Support Vector Machines
(Sparse Kernel Machines)

PPGEE/UFMG

Preliminaries

- Linear Classifiers:
  - Decision Boundary: corresponds to a $(p-1)$-dimension hyperplane in $\mathbb{R}^p$ [1]
  - $p = 2 \Rightarrow$ straight line
  - $p = 3 \Rightarrow$ plane
  - $p > 3 \Rightarrow$ Hyperplane
- One single data point $\vec{x}_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,p}\}$ lies in the hyperplane if

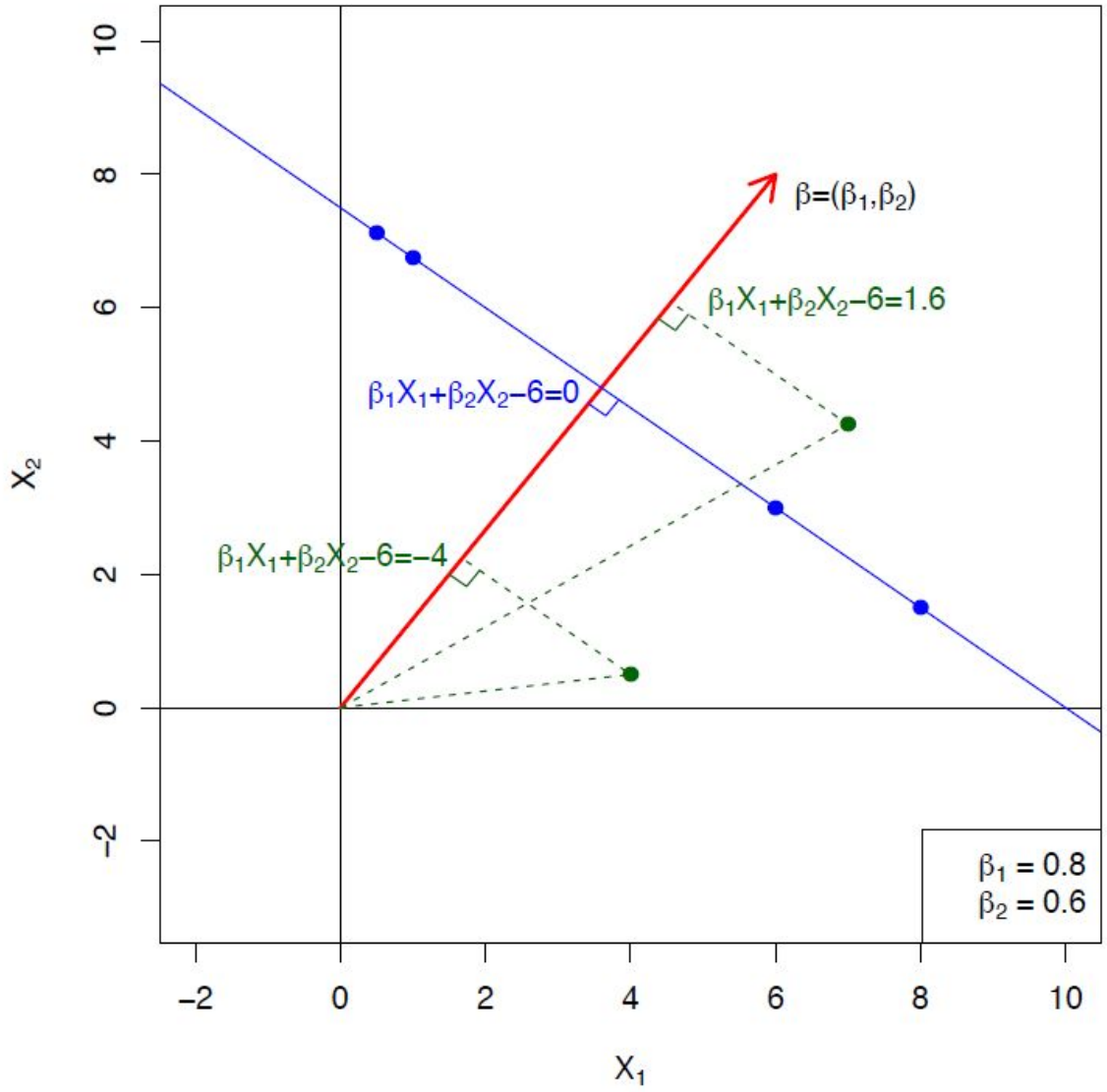
$$\beta_1 \cdot x_{i,1} + \beta_2 \cdot x_{i,2} + \ldots + \beta_p \cdot x_{i,p} + \beta_0 = 0$$

also written in a vector form

$$\vec{\beta}^T \cdot \vec{x}_i + \beta_0 = 0$$

[1] p is the number of the predictor variables (input space dimension)

Hyperplane in R^2



Classification using a Hyperplane

- A new notation for SVMs:
  - Consider the dataset $D = \{(\mathbf{x}_1, t_i), (\mathbf{x}_2, t_2) \ldots, (\mathbf{x}_N, t_N)\}$
  - where $\mathbf{x}_i = \{x_{i,1}, \ldots, x_{i,p}\}$ and $t_i = \{-1, +1\}$
- One can classify an observation $\mathbf{x}_i$ from the signal of hyperplane

