DATA MINING 2
Support Vector Machine

Riccardo Guidotti

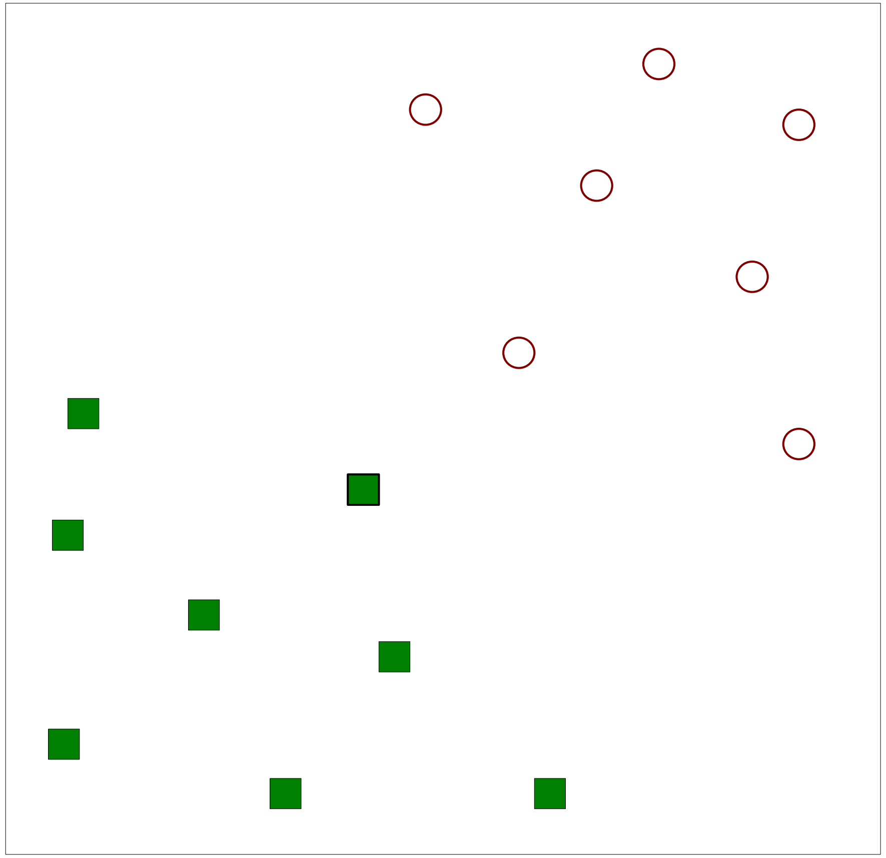
a.a. 2022/2023

Slides edited from Tan, Steinbach, Kumar, Introduction to Data Mining

UNIVERSITÀ DI PISA

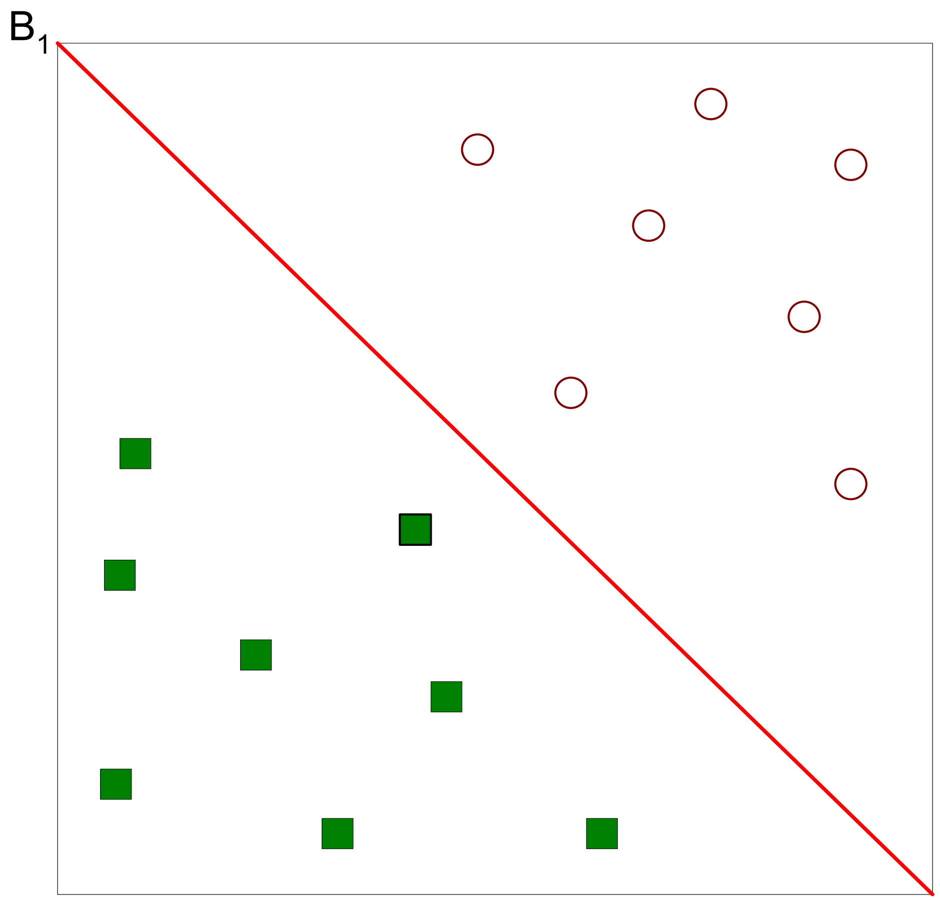
Maximum Margin Hyperplanes

- Find a linear hyperplane (decision boundary) that separates the data.



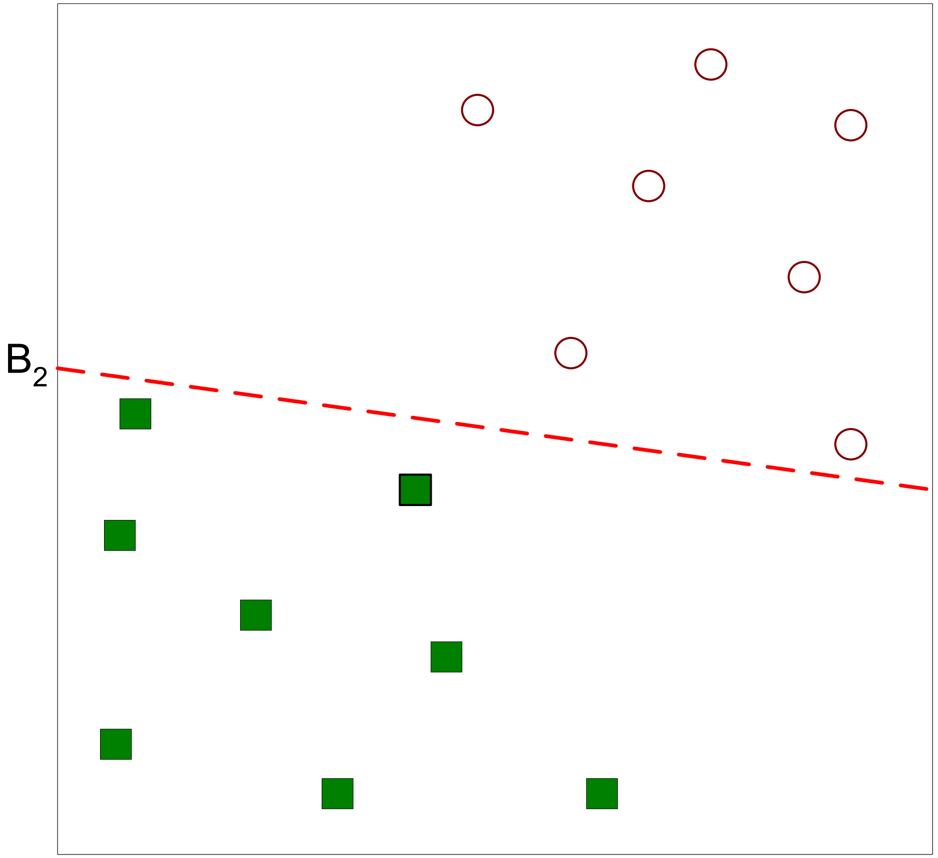
Maximum Margin Hyperplanes

- One possible solution.



