UNIVERSITAT DE BARCELONA

Introduction to Machine Learning

Master in Artificial Intelligence

UPC, UB, URV

UNIVERSITAT DE BARCELONA

Week 5

Course. Introduction to Machine Learning

Theory 5. Support Vector Machine

Dr. Maria Salamó Llorente

maria.salamo@ub.edu

Dept. Mathematics and Informatics,

Faculty of Mathematics and Informatics,

University of Barcelona (UB)

Introduction to Machine Learning

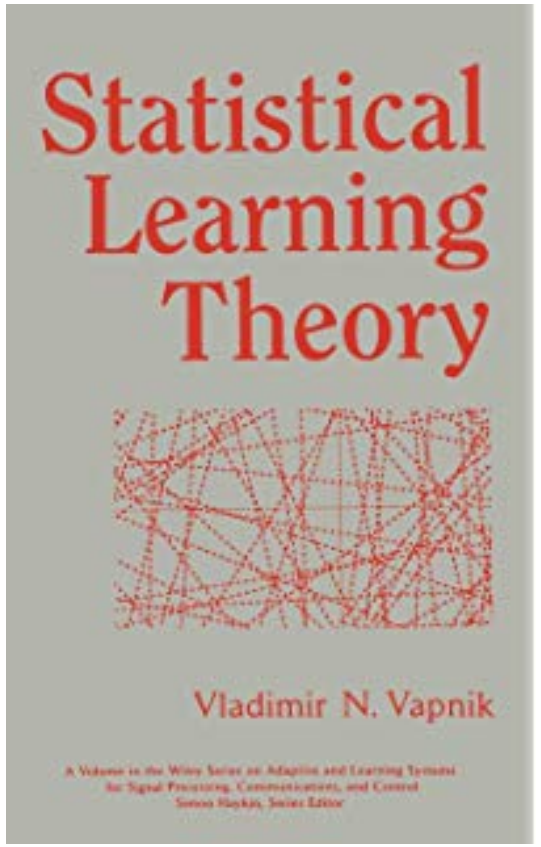
Supervised Learning					Decision Learning Theory	Unsupervised Learning			Machine Learning in practice	
Non Linear Decision				Linear Decision	Basic concepts of Decision Learning Theory	Cluster Analysis	Factor Analysis	Visualization	Applications of ML	Beyond ML
Lazy Learning (K-NN, IBL, CBR)	Overfitting, model selection	Feature selection	Kernel Learning	Perceptron, SVM	Bias/Variance, VC dimension, Practical advice of how to use learning algorithms	K-Means, Fuzzy C-means, EM	PCA, ICA	Self Organized Maps (SOM) , Multi-Dimensional Scaling	RecSys	Bias and Fairness in ML

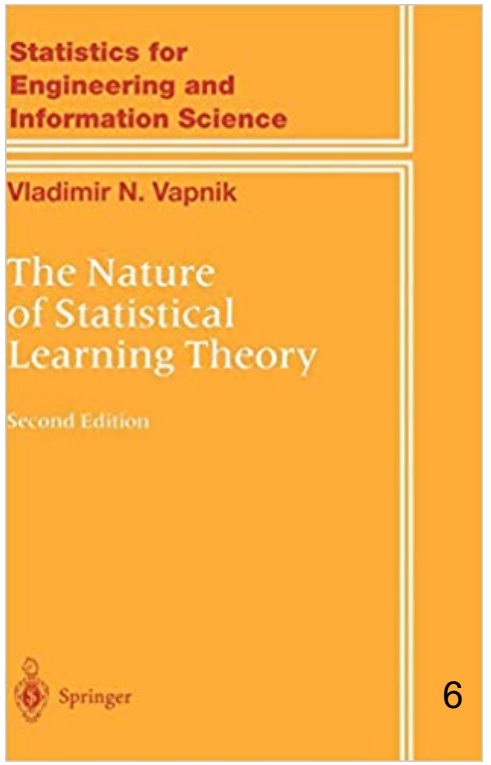
Contents

1. Introduction to Support Vector Machine

Introduction to SVM

- Theoretically well motivated algorithm: developed from **Statistical Learning Theory** (Vapnik & Chervonenkis) since the 60s.
- SVMs were introduced in **COLT-92** by Boser, Guyon & Vapnik. Become rather popular since
- **Empirically good performance**: successful applications in many fields (bioinformatics, text, image recognition, pattern analysis, …)





Introduction to SVM

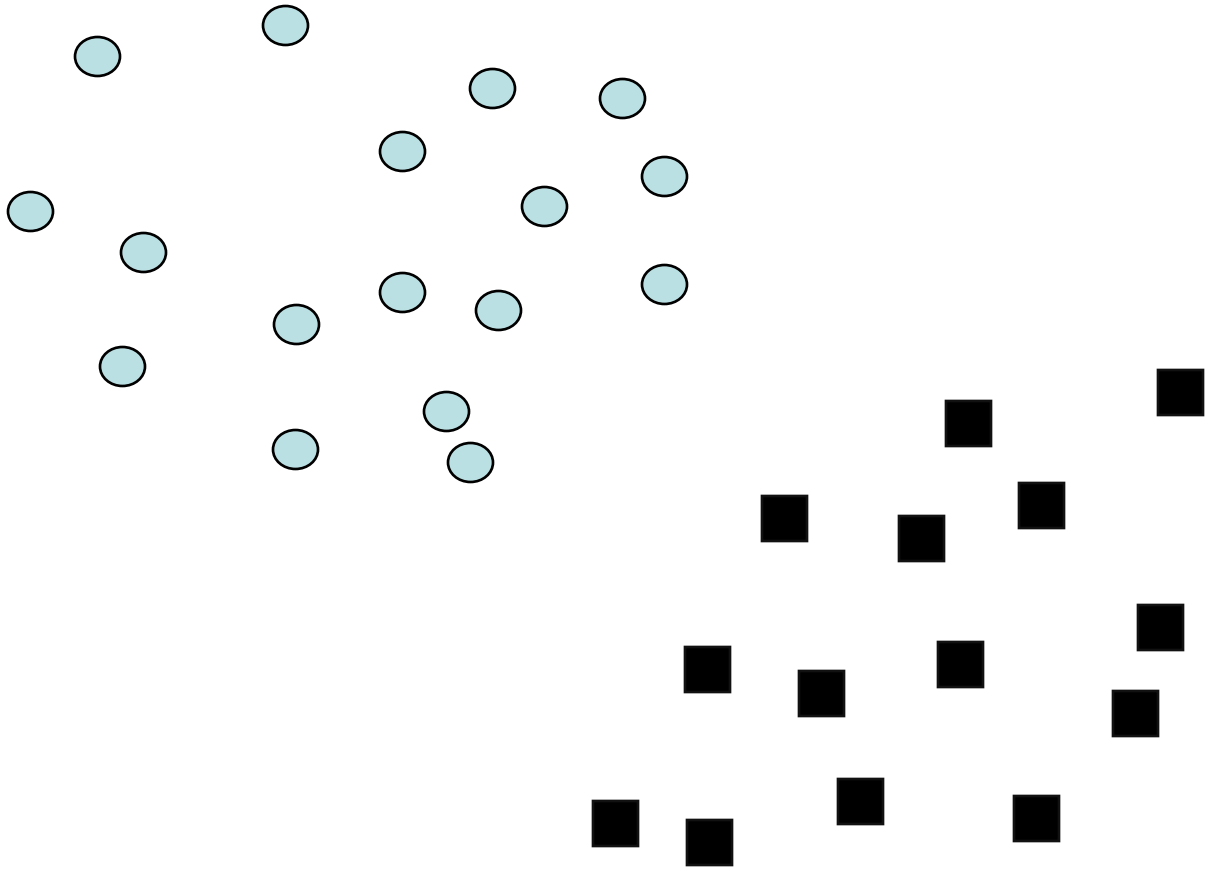
- SVM is a **supervised machine learning algorithm** which can be used for **classification** or **regression**
- The **objective** of the Support Vector Machine is to find **the best splitting boundary between data**
- They belong to the category of **linear classifiers**
  - It plots each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate
  - Classification is done by **finding the separating hyperplane** that differentiate the two classes very well

Hyperplane

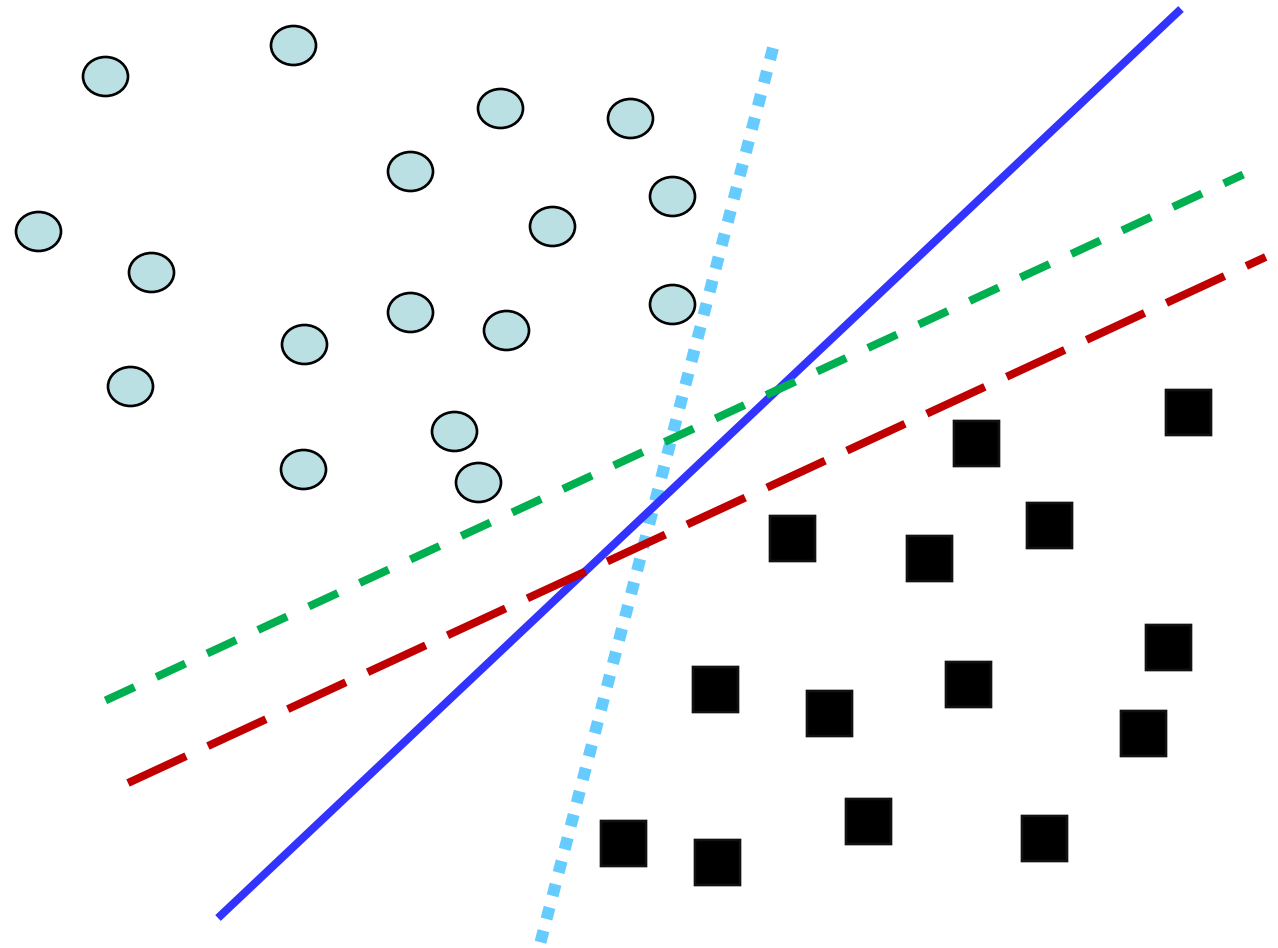
- A **hyperplane** is a generalization of a plane
  - In 2 dimensions: can separate by a line
  - In 3 dimensions: can separate by a plane
  - In higher dimensions: need *hyperplanes*

- A **hyperplane** is an affine subspace of dimension *n-1*, which divides the space into two half spaces which correspond to the inputs of 2 distinct classes