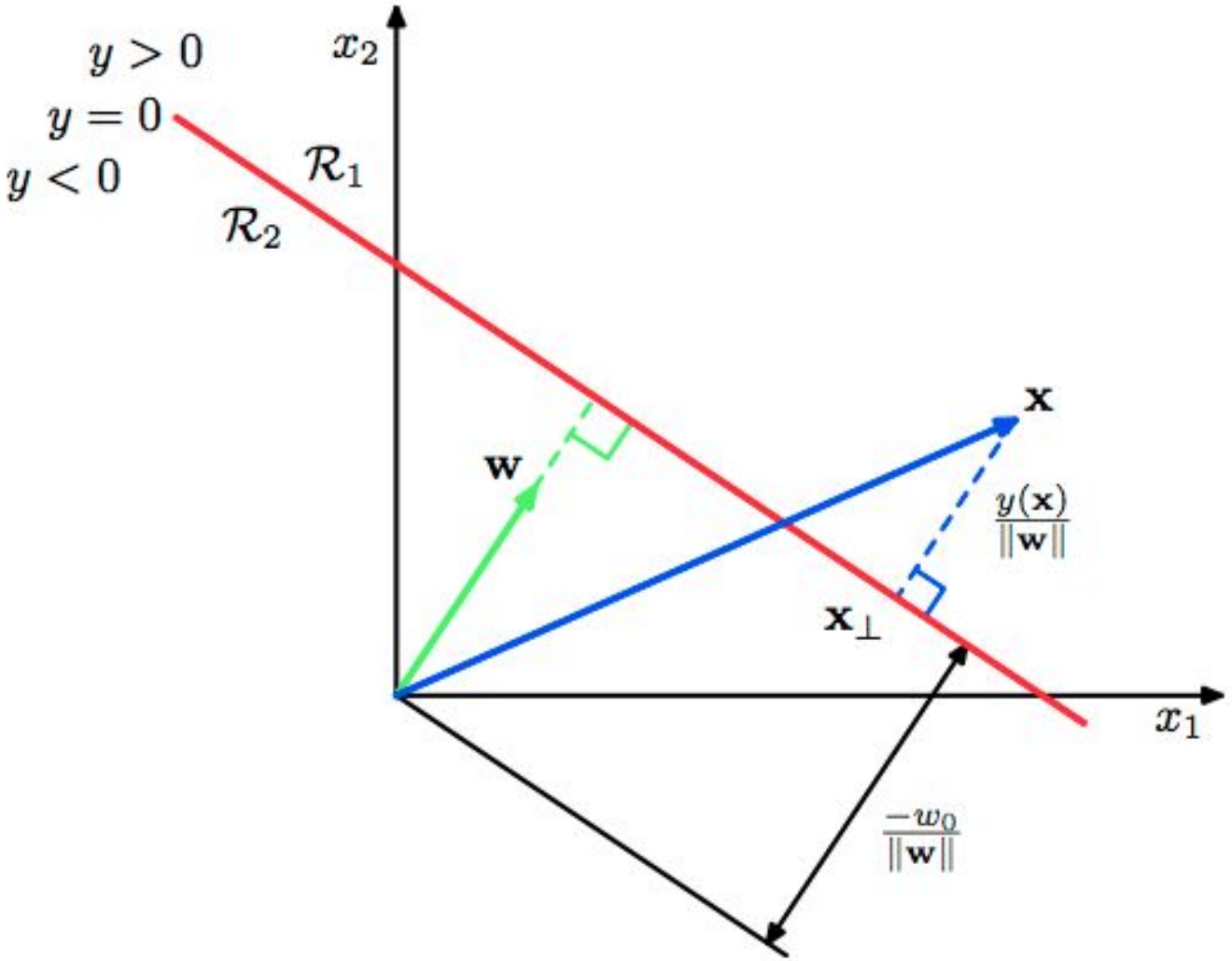
$$y_i = f(\mathbf{x}_i) = \mathbf{w}^T \cdot \mathbf{x}_i + b$$

$$+1 \text{ (positive) if } f(\mathbf{x}_i) \geq 0$$

$$-1 \text{ (negative) if } f(\mathbf{x}_i) < 0.$$

- Important: the magnitude of $f(\mathbf{x}_i)$ is proportional to distance of $\mathbf{x}_i$ to the hyperplane.

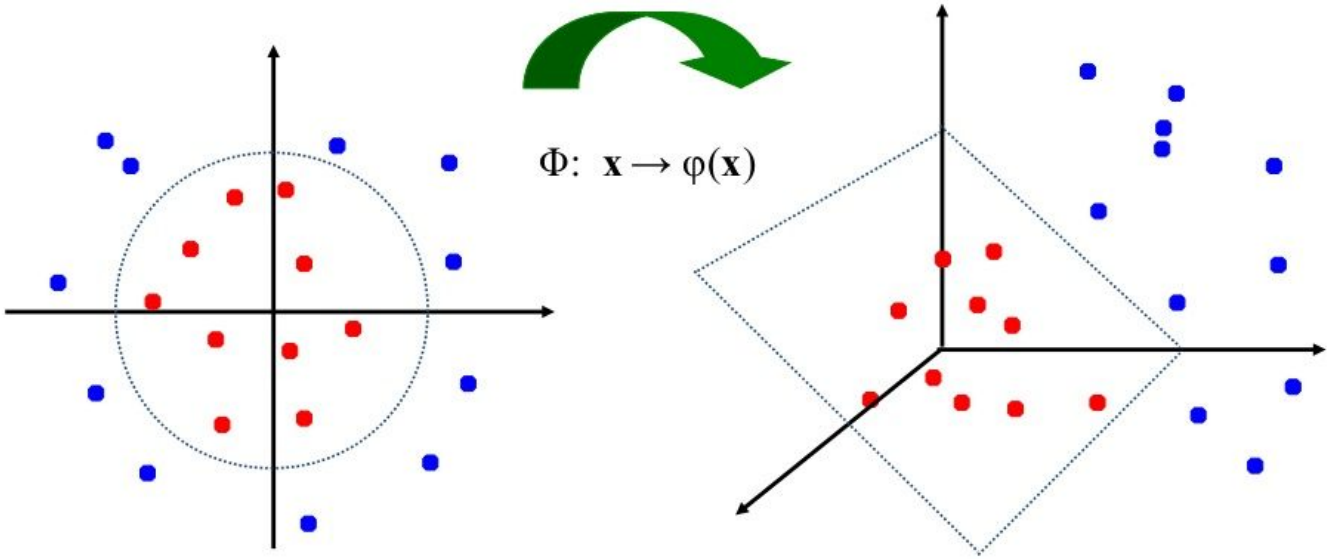
Geometry of Linear Decision Boundary



Preliminaries

Non-linear SVMs: Feature spaces

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



SVM - Problem Setting

Assume for the moment that the training data set is **linearly separable** in feature space, so that by definition there exists at least one choice of the parameters $\mathbf{w}$ and b such that

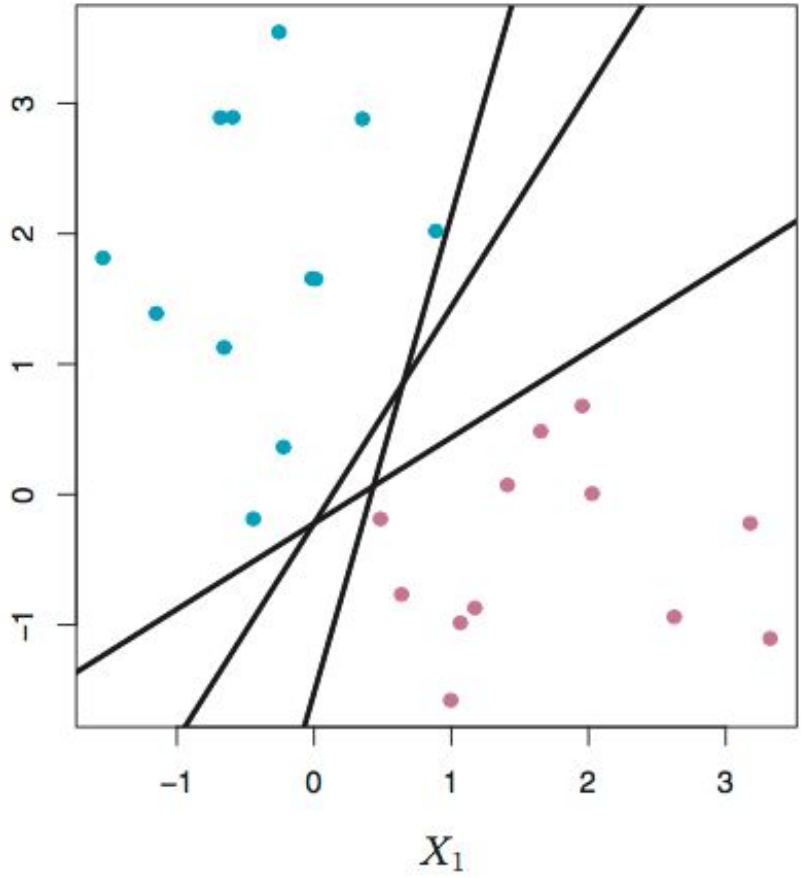
$$\begin{cases} \mathbf{w}^T\phi(\mathbf{x}_i) + b \geq 0 & \forall\, t_i = 1 \\ \mathbf{w}^T\phi(\mathbf{x}_i) + b < 0 & \forall\, t_i = -1 \end{cases}$$