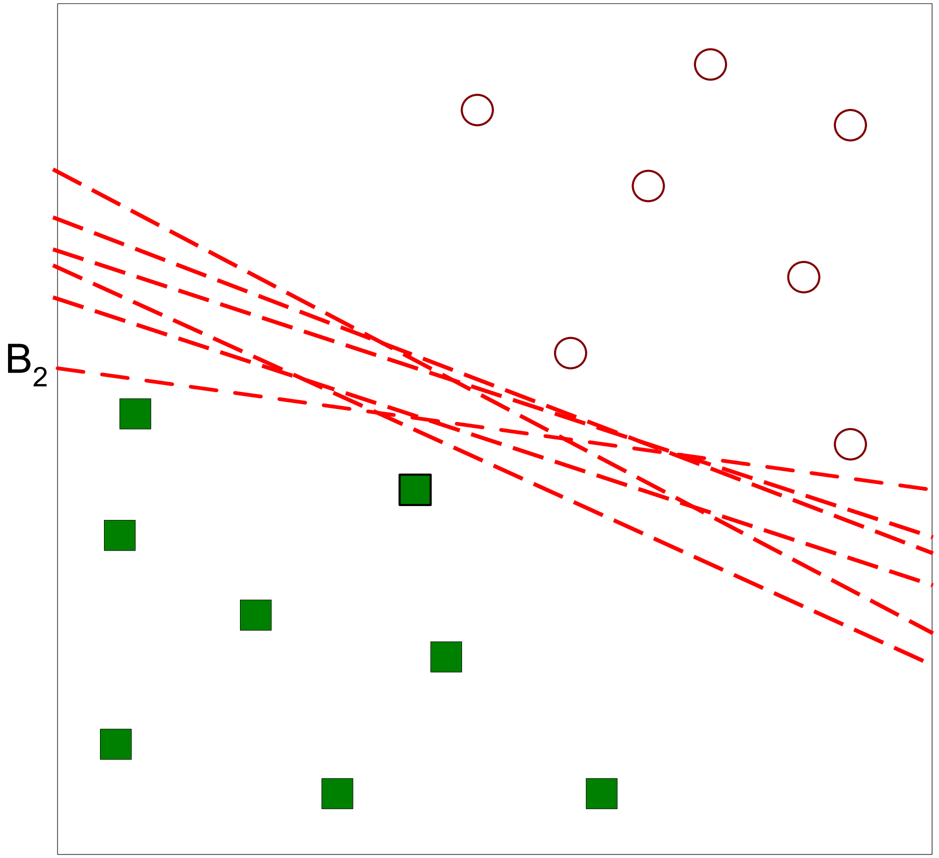
Maximum Margin Hyperplanes

- Another possible solution.



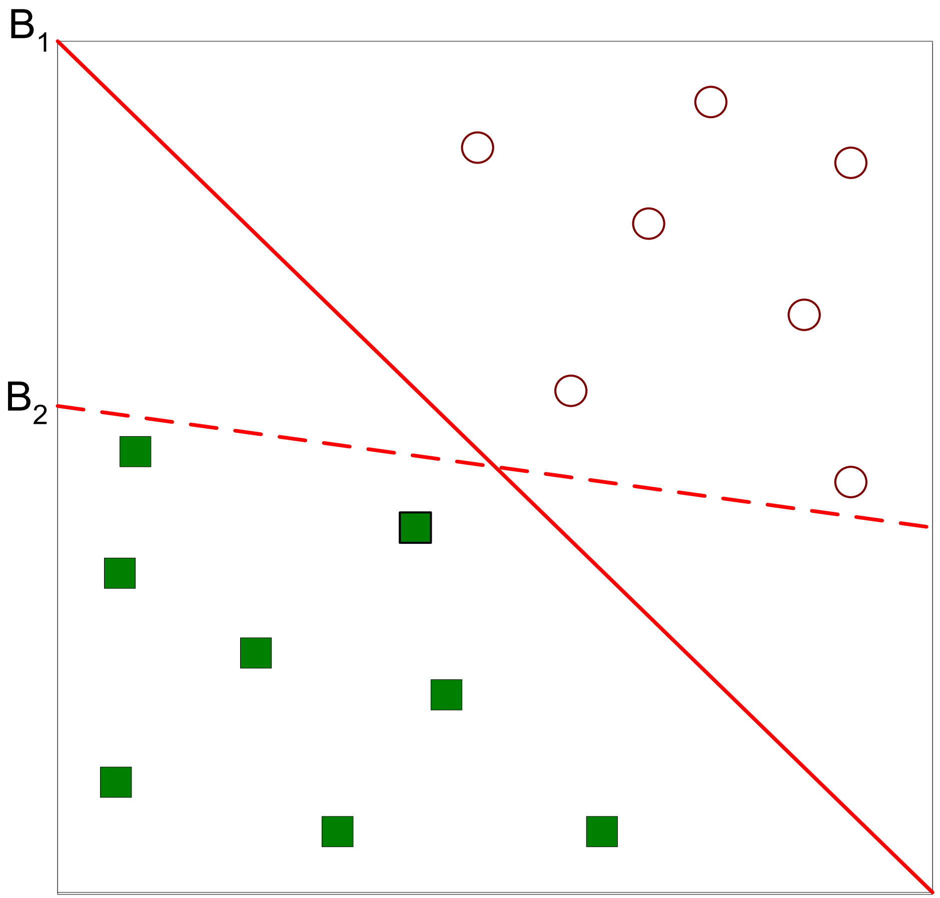
Maximum Margin Hyperplanes

- Other possible solutions.



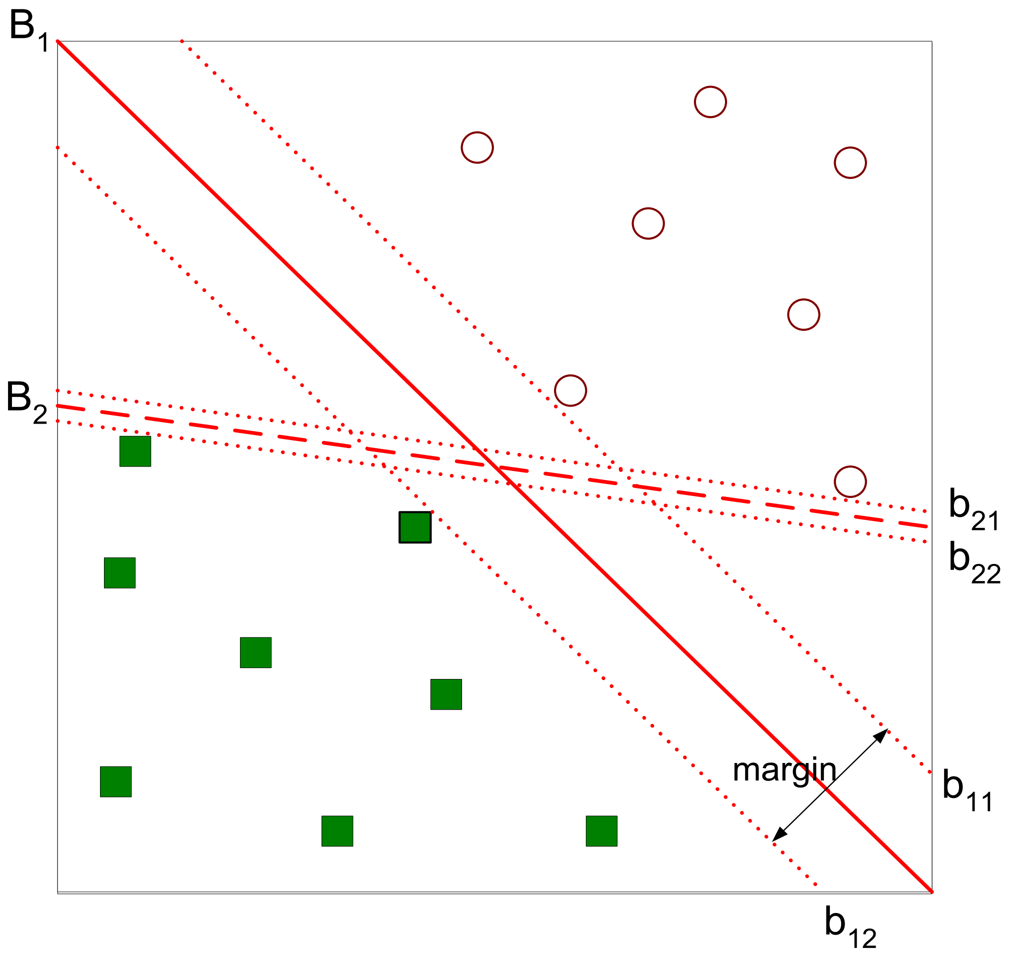
Maximum Margin Hyperplanes

- Let's focus on B_1 and B_2.
- Which one is better?
- How do you define better?



Maximum Margin Hyperplanes

- The best solution is the hyperplane that **maximizes** the **margin**.
- Thus, B_1 is better than B_2.



