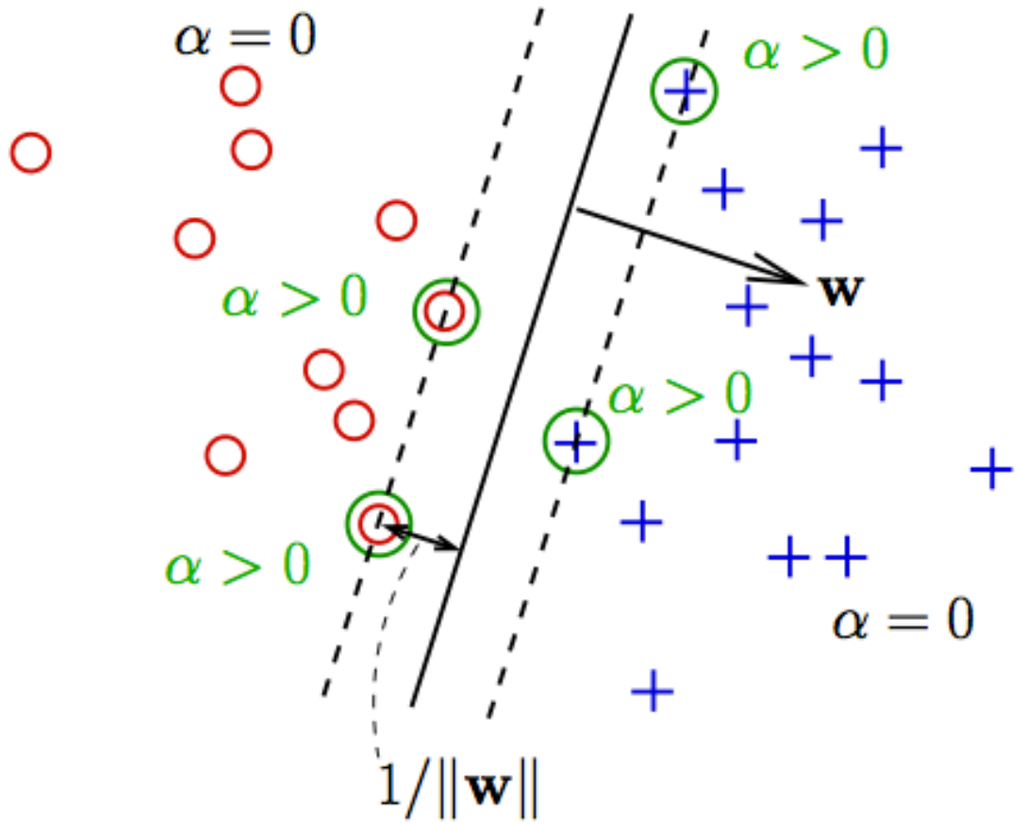
    and are called **Support Vectors**

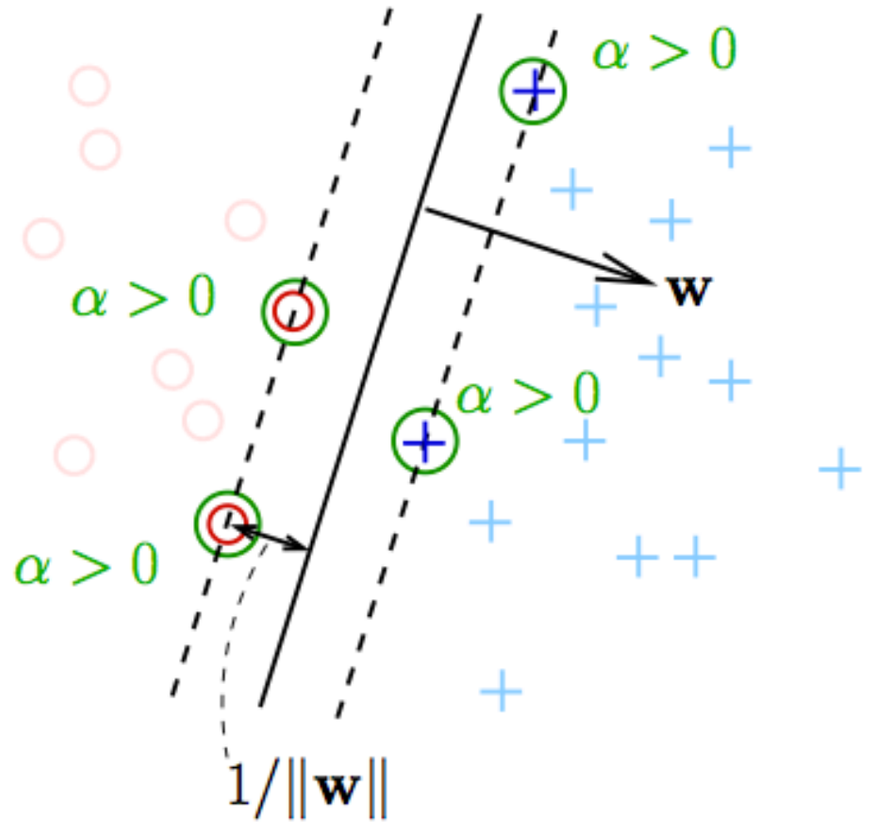


- An estimation of the LOO error is given by the proportion of support vectors $\text{Err}_{LOO} \leq \frac{|\{k|\alpha_k > 0\}|}{N}$

SUPPORT VECTORS

- **Only the support vectors determine the hyperplane** : this is why SVM are sparse machines !





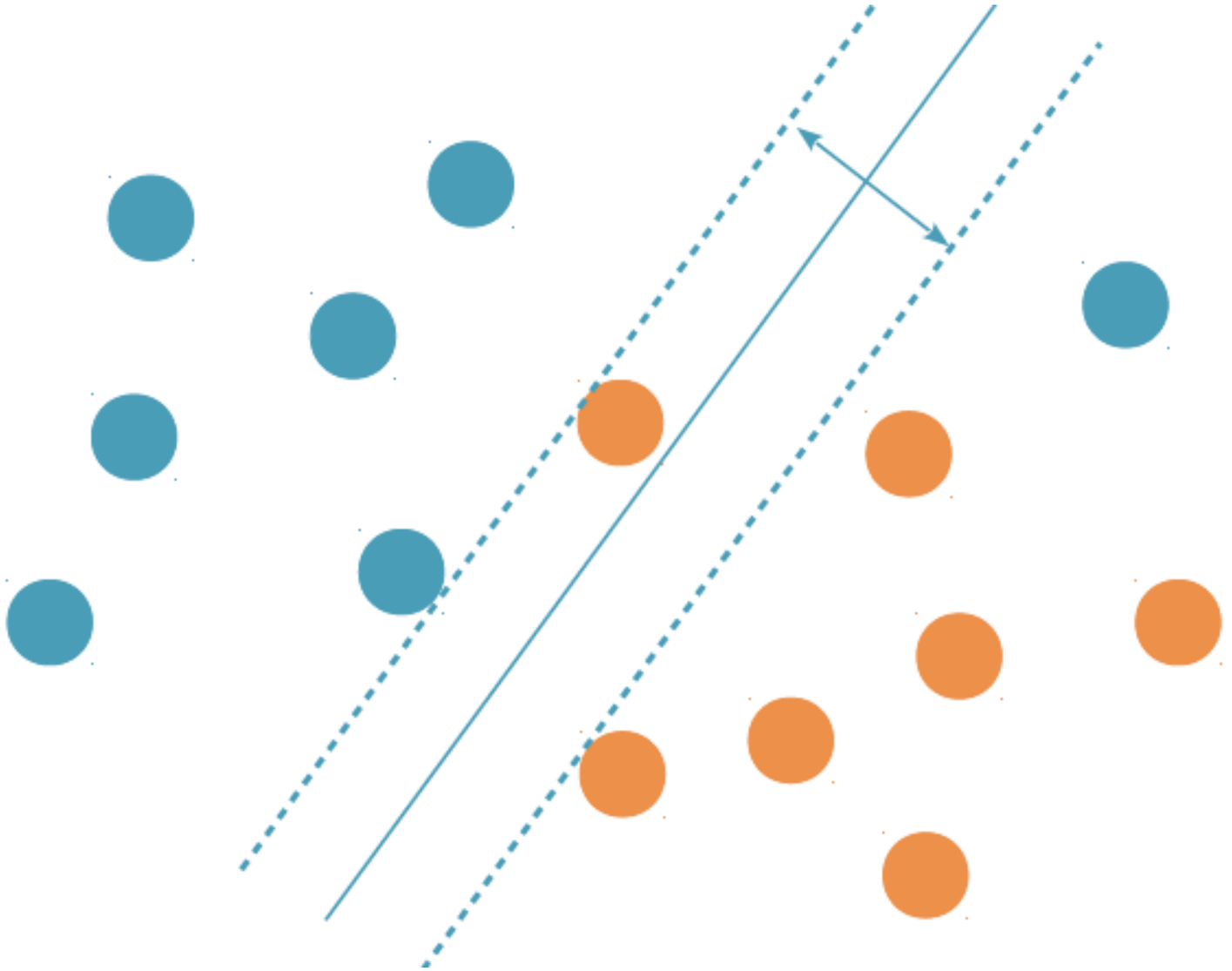
LEARNING OBJECTIVES

- Risk minimization
- Support Vector Machines
- **Soft-margin SVMs**
- Kernels and Non-linear SVMs
- Multi-class SVMs

SOFT MARGIN SVMS

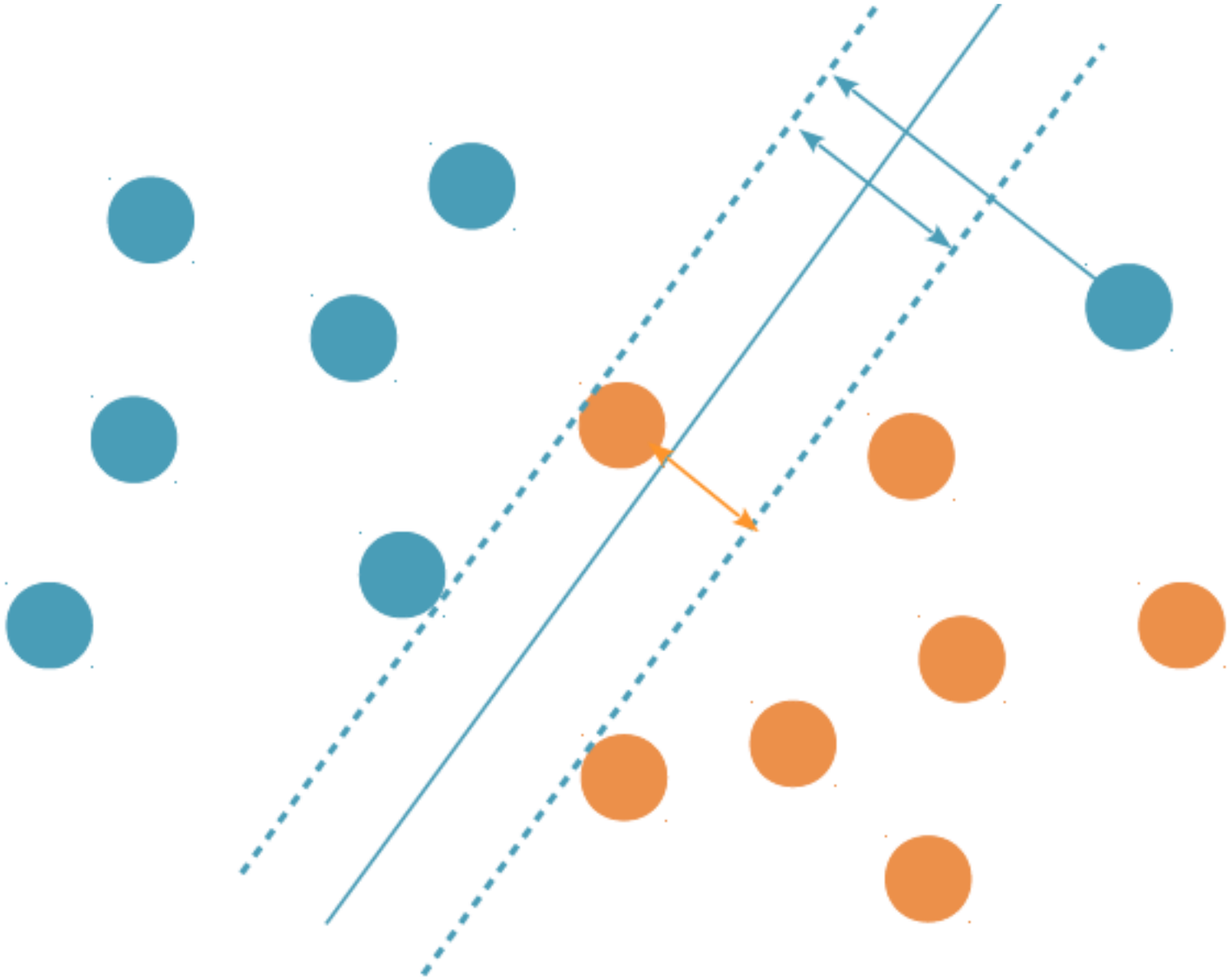
- What if the data are not linearly separable ?

SOFT MARGIN SVMS



SOFT MARGIN SVMS

- Clearly poses a problem for the margin !