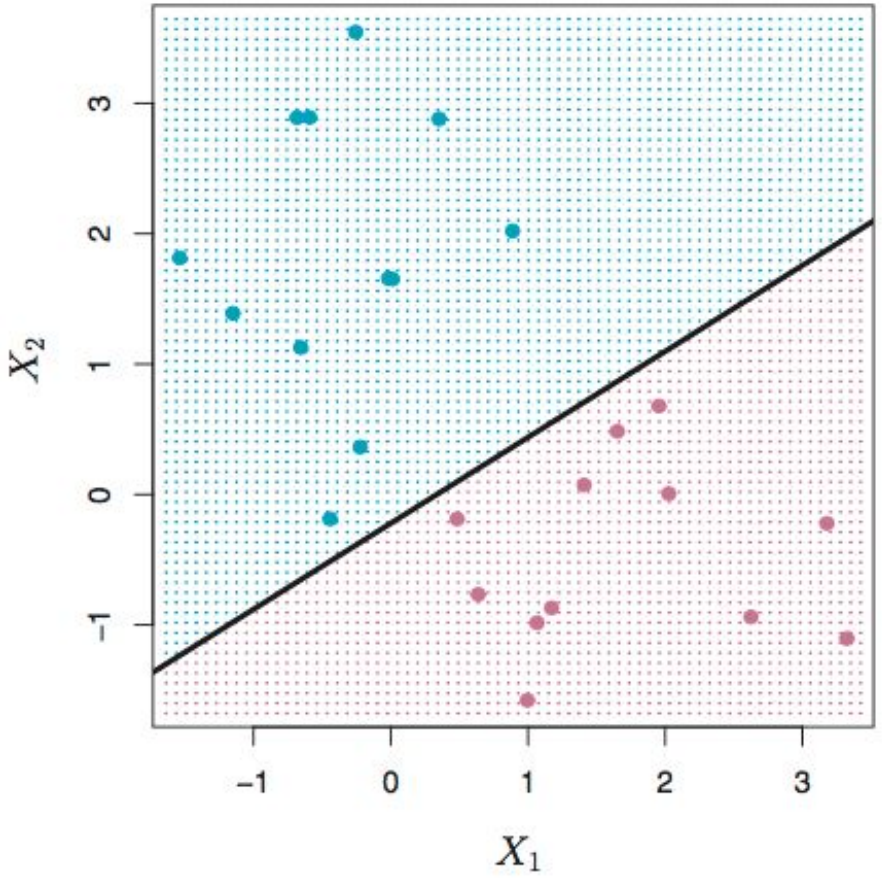
which can be briefly written as

$$t_i[\mathbf{w}^T\phi(\mathbf{x}_i) + b] \geq 0$$

There may of course exist many such solutions that separate the classes exactly.