Linear SVM: Separable Case

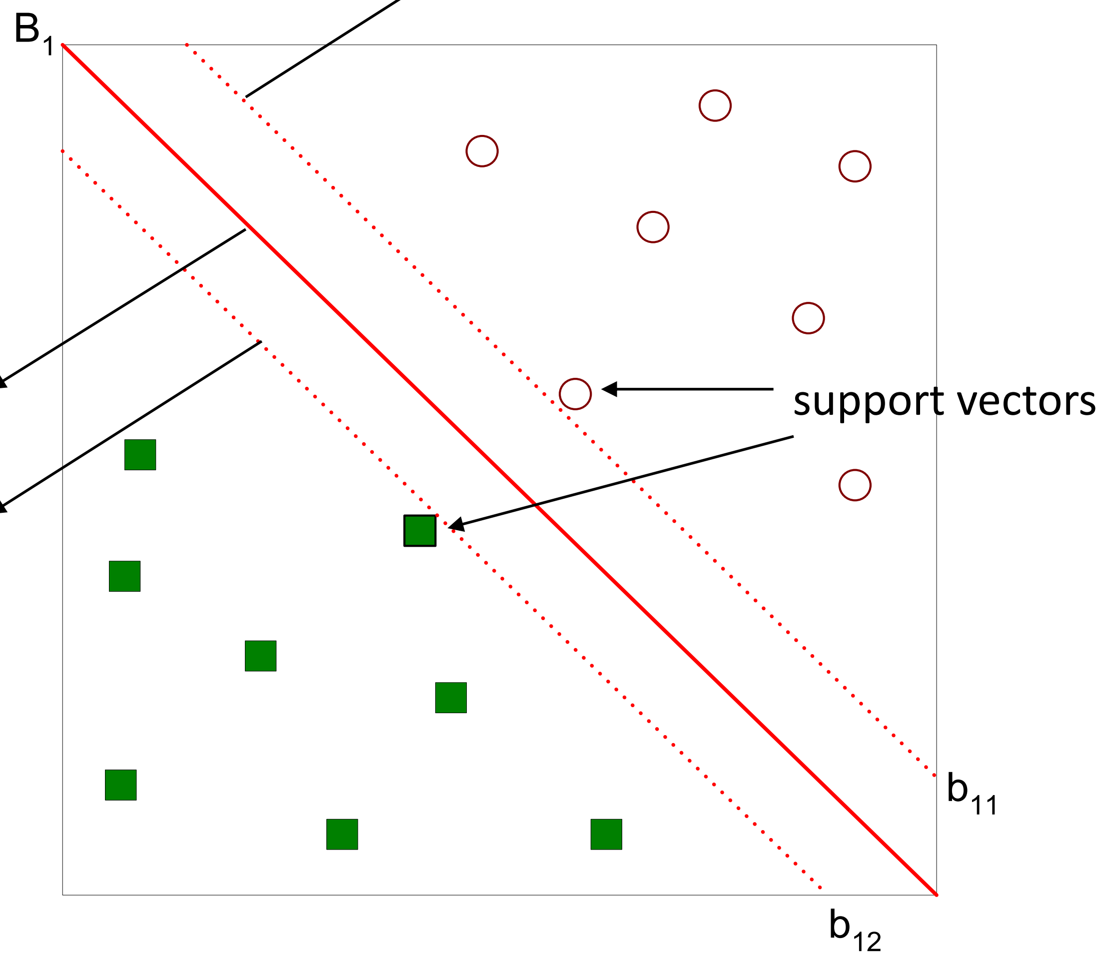
$\vec{w} \bullet \vec{x} + b = +1$

- A linear SVM is a classifier that searches for a hyperplane with the largest margin (a.k.a. maximal margin classifier).

decision boundary

$\vec{w} \bullet \vec{x} + b = 0$

$\vec{w} \bullet \vec{x} + b = -1$



- *w* and *b* have to be learned.

- Given *w* and *b* the classifiers work as

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

Example calculus dot product

w = [.3 .2]   x = [1 2]   b = -2

w · x + b = .3*1 + .2*2 +(-2) = -1.3

Linear SVM: Separable Case

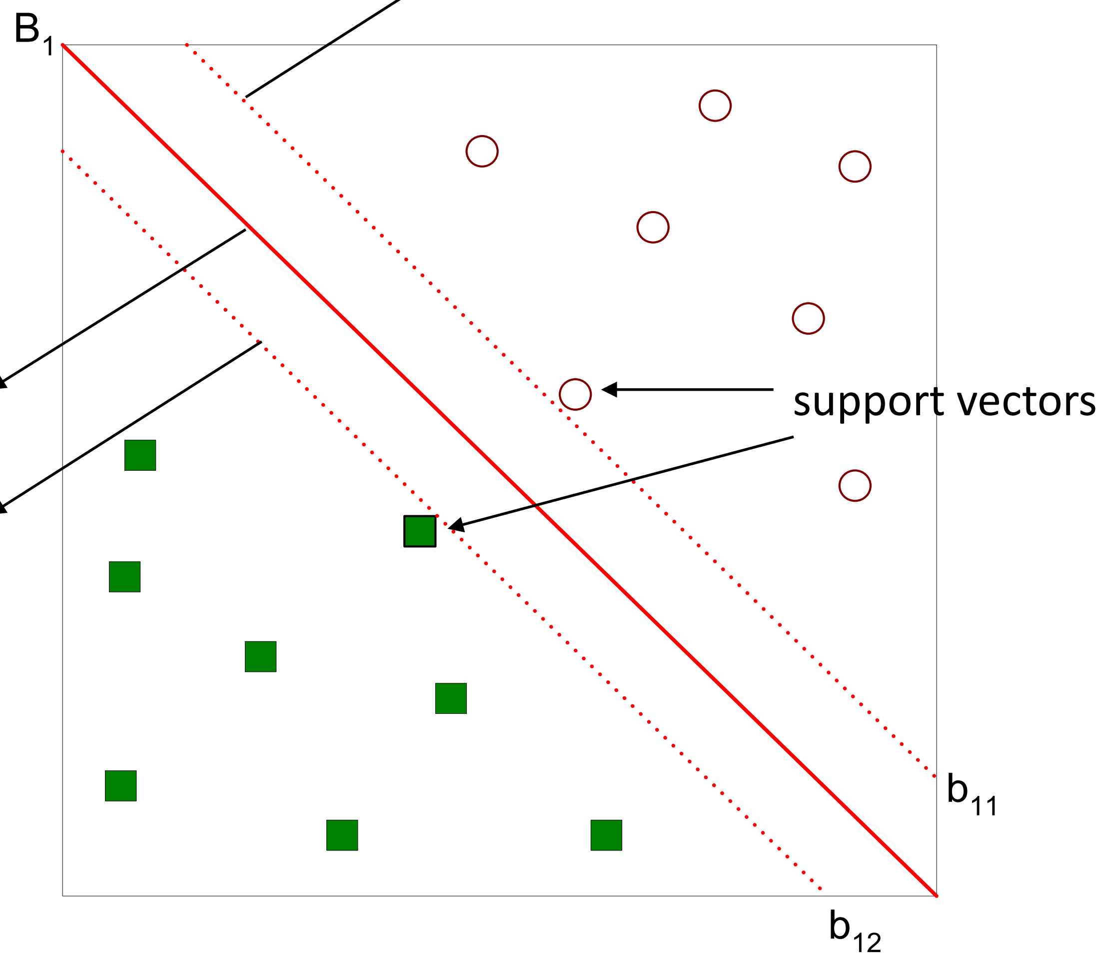
$\vec{w} \bullet \vec{x} + b = +1$

- What is the distance expression for a point *x* to a line *wx+b= 0* (the decision boundary)?

$$d(\mathbf{x}) = \frac{|\mathbf{x} \cdot \mathbf{w} + b|}{\sqrt{\|\mathbf{w}\|_2^2}} = \frac{|\mathbf{x} \cdot \mathbf{w} + b|}{\sqrt{\sum_{i=1}^{d} w_i^2}}$$