SOFT MARGIN SVMS

- When the data is not linearly separable (most of the cases)
- Find instead a trade-off between **large margin** and **few errors**

$$\min_f \left(\frac{1}{\text{margin}(f)} + C \times \text{error}(f) \right)$$

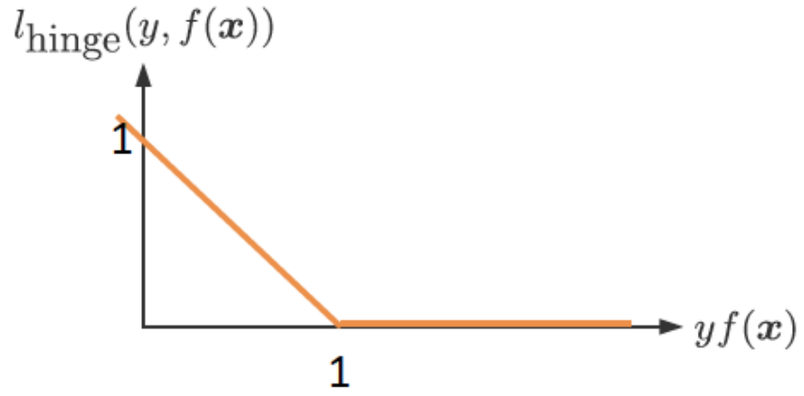
- How to measure the error ?
  - Consider how far from the hyperplane the example is

$$\begin{cases} 0 & \text{if } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \\ 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b) & \text{otherwise} \end{cases}$$

  - This is the hinge loss function

$$L_{hinge}(v, y) = \max(1 - yv, 0)$$

$$= \begin{cases} 0 & \text{if } yv \geq 1 \\ 1 - yv & \text{otherwise} \end{cases}$$



THE C PARAMETER

$$\min_f \left(\frac{1}{\text{margin}(f)} + C \times \text{error}(f) \right)$$

Large C

makes few errors

Small C

ensures a large margin

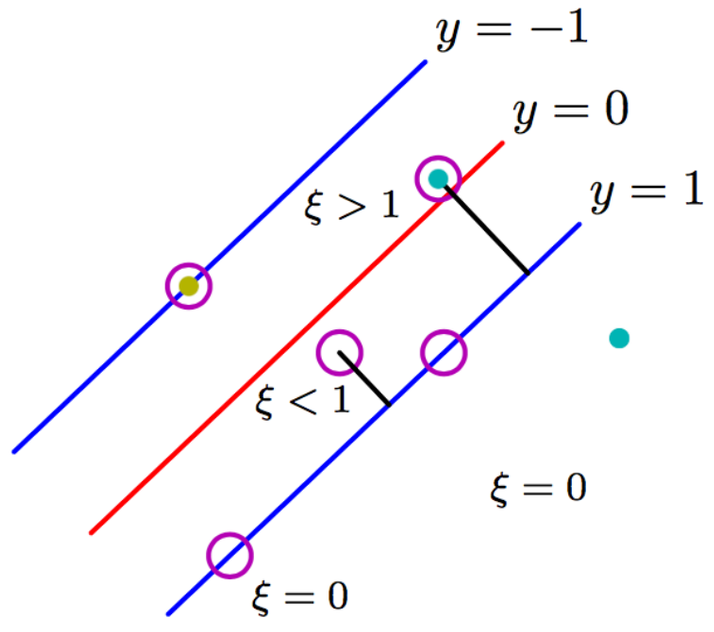
Intermediate C

finds a tradeoff

SLACK VARIABLES

- We want to allow some points to be on the wrong side of the hyperplane decision boundary i.e. to allow errors
- We want to increase linearly a penalty with the distance from the decision boundary
- Let's introduce **slack** variables $\xi_i \geq 0$ for each example

$$\xi_i \begin{cases} = 0 & \text{if correctly classified} \\ < 1 & \text{if in correct margin} \\ = 1 & \text{if on the decision boundary} \\ > 1 & \text{if misclassified} \end{cases}$$

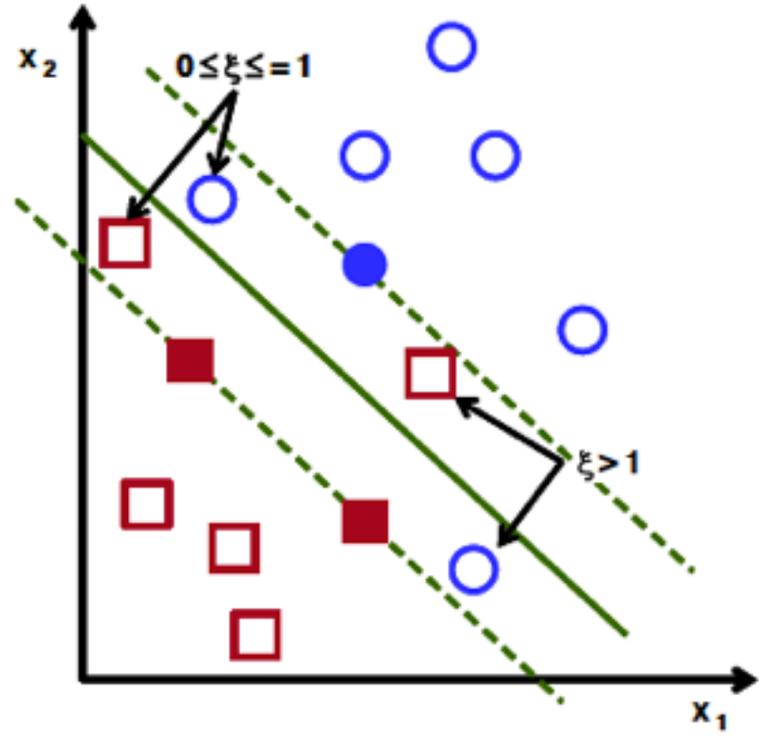


SLACK VARIABLES

- We introduce the slack variables into the classification constraints
- Originally we had $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$
- This corresponds to hard-margin SVMs
- If we release this constraint, with slack variables, we obtain soft-margin SVMs:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$$

  - When $\xi_i > 1$ the example is wrongly classified
  - When $0 \leq \xi_i \leq 1$ the example is in the margin but well classified



BACK TO OUR FORMULATION

- We had expressed our objective as

$$\min_f \left(\frac{1}{\text{margin}(f)} + C \times \text{error}(f) \right)$$

- which gives

$$\arg\min_{\mathbf{w},b} \left(\frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)) \right)$$

- This is not an explicit Quadratic Programming problem

- We can approximate it as

$$\arg\min_{\mathbf{w},b} \left(\frac{1}{2}\|\mathbf{w}\|^2 + \boxed{C \sum_{i=1}^{N} \xi_i} \right)$$

Upper bound on the number of misclassifications

$$\text{s.t. } y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

LAGRAGIAN OF THE SOFT-MARGIN SVM

- We have

$$\arg\min_{\mathbf{w},b}\left(\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i\right)$$

$$\text{s.t. } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

- and we look for the saddle point of the Lagrangian

$$L(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right) - \sum_{i=1}^{N}\beta_i\xi_i$$

with $\alpha_i \geq 0$ and $\beta_i \geq 0$

which is slightly different from the hard-margin SVMs :

$$L(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right) - \sum_{i=1}^{N}\beta_i\xi_i$$

ASSOCIATED KKT CONDITIONS

➤ The final problem formulation with KKT conditions is

$$L(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right) - \sum_{i=1}^{N}\beta_i\xi_i$$

$$\begin{aligned}
\alpha_i &\geq 0 \\
y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i &\geq 0 \\
\alpha_i\left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right) &= 0 \\
\beta_i &\geq 0 \\
\xi_i &\geq 0 \\
\beta_i\xi_i &= 0
\end{aligned}$$

COMPUTING THE GRADIENTS

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\beta})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\beta})}{\partial b} = \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\alpha},\boldsymbol{\xi},\boldsymbol{\beta})}{\partial \xi_i} = C - \alpha_i - \beta_i = 0$$

No change for *w* and *b* from the hard-margin SVMs !

$\alpha_i = C - \beta_i$ and since $\alpha_i \geq 0$ and $\beta_i \geq 0$

we have $\alpha_i \leq C$

DUAL FORMULATION OF THE SOFT-MARGIN SVM

$$L(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right) - \sum_{i=1}^{N}\beta_i\xi_i$$

rewritten as

$$L(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i y_i \mathbf{w}^T\mathbf{x}_i - b\sum_{i=1}^{N}\alpha_i y_i + \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\alpha_i\xi_i - \sum_{i=1}^{N}\beta_i\xi_i$$

grouping the ξ_i

$$L(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\beta}) = \underbrace{\frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i y_i \mathbf{w}^T\mathbf{x}_i}_{\text{the same than for hard-margin}} - b\underbrace{\sum_{i=1}^{N}\alpha_i y_i}_{0} + \sum_{i=1}^{N}\alpha_i + \sum_{i=1}^{N}\underbrace{(C - \alpha_i - \beta_i)}_{0}\xi_i$$

and we use

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i$$

as

$$\sum_{i=1}^{N}\alpha_i y_i = 0$$

as

$$C - \alpha_i - \beta_i = 0$$

DUAL FORMULATION OF THE SOFT-MARGIN SVM

- The dual problem to maximize is **the same** than for Hard-margin:

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

- with the constraints: $\sum_{i=1}^{N} \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$