SVM - Problem Setting





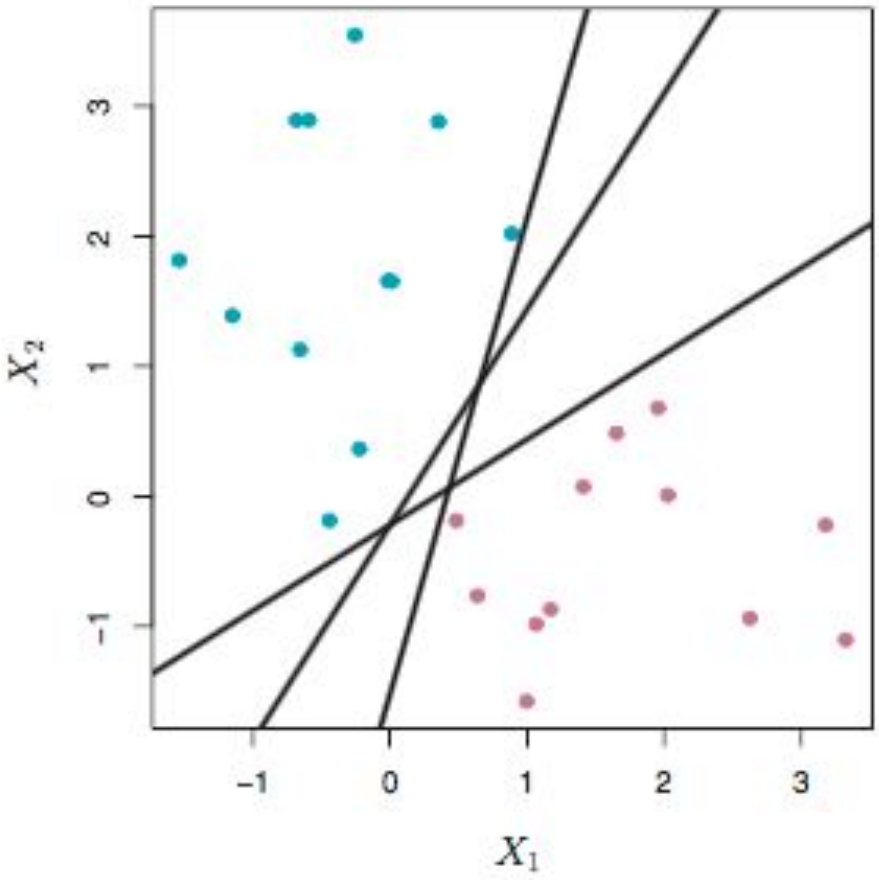
SVM - Problem Setting

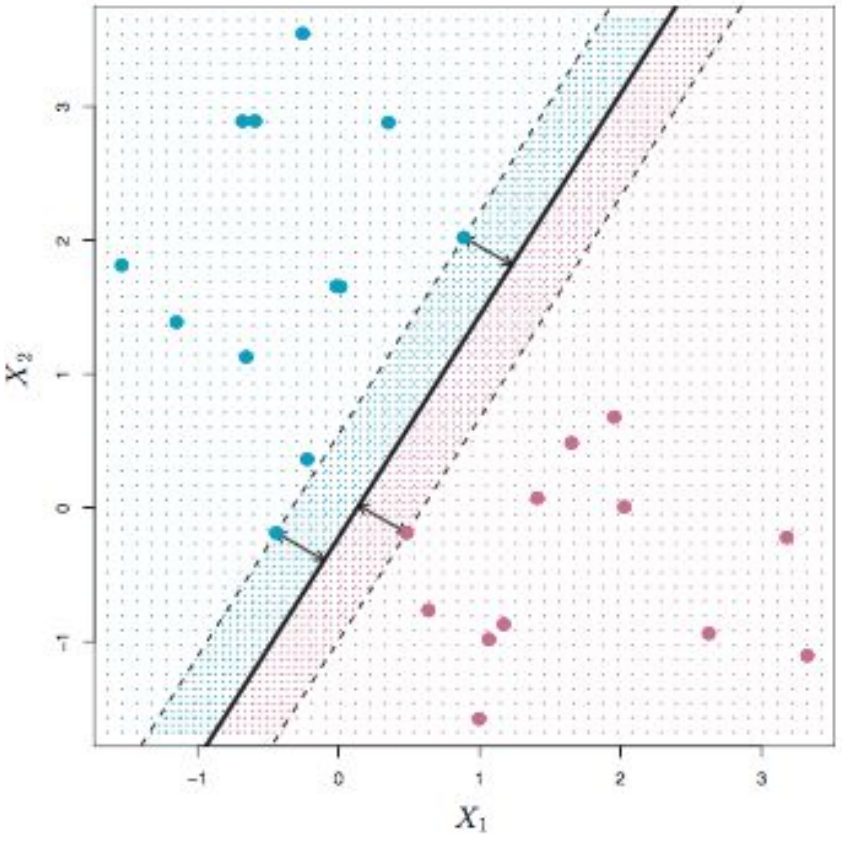
- in order to choice the decision boundary that gives the smallest generalization error, the SVM uses the concept of **margin**.
  - **margin**: is defined as the perpendicular distance between the decision boundary and the closest training patterns.

- maximizing the margin leads to a particular choice of decision boundary.

- Why maximum margin would lead to the **smallest generalization error**?
  - see Vapnik, Vladimir (2000). *The nature of statistical learning theory*. Springer.

SVM - Problem Setting





Hard Margin SVM

the perpendicular distance of a pattern $\mathbf{x}_i$ to the hyperplane is given by

$$\frac{t_i[\mathbf{w}^T\phi(\mathbf{x}_i)+b]}{||\mathbf{w}||}$$

margin (ρ): the perpendicular distance to the closest training pattern.

From all possible $(\mathbf{w}, b)$, SVM search for the solution that maximizes ρ. Then, the maximum margin can be found by solving

$$\max_{\mathbf{w},b} \quad \rho = \left\{ \frac{1}{||\mathbf{w}||} \min_i \left[t_i[\mathbf{w}^T\phi(\mathbf{x}_i)+b]\right] \right\}$$

$$\text{s.t.} \quad t_i[\mathbf{w}^T\phi(\mathbf{x}_i)+b] \geq 0 \quad i = 1, \ldots, N$$

Hard Margin SVM

to convert the earlier optimization problem into an equivalent problem that is much easier to solve, we make the rescaling

$$\mathbf{w} = \kappa \mathbf{w} \quad \text{and} \quad b = \kappa b \quad \text{with } \kappa > 0$$

so that the closest training pattern will have $t_i[\mathbf{w}^T \phi(\mathbf{x}_i) + b] = 1$. This is known as the canonical representation of the decision hyperplane.

Thus, the problem of finding $(\mathbf{w}, b)$ that maximizes the margin will be

$$\begin{aligned} \max_{\mathbf{w},b} \quad & \frac{1}{||\mathbf{w}||} \\ \text{s.t.} \quad & t_i[\mathbf{w}^T \phi(\mathbf{x}_i) + b] \geq 1 \quad \forall\, i = 1, \ldots, N \end{aligned}$$

or equivalently

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & t_i[\mathbf{w}^T \phi(\mathbf{x}_i) + b] \geq 1 \quad \forall\, i = 1, \ldots, N \end{aligned}$$

Hard Margin SVM

The SVM's Primal Optimization Problem: (hard-margin)

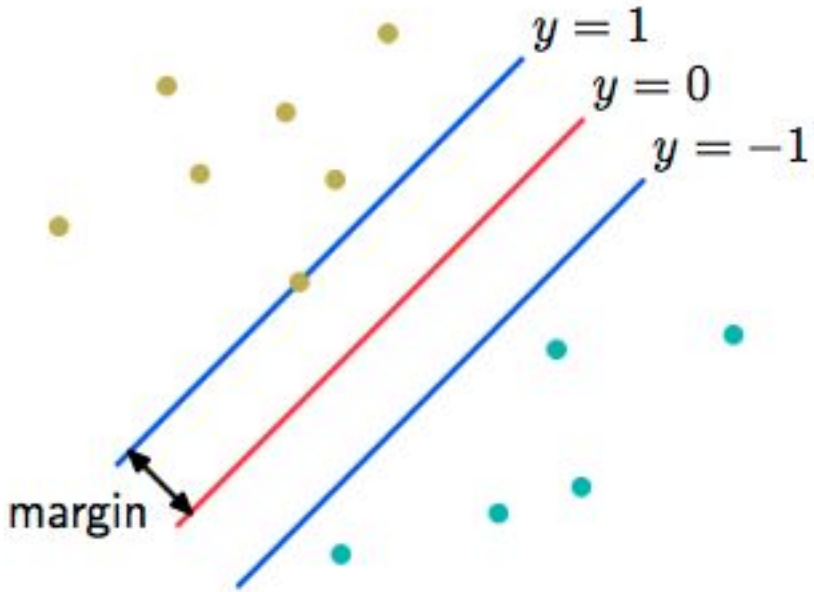
$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & t_i[\mathbf{w}^T\phi(\mathbf{x}_i) + b] \geq 1 \quad \forall\, i = 1, \ldots, N \end{aligned}$$