decision boundary
$\vec{w} \bullet \vec{x} + b = 0$

$\vec{w} \bullet \vec{x} + b = -1$

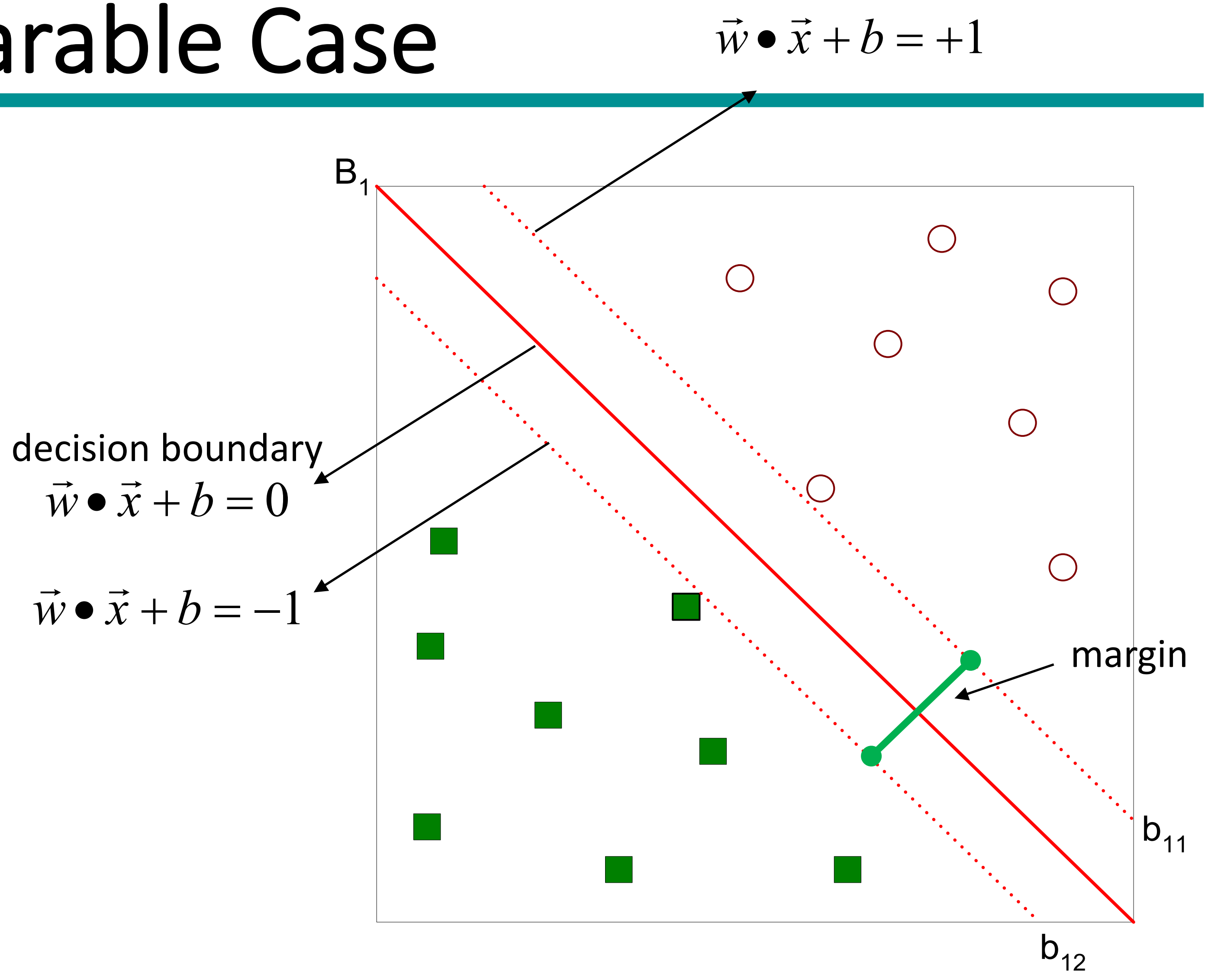


Linear SVM: Separable Case

- The distance between B_1 and b_{11} is $1/||w||$
- The distance between b_{11} and b_{12}, i.e., the margin is

$$\text{Margin} = \frac{2}{\|\vec{w}\|}$$

- In order to ***maximize the margin*** we need to minimize $||w||$



Learning a Linear SVM

- Learning the SVM model is equivalent to determining *w* and *b.*
- How to find *w* and *b?*
- Objective is to ***maximize the margin***.
- Which is equivalent to minimize
- Subject to to the following constraints
- This is a constrained optimization problem that can be solved using the *Lagrange* multiplier method.
- Introduce Lagrange multiplier λ (or α)

$$\text{Margin} = \frac{2}{\|\vec{w}\|}$$

$$L(\vec{w}) = \frac{\|\vec{w}\|^2}{2}$$

$$y_i = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 \end{cases}$$

=

$$y_i(\mathbf{w} \bullet \mathbf{x}_i + b) \geq 1, \quad i = 1,2,\ldots,N$$

Constrained Optimization Problem

Minimize $\| \mathbf{w} \| = \langle \mathbf{w} \cdot \mathbf{w} \rangle$ subject to $y_i(\langle \mathbf{x}_i \cdot \mathbf{w} \rangle + b) \geq 1$ for all i

Lagrangian method : maximize $\inf_{\mathbf{w}} L(\mathbf{w}, b, \alpha)$, where

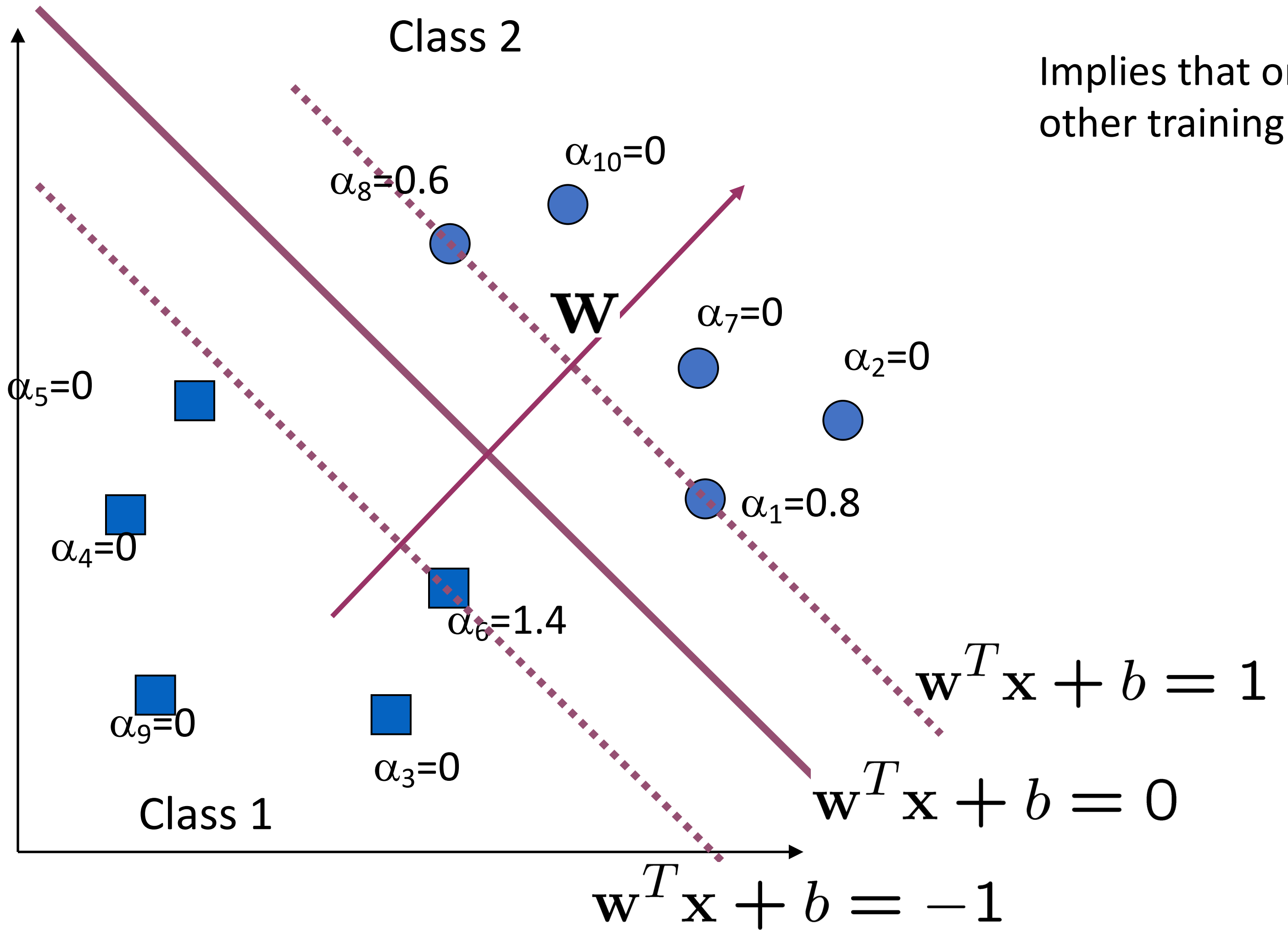
$$L(\mathbf{w}, b, \alpha) = \frac{1}{2} \| \mathbf{w} \| - \sum_i \alpha_i \left[\left(y_i (\mathbf{x}_i \cdot \mathbf{w}) + b \right) - 1 \right]$$

At the extremum, the partial derivative of L with respect both $\mathbf{w}$ and b must be 0. Taking the derivatives, setting them to 0, substituting back into L, and simplifying yields :