$\alpha_i = 0$ *« easy »*

Not a Support vector and well classified

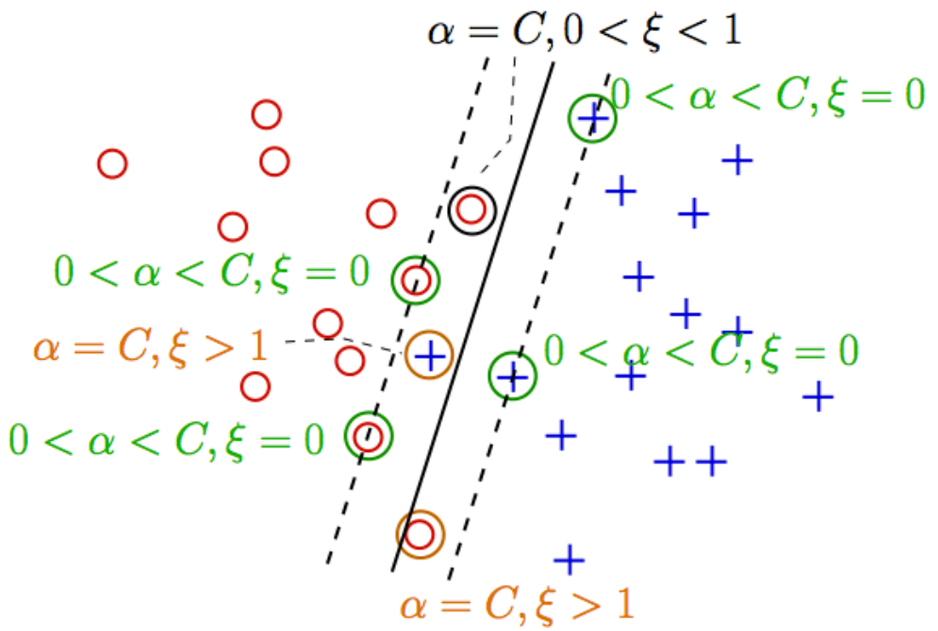
$0 < \alpha_i < C$ *« somewhat hard »*

The Support vector is on the margin and $\xi_i = 0$

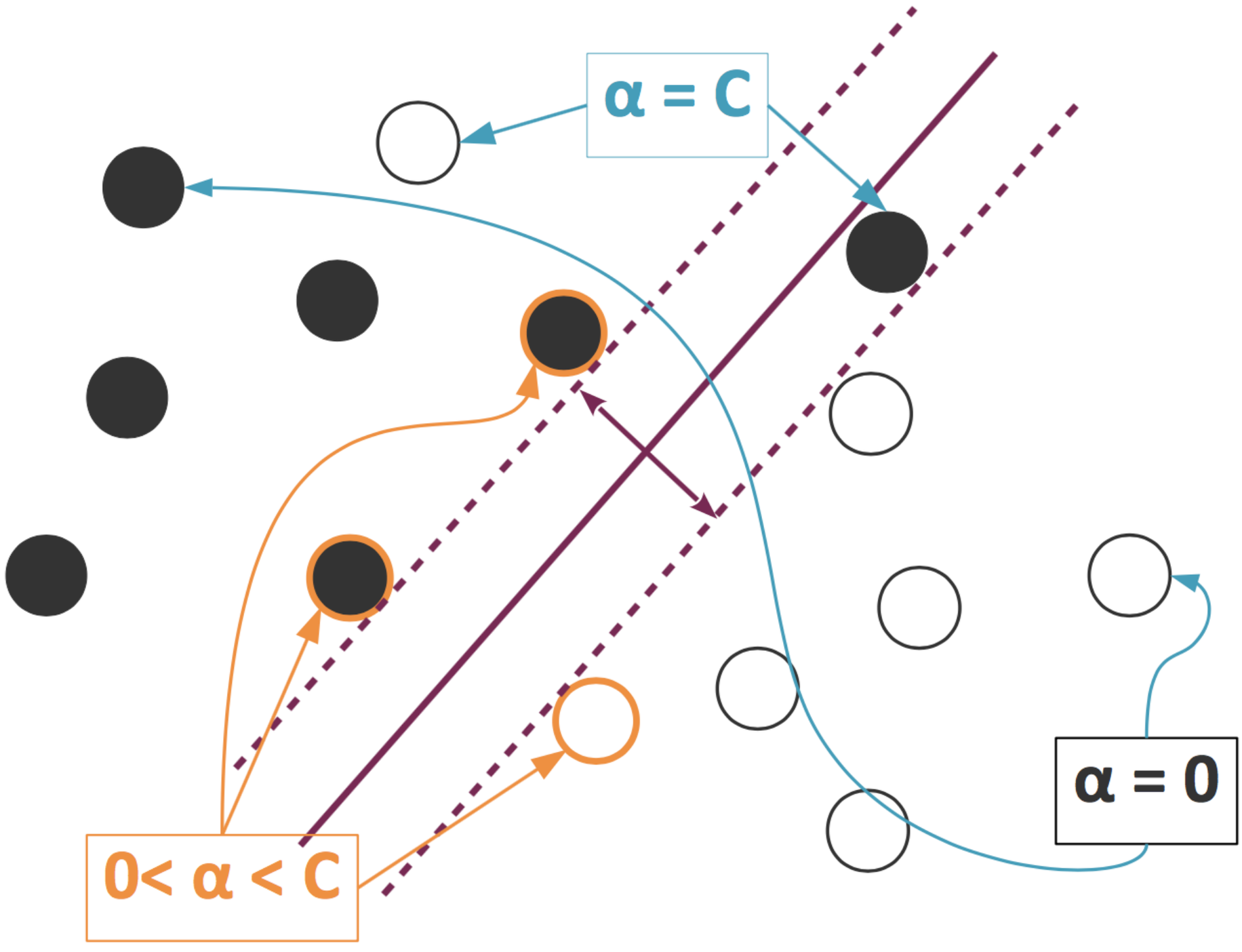
$\alpha_i = C$ *« hard »*

The Support vector is in the margin

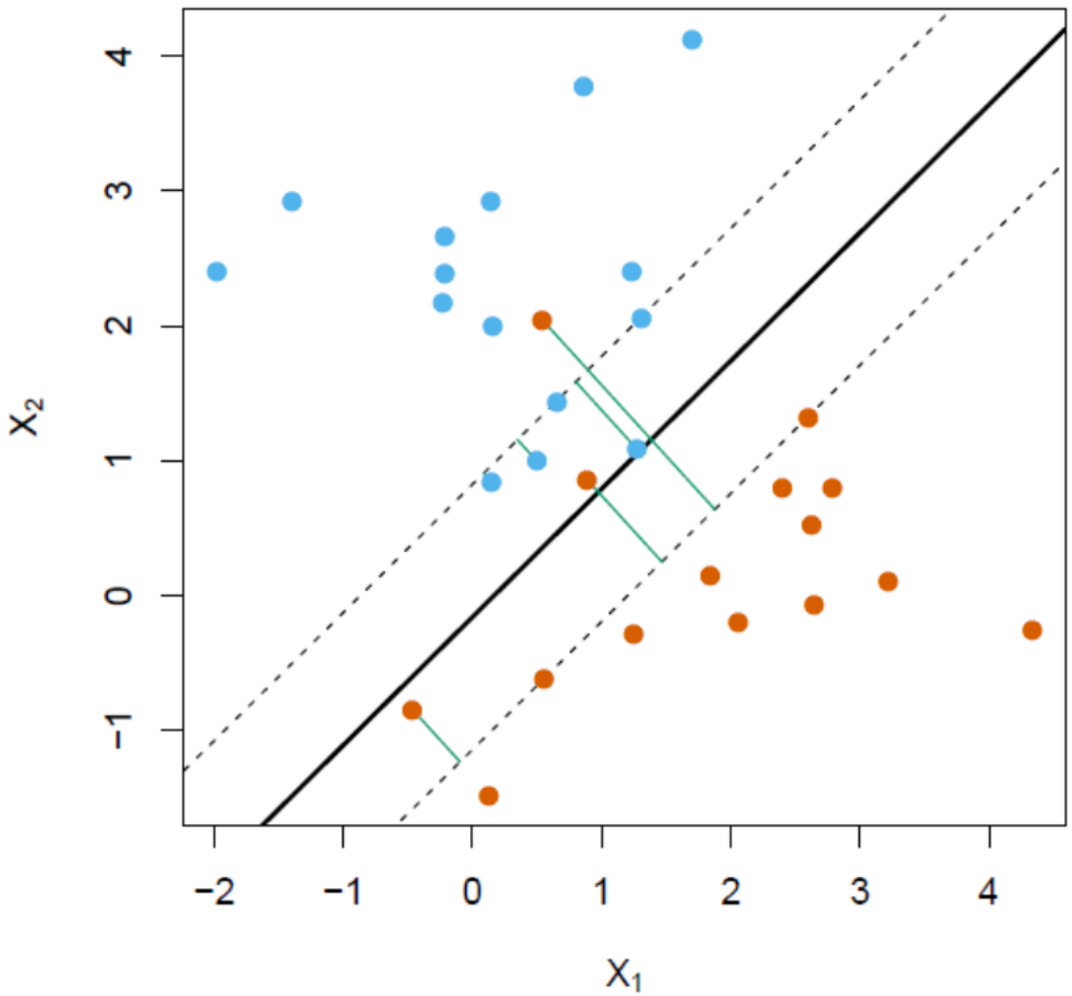
and misclassified $\xi_i > 1$ *or well classified* $0 < \xi_i < 1$



QUICK RECAP ON SV FOR THE SOFT-MARGIN SVMS

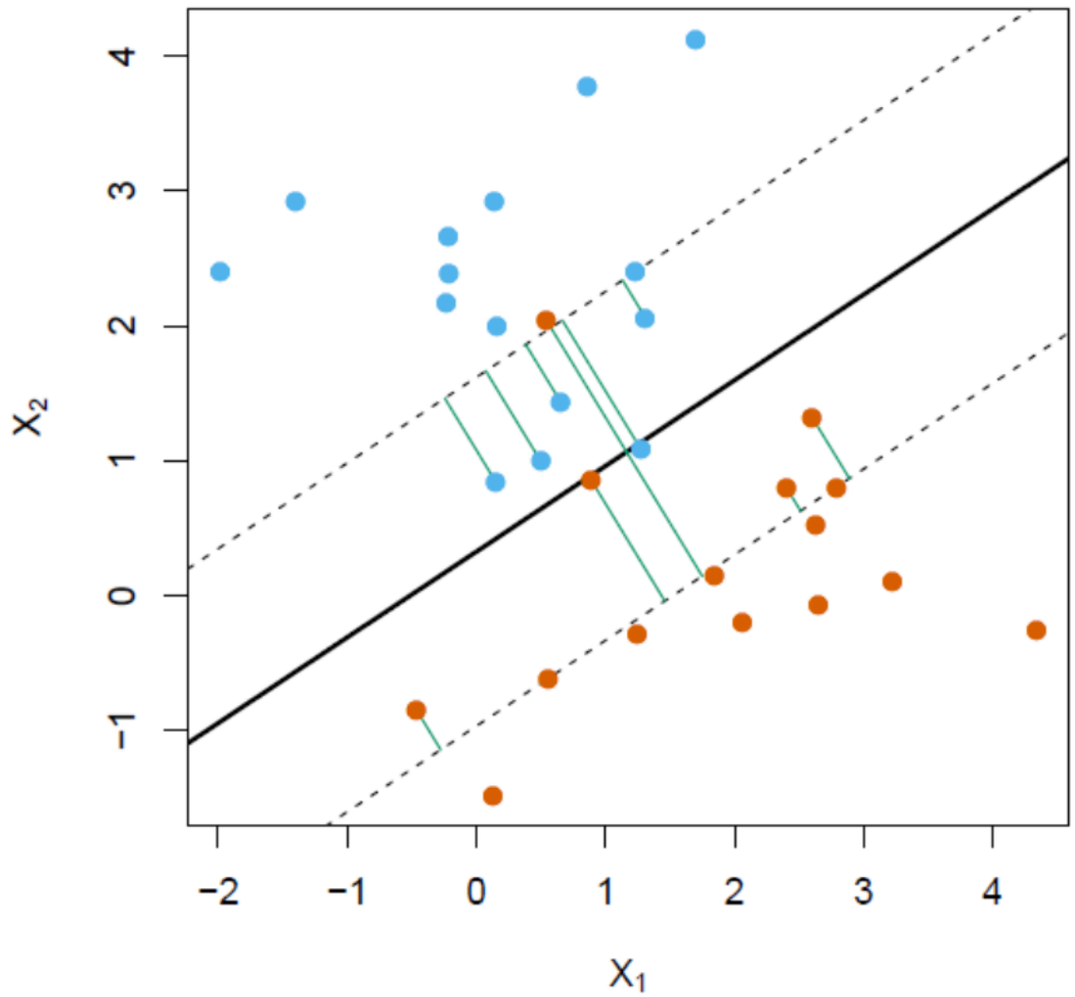


INFLUENCE OF THE PARAMETER C



Large C :

few errors, tiny margin



Small C :

more errors, large margin

TOWARDS NON LINEAR SVMS ?

- For a hard or soft margin, the dual Lagrangian optimization problem is:

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

- and the decision function is

$$f(\mathbf{x}) = sgn\left(\sum_{i=1}^{N} y_i \alpha_i \mathbf{x}_i^T \mathbf{x} + b \right)$$

- We can remark that this uses the training examples only through scalar products: $\mathbf{x}_i^T \mathbf{x}_j =< \mathbf{x}_i, \mathbf{x}_j >$
- We can make the most of this thanks to the **Kernel Theory**

  and define non-linear SVMs

LEARNING OBJECTIVES

- Risk minimization
- Support Vector Machines
- Soft-margin SVMs
- **Kernels and Non-linear SVMs**
- Multi-class SVMs

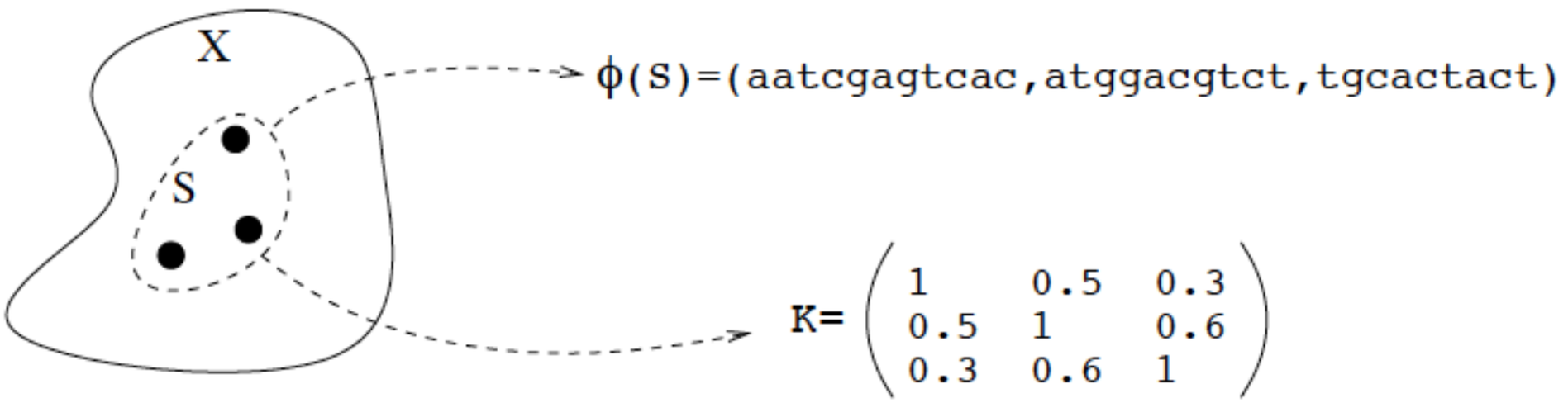
KERNELS

- Real data can be much more complicated than simple vectors of real values (strings, graphs, images, ...)
- How to develop **versatile** algorithms without any assumptions on the **type of data** ?
- The Kernel approach:
  - Develop methods using **pairwise comparisons**
  - By imposing constraints on the pairwise comparison function (positive definite kernels), we obtain a **general framework for learning from data**

REPRESENTATION BY PAIRWISE COMPARISONS

- Idea
  - Define a comparison function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$
  - Represent a set of N data points $\mathcal{S} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ by the $N \times N$ matrix:

$$[\mathbf{K}]_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$$ *called a Gram matrix*



REPRESENTATION BY PAIRWISE COMPARISONS

- K is always an $N \times N$ matrix, whatever the nature of data: **the same algorithm will work for any type of data** (vectors, strings, ...).
- Total **modularity** between the **choice of function *K*** and **the choice of the algorithm**.
- **Poor scalability** with respect to the dataset size (n^2 to compute and store K)... : some approximation strategies do exist
- We will restrict ourselves to a **particular class** of pairwise comparison functions.