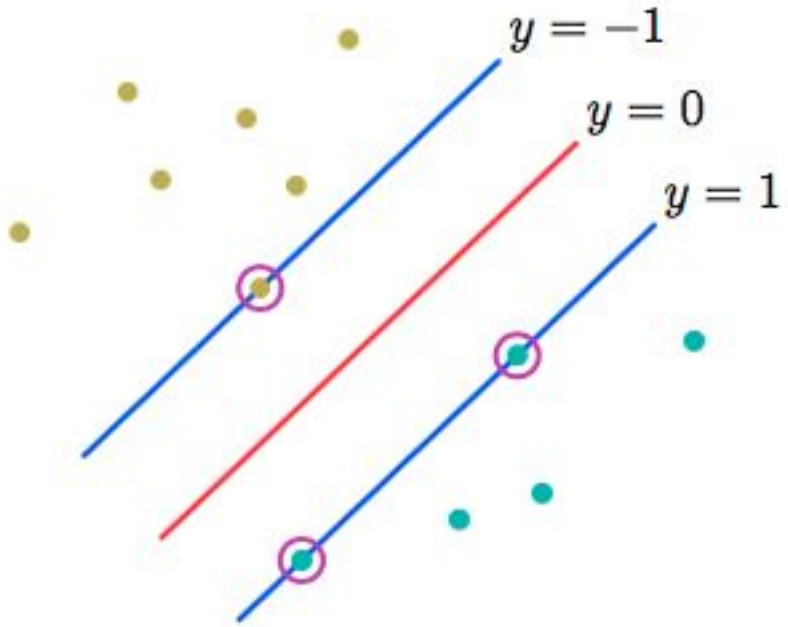
the constraints are said to be:

- **active**: for the patterns for which the equality holds.
- **inactive**: for the patterns for which the equality does not hold.

By definition, there will always be at least one active constraint, because there will always be a closest point, and once the margin has been maximized there will be **at least two active constraints**.

Hard Margin SVM





SVM's optimization problem

The SVM's Primal Optimization Problem: (hard-margin)

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & t_i[\mathbf{w}^T\phi(\mathbf{x}_i)+b] \geq 1 \quad \forall\, i = 1,\ldots,N \end{aligned}$$

Quadratic programming problem: minimizing a quadratic function subject to a set of linear inequality constraints.

In order to solve it, we introduce Lagrange multipliers $a_i > 0$, with one multiplier an for each of the constraints, giving the Lagrangian function

$$L(\mathbf{w},b,\mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} a_i[t_i(\mathbf{w}^T\phi(\mathbf{x}_i)+b)-1]$$

we should minimize $L(\mathbf{w},b,\mathbf{a})$ with respect to $\mathbf{w}$ and b, and maximizing with respect to $\mathbf{a}$.

SVM's optimization problem

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} a_i[t_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) - 1]$$

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} a_i t_i \mathbf{w}^T\phi(\mathbf{x}_i) - b\sum_{i=1}^{N} a_i t_i + \sum_{i=1}^{N} a_i$$

analyzing the partial derivatives $\frac{\partial L(\mathbf{w},b,\mathbf{a})}{\partial \mathbf{w}} = 0$ and $\frac{\partial L(\mathbf{w},b,\mathbf{a})}{\partial b} = 0$ give us

$$\mathbf{w} = \sum_{i=1}^{N} a_i t_i \phi(\mathbf{x}_i)^T$$

$$\sum_{i=1}^{N} a_i t_i = 0$$

SVM's optimization problem

Eliminating $\mathbf{w}$ and b from $L(\mathbf{w}, b, \mathbf{a})$ gives the **dual formulation** of the maximum margin problem in which

$$\max_{\mathbf{a}} \quad \tilde{L}(\mathbf{a}) = \sum_{i=1}^{N} a_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j t_i t_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

$$\text{s.t.} \quad a_i > 0 \quad \forall\, i = 1, \ldots, N$$

$$\sum_{j=1}^{N} a_i t_i = 0$$

Again, this takes the form of a quadratic programming problem.

Note: the dual formulation based on the inner product $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ allows to rewrite the problem using Kernels.

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

so that the maximum margin classifier can be applied efficiently to feature spaces whose dimensionality exceeds the number of data points, including infinite feature spaces.

SVM Prediction

Having solved the quadratic programming problem and found a value for **a**, one can then determine the SVM prediction function

$$y(\mathbf{x}) = \sum_{i=1}^{N} a_i t_i K(\mathbf{x}, \mathbf{x}_i) + b$$

and the classification is given by $sign(y(\mathbf{x}))$.

- Polynomial
- Gaussian Radial-basis function (RBF)
- Two-layer perceptron

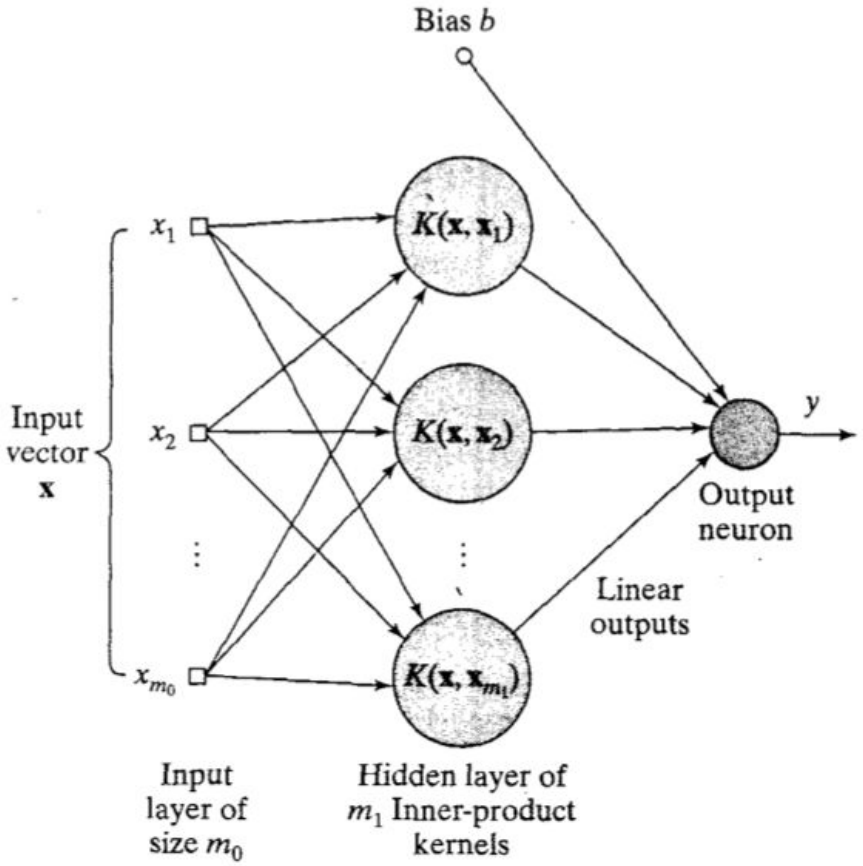
$$K(x, x_i) = (x^T x_i + 1)^p$$

$$K(x, x_i) = \exp\left(-\frac{1}{2\sigma^2}\|x - x_i\|^2\right)$$

$$K(x, x_i) = \tanh\left(\beta_0 x^T x_i + \beta_1\right)$$

SVM Prediction

$$y(\mathbf{x}) = \sum_{i=1}^{N} a_i t_i K(\mathbf{x}, \mathbf{x}_i) + b$$



Support Vectors

Karush-Kuhn-Tucker (KKT) conditions for the optimization problem

$$a_i \geq 0 \tag{1}$$

$$t_i y(\mathbf{x}_i) - 1 \geq 0 \tag{2}$$

$$a_i(t_i y(\mathbf{x}_i) - 1) = 0 \tag{3}$$

From condition (3), for every training pattern either, $a_i = 0$ or $t_i y(\mathbf{x}_i) = 1$.