Maximize $\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle$

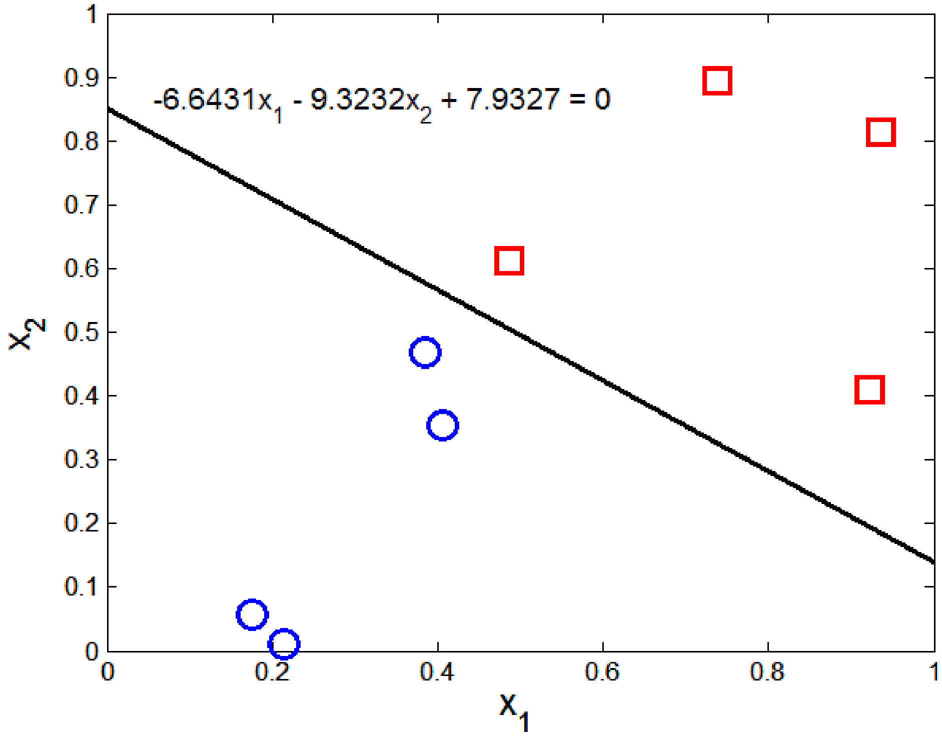
subject to $\sum_i y_i \alpha_i = 0$ and $\alpha_i \geq 0$

A Geometrical Interpretation



Implies that only support vectors matter; other training examples are ignorable.

Example of Linear SVM



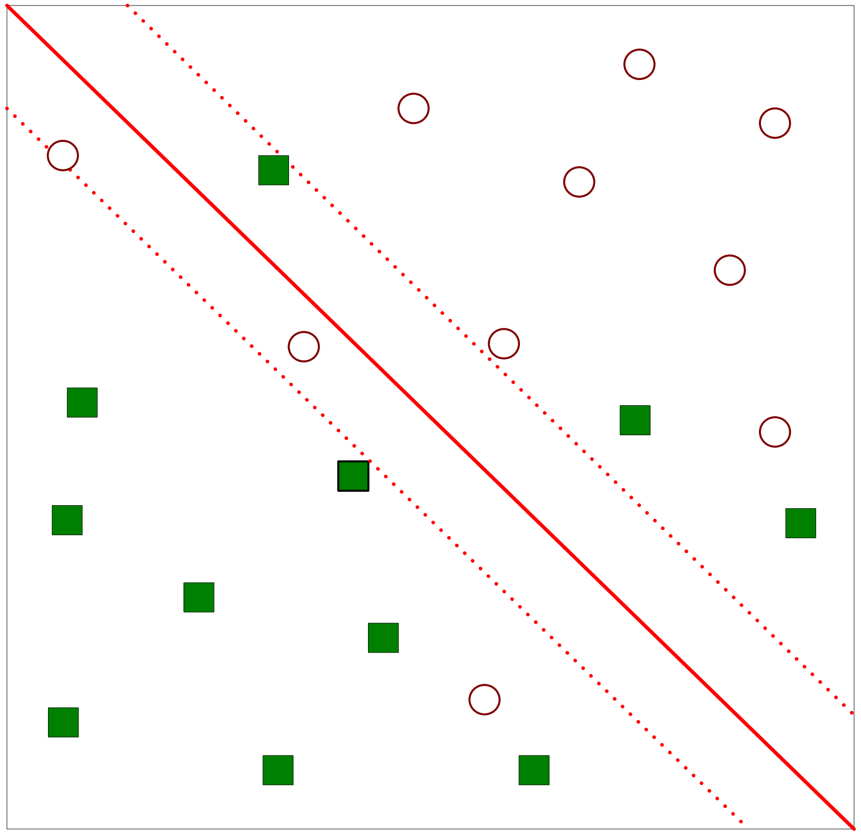
Support vectors

x1	x2	y	λ
0.3858	0.4687	1	65.5261
0.4871	0.611	-1	65.5261
0.9218	0.4103	-1	0
0.7382	0.8936	-1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	-1	0
0.2146	0.0099	1	0

$\lambda = \alpha$

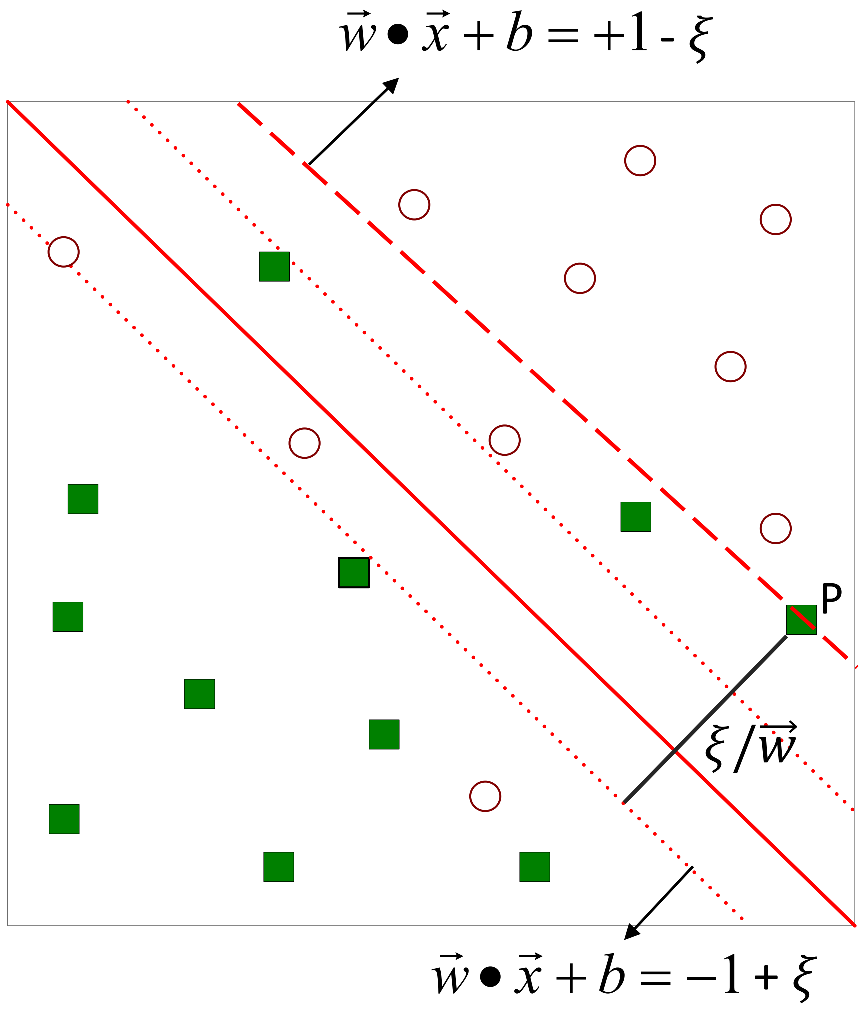
Linear SVM: Non-separable Case

- What if the problem is not linearly separable?
- We must allow for errors in our solution.



Slack Variables

- The inequality constraints must be relaxed to accommodate the nonlinearly separable data.
- This is done introducing slack variables ξ (xi) into the constrains of the optimization problem.
- ξ provides an estimate of the error of the decision boundary on the misclassified training examples.