Identifying the right hyperplane

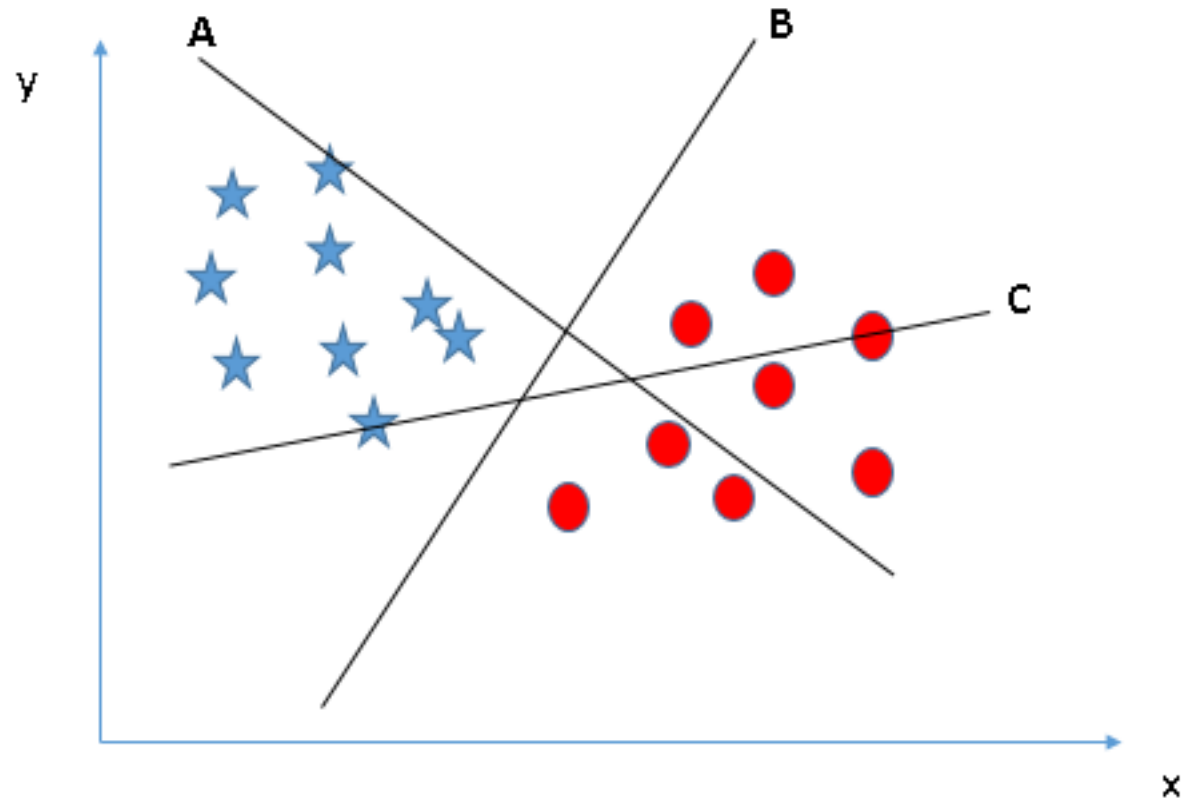


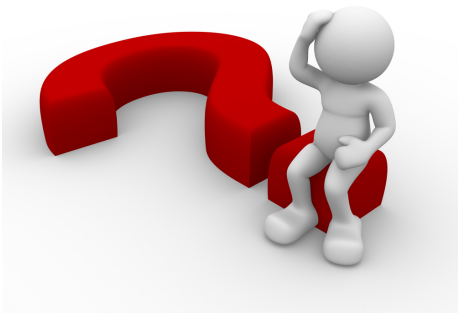
Identifying the right hyperplane



Identifying the right hyperplane

Scenario 1

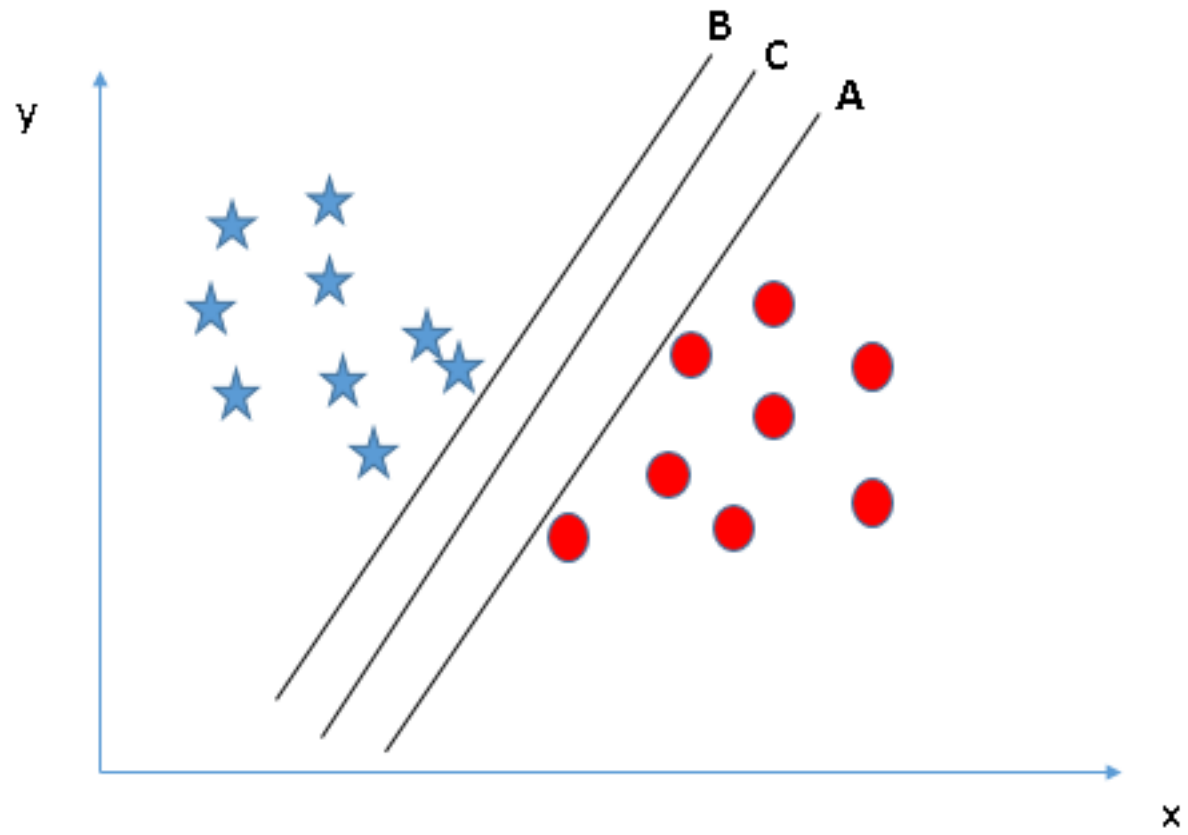


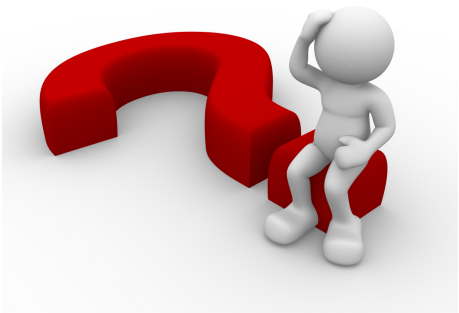


Select the hyperplane that separates/segregates the two clases better?

Identifying the right hyperplane

Scenario 2

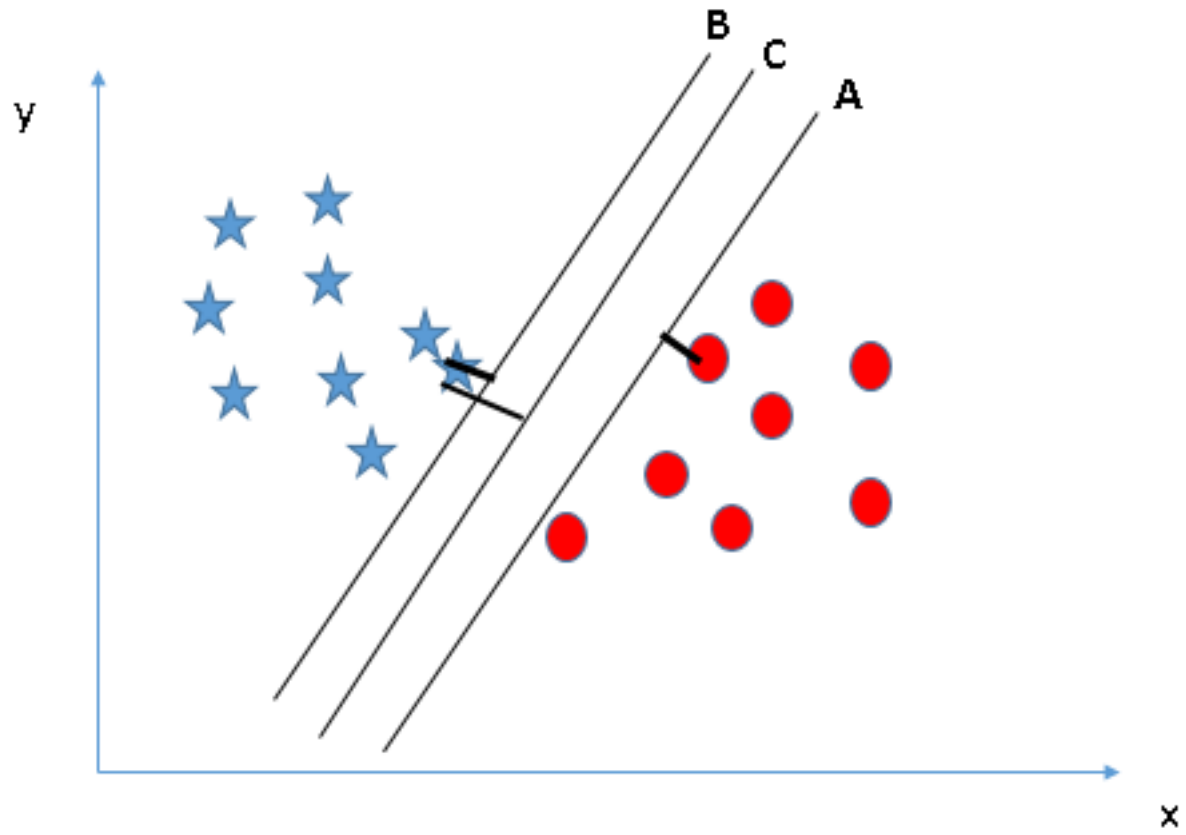




How can we identify the right hyperplane?

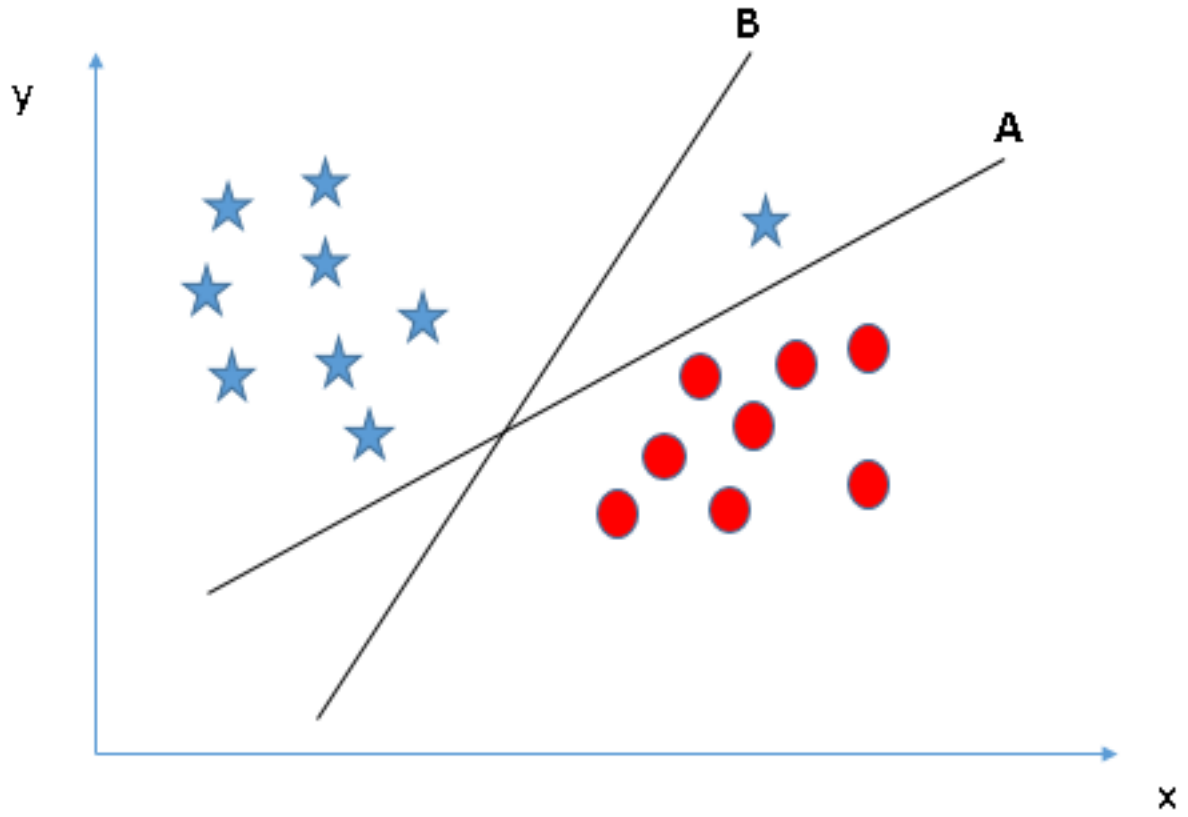
Identifying the right hyperplane

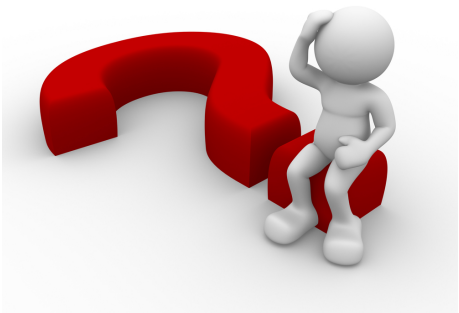
Scenario 2



Identifying the right hyperplane

Scenario 3

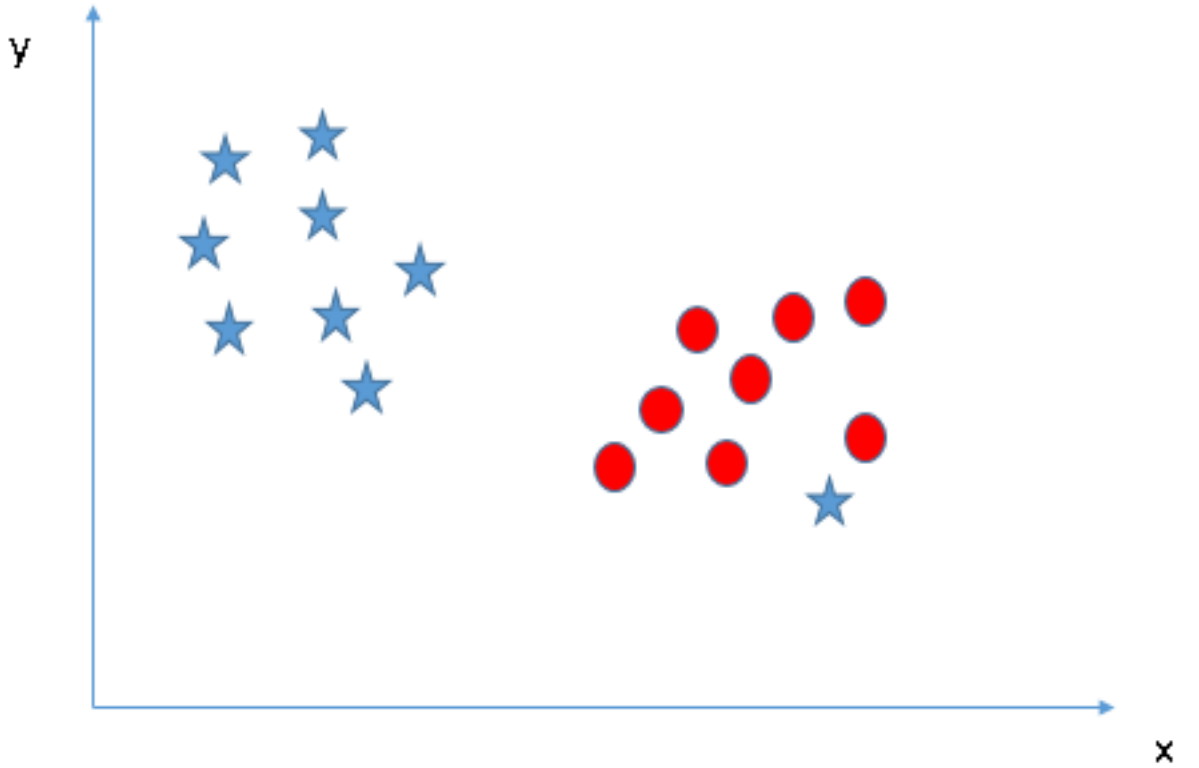


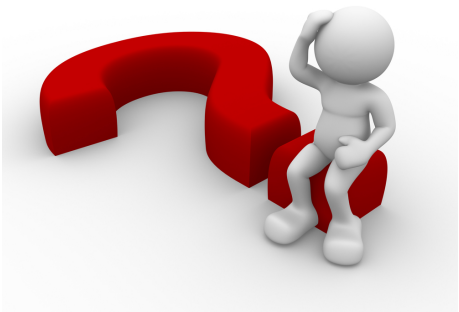


Use the rules discussed to select the right hyperplane

Identifying the right hyperplane

Scenario 4

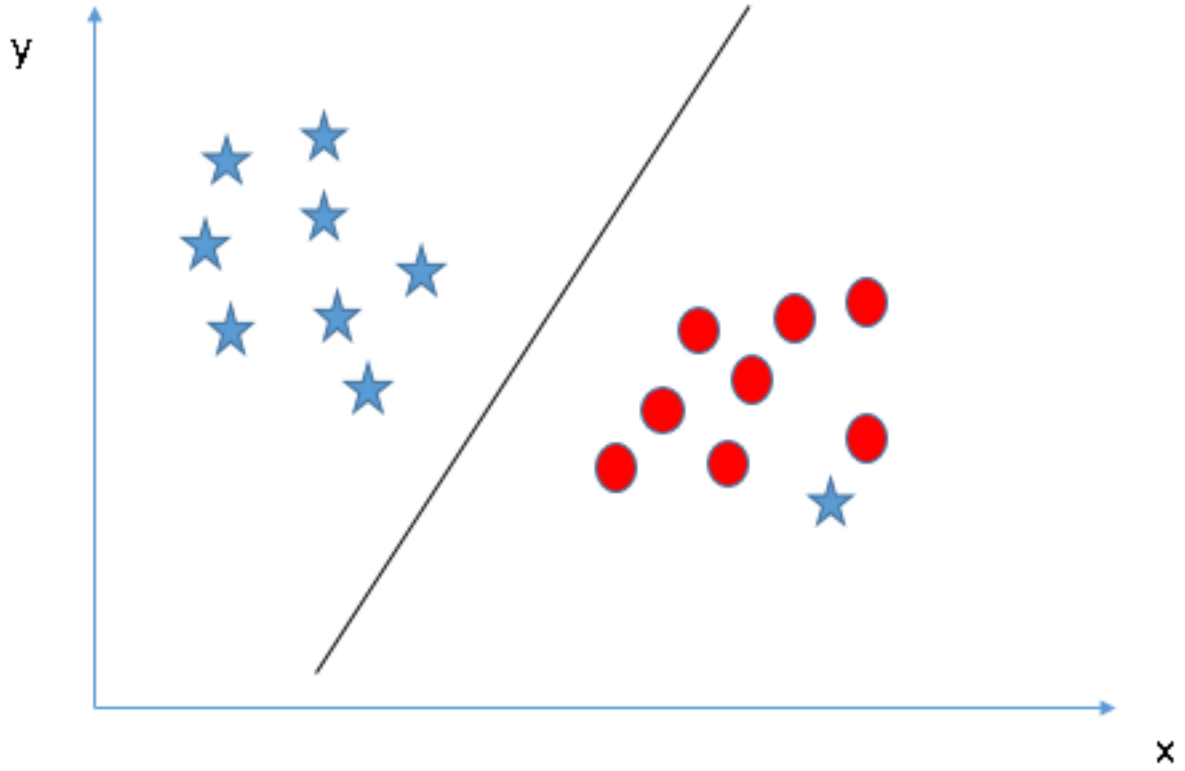




Select the hyperplane that separates/segregates the two clases better?