POSITIVE DEFINITE (P.D.) KERNELS

➤ A positive definite (p.d.) kernel on a set $\mathcal{X}$ is a function

$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that is symmetric

$$\forall (\mathbf{x}, \mathbf{x}) \in \mathcal{X}^2, \quad K(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}', \mathbf{x})$$

and which satisfies, for all $N \in \mathbb{N}$, $(\mathbf{x}_1, \cdots, \mathbf{x}_N) \in \mathcal{X}^N$ and $(a_1, \cdots, a_N) \in \mathbb{R}^N$:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$$

Equivalently, a Kernel $\mathbf{K}$ is p.d. iff the similarity matrix $[\mathbf{K}]_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ is positive semidefinite : $\mathbf{a}^T \mathbf{K} \mathbf{a} \geq 0 \quad \forall \mathbf{a} \in \mathbb{R}^N$

ONE SIMPLE P.D. KERNEL

- Let $\mathcal{X} = \mathbb{R}^d$. The function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined by:

$$\forall (\mathbf{x}, \mathbf{x}) \in \mathcal{X}^2, \quad K(\mathbf{x}, \mathbf{x}') =< \mathbf{x}, \mathbf{x}' >= \mathbf{x}^T \mathbf{x}'$$

  is positive definite (it is called the **linear kernel**)

Proof:

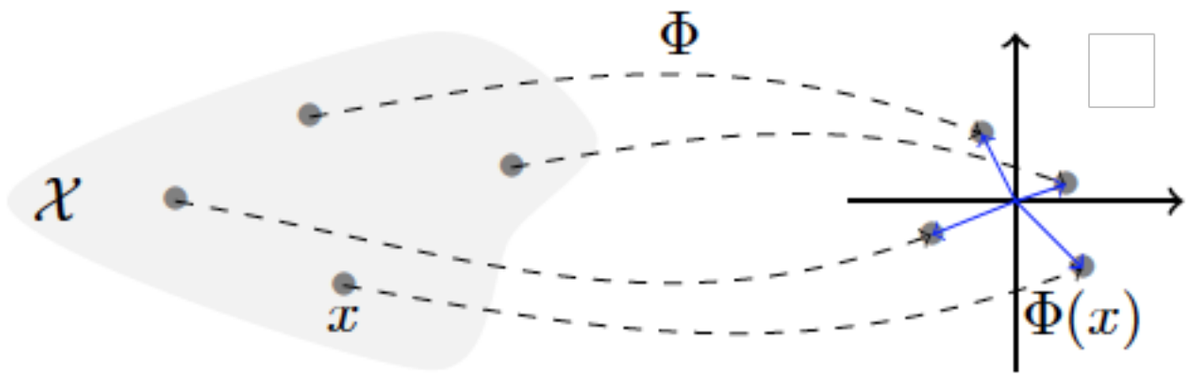
Symmetric: $\mathbf{x}^T \mathbf{x}' = \mathbf{x}'^T \mathbf{x}$

Positive:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j \mathbf{x}_i^T \mathbf{x}_j$$

$$= \left(\sum_{i=1}^{N} a_i \mathbf{x}_i \right)^T \left(\sum_{j=1}^{N} a_j \mathbf{x}_j \right) = \| \sum_{i=1}^{N} a_i \mathbf{x}_i \|^2 \geq 0$$

A MORE AMBITIOUS P.D. KERNEL



- Let $\mathcal{X}$ be any set, and $\Phi : \mathcal{X} \to \mathbb{R}^d$. Then the function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined as follows is p.d. :

$$\forall (\mathbf{x}, \mathbf{x}') \in \mathcal{X}^2, \quad K(\mathbf{x}, \mathbf{x}') = < \Phi(\mathbf{x}), \Phi(\mathbf{x}') >$$

- The kernel between two vectors *x* and *x'* is the inner product of their projection into some Hilbert space (a feature space).
- Proof: similar than for the linear kernel

EXAMPLE : A POLYNOMIAL KERNEL

- Consider this polynomial kernel on $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$

$$K(\mathbf{x}, \mathbf{x}') = \left\langle \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \begin{bmatrix} x_1' \\ x_2' \end{bmatrix} \right\rangle^2 = (x_1 x_1' + x_2 x_2')^2$$

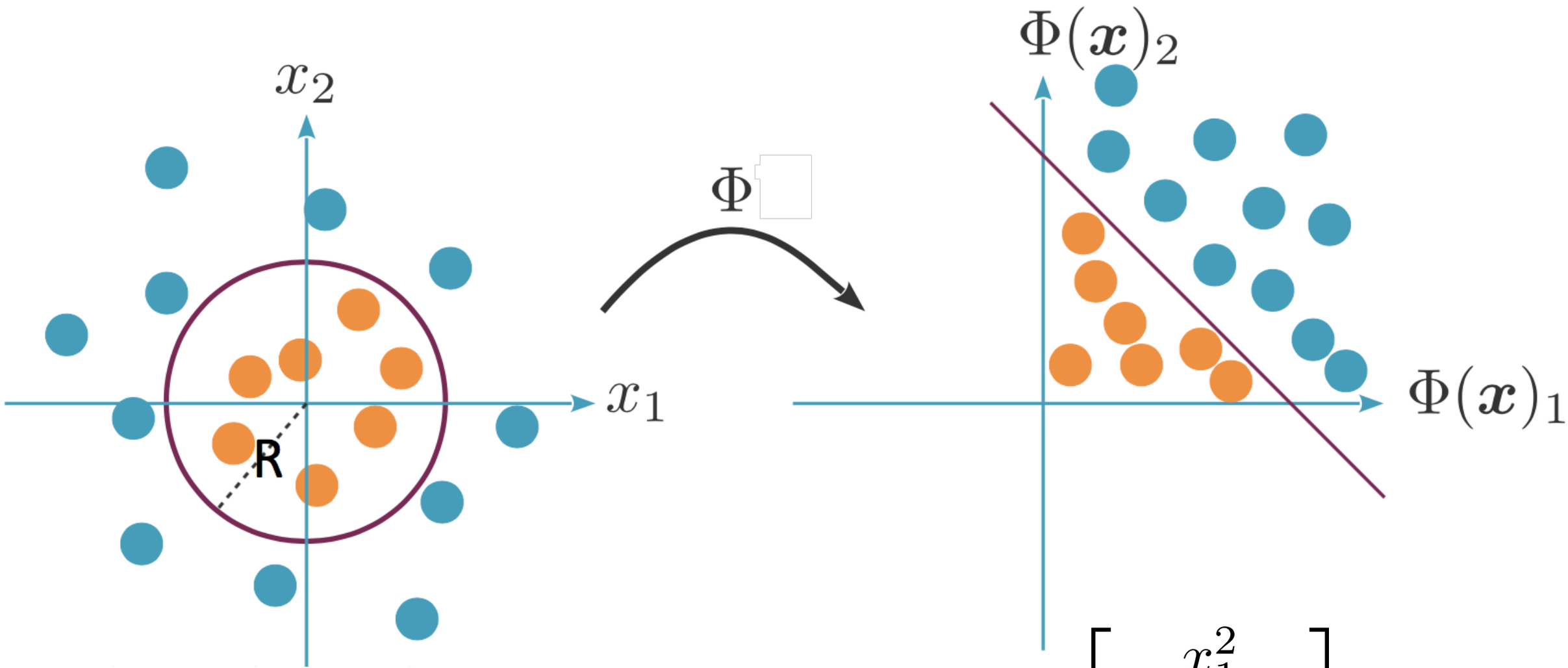
$$= x_1^2 x_1'^2 + 2 x_1 x_2 x_1' x_2' + x_2^2 x_2'^2$$

$$= \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$$

with $\Phi(\mathbf{x}) = \begin{bmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{bmatrix} \in \mathbb{R}^3$

THE PROJECTION INTO THE FEATURE SPACE

- A linear separator in the feature space can separate data from the two classes that were not linearly separable in the original space



$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$$

$$\Phi(\mathbf{x}) = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix} \in \mathbb{R}^3$$

COVER THEOREM

➤ <u>Theorem (Cover, 1965):</u>

« A complex pattern-classification problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated. »

- A naïve application of this theorem is not interesting because we will be subject to the curse of dimensionality
- Thanks to the kernel trick we can implicitly operate into a space of larger dimensions without explicitly computing the projection

$$K(\mathbf{x}, \mathbf{x}') = \left\langle \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} \right\rangle^2 = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$$

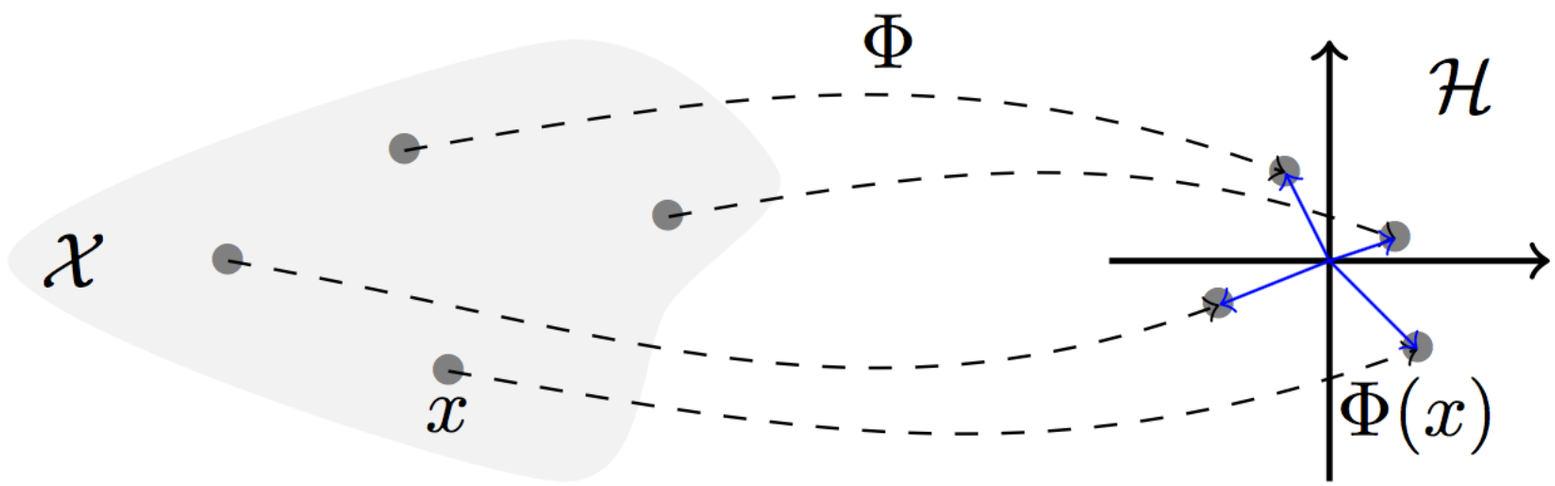
KERNELS AS INNER PRODUCTS

➤ <u>Theorem (Aronszjan, 1950):</u>

K is a p.d. kernel on the set $\mathcal{X}$ iff three exists a Hilbert space $\mathcal{H}$ and a mapping $\Phi : \mathcal{X} \to \mathcal{H}$

such that for any $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$

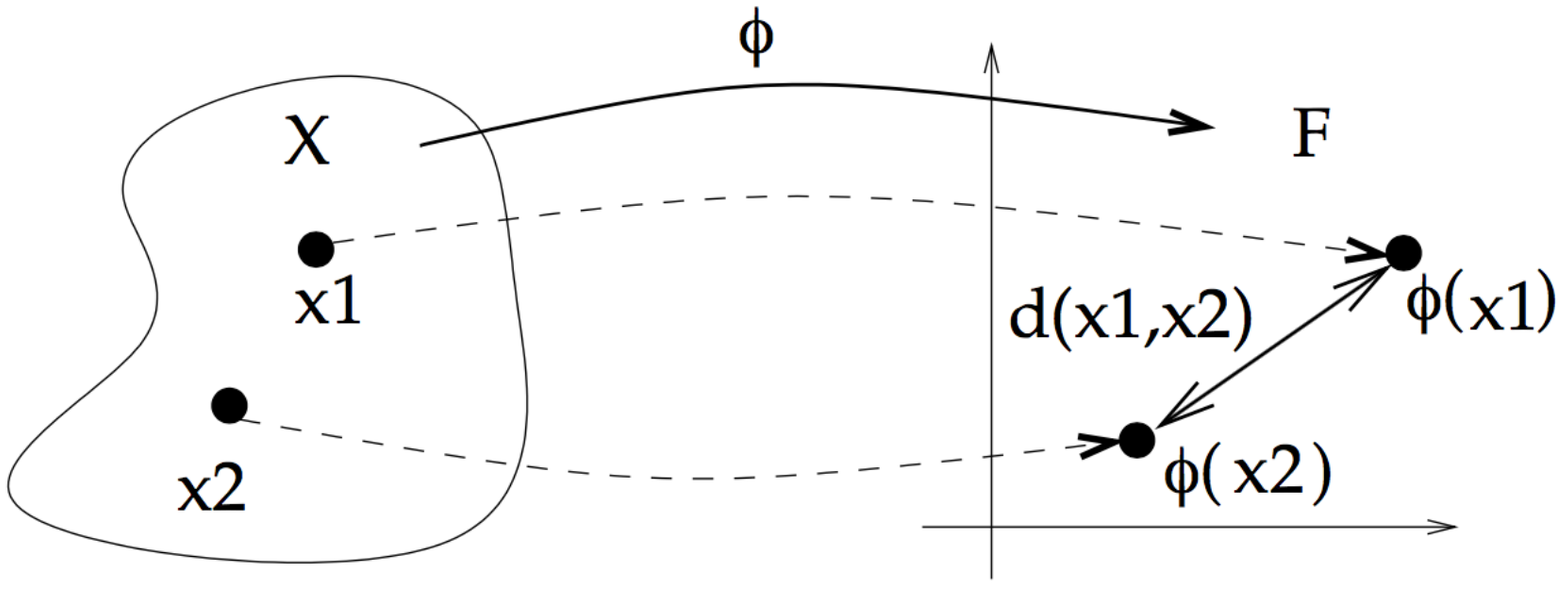
$$K(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{\mathcal{H}}$$



THE KERNEL TRICK

- **Any algorithm** to process finite-dimensional vectors that can be expressed **only in terms of pairwise inner products** can be applied to potentially infinite-dimensional vectors in the feature space of a p.d. kernel by **replacing each inner product evaluation by a kernel evaluation**.
- This is called the « **kernel trick** » and it has huge practical applications
- Vectors in the feature space are only manipulated **implicitly**, through pairwise inner products.