- any pattern for which $a_i = 0$ plays no role in making predictions for new data points; its effect in the SVM output will be null.

- the remaining data points are called support vectors, and because they satisfy $t_i y(\mathbf{x}_i) = 1$, they correspond to points that lie on the maximum margin hyperplanes in feature space (contour lines).

Note: once the model is trained, a significant proportion of the data points can be discarded and only the support vectors retained.

Bias (threshold)

The bias parameter b is determined by noting that any support vector $\mathbf{x}_i$ satisfies $t_i y(\mathbf{x}_i) = 1$.

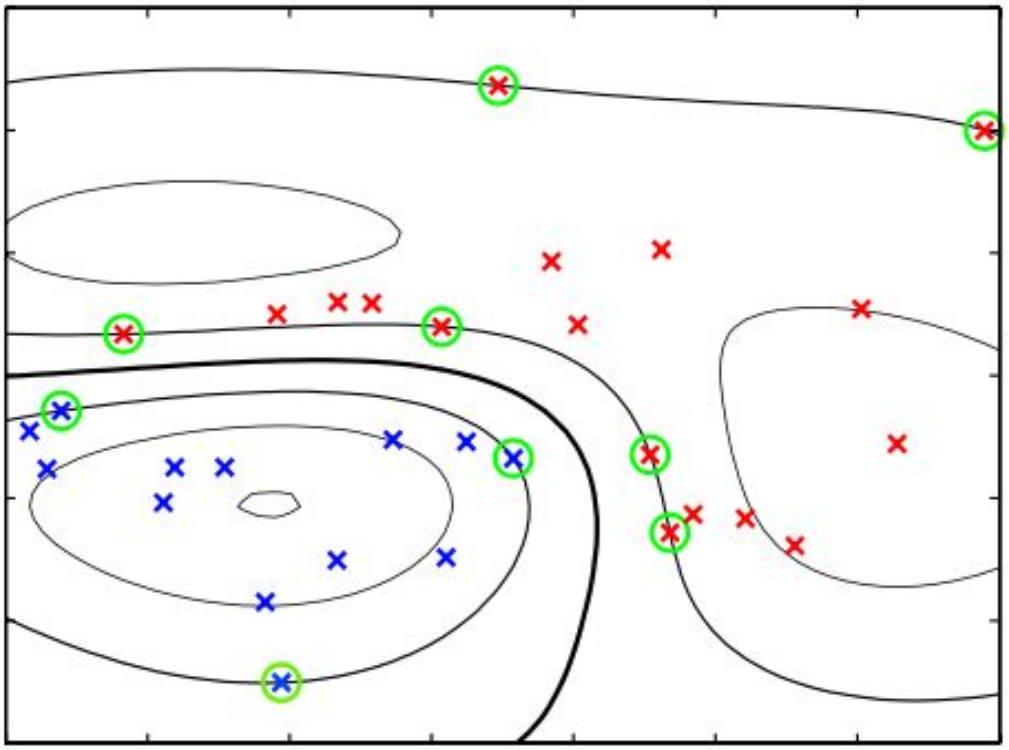
Choosing an arbitrary support vector from the positive class (for which $t_1 = 1$), we have

$$\sum_{m \in \mathbb{S}} a_m t_m K(\mathbf{x}_i, \mathbf{x}_m) + b = 1$$

$$b = 1 - \sum_{m \in \mathbb{S}} a_m t_m K(\mathbf{x}_i, \mathbf{x}_m)$$

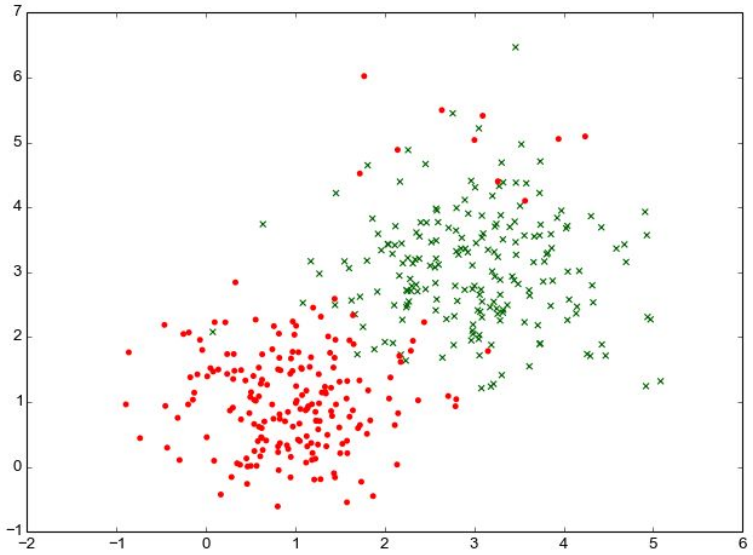
where $\mathbb{S}$ denotes the set of indices of the support vectors.

SVM Example



contours of constant y(x) obtained from a support vector machine having a Gaussian kernel function (extracted from Bishop, 2006)

Overlapping Class Distributions

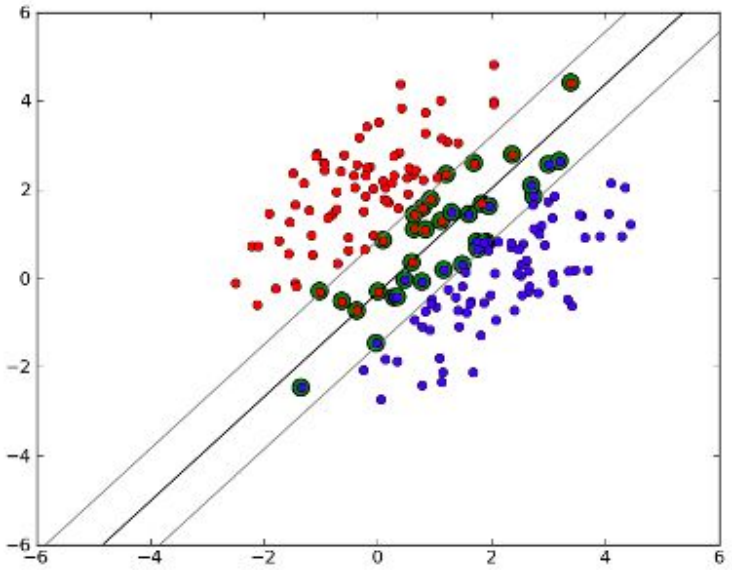


So far, we have assumed that the training data points are linearly separable in the feature space $\phi(\mathbf{x})$.