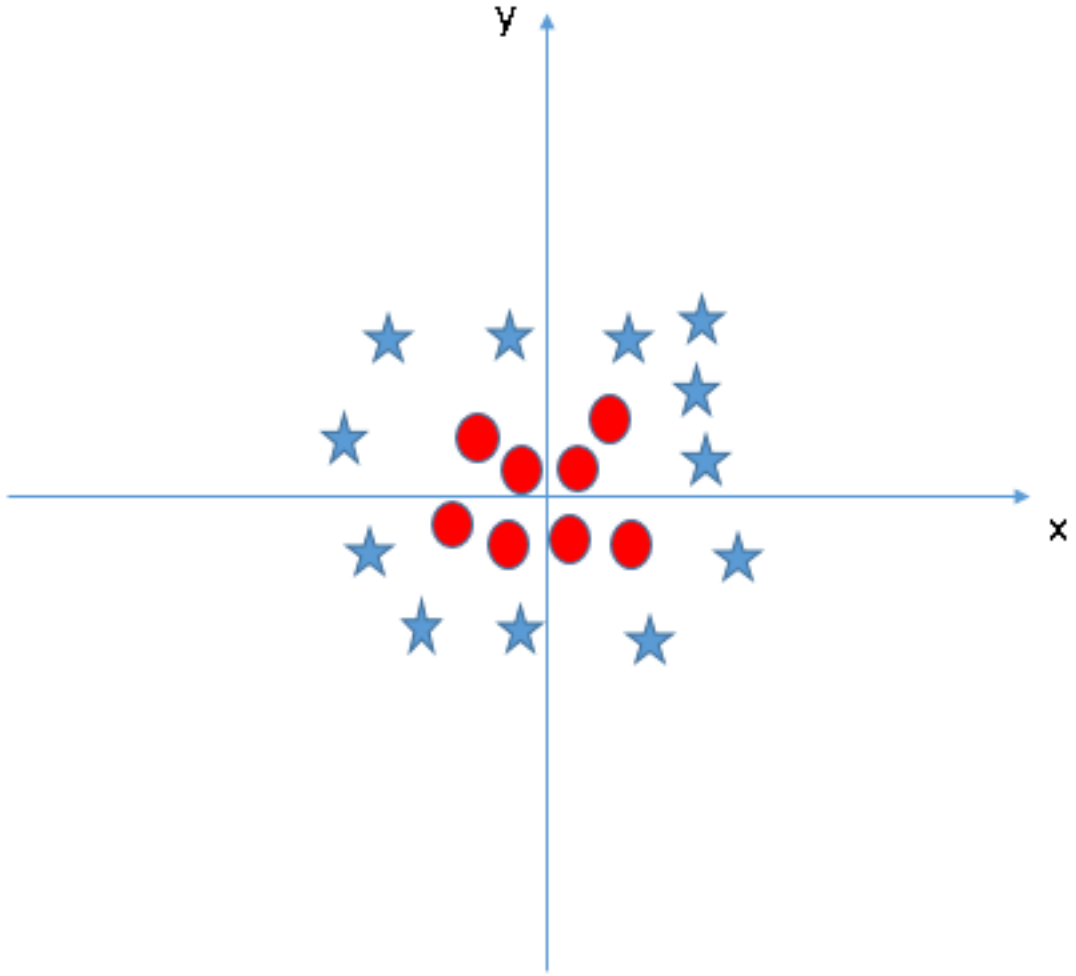
Identifying the right hyperplane

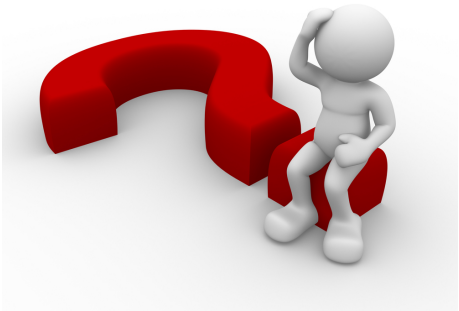
Scenario 4



Identifying the right hyperplane

Scenario 5

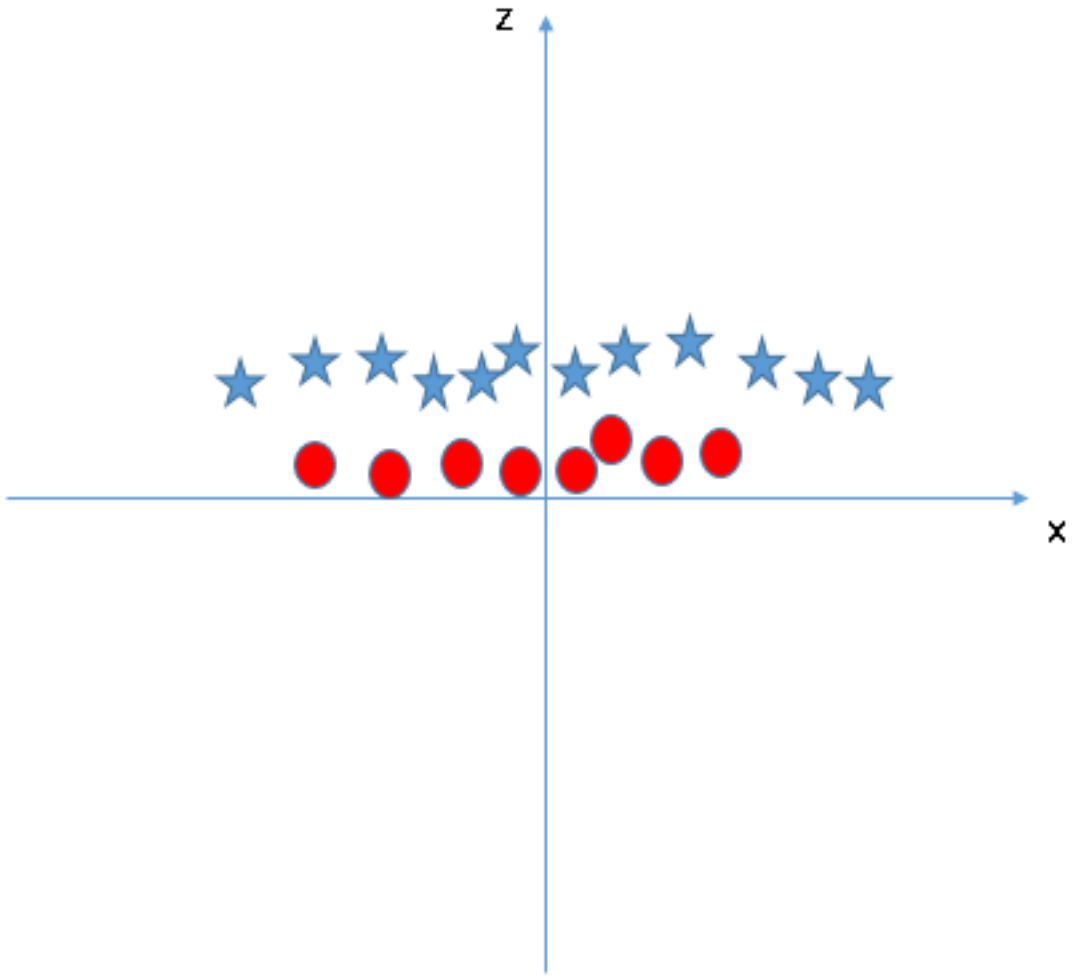




Select the hyperplane that separates/segregates the two clases better?

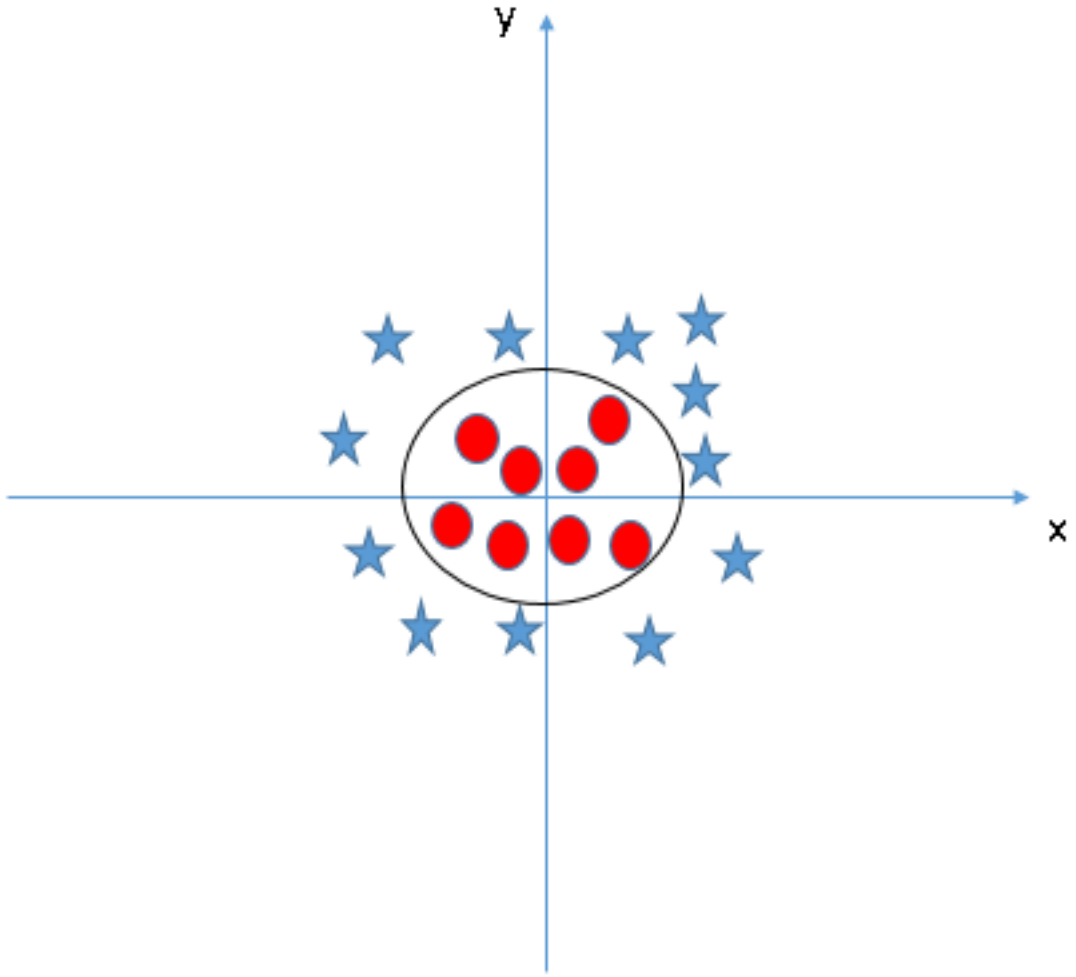
Identifying the right hyperplane

Scenario 5



Identifying the right hyperplane

Scenario 5

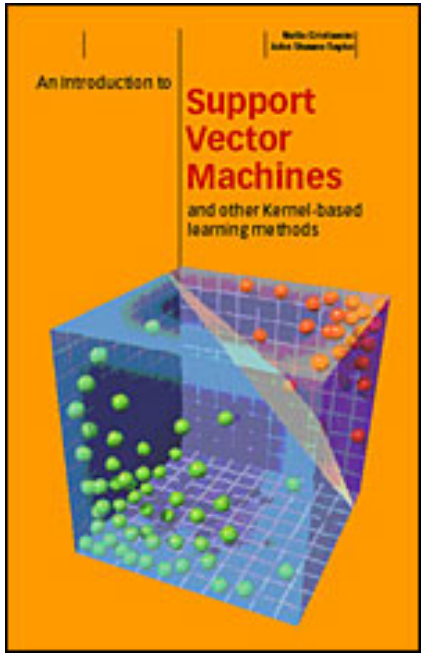


2. Defining SVM conceptually

Definition

Formal Definition: 'Support Vector Machine is a system for efficiently training linear learning machines in kernel-induced feature spaces, while respecting the insights of generalization theory and exploiting optimization theory.'

- **AN INTRODUCTION TO SUPPORT VECTOR MACHINES (and other kernel-based learning methods)**
  N. Cristianini and J. Shawe-Taylor
  Cambridge University Press
  2000 ISBN: 0 521 78019 5
- **Kernel Methods for Pattern Analysis**
  John Shawe-Taylor & Nello Cristianini
  Cambridge University Press, 2004



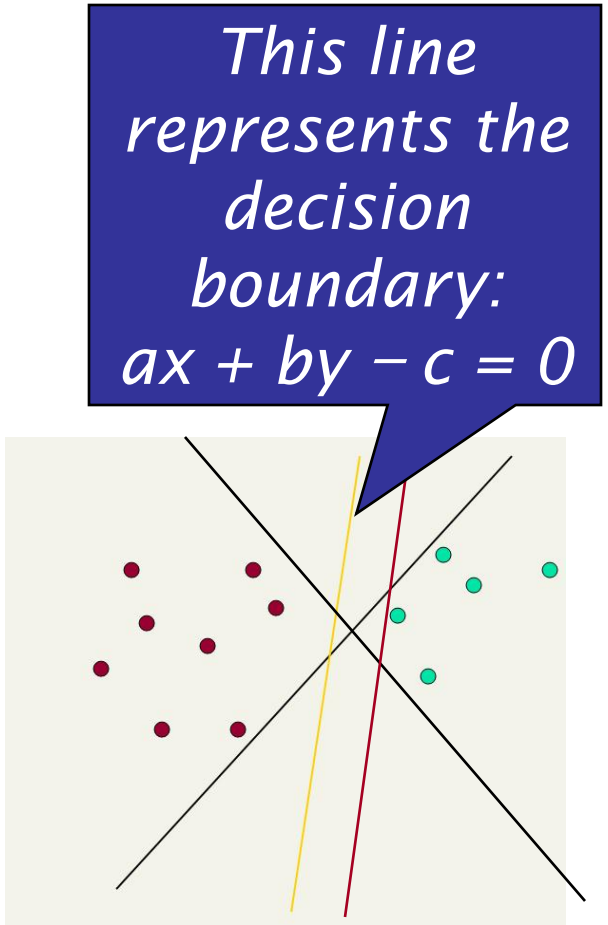
Concepts: 2D Case

The aim of the SVM classification is to devise a computationally efficient way of learning "good" separating hyperplanes in a high dimensional feature space