Learning a Non-separable Linear SVM

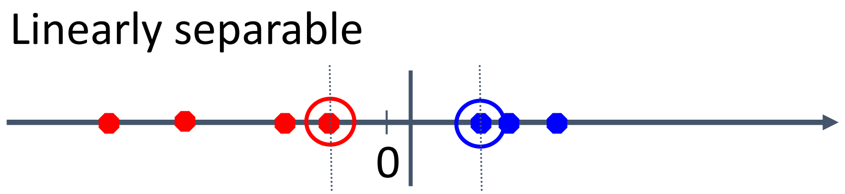
- Objective is to minimize
- Subject to to the constraints
- where *C* and *k* are user-specified parameters representing the penalty of misclassifying the training instances
- Lagrangian multipliers are constrained to $0 \leq \lambda \leq C$.

$$L(w) = \frac{\|\vec{w}\|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k\right)$$

$$y_i = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 - \xi_{\mathrm{i}} \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 + \xi_{\mathrm{i}} \end{cases}$$

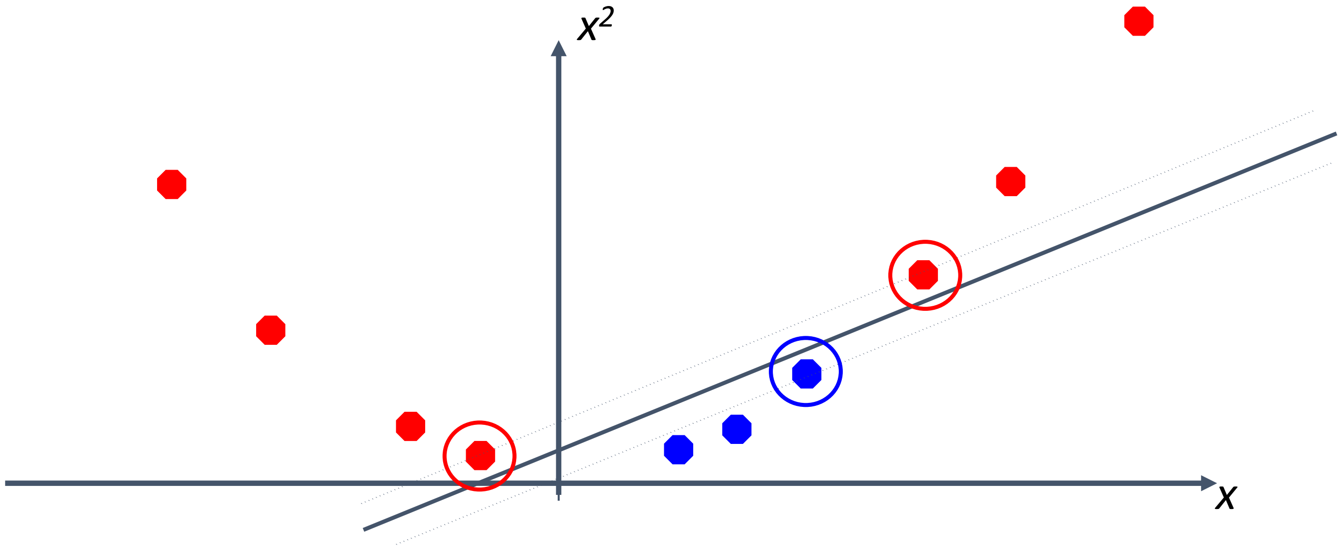
https://scikit-learn.org/stable/auto_examples/svm/plot_linearsvc_support_vectors.html#sphx-glr-auto-examples-svm-plot-linearsvc-support-vectors-py

Non-linear SVM



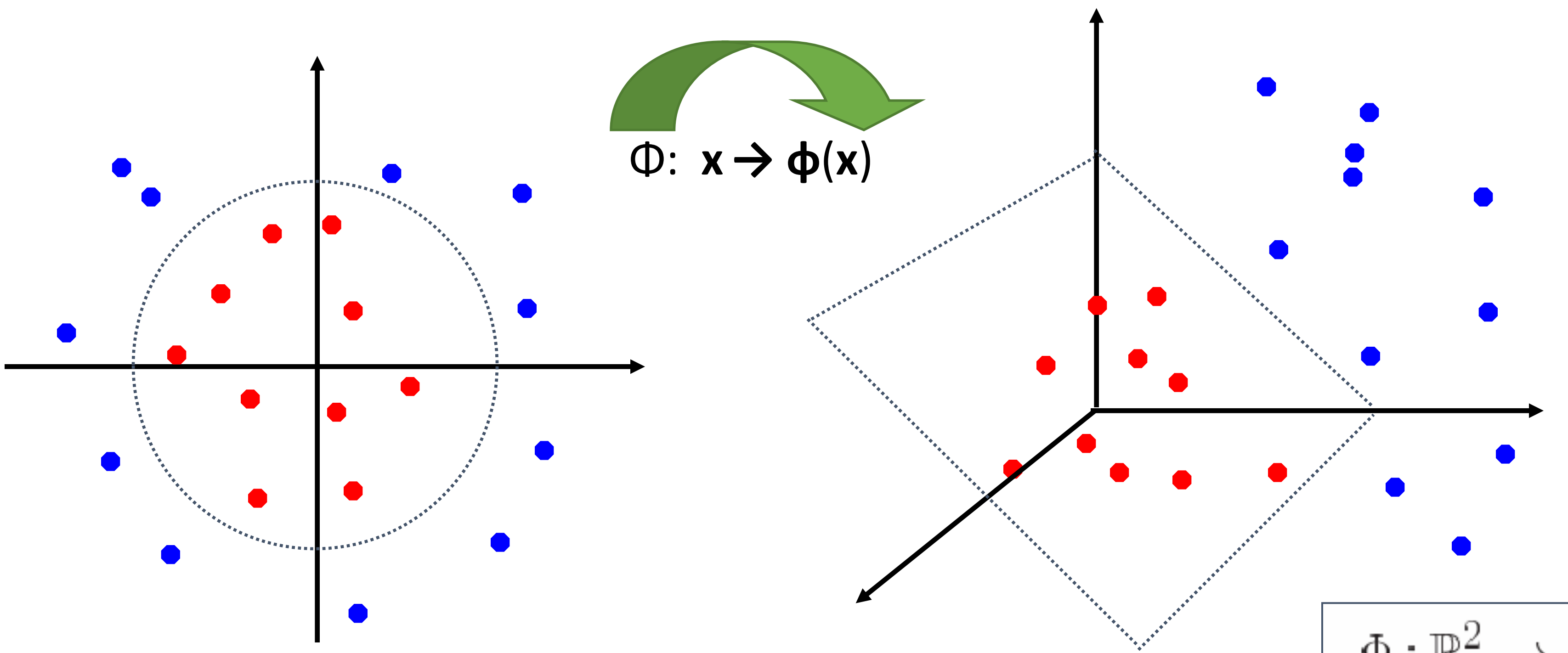
- What if the decision boundary is not linear?

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



Non-linear SVMs: Feature Spaces

Idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable.



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

Non-linear SVM

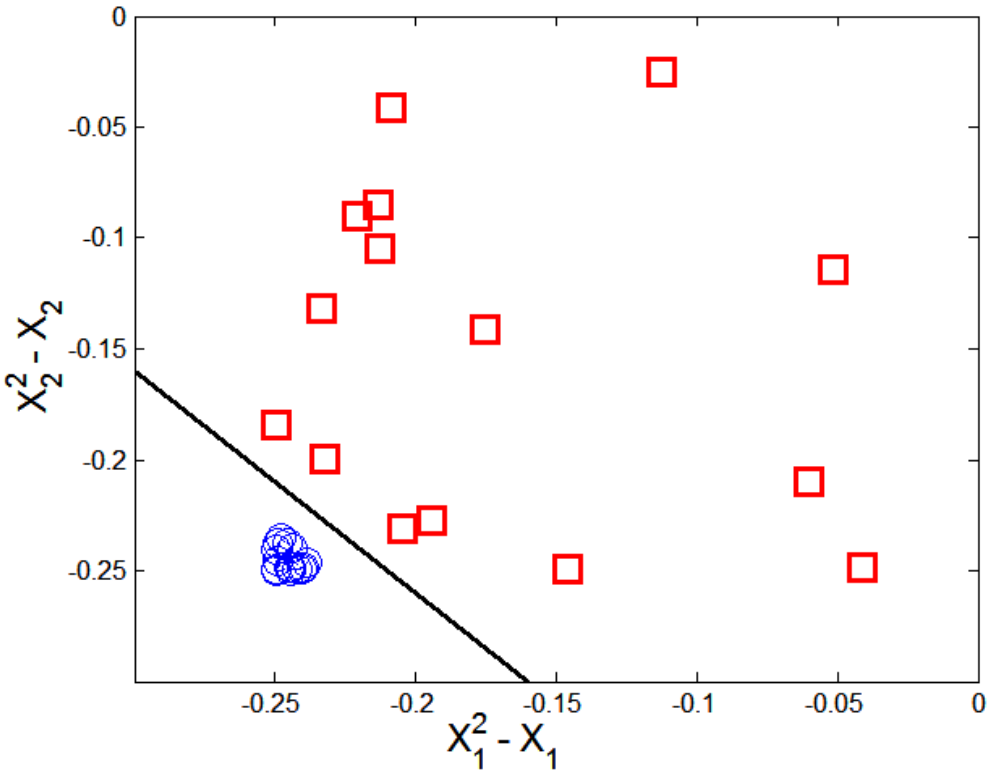
- The trick is to transform the data from its original space x into a new space $\Phi(x)$ (phi) so that a linear decision boundary can be used.