The resulting support vector machine will give exact separation of the training data in the original input space x, although the corresponding decision boundary will be nonlinear.

In practice, however, the class-conditional distributions may overlap, in which case exact separation of the training data can lead to poor generalization.

Overlapping Class Distributions



The idea is to modify the SVM original formulation so that data points are allowed to be on the 'wrong side' of the margin boundary, but with a penalty that increases with the distance from that boundary.

Penalty is a linear function of this distance!

But how this can be implemented?

Soft Margin SVM

N slack variables $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_N$ are introduced to the problem.

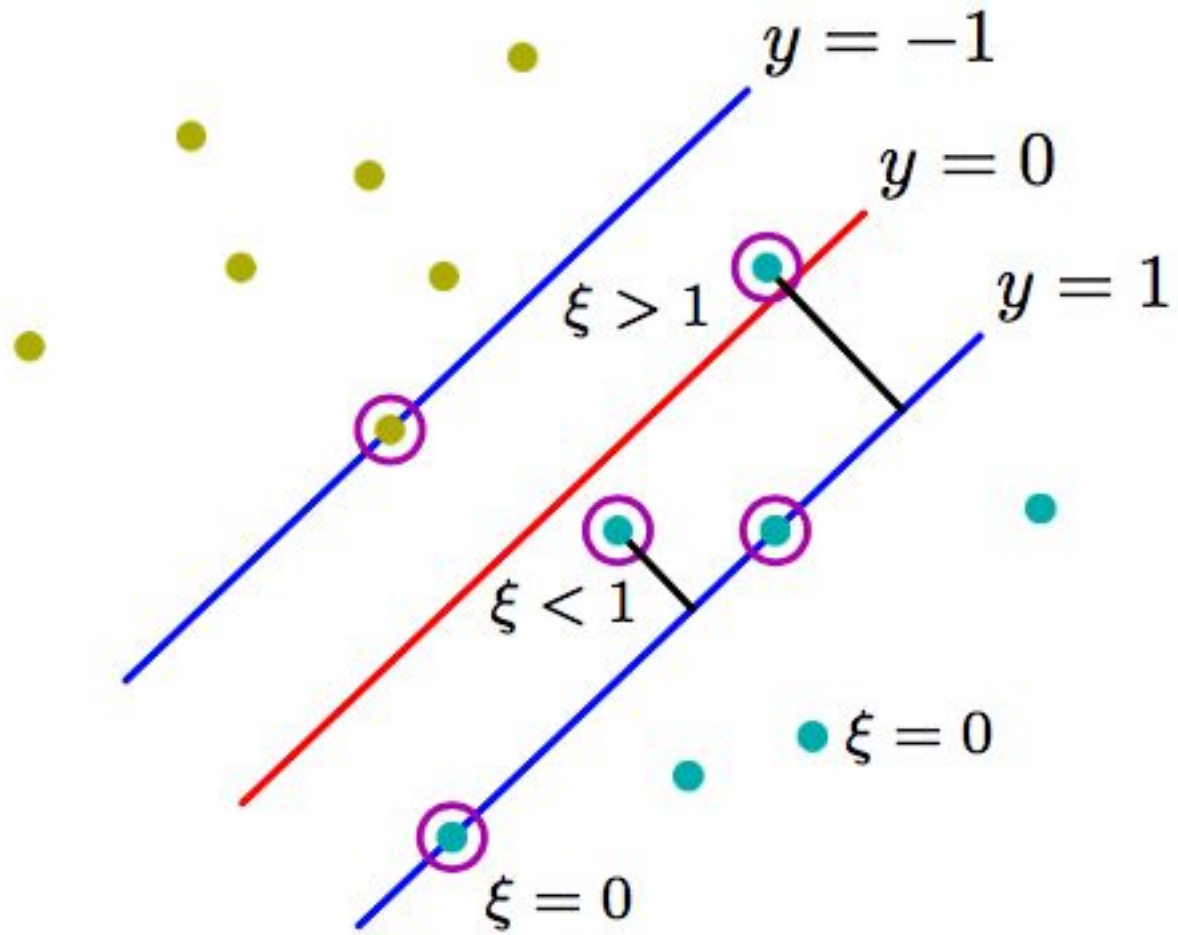
there is one slack variable ε_i for each training pattern.

$$\varepsilon_i = \begin{cases} 0 & \text{if } t_i(\mathbf{w}^T \phi(\mathbf{x}_i) + b) \geq 1, \\ |t_i - (\mathbf{w}^T \phi(\mathbf{x}_i) + b)| & \text{otherwise.} \end{cases}$$

such that

- $\epsilon_i = 0$: for (correctly classified) patterns that are either on the margin or on the correct side of the margin;
- $0 < \epsilon_i \leq 1$: for patterns that lie inside the margin, but on the correct side of the decision boundary;
- $\epsilon_i > 1$: for patterns that lie on the wrong side of the decision boundary and are misclassified;

Soft Margin SVM



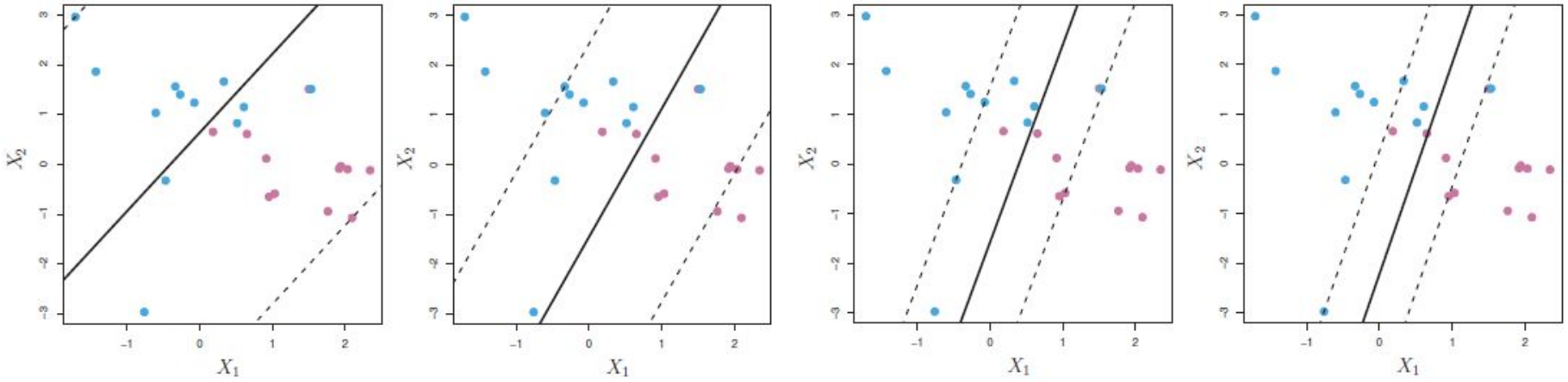
Soft Margin SVM

- The SVM's Primal Optimisation Problem: (soft-margin)

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\varepsilon_i \\ \text{s.t.} \quad & t_i[\mathbf{w}^T\phi(\mathbf{x}_i) + b] \geq 1 - \varepsilon_i \quad \forall\, i = 1, \ldots, N \end{aligned}$$

- the parameter $C > 0$ is a limiting value for the sum of the slack variables $\sum_{i=1}^{N}\varepsilon_i$.
- thus, C controls the amount and the severity of the allowed margin violations.
  - when C is large: the margin is increased. More errors are allowed.
  - when C is small (tends to zero): the margin becomes more rigid.