Find a, b, c such that

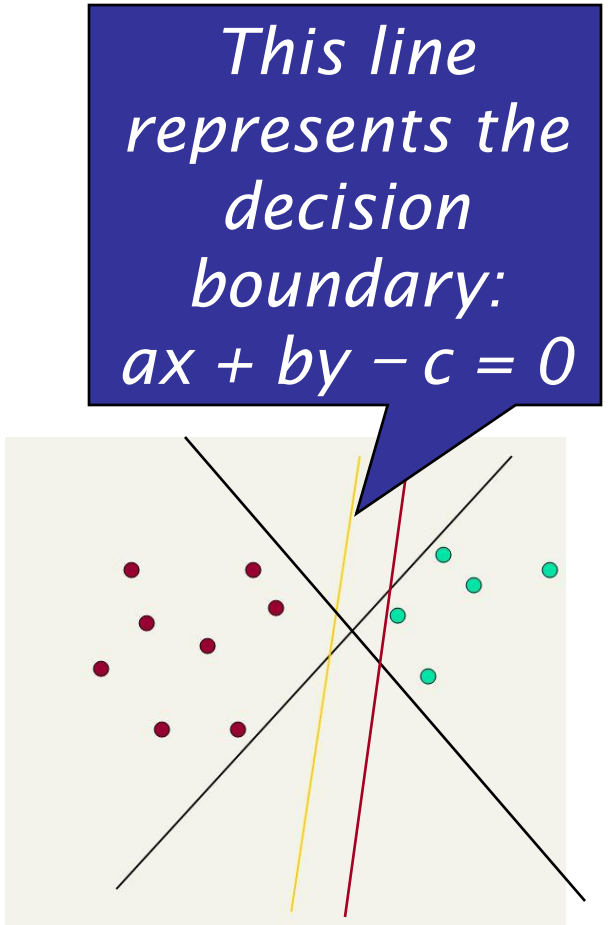
$ax + by \geq c$ **for red points**

$ax + by \leq (or < c)$ **for green points**



Concepts of SVM

- Lots of possible solutions for *a, b, c.*
- Some methods find a separating hyperplane, but not the optimal one [according to some criterion of expected goodness]
  - E.g., perceptron
- Support Vector Machine (SVM) finds an optimal solution.
  - Maximizes the distance between the hyperplane and the "difficult points" close to decision boundary
  - One intuition: if there are no points near the decision surface, then there are no very uncertain classification decisions



Concepts SVM

- By "*good hyperplanes*" we will understand the ones optimizing the generalization bounds
  - Different generalization bounds exist
    - Optimize the maximal margin
    - Margin distribution
    - Number of support vectors, etc…
  - The most common and well stablished is to minimizing the norm of the weight vector
- The simplest model of SVM is **a maximal margin classifier**

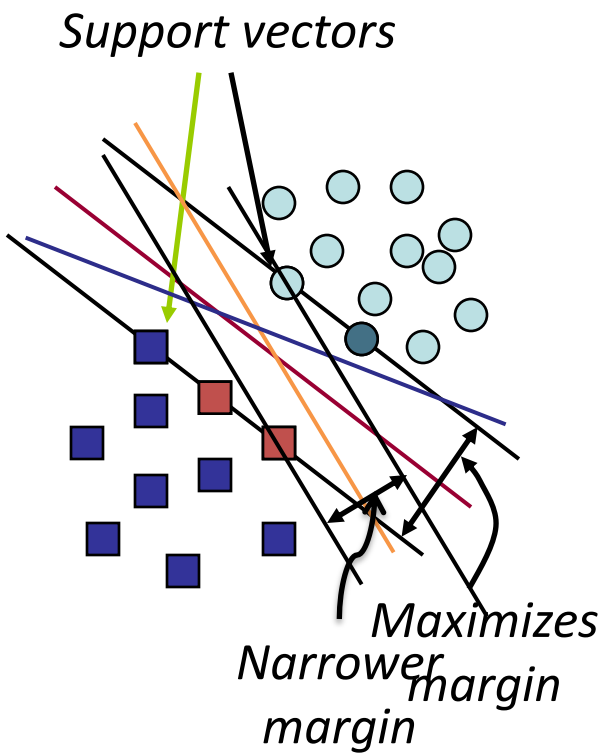
Maximal Margin Classifier

- It works only for data that is linearly separable
  - Hence can not be used in many real-world problems
  - However, it is the most simple to understand the building blocks of SVM

- The maximal margin classifier optimizes the generalization error by separating the data with the maximal margin hyperplane,
  - Given that the generalization does not depend on the dimensionality of the space, this separation can be sought in any kernel-induced feature space

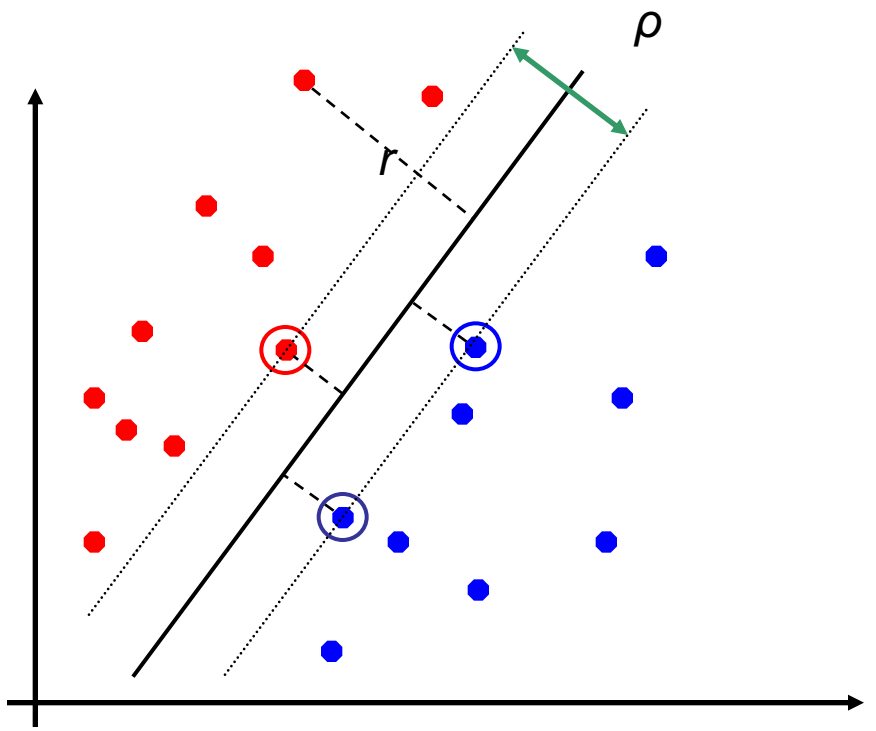
Margin and Support Vectors

- SVMs maximize the ***margin*** around the separating hyperplane.
  - A.k.a. large margin classifiers
- The decision function is fully specified by a subset of training samples, *the* ***support vectors***.
- Solving SVMs is a ***quadratic programming*** problem
- Seen by many as the most successful current classification method



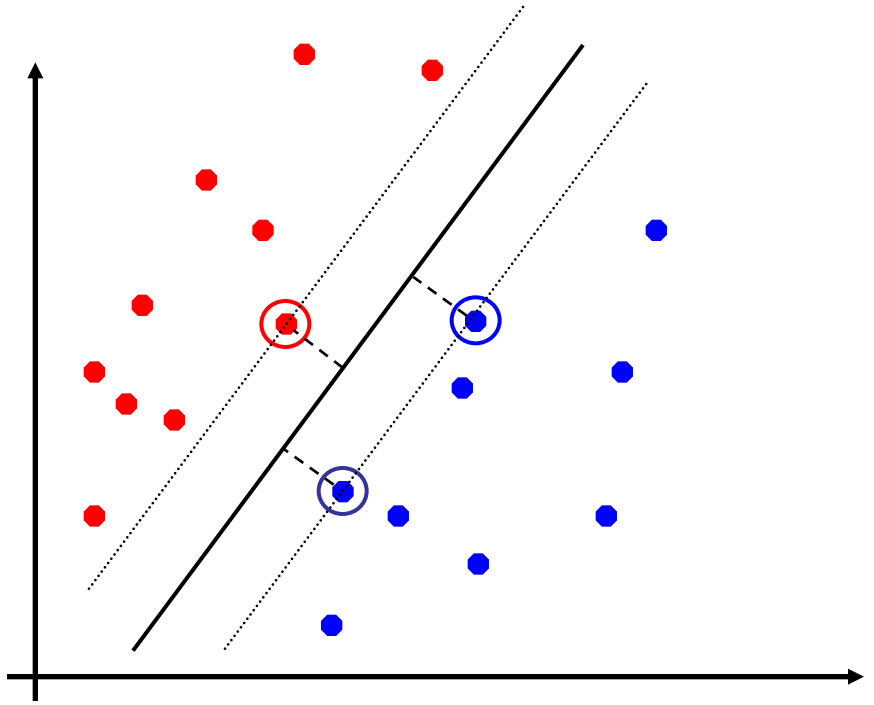
Classification Margin

- Distance from example data, x, to the separator is $r = \frac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|}$
- Data closest to the hyperplane are ***support vectors***.
- ***Margin*** ρ of the separator is the width of separation between classes.



Maximum Margin Classification

- Maximizing the margin is good according to intuition and theory.
- Implies that only support vectors are important; other training examples are ignorable.

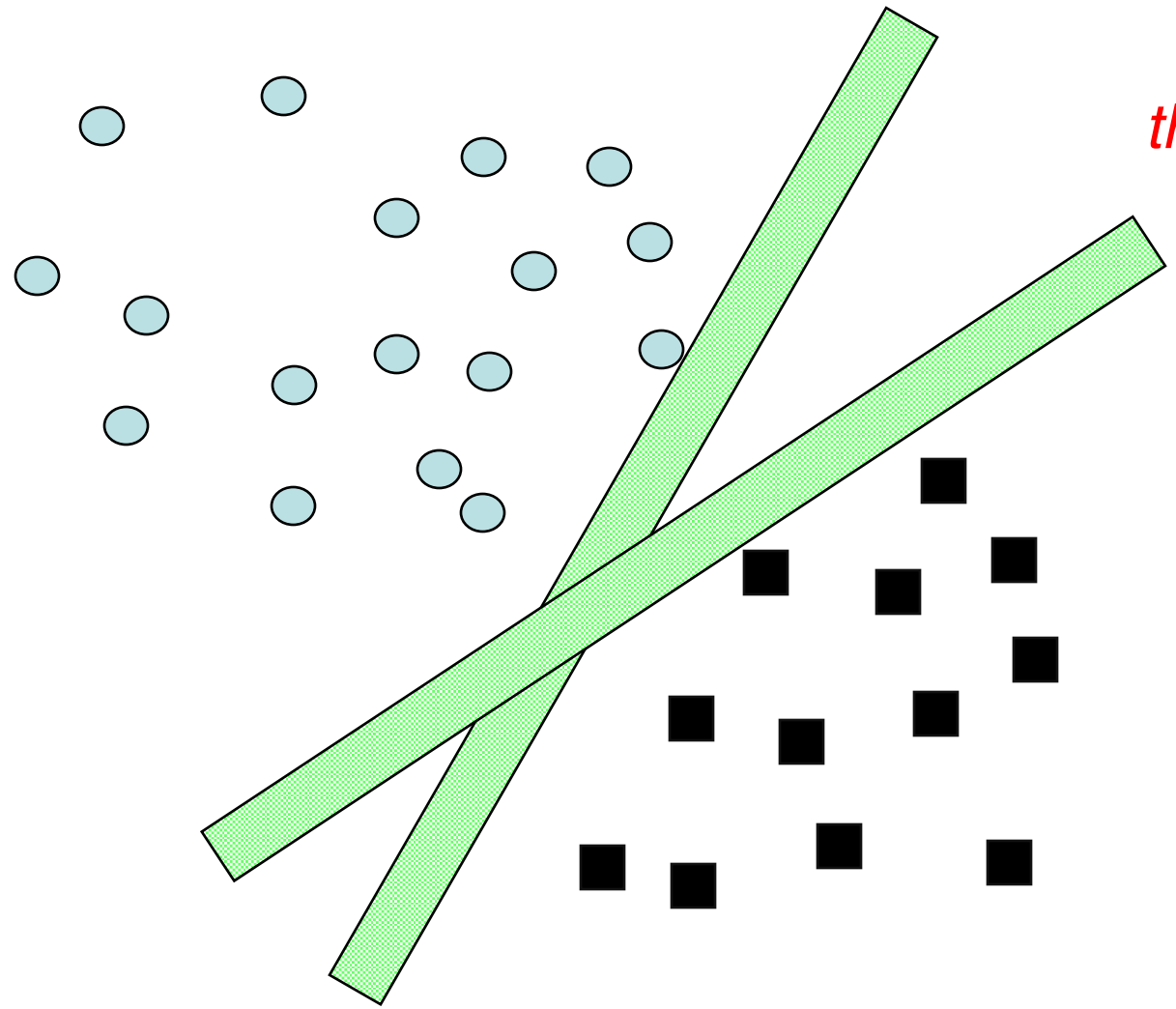


3. Linear SVM mathematically

Statistical Learning Theory

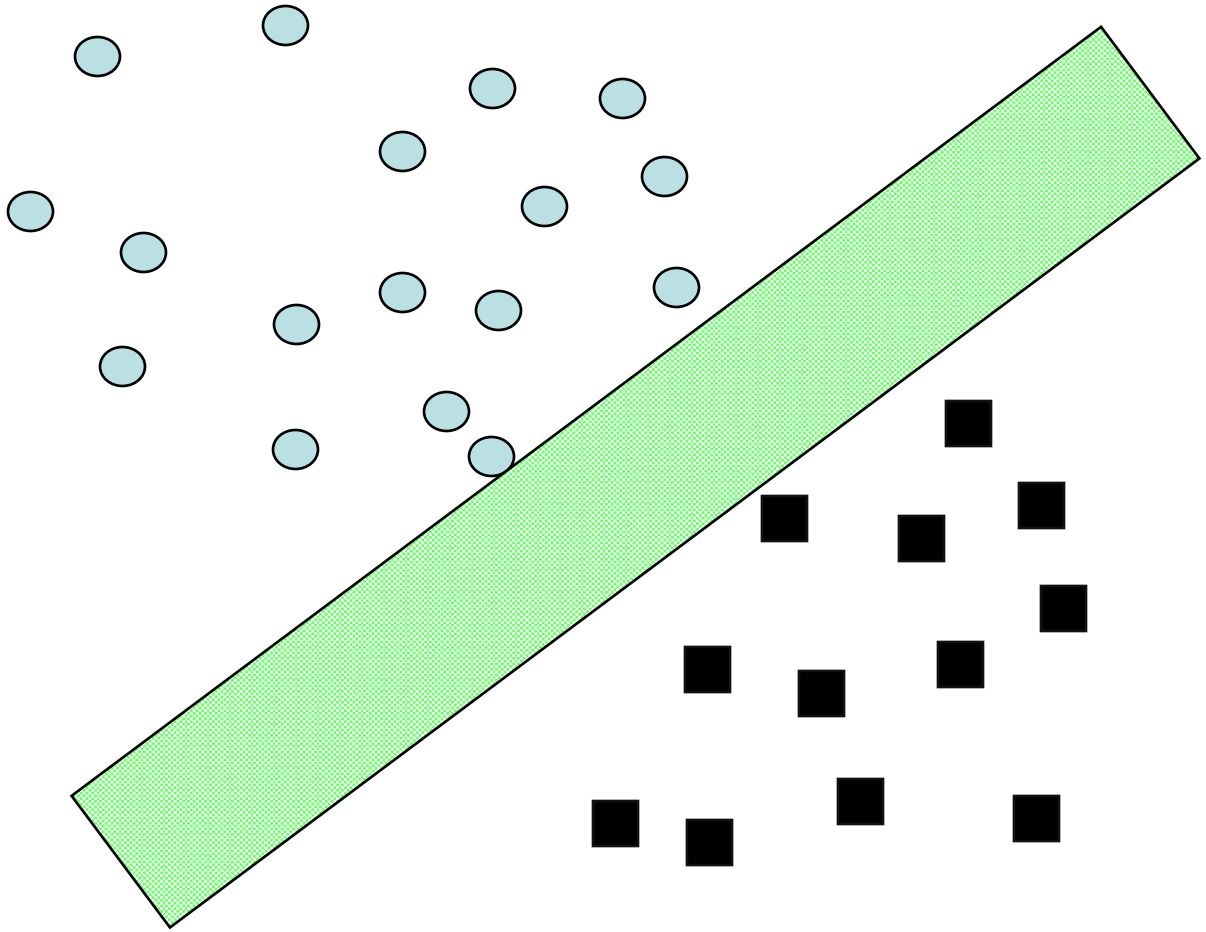
- **Generalization theory** gives guidance to prevent overfitting
  - Misclassification error and the function complexity bound **generalization error**.
  - Maximizing margins minimizes complexity.
  - "Eliminates" overfitting.
- **Optimization theory** provides mathematical techniques to find hyperplanes
  - Solution depends only on *Support Vectors* not on the number of attributes.