EXAMPLE : COMPUTING DISTANCES IN THE FEATURE SPACE

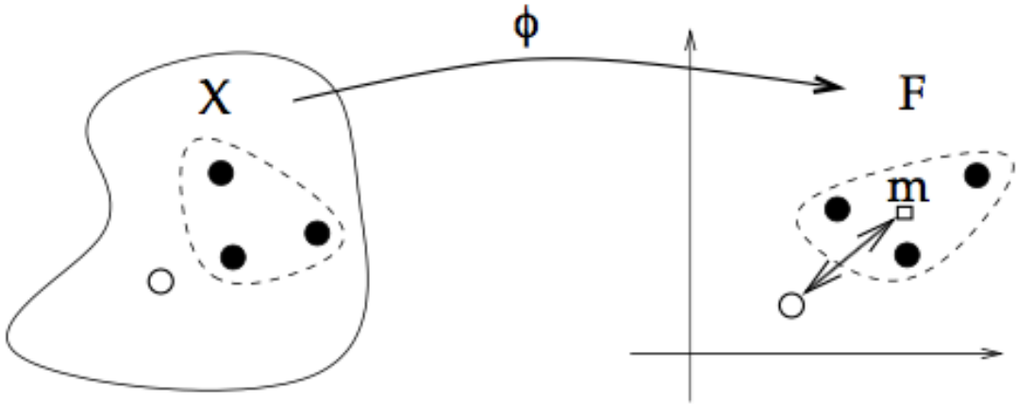


$$
\begin{aligned}
d_K(\mathbf{x}_1, \mathbf{x}_2)^2 &= \|\Phi(\mathbf{x}_1) - \Phi(\mathbf{x}_2)\|^2_{\mathcal{H}} \\
&= \langle \Phi(\mathbf{x}_1) - \Phi(\mathbf{x}_2), \Phi(\mathbf{x}_1) - \Phi(\mathbf{x}_2) \rangle_{\mathcal{H}} \\
&= \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_1) \rangle_{\mathcal{H}} - 2 \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2) \rangle_{\mathcal{H}} + \langle \Phi(\mathbf{x}_2), \Phi(\mathbf{x}_2) \rangle_{\mathcal{H}} \\
&= K(\mathbf{x}_1, \mathbf{x}_1) - 2K(\mathbf{x}_1, \mathbf{x}_2) + K(\mathbf{x}_2, \mathbf{x}_2)
\end{aligned}
$$

EXAMPLE : DISTANCE BETWEEN A POINT AND A SET

- Given a set of points $\mathcal{S} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$
- The barycenter is defined in the feature space $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x}_i)$
- The distance between x and the set is

$$d_K(x, \mathcal{S}) = \|\Phi(\mathbf{x}) - \boldsymbol{\mu}\| = \|\Phi(\mathbf{x}) - \frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x}_i)\|$$



The barycenter only exists in the feature space: it does not necessarily have a pre-image such that $\Phi(\bar{\mathbf{x}}) = \boldsymbol{\mu}$

$$d_K(x, \mathcal{S}) = \sqrt{K(\mathbf{x}, \mathbf{x}) - \frac{2}{N}\sum_{i=1}^{N}K(\mathbf{x}, \mathbf{x}_i) + \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}K(\mathbf{x}_i, \mathbf{x}_j)}$$

Can be used to define a Kernelized K-Means

KERNEL EXAMPLES

- Linear kernels

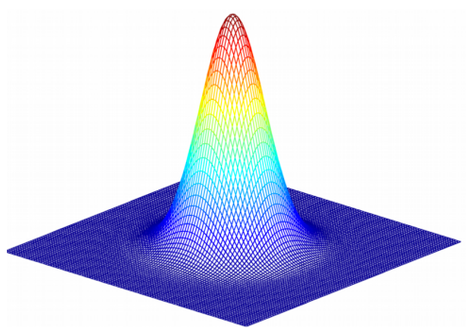
$$K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$$

- Polynomial kernels

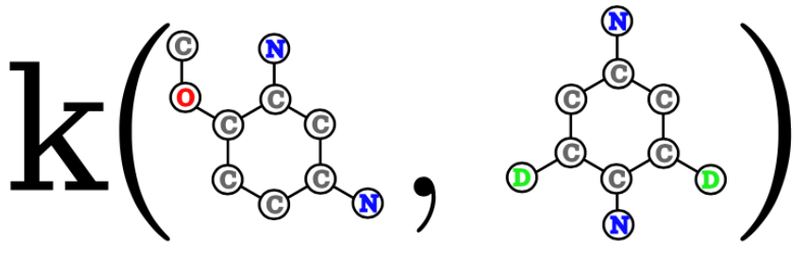
$$K(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}' + 1)^p$$

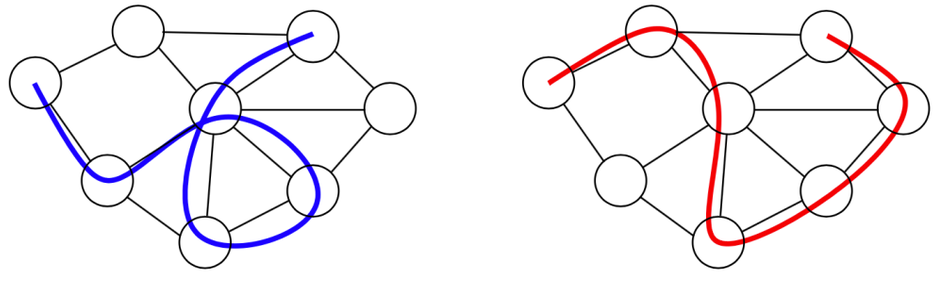
- Gaussian kernels

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$



- For more complex data structures such as graphs (molecules, social networks, ...), some specific kernels can be defined, by e.g. comparing paths into the graph





NON LINEAR SVM : FORMULATION

- Let $K(\cdot,\cdot)$ be a positive definite kernel in a Hilbert space $\mathcal{H}$
- There exists a function $\Phi : \mathcal{X} \to \mathcal{H}$ defined such that

  $\forall \mathbf{x}, \mathbf{x}' \in \mathcal{H}$ we have $K(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{\mathcal{H}}$
- Non linear SVMs are:

$$\arg \min_{\mathbf{w} \in \mathcal{H}, b} \left(\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \xi_i \right)$$

$$\text{s.t. } y_i(\mathbf{w}^T \phi(\mathbf{x}_i) + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

OPTIMAL HYPERPLANE FORMULATION

- The hyperplane equation in the projected space is

$$\mathbf{w}^T \phi(\mathbf{x}) + b = 0$$

- By using $\phi_0(\mathbf{x}) = b$ we rewrite it as $\mathbf{w}^T \phi(\mathbf{x}) = 0$

  (we augment the feature vector with a constant dimension)

- The optimal hyperplane is given by $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \phi(\mathbf{x}_i)$
- Putting all this together we have

$$\mathbf{w}^T \phi(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) = 0$$

- Classifying an example requires the kernel function with the support vectors: $\alpha_i \neq 0$

DUAL FORMULATION

- The Lagragian dual problem for the non linear SVM is

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

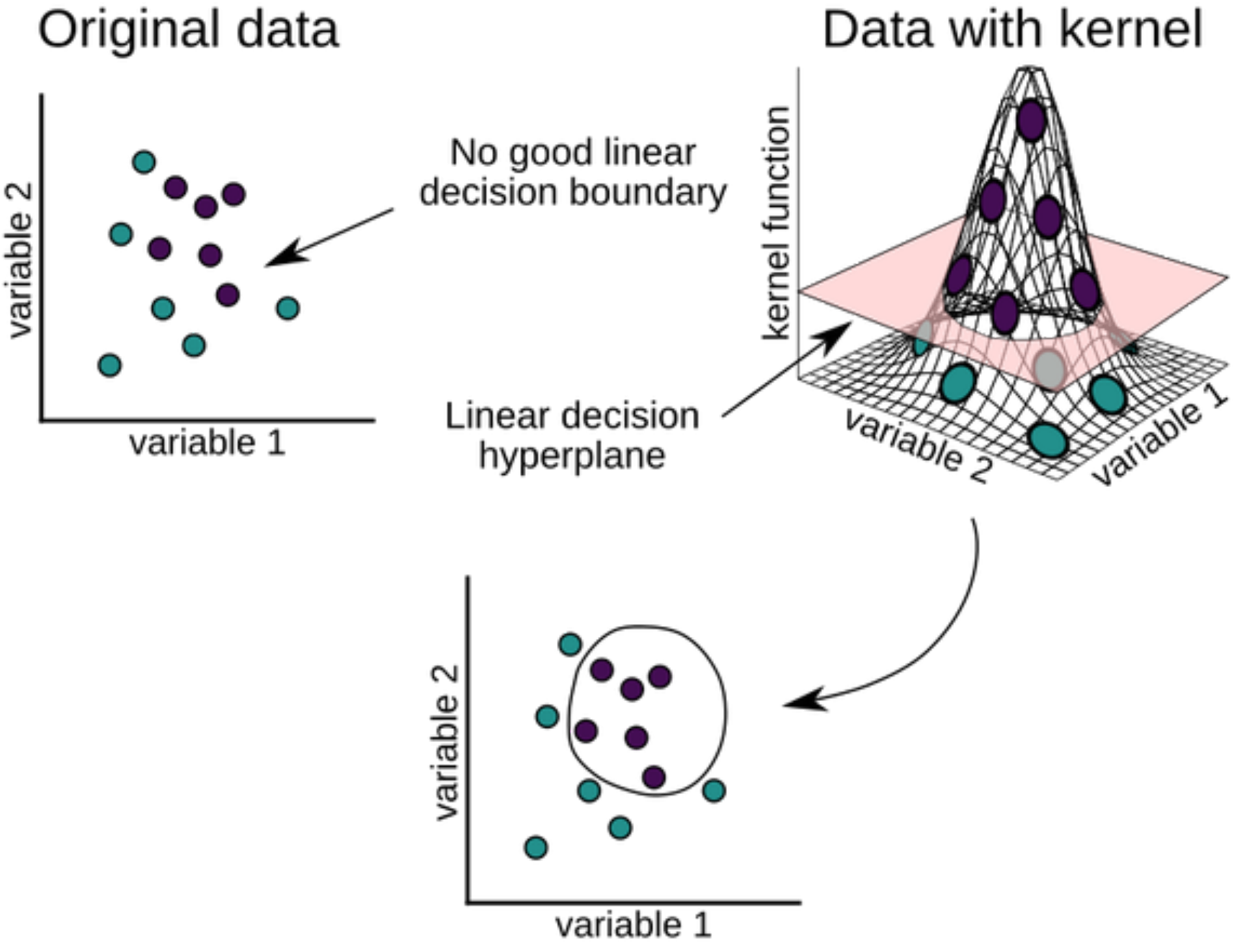
$$= \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

- with the constraints $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$
- The prediction function is:

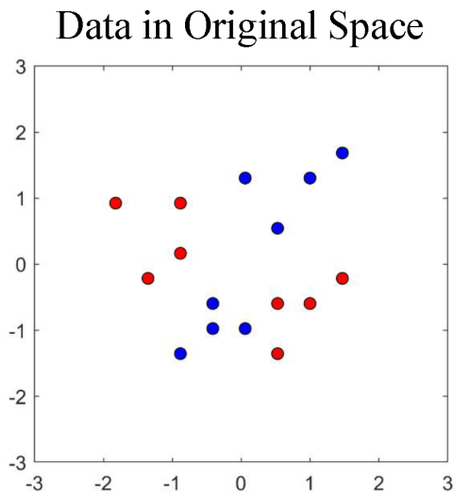
$$D(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \right)$$

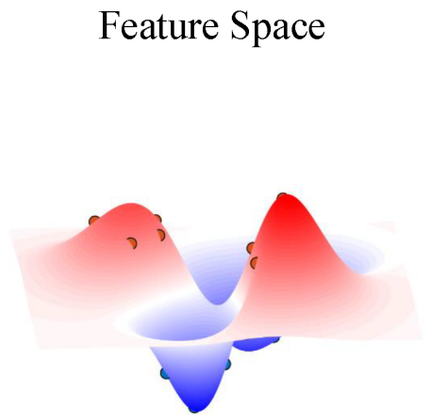
A non linear SVM is linear SVM with a non linear kernel