$$x_1^2 - x_1 + x_2^2 - x_2 = -0.46.$$

$$\Phi : (x_1, x_2) \longrightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1).$$

$$w_4 x_1^2 + w_3 x_2^2 + w_2\sqrt{2}x_1 + w_1\sqrt{2}x_2 + w_0 = 0.$$

- Decision boundary $\vec{w} \bullet \Phi(\vec{x}) + b = 0$



Learning a Nonlinear SVM

- Optimization problem

$$\min_{w} \frac{\|\mathbf{w}\|^2}{2}$$

$$subject\ to \quad y_i(w \cdot \Phi(x_i) + b) \geq 1,\ \forall\{(x_i, y_i)\}$$

- Which leads to the same set of equations but involve $\Phi(x)$ instead of x.

$$f(\mathbf{z}) = sign(\mathbf{w} \cdot \Phi(\mathbf{z}) + b) = sign(\sum_{i=1}^{n} \lambda_i y_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{z}) + b).$$

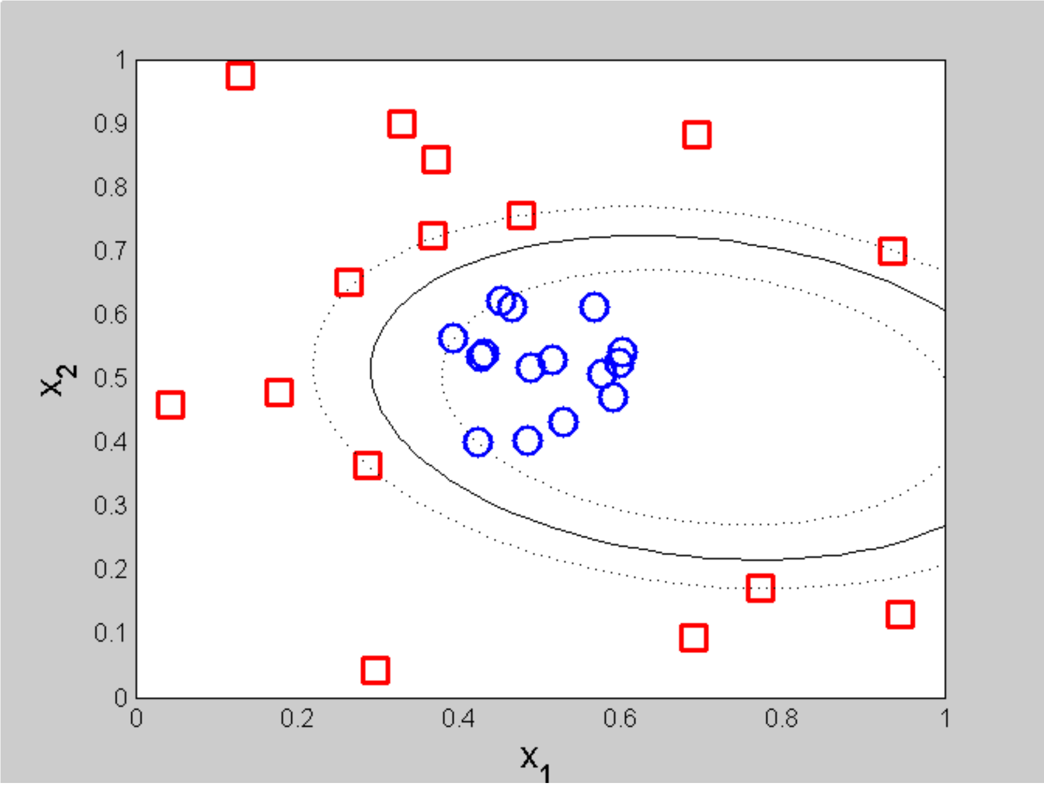
Issues:

- What type of mapping function Φ should be used?
- How to do the computation in high dimensional space?
  - Most computations involve dot product $\Phi(x) \cdot \Phi(x)$
  - Curse of dimensionality?

The Kernel Trick

- $\Phi(x) \cdot \Phi(x) = K(x_i, x_j)$
- $K(x_i, x_j)$ is a kernel function (expressed in terms of the coordinates in the original space)
- Examples:

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T\mathbf{y} + 1)^d$$
$$K(\mathbf{x}, \mathbf{y}) = \exp(-||\mathbf{x} - \mathbf{y}||^2/(2\sigma^2))$$
$$K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa\mathbf{x}^T\mathbf{y} + \theta)$$



https://scikit-learn.org/stable/auto_examples/svm/plot_svm_kernels.html#sphx-glr-auto-examples-svm-plot-svm-kernels-py
https://scikit-learn.org/stable/auto_examples/exercises/plot_iris_exercise.html#sphx-glr-auto-examples-exercises-plot-iris-exercise-py

Examples of Kernel Functions

- Polynomial kernel with degree *d*

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^d$$

- Radial basis function kernel with width σ

$$K(\mathbf{x}, \mathbf{y}) = \exp(-||\mathbf{x} - \mathbf{y}||^2 / (2\sigma^2))$$

  - Closely related to radial basis function neural networks
  - The feature space is infinite-dimensional
- Sigmoid with parameter κ and θ $\quad K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x}^T \mathbf{y} + \theta)$
  - It does not satisfy the Mercer condition on all κ and θ
- Choosing the Kernel Function is probably the most tricky part of using SVM.

The Kernel Trick

- The linear classifier relies on inner product between vectors $K(\mathbf{x}_i,\mathbf{x}_j)=\mathbf{x}_i^{\mathbf{T}}\mathbf{x}_j$
- If every datapoint is mapped into high-dimensional space via some transformation Φ: $\mathbf{x} \rightarrow \phi(\mathbf{x})$, the inner product becomes:

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

- A *kernel function* is a function that is equivalent to an inner product in some feature space.
- Example:

  2-dimensional vectors $\mathbf{x}=[x_1 \;\; x_2]$; let $K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^{\mathbf{T}}\mathbf{x}_j)^2$,

  Need to show that $K(\mathbf{x}_i,\mathbf{x}_j)= \boldsymbol{\phi}(\mathbf{x}_i)^{\mathbf{T}}\boldsymbol{\phi}(\mathbf{x}_j)$:

  $K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^{\mathbf{T}}\mathbf{x}_j)^2 = 1+ x_{i1}^2x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}=$

  $= [1 \;\; x_{i1}^2 \;\; \sqrt{2}\, x_{i1}x_{i2} \;\; x_{i2}^2 \;\; \sqrt{2}x_{i1} \;\; \sqrt{2}x_{i2}]^{\mathbf{T}} [1 \;\; x_{j1}^2 \;\; \sqrt{2}\, x_{j1}x_{j2} \;\; x_{j2}^2 \;\; \sqrt{2}x_{j1} \;\; \sqrt{2}x_{j2}] =$

  $= \boldsymbol{\phi}(\mathbf{x}_i)^{\mathbf{T}}\boldsymbol{\phi}(\mathbf{x}_j)$, where $\boldsymbol{\phi}(\mathbf{x}) = [1,\; x_1^2,\; \sqrt{2}\, x_1x_2,\; x_2^2,\; \sqrt{2}x_1,\; \sqrt{2}x_2]$

- Thus, a kernel function *implicitly* maps data to a high-dimensional space (without the need to compute each $\boldsymbol{\phi}(\mathbf{x})$ explicitly).

The Kernel Trick

$$f(\mathbf{z}) = sign(\mathbf{w} \cdot \Phi(\mathbf{z}) + b) = sign(\sum_{i=1}^{n} \lambda_i y_i \boxed{K(\mathbf{x}_i,\mathbf{z})} + b).$$

Advantages of using kernel:

- Don't have to know the mapping function Φ.
- Computing dot product $\Phi(x) \cdot \Phi(y)$ in the original space avoids curse of dimensionality.

Not all functions can be kernels

- Must make sure there is a corresponding Φ in some high-dimensional space.
- *Mercer's theorem* (see textbook) that ensures that the kernel functions can always be expressed as the dot product in some high dimensional space.

Mercer theorem: the function must be "positive-definite"

This implies that the *n* by *n* kernel matrix, in which the *(i,j)*-th entry is the $K(x_i, x_j)$, is always positive definite

This also means that optimization problem can be solved in polynomial time!