Margins and Complexity



Skinny margin
is more flexible
thus more complex

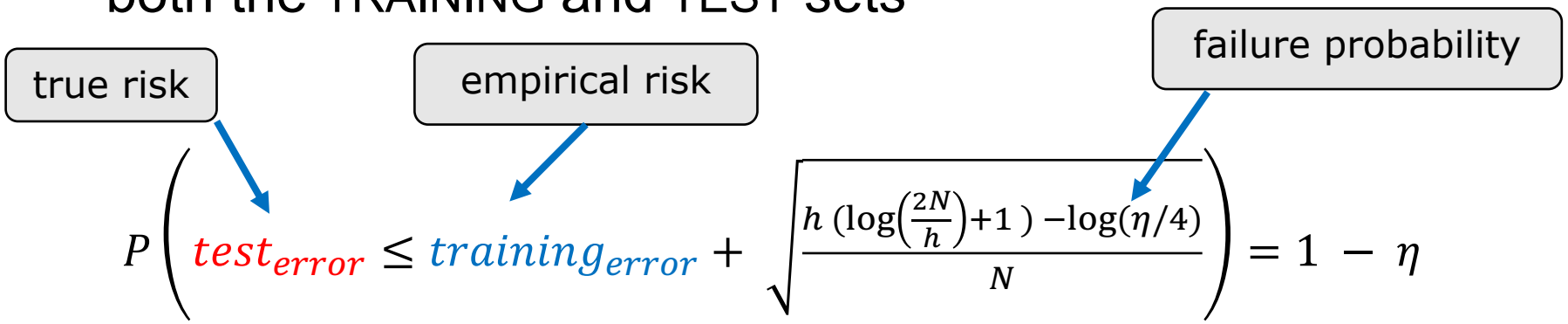
Margins and Complexity



Fat margin
is less complex

VC dimension

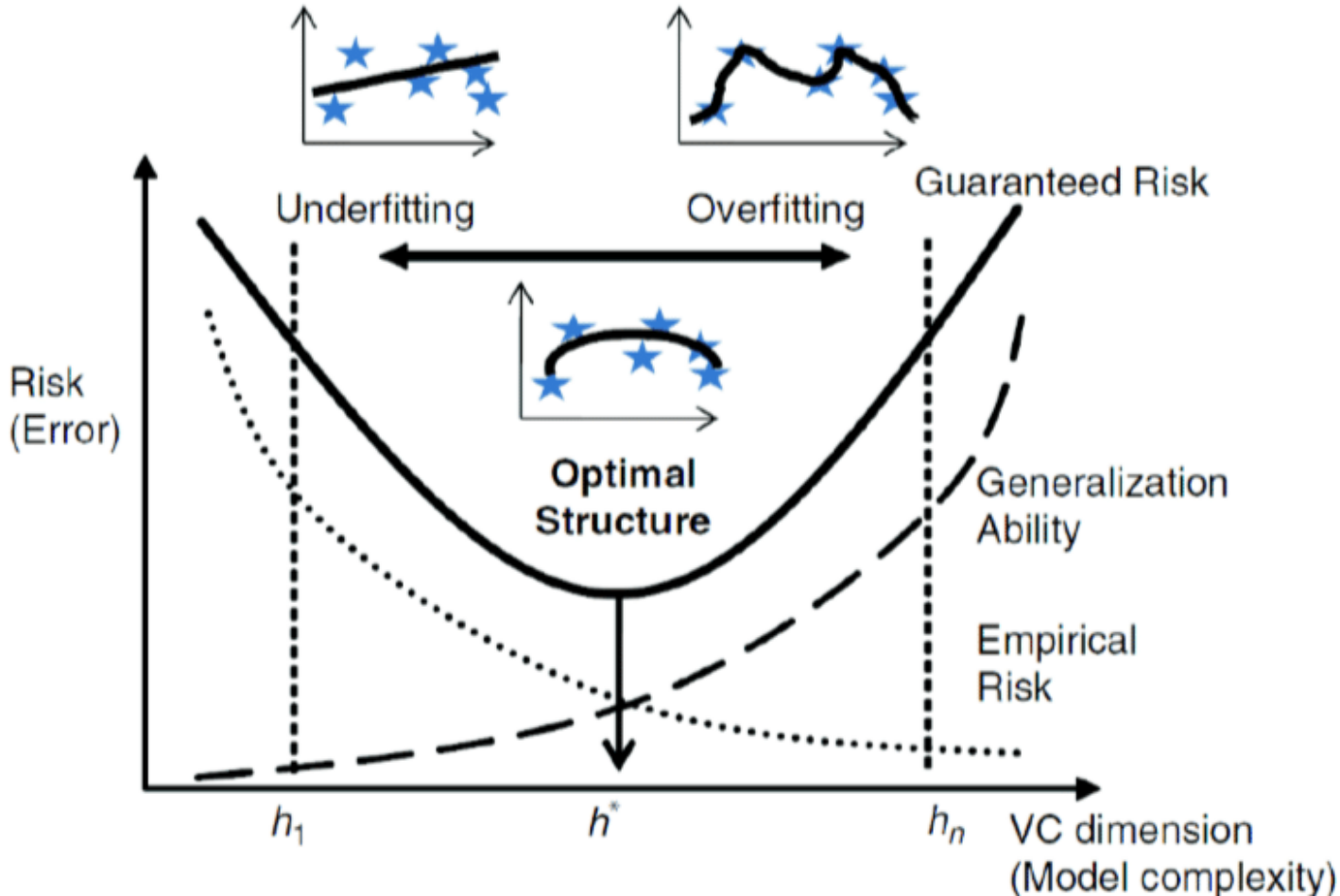
- The Vapnik-Chervonenkis dimension
  - Model complexity determines the PERFORMANCE/COST on both the TRAINING and TEST sets



  - h quantifies the complexity of the model (VC dimension)
  - N size on training data
  - The expression shows that the error calculated on the test set has an upper bound

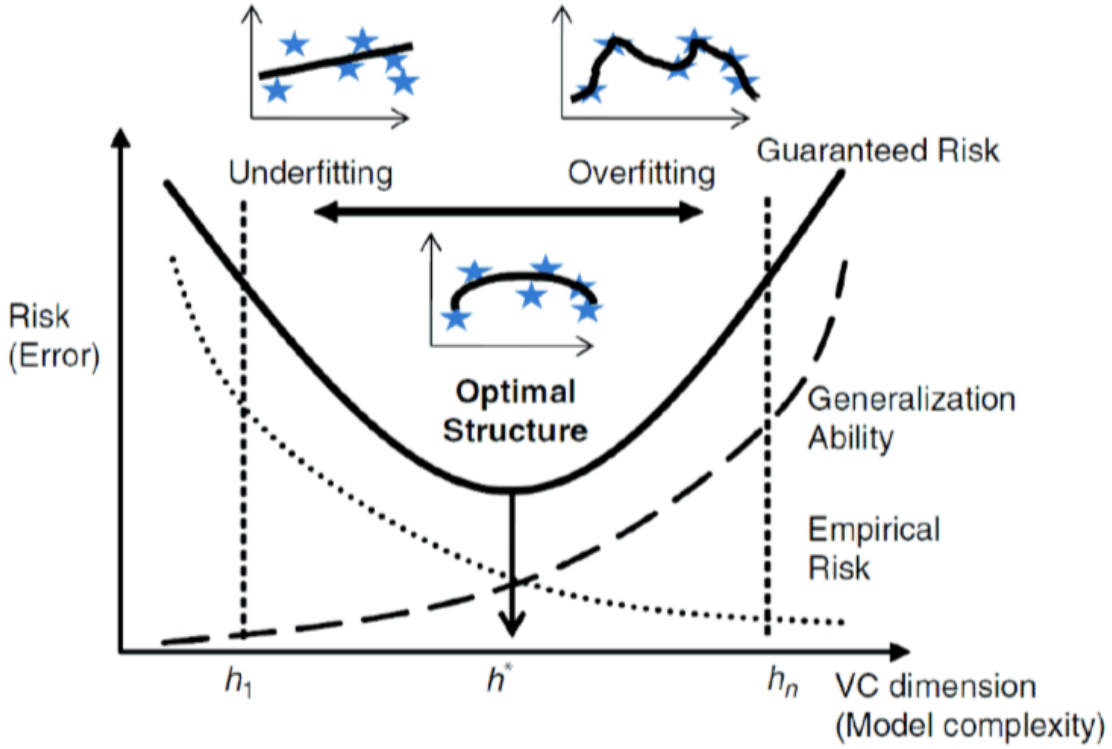
VC dimension

$$P\left(test_{error} \leq training_{error} + \sqrt{\frac{h\left(\log\left(\frac{2N}{h}\right)+1\right) - \log(\eta/4)}{N}} \right) = 1 - \eta$$



VC dimension

$$P\left(test_{error} \leq training_{error} + \sqrt{\frac{h\left(\log\left(\frac{2N}{h}\right)+1\right) - \log(\eta/4)}{N}} \right) = 1 - \eta$$



For **LINEAR CLASSIFIERS**

m: dimensions/features

$$h = m + 1$$

Mathematical definition

Given a **linearly separable** training sample

$$s = ((x_1, y_1), \cdots, (x_n, y_n))$$

where $x_i \in \mathbb{R}^n \; and \; y_i \in \{+1, -1\}$

the hyperplane (w, b) that solves the optimization problem

$$\begin{aligned} \boldsymbol{minimize}_{w,b} \quad & < w \cdot w >, \\ subject\ to \quad & y_i(< w \cdot x_i > + b) \geq 1, \\ & i = 1, \cdots, n \end{aligned}$$

realizes the maximal margin hyperplane with geometric margin

$$\rho = \frac{1}{\|w\|}$$

Hyperplane restrictions

- Assume that all data is at least distance 1 from the hyperplane, then the following two constraints follow for a training set $\{(\mathbf{x_i}, y_i)\}$

$$w^T x_i + b \geq 1 \quad if \; y_i = 1$$
$$w^T x_i + b \leq -1 \quad if \; y_i = -1$$

- For **support vectors**, the inequality becomes an equality
- Then, since each example's distance from the hyperplane

is $r = \dfrac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|}$ and the margin is: $\rho = \dfrac{1}{\|w\|}$

Maximum Margin: Formalization

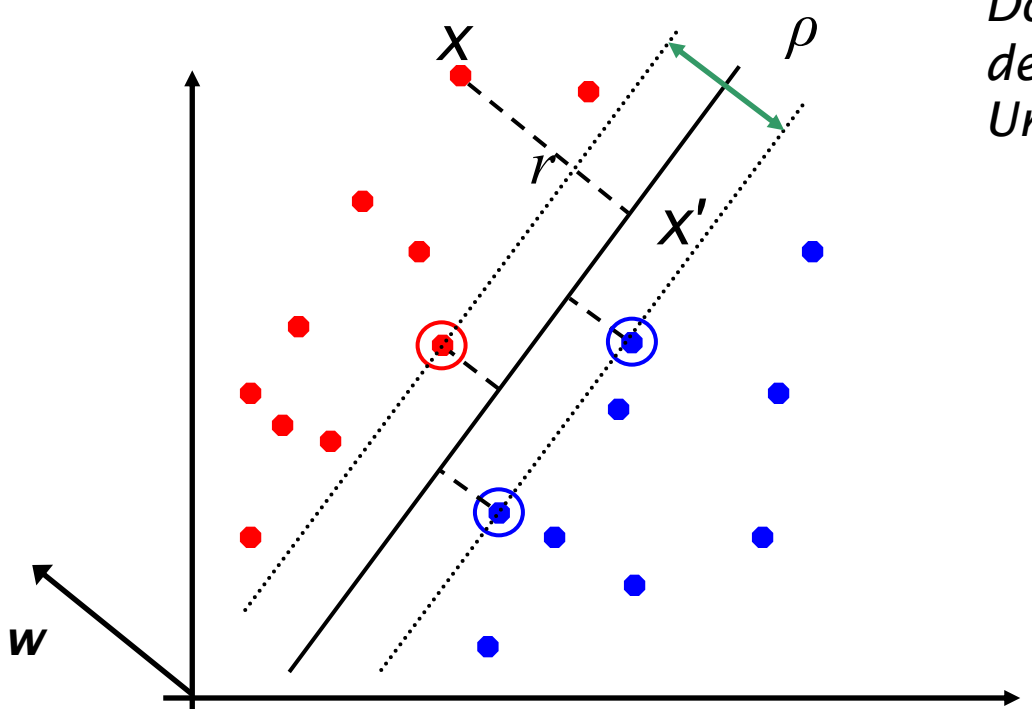
- **w**: decision hyperplane normal vector
- $\mathbf{x}_i$: data point *i*
- y_i: class of data point *i* (+1 or -1) NB: Not 1/0

- Classifier is: $f(\mathbf{x}_i) = \text{sign}(\mathbf{w}^T\mathbf{x}_i + b)$

- Functional margin of $\mathbf{x}_i$ is: $y_i (\mathbf{w}^T\mathbf{x}_i + b)$
  - But note that we can increase this margin simply by scaling **w**, **b**….
- Functional margin of dataset is twice the minimum functional margin for any point

Geometric Margin

- Distance from example to the separator is $r = y\dfrac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|}$
- Examples closest to the hyperplane are ***support vectors***.
- ***Margin*** ρ of the separator is the width of separation between support vectors of classes.



Derivation of finding r:
Dotted line $\mathbf{x}'-\mathbf{x}$ *is perpendicular to decision boundary so parallel to* $\mathbf{w}$.
Unit vector is $\mathbf{w}/|\mathbf{w}|$, *so line is* $r\mathbf{w}/|\mathbf{w}|$.

$\mathbf{x}' = \mathbf{x} - yr\mathbf{w}/|\mathbf{w}|$.
$\mathbf{x}'$ *satisfies* $\mathbf{w}^T\mathbf{x}'+b = 0$.
So $\mathbf{w}^T(\mathbf{x} - yr\mathbf{w}/|\mathbf{w}|) + b = 0$
Recall that $|\mathbf{w}| = \sqrt{\mathbf{w}^T\mathbf{w}}$.
So $\mathbf{w}^T\mathbf{x} - yr|\mathbf{w}| + b = 0$

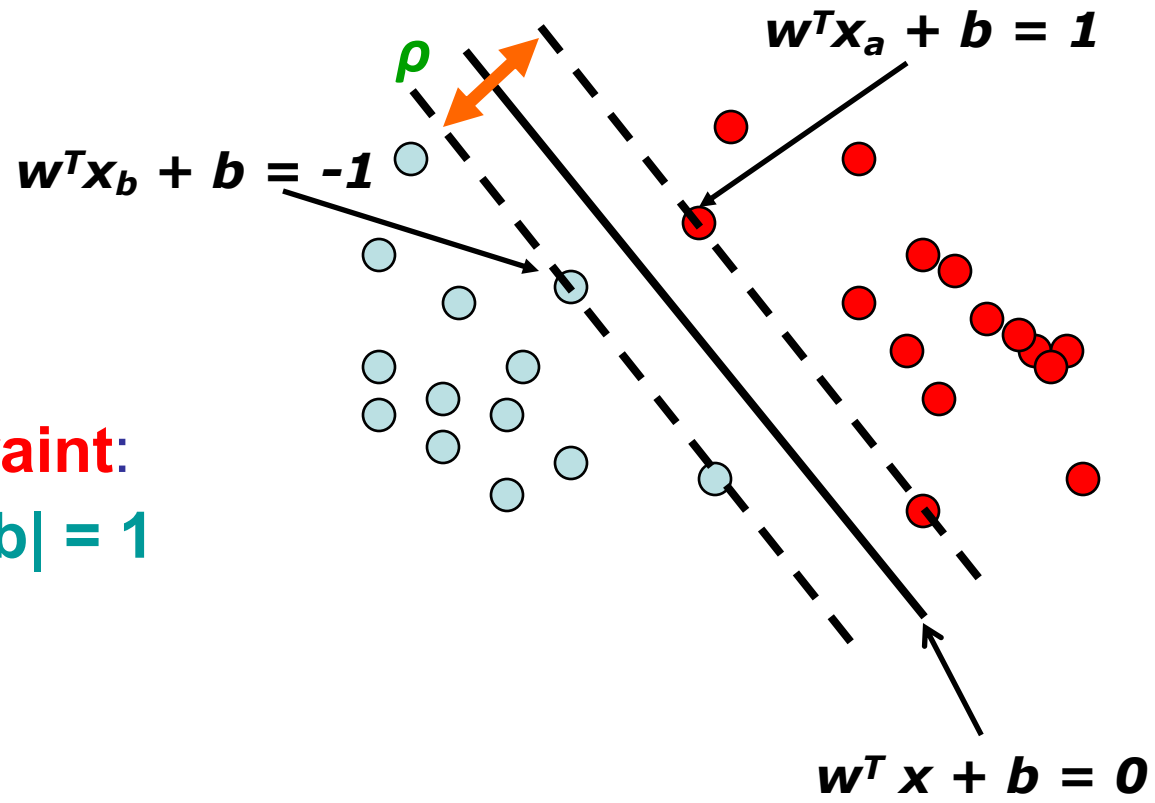
So, solving for r gives:

$$r = y(\mathbf{w}^T\mathbf{x} + b)/|\mathbf{w}|$$

Linear Support Vector Machine

- **Hyperplane**
  $\mathbf{w}^T \mathbf{x} + b = 0$
- **Extra scale constraint**:
  $\min_{i=1,\ldots,n} |\mathbf{w}^T \mathbf{x}_i + b| = 1$
- This implies:
  $\mathbf{w}^T(\mathbf{x}_a - \mathbf{x}_b) = 2$
  $\rho = ||\mathbf{x}_a - \mathbf{x}_b||_2 = 2/||\mathbf{w}||_2$