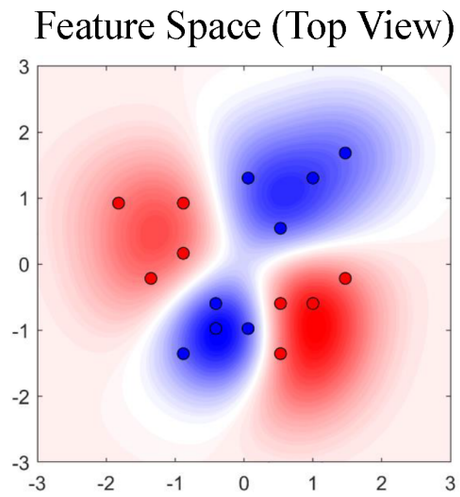
NON LINEAR SVMS ARE LINEAR SVMS IN THE FEATURE SPACE

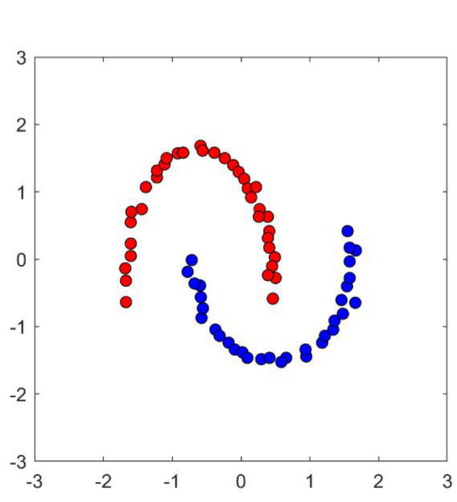


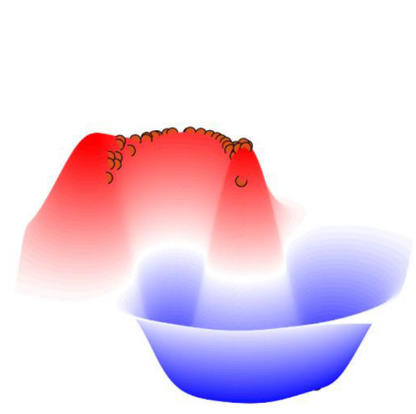
NON LINEAR SVMS ARE LINEAR SVMS IN THE FEATURE SPACE

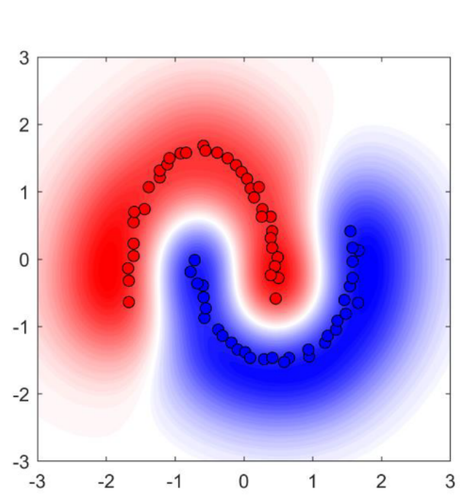


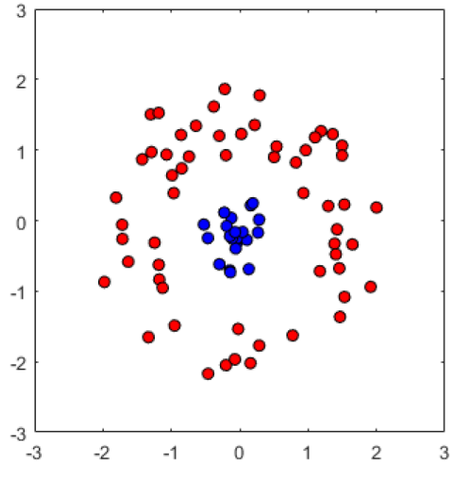


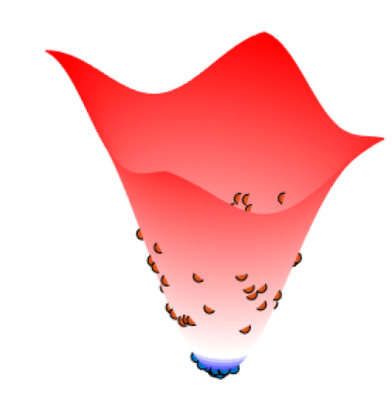


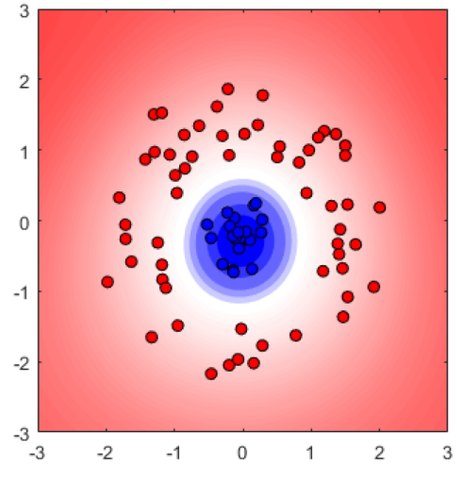




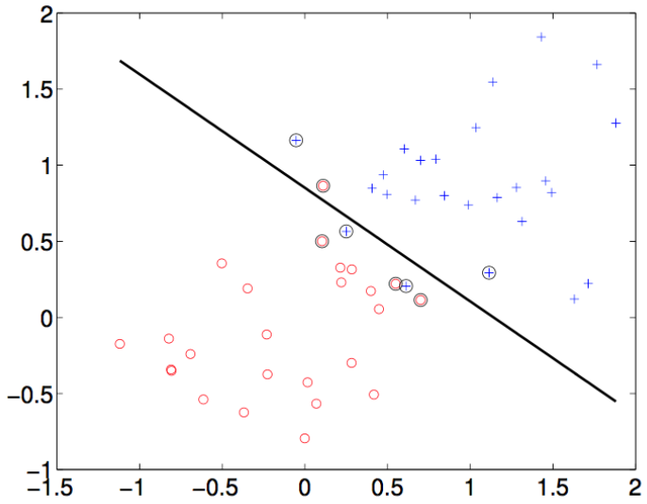




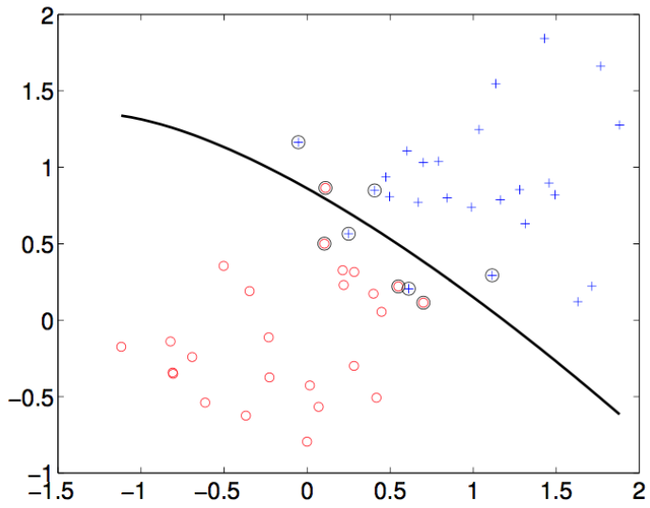




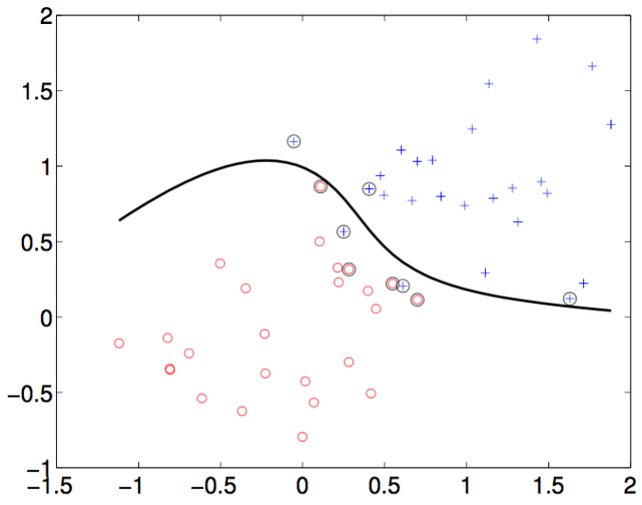
LINEAR VERSUS NONLINEAR



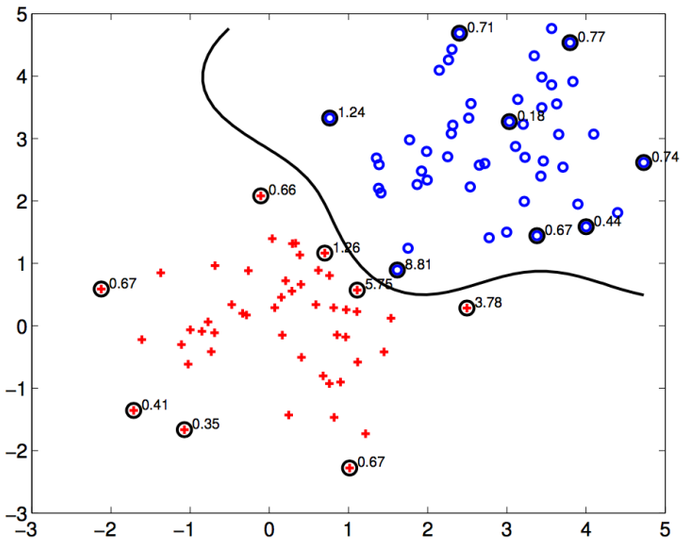
linear



2^{nd} order polynomial

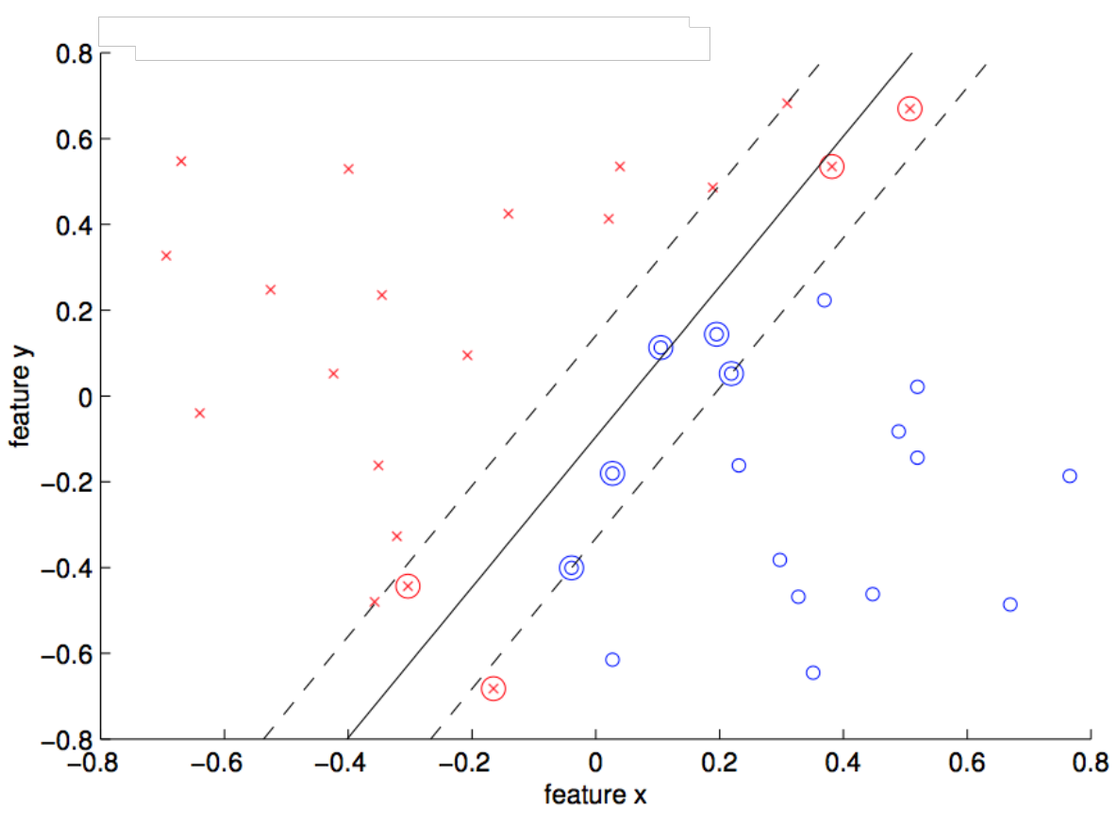


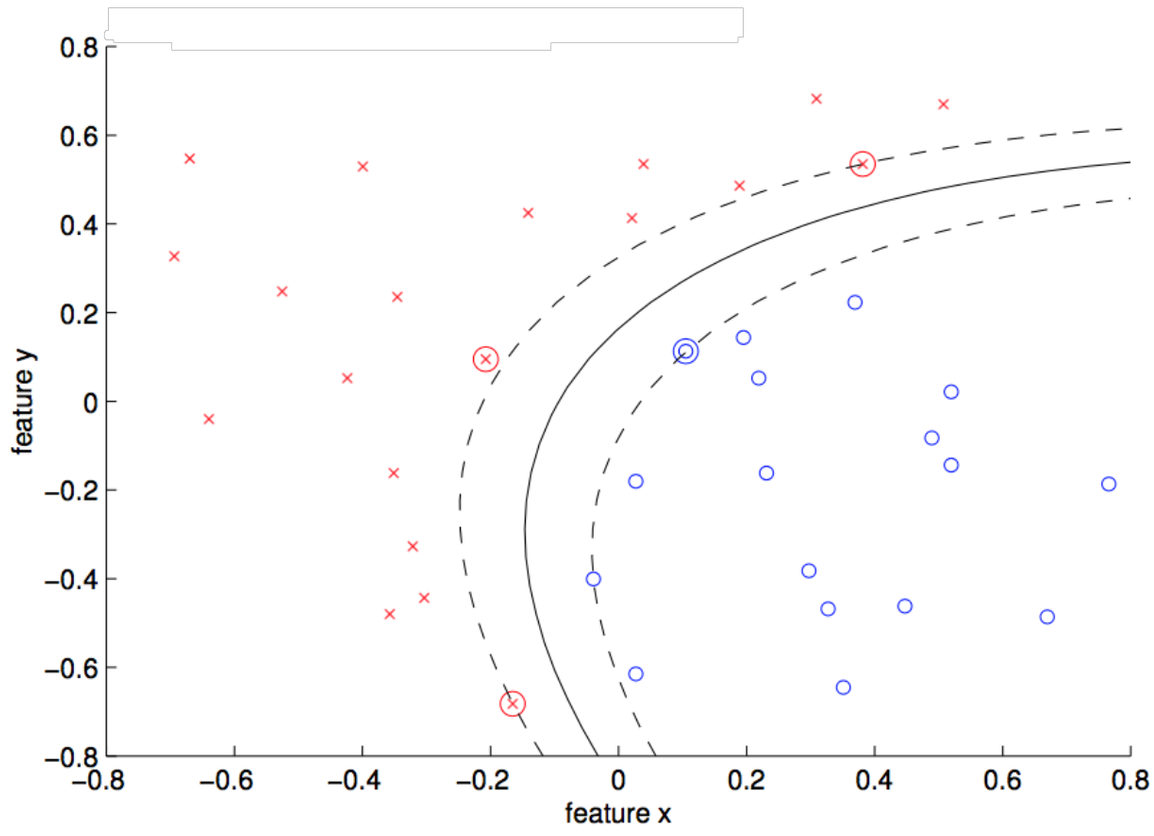
4^{th} order polynomial



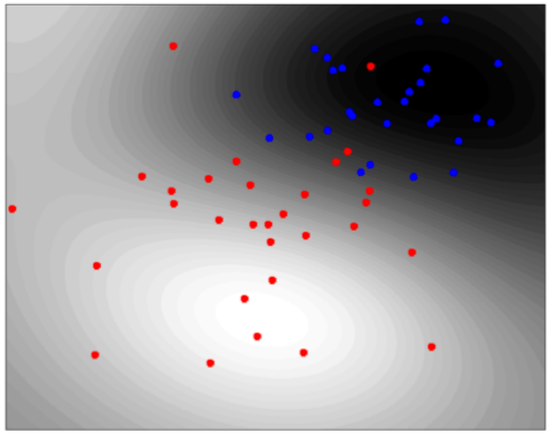
Radial basis kernel

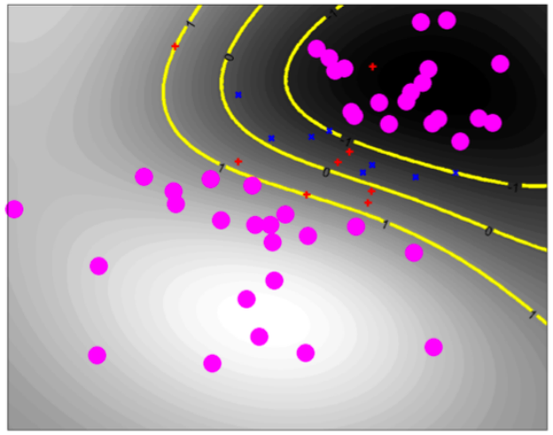
LINEAR VERSUS NONLINEAR

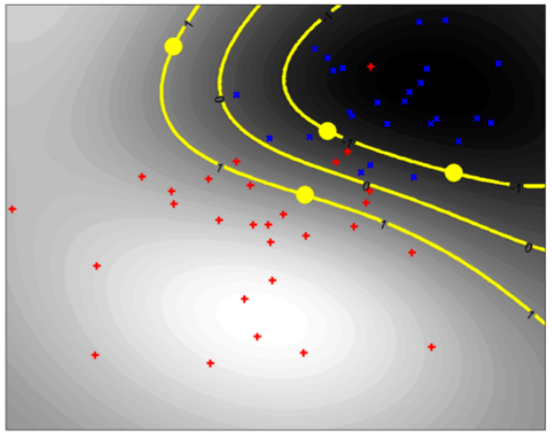


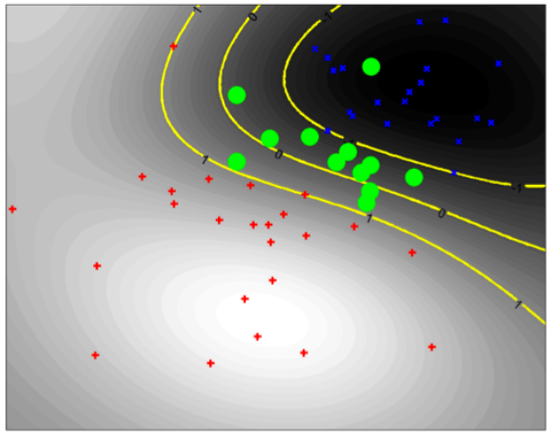


STILL SPARSE !



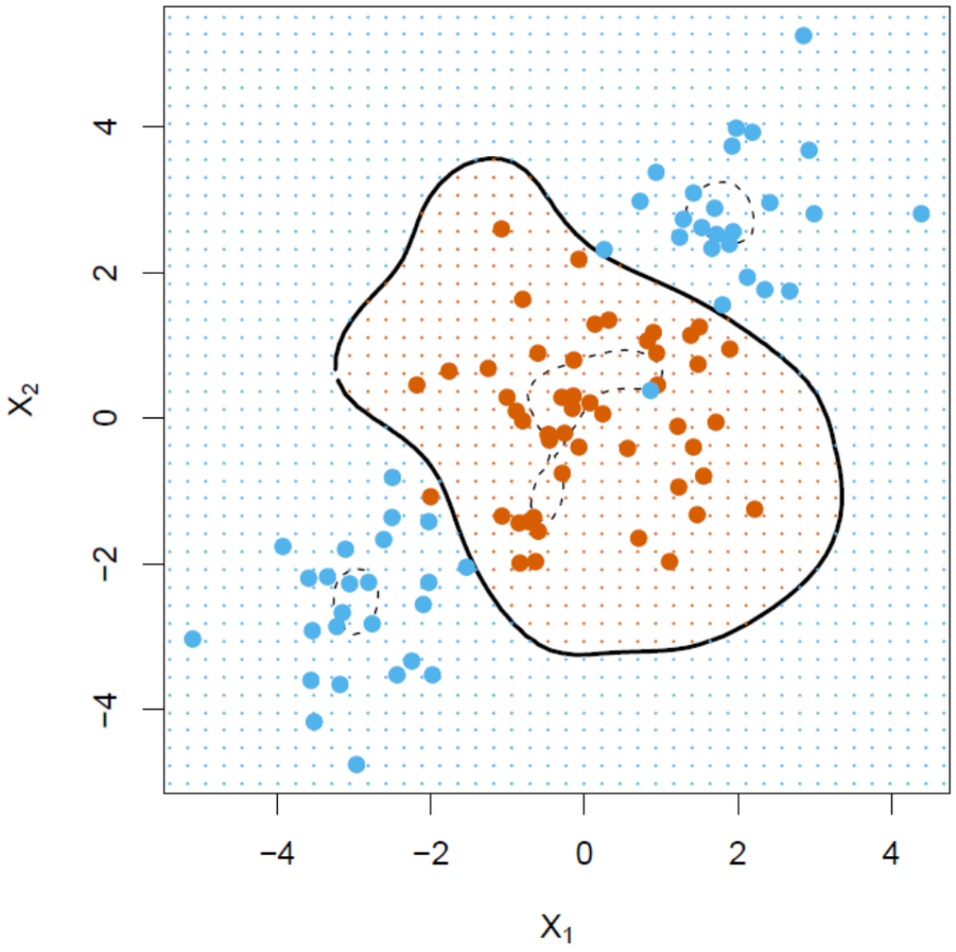






useless data well classified $\alpha = 0$	important data support $0 < \alpha < C$	suspicious data $\alpha = C$

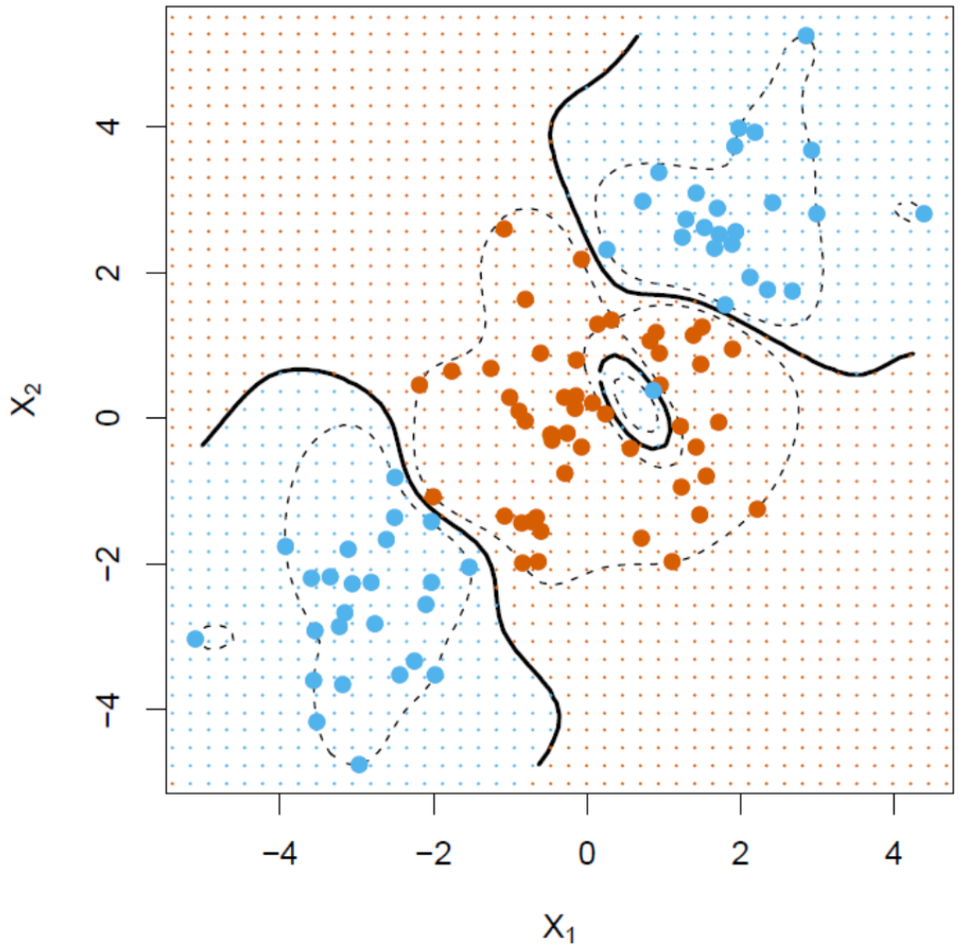
INFLUENCE OF THE PARAMETER C



Small C :

more errors, large margin

under fitting risk



Large C :

few errors, tiny margin

overfitting risk

SUPPORT VECTOR REGRESSION (SVR)

- We have a training dataset $(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}_1, y_i), \cdots, (\mathbf{x}_N, y_N)\}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$
- We seek a linear function $f : \mathcal{X} \to \mathbb{R}$ that gives the best fit with at most an ε deviation from the target
- This means that a tube of radius ε is fitted to the data
- The linear regression model is $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$
- The problem is formulated as minimizing

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}|y_i - f(\mathbf{x}_i)|_\varepsilon$$

with $$|y - f(\mathbf{x})|_\varepsilon = \max(0, |y - f(\mathbf{x})| - \varepsilon)$$

SUPPORT VECTOR REGRESSION (SVR)

- This is reformulated using slack variables to allow points to be outside of the tube
- We need two types of slack variables $\xi, \hat{\xi}$ for the cases

$$f(\mathbf{x}_i) - y_i > \varepsilon \quad \textit{and} \quad y_i - f(\mathbf{x}_i) > \varepsilon$$