Constrained Optimization Problem with Kernel

Minimize $\| \mathbf{w} \| = \langle \mathbf{w} \cdot \mathbf{w} \rangle$ subject to $y_i(\langle \mathbf{x}_i \cdot \mathbf{w} \rangle + b) \geq 1$ for all i

Lagrangian method: maximize $\inf_{\mathbf{w}} L(\mathbf{w}, b, \alpha)$, where

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2} \| \mathbf{w} \| - \sum_i \alpha_i \left[\left(y_i (\mathbf{x}_i \cdot \mathbf{w}) + b \right) - 1 \right]$$

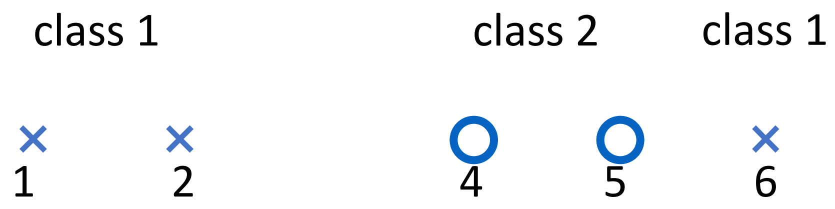
At the extremum, the partial derivative of L with respect both $\mathbf{w}$ and b must be 0. Taking the derivatives, setting them to 0, substituting back into L, and simplifying yields:

$$\text{Maximize} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \boxed{K(\mathbf{x}_i, \mathbf{x}_j)}$$

subject to $\sum_i y_i \alpha_i = 0$ and $\alpha_i \geq 0$

$\lambda = \alpha$

Example



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 values 1, 2, 6 as class 1 and 4, 5 as class 2
  - $\Rightarrow y_1=1$, $y_2=1$, $y_3=-1$, $y_4=-1$, $y_5=1$
- We use the polynomial kernel of degree 2
  - **$K(x,z) = (xz+1)^2$**
  - C is set to 100
- We first find α_i (i=1, …, 5) by

$$\text{max.} \quad \sum_{i=1}^{5} \alpha_i - \frac{1}{2} \sum_{i=1}^{5} \sum_{j=1}^{5} \alpha_i \alpha_j y_i y_j \boxed{(x_i x_j + 1)^2}$$

$$\text{subject to } 100 \geq \alpha_i \geq 0, \sum_{i=1}^{5} \alpha_i y_i = 0$$

Example

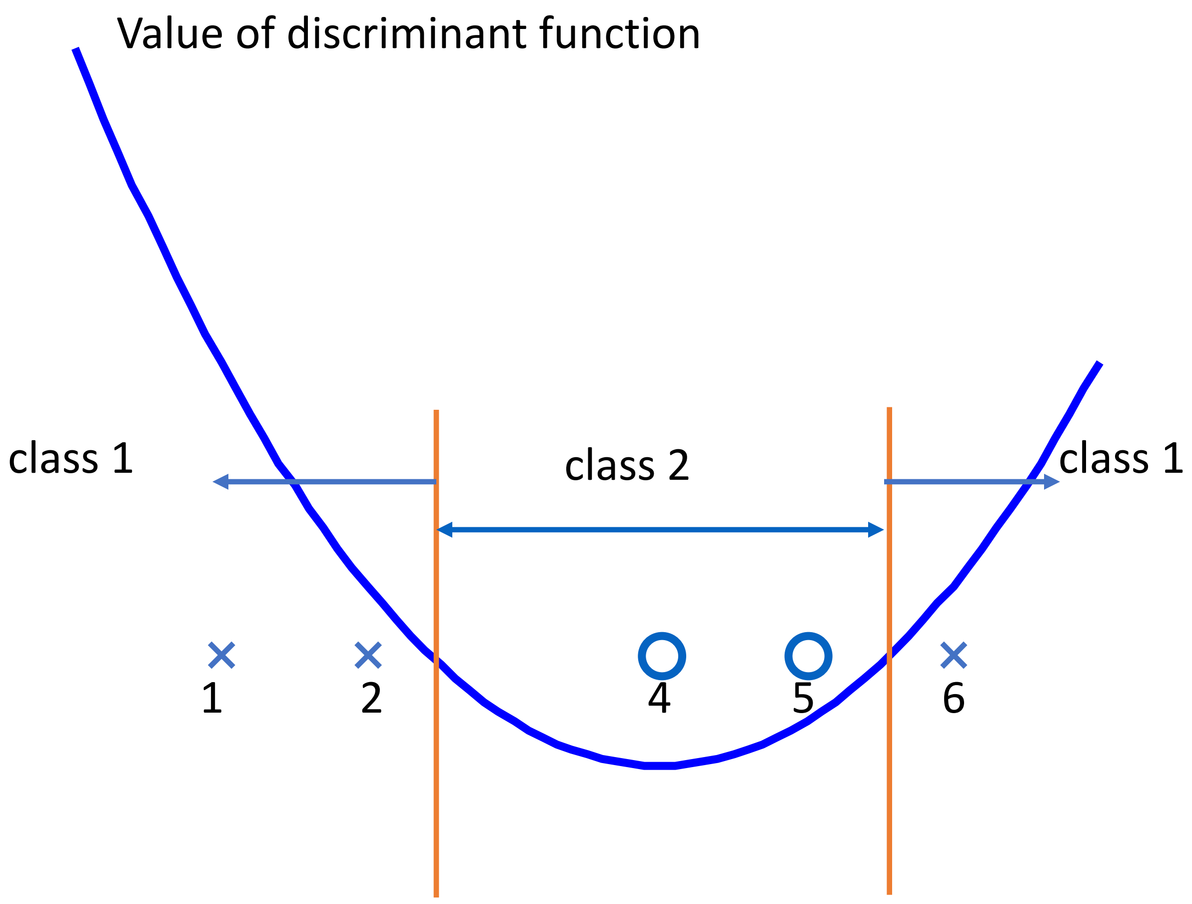
- We get
  - $\alpha_1=0$, $\alpha_2=2.5$, $\alpha_3=0$, $\alpha_4=7.333$, $\alpha_5=4.833$
  - Note that the constraints are indeed satisfied
  - The support vectors are $\{x_2=2, x_4=5, x_5=6\}$
- The discriminant function is

$$
\begin{aligned}
& f(z) \\
&= 2.5(1)(2z+1)^2 + 7.333(-1)(5z+1)^2 + \overset{\alpha_5}{4.833}(\overset{y_5}{1})\overset{K(z,x_5)}{(6z+1)^2} + b \\
&= 0.6667z^2 - 5.333z + b
\end{aligned}
$$

- *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 $\phi(\mathbf{w})^T\phi(\mathbf{x}) + b = -1$

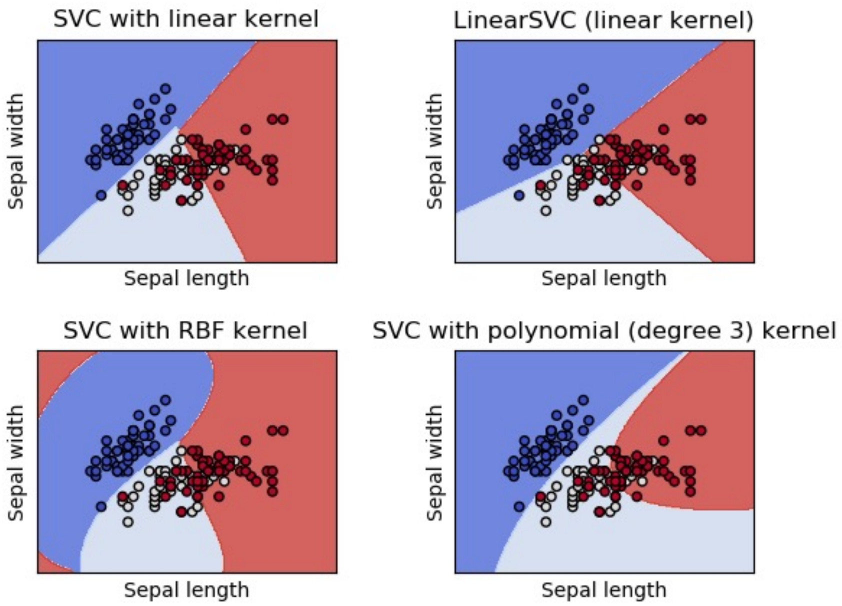
- All three give b=9 $\Longrightarrow$ $f(z) = 0.6667z^2 - 5.333z + 9$

Example


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

Support Vector Machine (SVM)

- SVM represents the decision boundary using a subset of the training examples, known as the **support vectors**.
- The basic idea behind SVM lies within the concept of **maximal margin hyperplane.**



Characteristics of SVM

- Since the learning problem is formulated as a convex optimization problem, efficient algorithms are available to find the **global** minima of the objective function (many of the other methods use greedy approaches and find **locally** optimal solutions).
- Overfitting is addressed by maximizing the margin of the decision boundary, but the user still needs to provide the type of kernel function and cost function.
- Difficult to handle missing values.
- Robust to noise.
- High computational complexity for building the model.

References

- Support Vector Machine (SVM). Chapter 5.5. Introduction to Data Mining.
- http://www.kernel-machines.org/
- http://www.support-vector.net/
- An Introduction to Support Vector Machines. N. Cristianini and J. Shawe-Taylor.
- C.J.C. Burges: A tutorial on Support Vector Machines. Data Mining and Knowledge Discovery 2:121-167, 1998.