The margin is $1/||\mathbf{w}||_2$



The optimization problem

- Formulation of the problem

Find $\boldsymbol{w}$ *and* $\boldsymbol{b}$ *such that* $\rho = \frac{1}{\|w\|}$

is maximized and for all $\{x_i, y_i\}$ *satisfying the following constraints*

$$\boldsymbol{w^T x + b \geq +1 \ \text{for} \ y_i = 1 \, ; w^T x + b \leq 1 \ \text{for} \ y_i = -1}$$

- The **optimization theory** establishes that an optimization problem, called **primal**, has a **DUAL** form if the function to be optimized and the restrictions are strictly **convex**
  - Under these circumstances, solving the dual problem let obtaining the solution to the primal problem

The optimization problem

- The next function

$$\min f(w) = \frac{1}{2}\|w\|^2 = \frac{1}{2} < w, w >$$

Satisfies the criterion of convexity and thus, it has a DUAL

- **STEPS** to transform our optimization problem:
  1. We build an optimization problem without restrictions using **Lagrange function**
  2. We apply **Karush-Kuhn-Tucker (KKT) conditions**

The optimization problem

- Then we can formulate the ***quadratic optimization problem***:

Find $\boldsymbol{w}$ *and* $\boldsymbol{b}$ *such that* $\rho = \frac{1}{\|w\|}$

is maximized and for all $\{x_i, y_i\}$ *satisfying the following constraints*

$$\boldsymbol{w^T x + b \geq +1 \ \text{for} \ y_i = 1 \, ; w^T x + b \leq 1 \ \text{for} \ y_i = -1}$$

A better formulation:

Find $\boldsymbol{w}$ *and* $\boldsymbol{b}$ *such that* $\boldsymbol{\Phi(w) = \frac{1}{2} w^T w}$ *is minimized and*

for all $\{x_i, y_i\}$ *satisfying the following constraints*

$$y_j \ (\boldsymbol{w^T x_i + b}) \geq \boldsymbol{1}$$

Solving the Optimization Problem

Find $\boldsymbol{w}$ *and* $\boldsymbol{b}$ *such that* $\boldsymbol{\Phi(w) = \frac{1}{2} w^T w}$ *is minimized and*

for all $\{x_i, y_i\}$ *satisfying the following constraints*

$$y_j \ (\boldsymbol{w^T x_i + b}) \geq \mathbf{1}$$

- Need to optimize a ***quadratic*** function subject to *linear* constraints.
- Quadratic optimization problems are a well-known class of mathematical programming problems, and many (intricate) algorithms exist for solving them (with many special ones built for SVMs).
- The solution involves constructing a ***dual problem*** where a *Lagrange multiplier* α_i is associated with every constraint in the **primary problem**:

Find $\alpha_1 \ldots \alpha_N$ *such that*

$\boldsymbol{Q(\alpha)} = \sum_i \ \alpha_i - \frac{1}{2} \sum_i \ \alpha_i \alpha_j y_i y_j x_i \ x_j$ is maximized s.t.

$$(1) \ \sum_i \ \alpha_i y_i = 0$$

$$(2) \ \alpha_i \geq 0 \ \text{ for all } \alpha_i$$

The Optimization Problem Solution

- The solution has the form:

$$\boldsymbol{w} = \sum_i \boldsymbol{\alpha_i y_i x_i} \qquad \boldsymbol{b} = \boldsymbol{y_k} - \boldsymbol{w_k^T x_k} \text{ for any } x_k \text{ such that } \alpha_k \neq 0$$

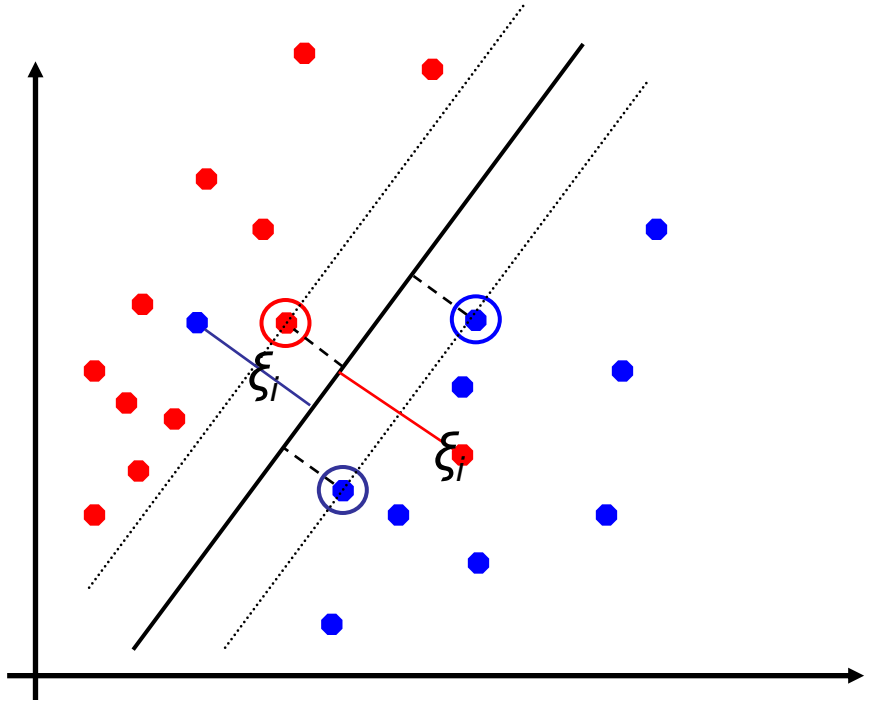
- Each non-zero α_i indicates that corresponding $\mathbf{x_i}$ is a support vector. Then the classifying function will have the form:

$$\boldsymbol{f(x)} = \sum \boldsymbol{\alpha_i y_i x_i^T x} + \boldsymbol{b}$$

- Notice that it relies on an *inner product* between the test point **x** and the support vectors $\mathbf{x_i}$
- Also keep in mind that solving the optimization problem involved computing the inner products $\mathbf{x_i^T x_j}$ between all pairs of training points!

Soft Margin Classification

- What to do if the training set is not linearly separable?
  - *Slack variables* ξ_i can be added to allow misclassification of difficult or noisy examples.
  - Allow some errors
    - Let some points be moved to where they belong to, at a cost
  - Still, try to minimize training set errors, and, to place hyperplane "far" from each class (large margin)



Soft Margin Mathematical definition

Given a linearly separable training sample

$$s = ((x_1, y_1), \cdots, (x_n, y_n))$$

where $x_i \in \mathbb{R}^n \; and \; y_i \in \{+1, -1\}$

the hyperplane (w, b) that solves the optimization problem

$$\begin{aligned} \boldsymbol{minimize}_{w,b} \quad & < w \cdot w >, \\ subject\ to \quad & y_i(< w \cdot x_i > + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, i = 1, \cdots, n \end{aligned}$$

realizes the maximal margin hyperplane with geometric margin $\rho = \frac{1}{\|w\|}$

Soft Margin Classification Mathematically

- The old formulation:

Find $\boldsymbol{w}$ *and* $\boldsymbol{b}$ *such that* $\boldsymbol{\Phi(w) = \frac{1}{2} w^T w}$ *is minimized and*

for all $\{x_i, y_i\}$ *satisfying the following constraints*

$$y_j \ (\boldsymbol{w^T x_i + b}) \geq \mathbf{1}$$

- The new formulation incorporating slack variables:

Find $\boldsymbol{w}$ *and* $\boldsymbol{b}$ *such that* $\boldsymbol{\Phi(w) = \frac{1}{2} w^T w + C \sum \xi_i}$ *is minimized*

and for all $\{x_i, y_i\}$ *satisfying the following constraints*

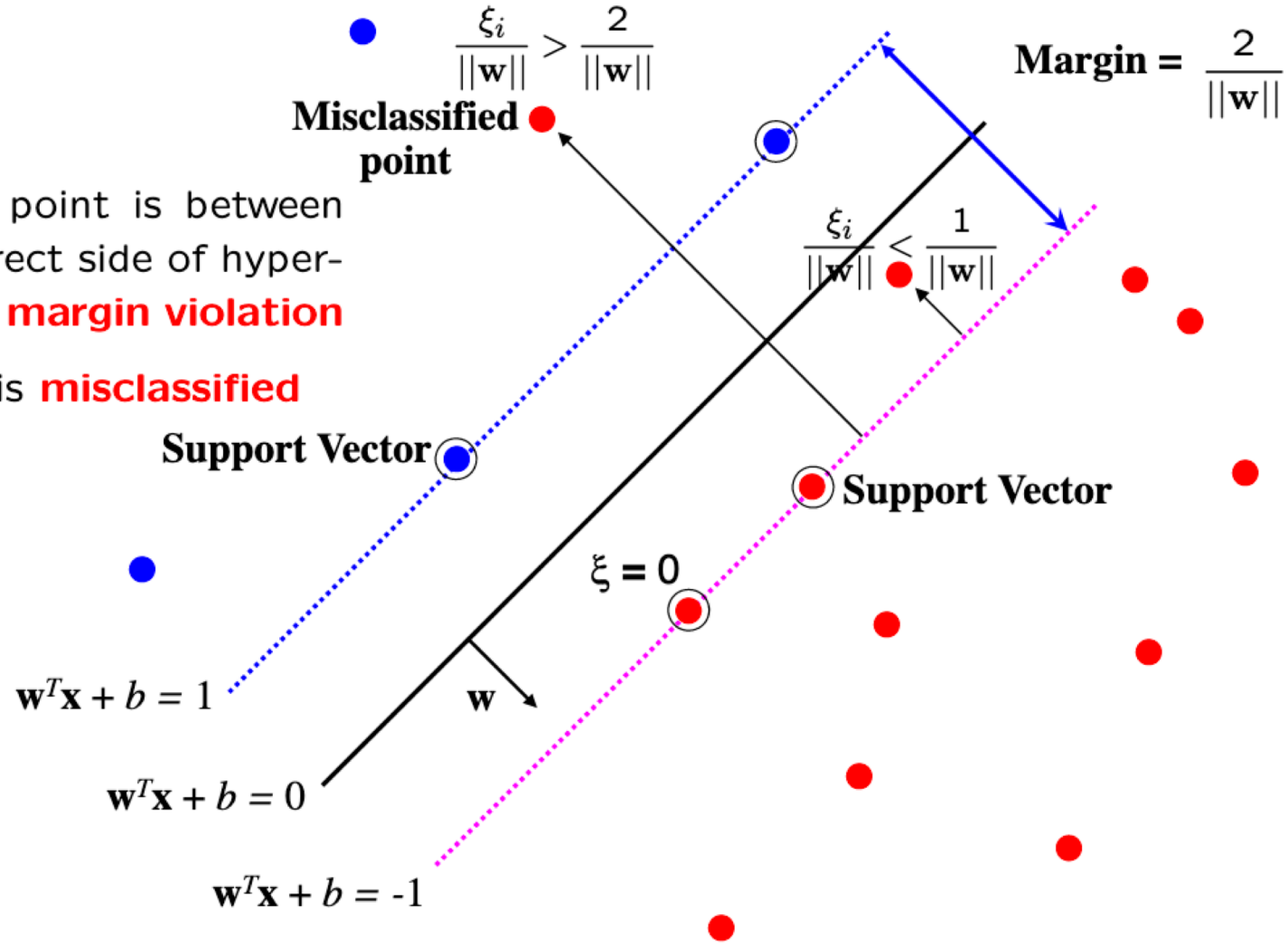
$$y_j \ (\boldsymbol{w^T x_i + b}) \geq \mathbf{1} - \boldsymbol{\xi_i} \text{ and } \boldsymbol{\xi_i} \geq \mathbf{0} \ for \ all \ i$$

- Parameter *C* can be viewed as a way to control overfitting (a regularization term)

"Slack" variables

$\xi_i \geq 0$

- for $0 < \xi \leq 1$ point is between margin and correct side of hyper-plane. This is a **margin violation**
- for $\xi > 1$ point is **misclassified**



Soft Margin Classification – Solution

- The dual problem for soft margin classification:

Find $\alpha_1 \ldots \alpha_N$ *such that*

$$\boldsymbol{Q(\alpha)} = \sum_i \alpha_i - \frac{1}{2}\sum_i \alpha_i \alpha_j y_i y_j x_i \; x_j \text{ is maximized s.t.}$$

$$(1)\ \sum_i \boldsymbol{\alpha_i y_i} = \mathbf{0}$$

$$(2)\ \mathbf{0} \le \boldsymbol{\alpha_i} \le \boldsymbol{C} \ \textbf{for all } \boldsymbol{\alpha_i}$$

- Neither slack variables ξ_i nor their Lagrange multipliers appear in the dual problem!
- Again, $\mathbf{x_i}$ with non-zero α_i will be support vectors.
- Solution to the dual problem is:

w *and* ***b*** *are not needed explicitly for classification!*

$$\boldsymbol{w} = \sum_i \boldsymbol{\alpha_i y_i x_i}$$

$$\boldsymbol{b} = \boldsymbol{y_k}(\mathbf{1} - \boldsymbol{\xi_k}) - \boldsymbol{w_k^T x_k} \text{ where k} = arg\max_k \alpha_k$$

$$\boldsymbol{f(x)} = \sum \boldsymbol{\alpha_i y_i x_i^T x} \; + \boldsymbol{b}$$

"Soft" margin solution

- C is a **regularization** parameter:
  - Small C allows constraints to be easily ignored $\longrightarrow$ large margin
  - Large C makes constraints hard to ignore $\longrightarrow$ narrow margin
  - $C = \infty$ enforces all constraints: hard margin

- This is still a quadratic optimization problem and there is a unique minimum. Note there is only one parameter, C

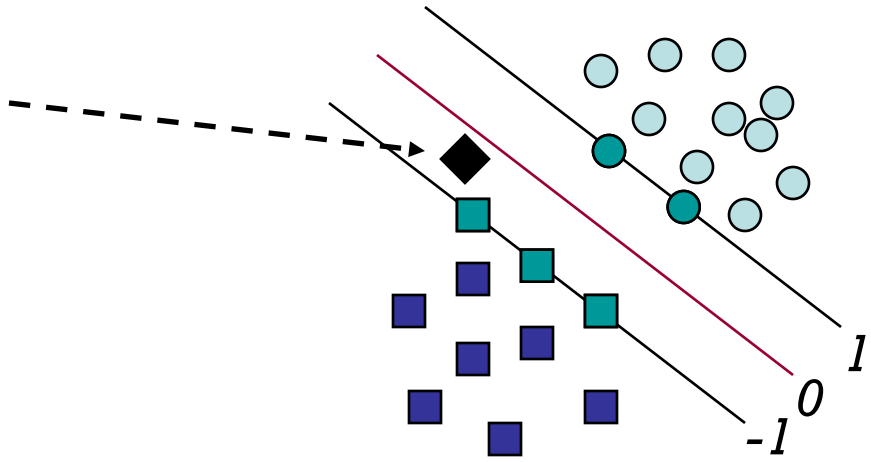
Classification with SVMs

- Given a new point **x**, we can score its projection onto the hyperplane normal:
  - I.e., compute score: $\boldsymbol{w^T x + b = \sum \alpha_i y_i x_i^T x + b}$
    - Decide class based on whether < or > 0
  - Can set confidence threshold *t*.