- Above the ε-tube

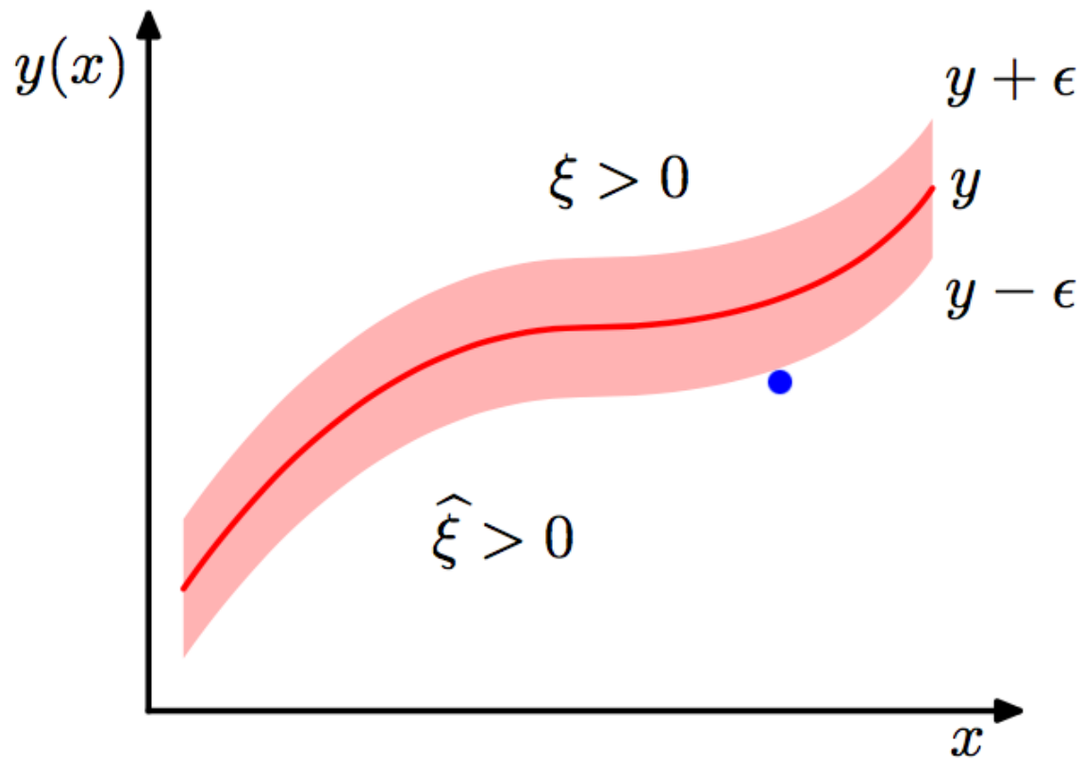
$$\xi > 0, \hat{\xi} = 0$$

- Below the ε-tube

$$\xi = 0, \hat{\xi} > 0$$

- Inside the ε-tube

$$\xi = \hat{\xi} = 0$$



SLACK VARIABLES FOR SVR

- The condition for a point to be in the tube is

$$f(\mathbf{x}_i) - \varepsilon \leq y_i \leq f(\mathbf{x}_i) + \varepsilon$$

- With the slack variables we get

$$f(\mathbf{x}_i) - \varepsilon - \hat{\xi}_i \leq y_i \leq f(\mathbf{x}_i) + \varepsilon + \xi_i$$

- We can reformulate the problem as minimizing

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}(\xi_i + \hat{\xi}_i) \qquad \textit{under the constraints}$$

$$y_i \leq f(\mathbf{x}_i) + \varepsilon + \xi_i$$

$$y_i \geq f(\mathbf{x}_i) - \varepsilon - \hat{\xi}_i$$

$$\hat{\xi}_i \geq 0 \qquad \xi_i \geq 0$$

SVR DUAL FORMULATION

- Using Lagrange multipliers and KKT conditions we can express the dual formulation as maximizing

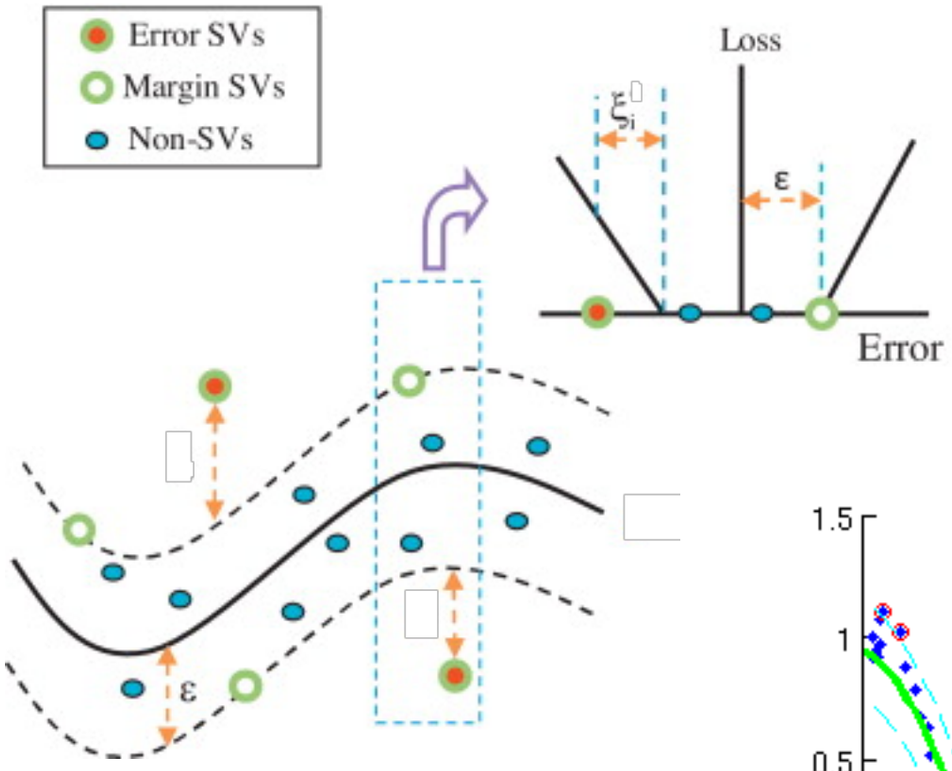
$$L(\boldsymbol{\alpha}, \hat{\boldsymbol{\alpha}}) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i - \hat{\alpha}_i)(\alpha_j - \hat{\alpha}_j)K(\mathbf{x}_i, \mathbf{x}_j)$$
$$-\varepsilon\sum_{i=1}^{N}(\alpha_i + \hat{\alpha}_i) + \sum_{i=1}^{N}(\alpha_i - \hat{\alpha}_i)y_i$$

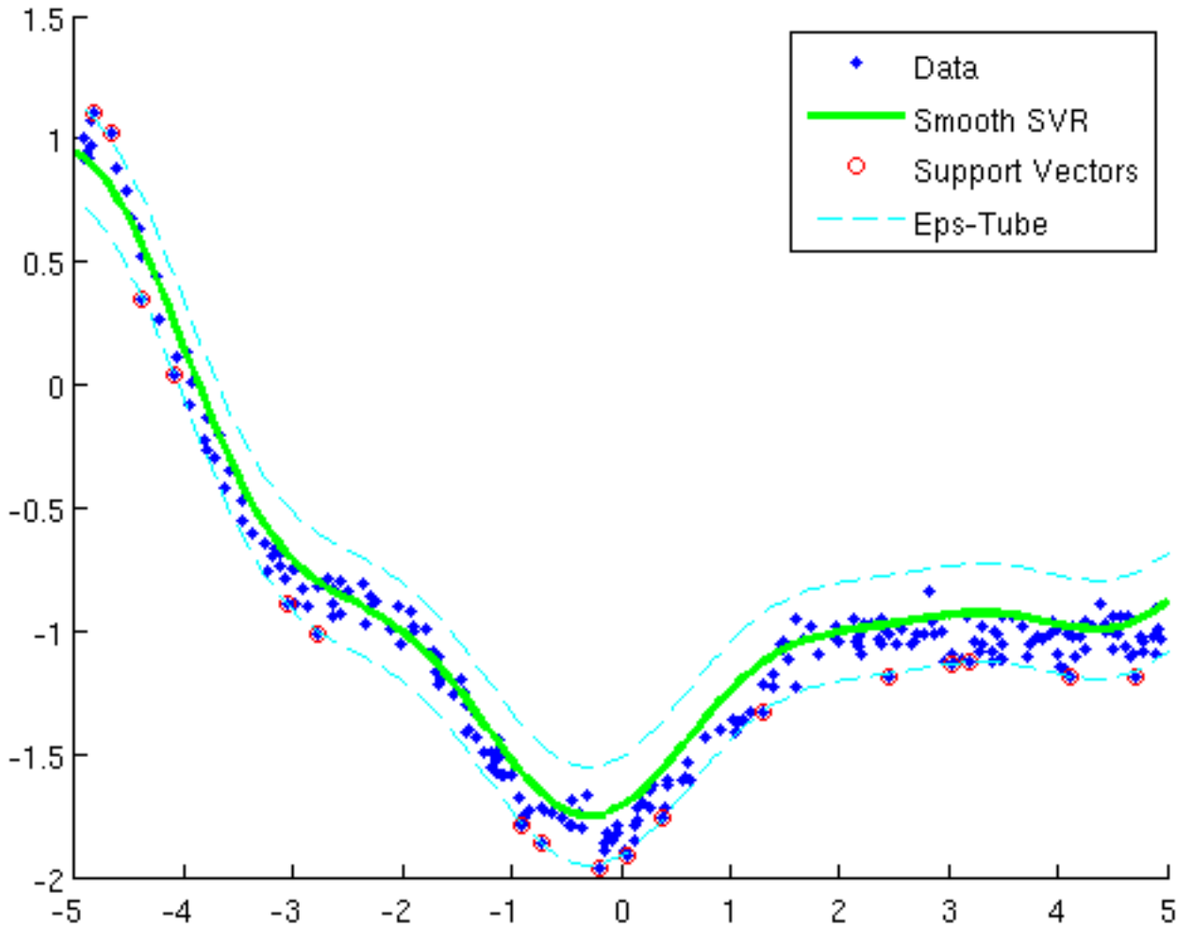
$$\textit{with} \quad 0 \leq \alpha_i \leq C \qquad 0 \leq \hat{\alpha}_i \leq C \qquad \sum_{i=1}^{N}(\alpha_i - \hat{\alpha}_i) = 0$$

- New predictions can be made with

$$D(\mathbf{x}) = \sum_{i=1}^{N}(\alpha_i - \hat{\alpha}_i)K(\mathbf{x}_i, \mathbf{x}) + b$$

SVR


Error SVs
Margin SVs
Non-SVs
Loss
ξi
ε
Error
b
ε


Data
Smooth SVR
Support Vectors
Eps-Tube
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-5
-4
-3
-2
-1
0
1
2
3
4
5

LEARNING OBJECTIVES

- Risk minimization
- Support Vector Machines
- Soft-margin SVMs
- Kernels and Non-linear SVMs
- **Multi-class SVMs**

MULTICLASS CLASSIFICATION

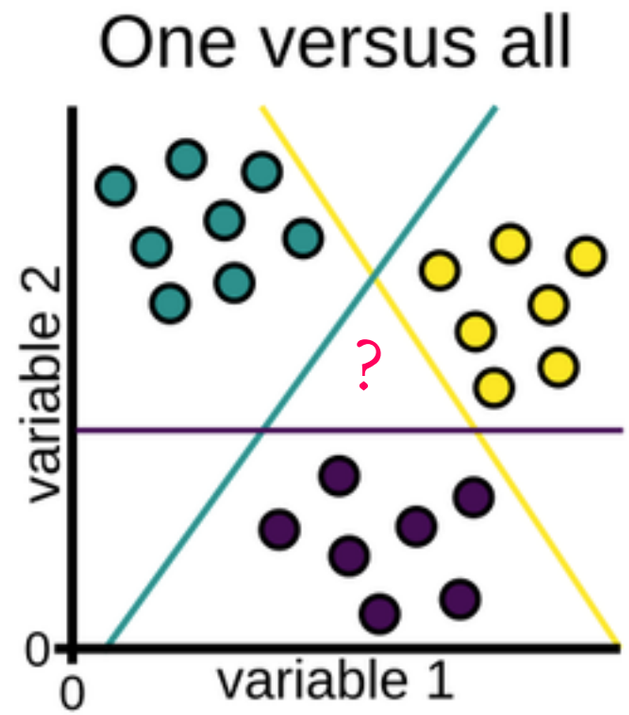
- SVMs are a 2 class classifier
- Common approaches consist in
  - dividing the problem into several binary classifications sub problems
  - combining the sub-classifications results to take a decision
- A typical approach is
  - **One versus All** : one class versus all the other

ONE VERSUS ALL

- For a classification problem into *K* classes, *K* separate SVMs are considered
- The k^{th} SVM is trained using the data from the class C_k as the positive examples and the data from the remaining *K-1* classes as negative examples
- The SVM with the highest output is selected as the winning class

$$y(\mathbf{x}) = \arg\max_k y_k(\mathbf{x})$$

- Typical problems:
  - Can lead to inconsistent results
  - No guaranty that the outputs of the classifiers are on the same scale : use probabilities instead !
  - The training sets are imbalanced : more negative examples than positive ones



PROBABILISTIC OUTPUT FOR SVMS

- The outputs of SVMs are NOT probabilistic outputs
- Platt has proposed a scaling algorithm for that:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(Af(\mathbf{x}) + B)}$$

- It performs a logistic transformation of the classifier scores using two learned parameters A and B
- A and B are estimated by minimizing the cross-entropy error function on a training set consisting of pairs of values from the SVM output and the target classes $(y(\mathbf{x}_i), y_i)$

EXAMPLE FOR A ONE VERSUS ALL APPROACH