Score > t: yes

Score < -t: no

Else: don't know



Linear SVMs: Summary

- The classifier is a *separating hyperplane.*
  - Most "important" training points are support vectors; they define the hyperplane.

- Quadratic optimization algorithms can identify which training points $\mathbf{x_i}$ are support vectors with non-zero Lagrangian multipliers α_i.

- Both in the dual formulation of the problem and in the solution training points appear only inside inner products:

Find $\alpha_1 \ldots \alpha_N$ *such that*

$\boldsymbol{Q(\alpha)} = \sum_i \alpha_i - \frac{1}{2}\sum_i \alpha_i\alpha_j y_i y_j x_i x_j$ is maximized s.t.

$(1)\ \sum_i \boldsymbol{\alpha_i y_i = 0}$

$(2)\ \mathbf{0 \le \alpha_i \le C}$ **for all** $\boldsymbol{\alpha_i}$

$$\boldsymbol{f(x) = \sum \alpha_i y_i x_i^T x + b}$$

What else?

Look at the video

https://youtu.be/efR1C6CvhmE

TO READ:

A Tutorial on Support Vector Machines for Pattern Recognition
Christopher J.C. Burges

Sections 2 and 3.

With the exception of Section 4, the remaining of the sections are optional

4. Non-linear SVM mathematically

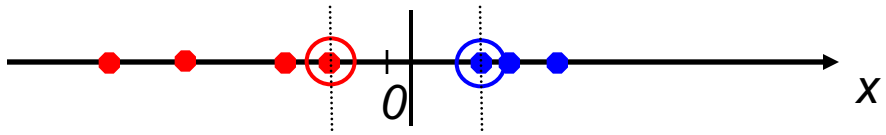
Extension to Non-linear Decision Boundary

- So far, we have only considered large-margin classifier with a linear decision boundary
- How to generalize it to become nonlinear?

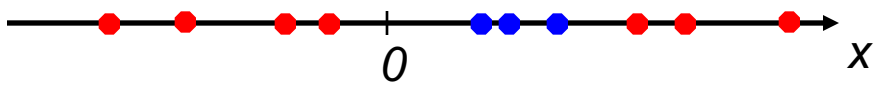
- **Key idea**: transform x_i to a higher dimensional space to "make life easier"
  - Input space: the space the point $\boldsymbol{x_i}$ are located
  - Feature space: the space of $\phi(\boldsymbol{x_i})$ after transformation

Non-linear SVMs

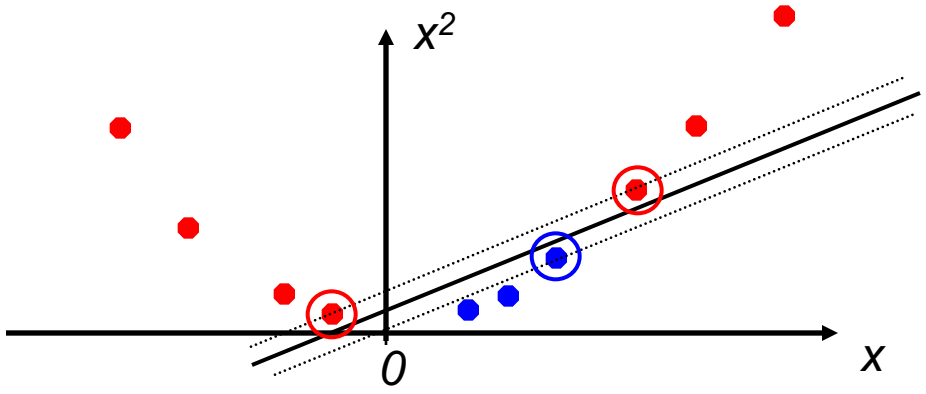
- Datasets that are linearly separable with some noise work out great:



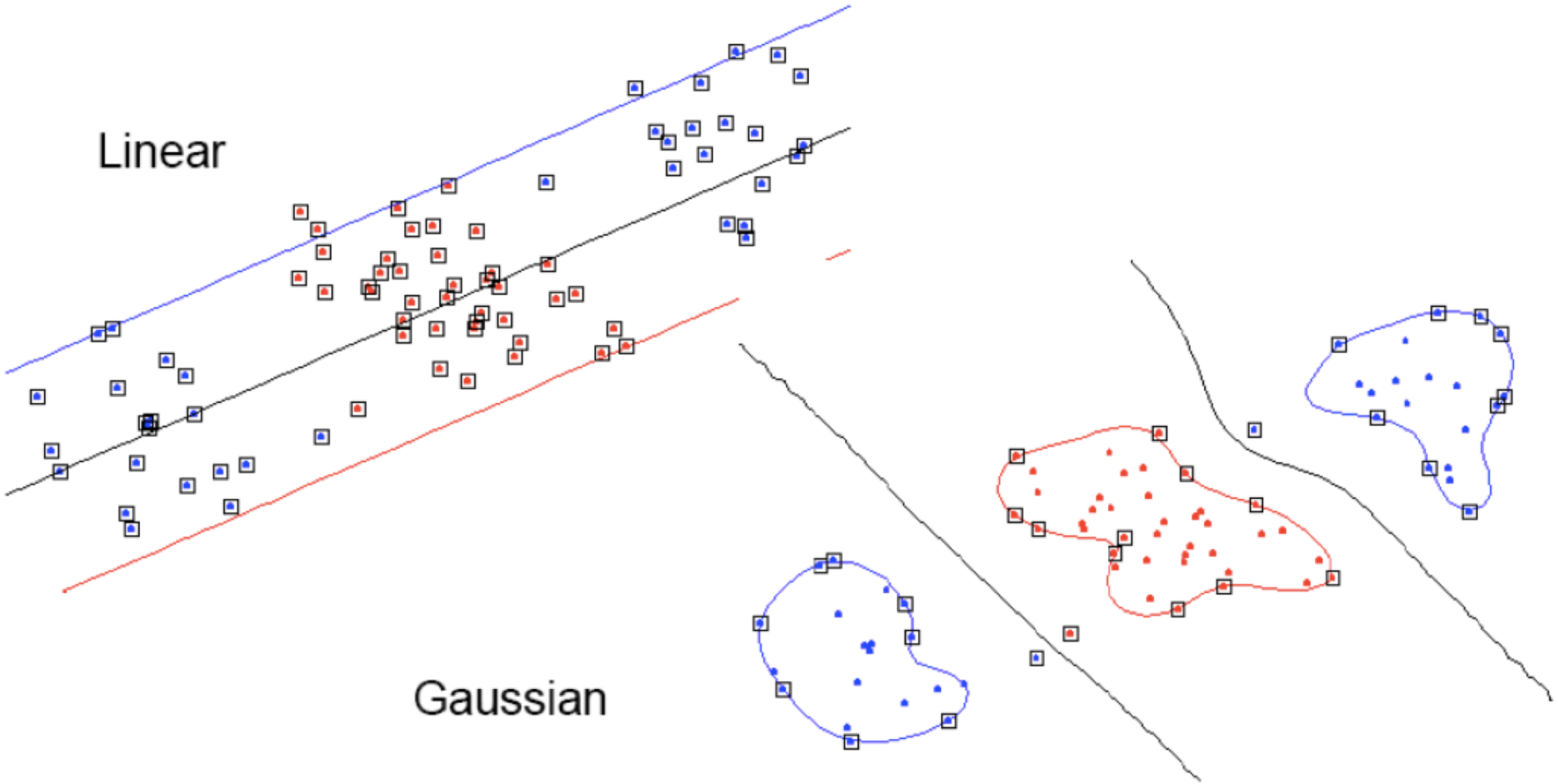
- But what are we going to do if the dataset is just too hard?



- How about… mapping data to a higher-dimensional space:

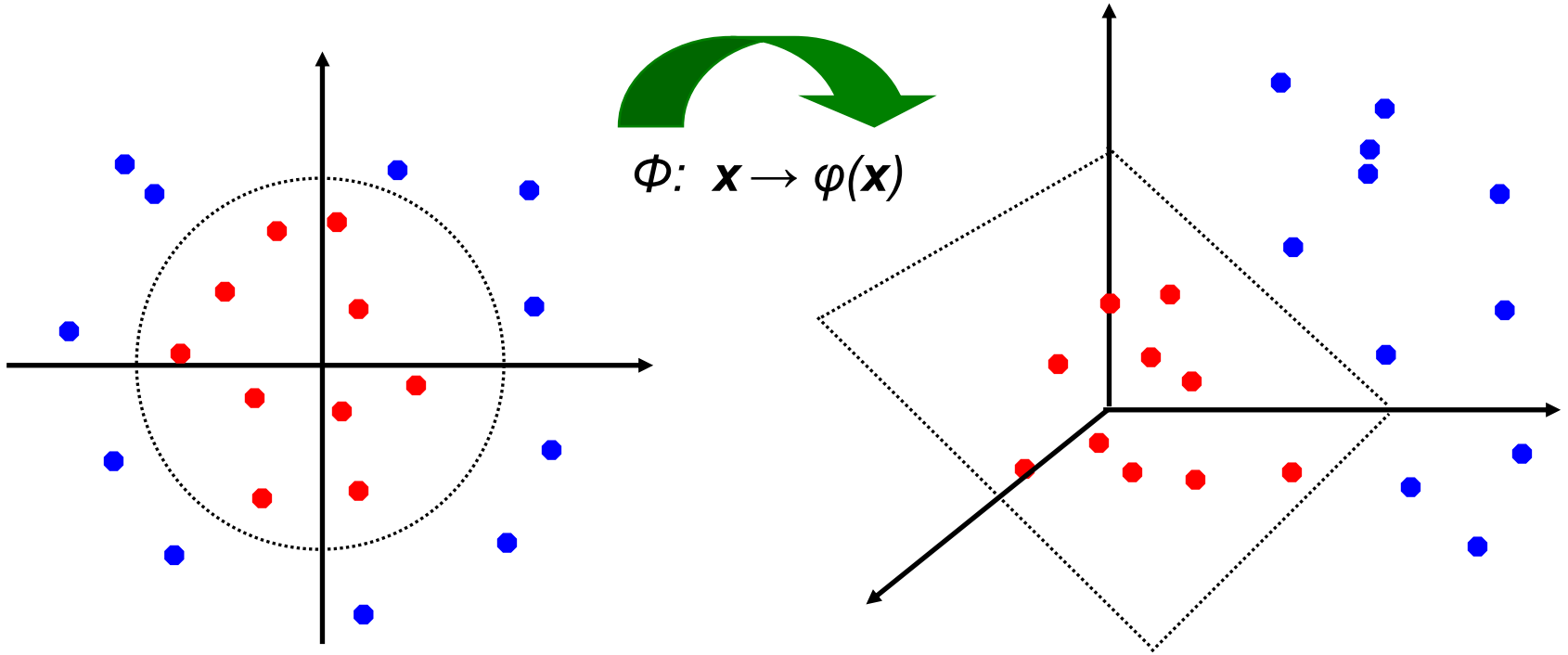


Non-linear SVMs

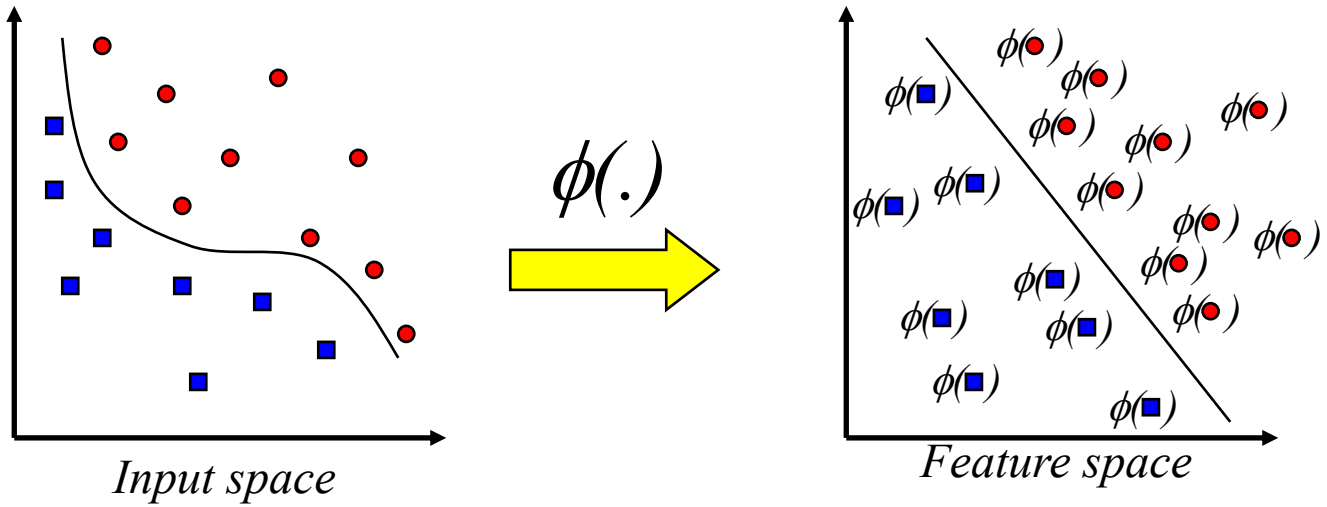


Non-linear SVMs: Feature spaces

- **General idea**: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



Transforming the Data



Note: feature space is of higher dimension than the input space in practice

- Computation in the feature space can be costly because it is high dimensional
  - The feature space is typically infinite-dimensional!
- The kernel trick comes to rescue

The Kernel Trick

- Recall the SVM optimization problem

Find $\alpha_1 \ldots \alpha_N$ *such that*

$\boldsymbol{Q(\alpha)} = \sum_i \alpha_i - \frac{1}{2}\sum_i \alpha_i\alpha_j y_i y_j x_i x_j$ is maximized s.t.

$(1)\ \sum_i \boldsymbol{\alpha_i y_i} = \mathbf{0}\ ,\quad (2)\ \mathbf{0} \leq \boldsymbol{\alpha_i} \leq \boldsymbol{C}\ \textbf{for all}\ \boldsymbol{\alpha_i}$

- The data points only appear as inner product
- As long as we can calculate the inner product in the feature space, we do not need the mapping explicitly
- Many common geometric operations (angles, distances) can be expressed by inner products
- Define the kernel function *K* by

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

An Example for $\phi(.)$ and K(.,.)

- Suppose $\phi(.)$ is given as follows

$$\phi(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

- An inner product in the feature space is

$$\langle \phi(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}), \phi(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}) \rangle = (1 + x_1y_1 + x_2y_2)^2$$

- So, if we define the kernel function as follows, there is no need to carry out $\phi(.)$ explicitly

$$K(\mathbf{x}, \mathbf{y}) = (1 + x_1y_1 + x_2y_2)^2$$

- This use of kernel function to avoid carrying out $\phi(.)$ explicitly is known as the **kernel trick**

More on Kernel Functions

- Not all similarity measures can be used as kernel function, however
  - The kernel function needs to satisfy the **Mercer function**,
    - ***Every positive definite symmetric function is a kernel***
- This implies that
  - the *n* by *n* kernel matrix,
  - in which the (i,j)-th entry is the $K(\mathbf{x}_i, \mathbf{x}_j)$, is always positive definite
- This also means that optimization problem can be solved in polynomial time!

Examples of Kernel Functions

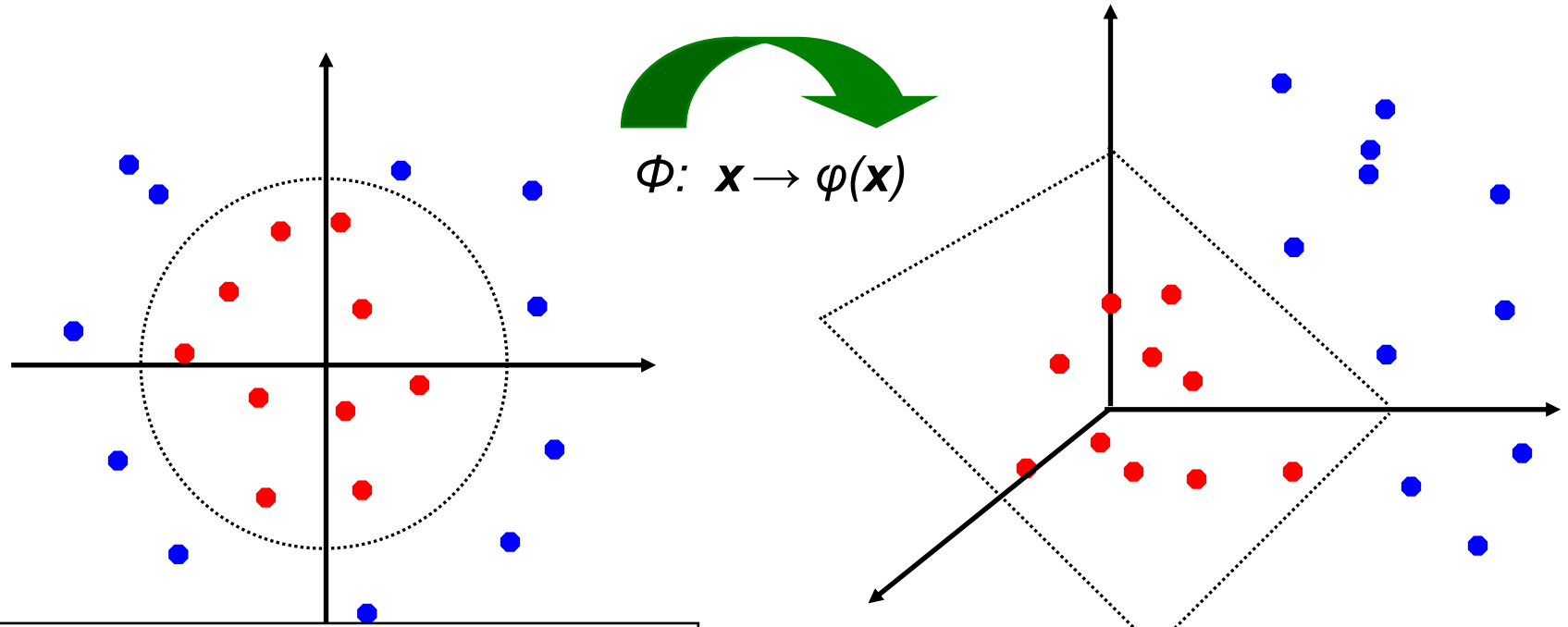
- Why use kernels?
  - Make non-separable problem separable
  - Map data into better representational space

- Common Kernels
  - **Linear**: $K(x_i, x_j) = < x_i \cdot x_j >$
  - **Polynomial of power p**: $K(x_i, x_j) = (\tau + \gamma x_i \cdot x_j)_p$
  - **Gaussian** (radial-basis function network): $K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}, \gamma > 0$
  - **Sigmoidal**: $K(x_i, x_j) = \tanh(\gamma < x_i x_j > + \tau)$

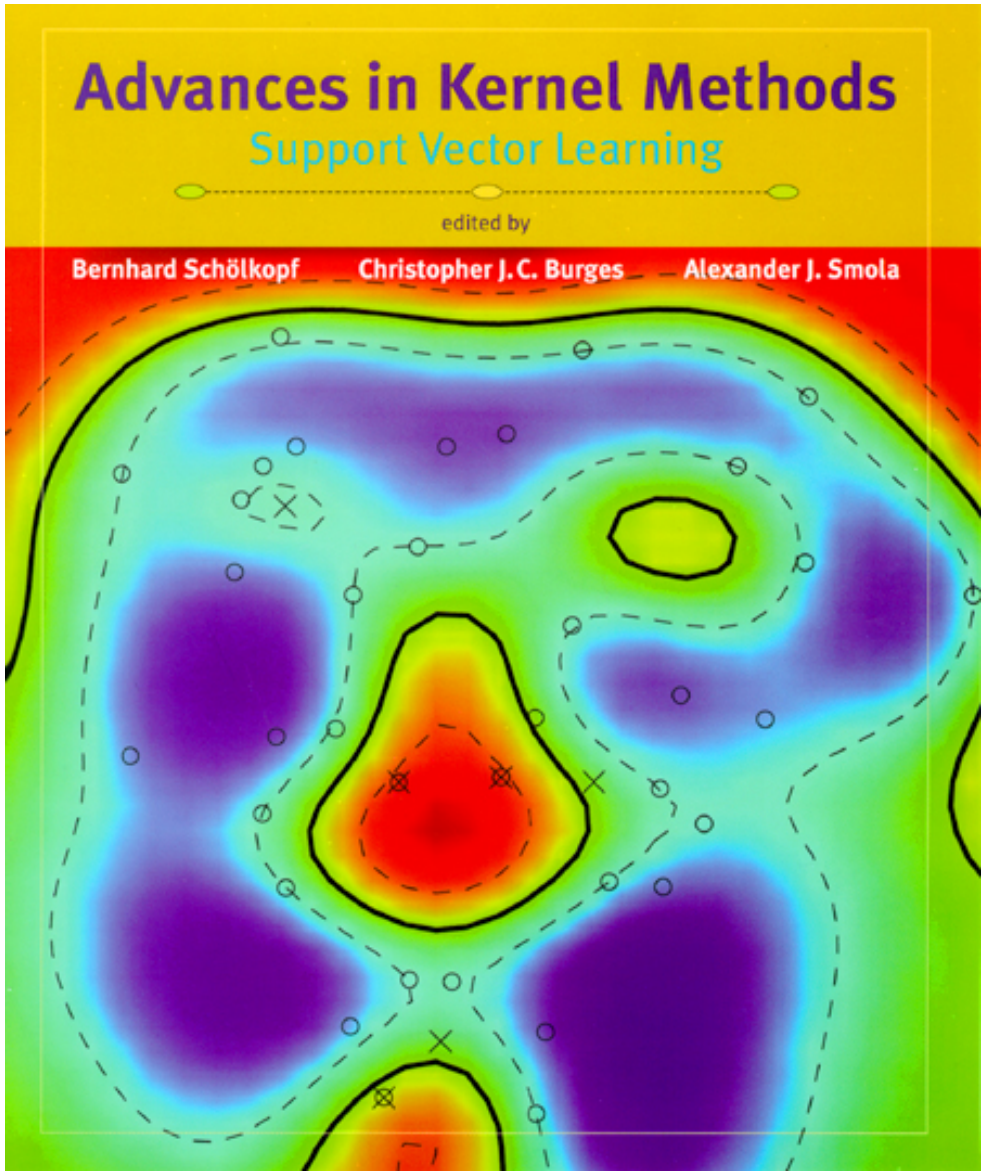
The parameters $\gamma, \tau, and\ p$ are called the parameters of the kernel.

Non-linear SVMs: Feature spaces

- **General idea**: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$


Advances in Kernel Methods
Support Vector Learning
edited by
Bernhard Schölkopf
Christopher J.C. Burges
Alexander J. Smola

Non-linear SVMs Mathematically

- Dual problem formulation:

Find $\alpha_1 \ldots \alpha_N$ *such that*

$\boldsymbol{Q(\alpha)} = \sum_i \ \alpha_i \ - \frac{1}{2}\sum_i \ \alpha_i \alpha_j y_i y_j K(x_i \ x_j)$ is maximized s.t.

$(1)\ \sum_i \ \boldsymbol{\alpha_i y_i = 0}\ ,\quad (2)\ \mathbf{0} \leq \boldsymbol{\alpha_i}\ \textbf{for all}\ \boldsymbol{\alpha_i}$

- The solution is:

$$\boldsymbol{f(x) = \sum \alpha_i y_i K(x_i^T x\ \ ) + b}$$

- Optimization techniques for finding $\alpha_i' s$ remain the same!

Example

- Suppose we have 5 one-dimensional data points
  - x_1=1, x_2=2, x_3=4, x_4=5, x_5=6, with 1, 2, 5 as class 1 and 3, 4 as class 2 ⇒ y_1=1, y_2=1, y_3=-1, y_4=-1, y_5=1
- We use the polynomial kernel of degree 2
  - $K(x,y) = (xy+1)^2$
  - C is set to 100
- We first find α_i (i=1, …, 5) by

Find $\alpha_1 \ldots \alpha_5$ *such that*

$$\boldsymbol{Q(\alpha)} = \sum_{i=1}^{5} \alpha_i - \frac{1}{2}\sum_{i=1}^{5}\sum_{j=1}^{5} \alpha_i \alpha_j y_i y_j (x_i \; x_j + 1)^2 \text{ is maximized}$$

$$\text{s.t.} \quad (1)\ \sum_{i=1}^{5} \boldsymbol{\alpha_i y_i} = \mathbf{0}\,, \quad (2)\ \mathbf{0} \le \boldsymbol{\alpha_i} \le \mathbf{100}$$

Example

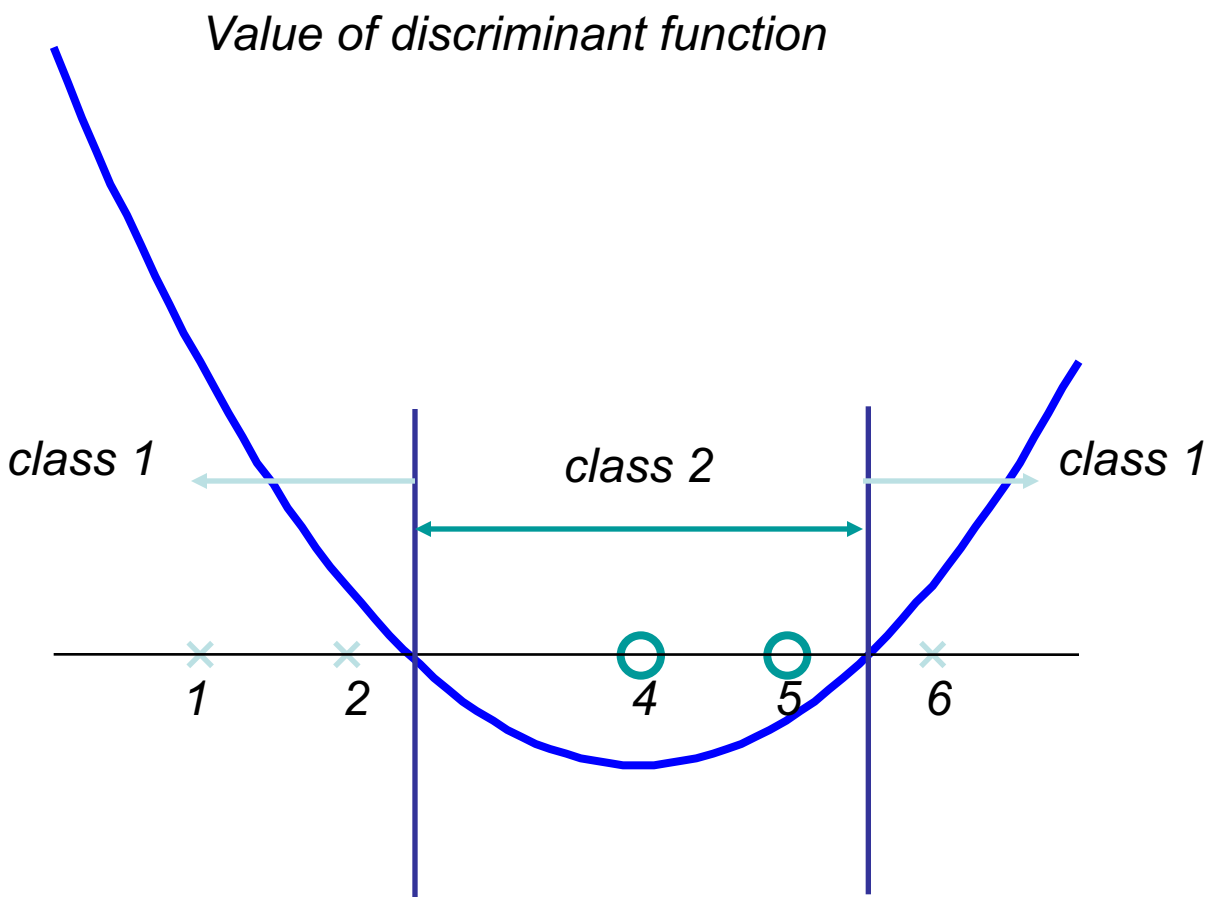
- By using a Quadratic Problem solver, we get
  - α_1=0, α_2=2.5, α_3=0, α_4=7.333, α_5=4.833
  - Note that the constraints are indeed satisfied
  - The support vectors are **{x_2=2, x_4=5, x_5=6}**
- The discriminant function is

$$f(z) = 2.5(1)(2z+1)^2 + 7.333(-1)(5z+1)^2 + 4.833(1)(6z+1)^2 + b$$
$$= 0.6667z^2 - 5.333z + b$$

(α_5, y_5, $K(z, x_5)$)

- b is recovered by solving f(2)=1 or by f(5)=-1 or by f(6)=1, as x_2 and x_5 lie on the line $\phi(\mathbf{w})^T\phi(\mathbf{x}) + b = 1$ and x_4 lies on the line $\phi(\mathbf{w})^T\phi(\mathbf{x}) + b = -1$
- All three give b=9 $\rightarrow$ $f(z) = 0.6667z^2 - 5.333z + 9$

Example


Value of discriminant function
class 1
class 2
class 1
1
2
4
5
6

What else?

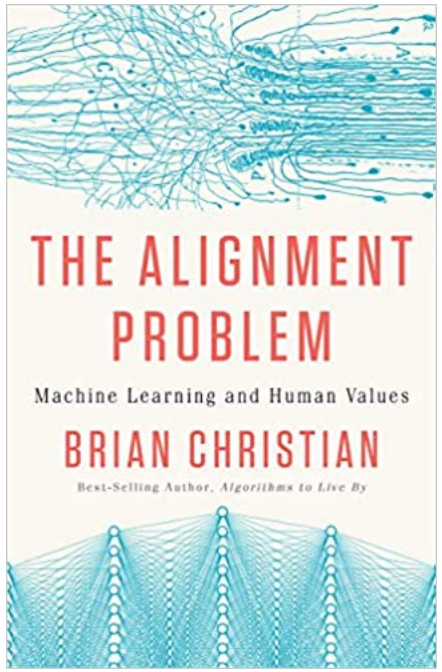
TO READ:

A Tutorial on Support Vector Machines for Pattern Recognition
Christopher J.C. Burges

Section 4

With the exception of Sections 2 and 3, the remaining of the sections are optional

A final reading



The Alignment Problem: Machine Learning and Human Values by Brian Christian

Today's "machine-learning" systems, trained by data, are so effective that we've invited them to see and hear for us—and to make decisions on our behalf. But alarm bells are ringing. Recent years have seen an eruption of concern as the field of machine learning advances. When the systems we attempt to teach will not, in the end, do what we want or what we expect, ethical and potentially existential risks emerge. Researchers call this the alignment problem.

Systems cull résumés until, years later, we discover that they have inherent gender biases. Algorithms decide bail and parole—and appear to assess Black and White defendants differently. We can no longer assume that our mortgage application, or even our medical tests, will be seen by human eyes. And as autonomous vehicles share our streets, we are increasingly putting our lives in their hands.

The mathematical and computational models driving these changes range in complexity from something that can fit on a spreadsheet to a complex system that might credibly be called "artificial intelligence." They are steadily replacing both human judgment and explicitly programmed software.

UNIVERSITAT DE BARCELONA

Week 5

Course. Introduction to Machine Learning

Theory 5. Support Vector Machine

Dr. Maria Salamó Llorente

maria.salamo@ub.edu

Dept. Mathematics and Informatics,

Faculty of Mathematics and Informatics,

University of Barcelona (UB)