Soft Margin SVM

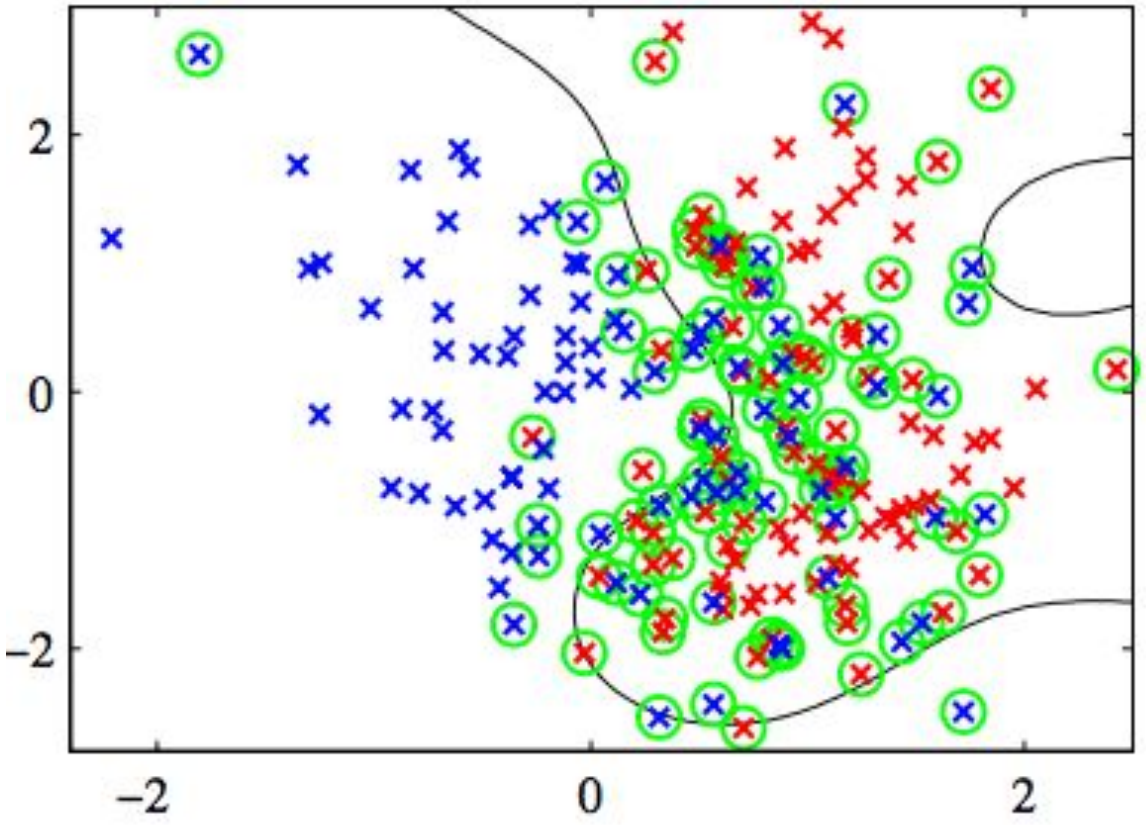


- the parameter $C > 0$ controls the trade-off between the bias and variance of the model.
- When C is large, the margin is wide, and several observations become support vectors. The classifier has potentially a low variance and a high bias.
- When C is small, fewer observations become support vectors, leading to a classifier with potentially high variance and low bias.

Soft Margin SVM

- Selection of the parameter C for SVMs - Cross-Validation:
  - Choose a grid of C values and compute cross-validation error rate for each value of C.
  - Then, select the tuning parameter C for which the cross-validation error rate is smallest.
  - Finally, the SVM model is re-fit using all the available observations and the selected value of the C parameter.

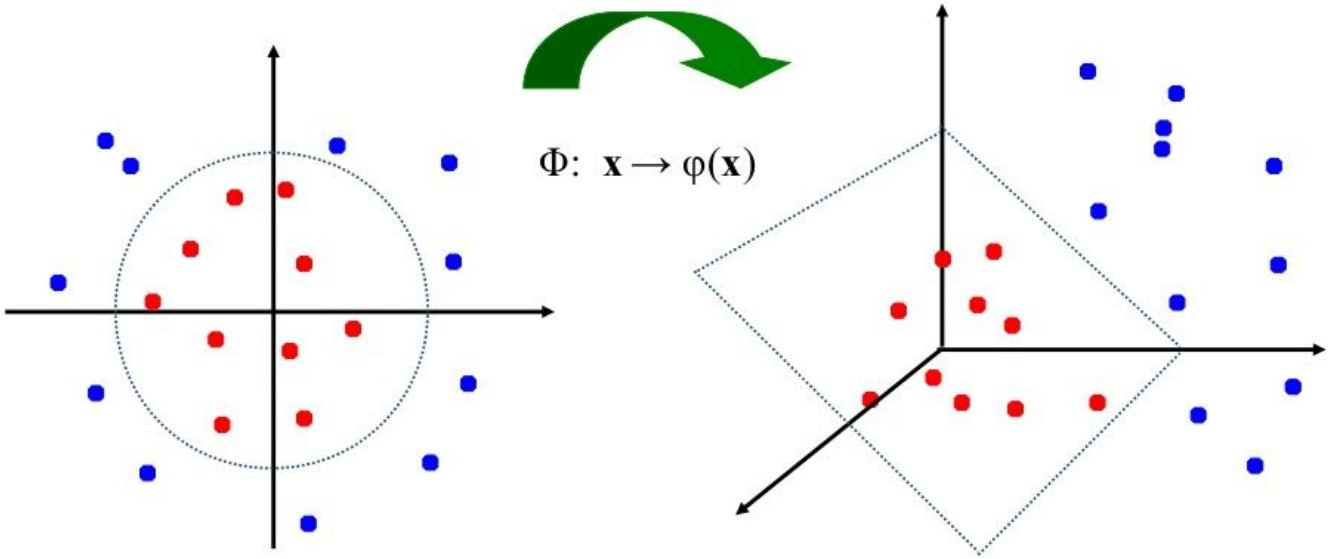
SVM Example (class overlapping)



Non-linear SVM

Non-linear SVMs: Feature spaces

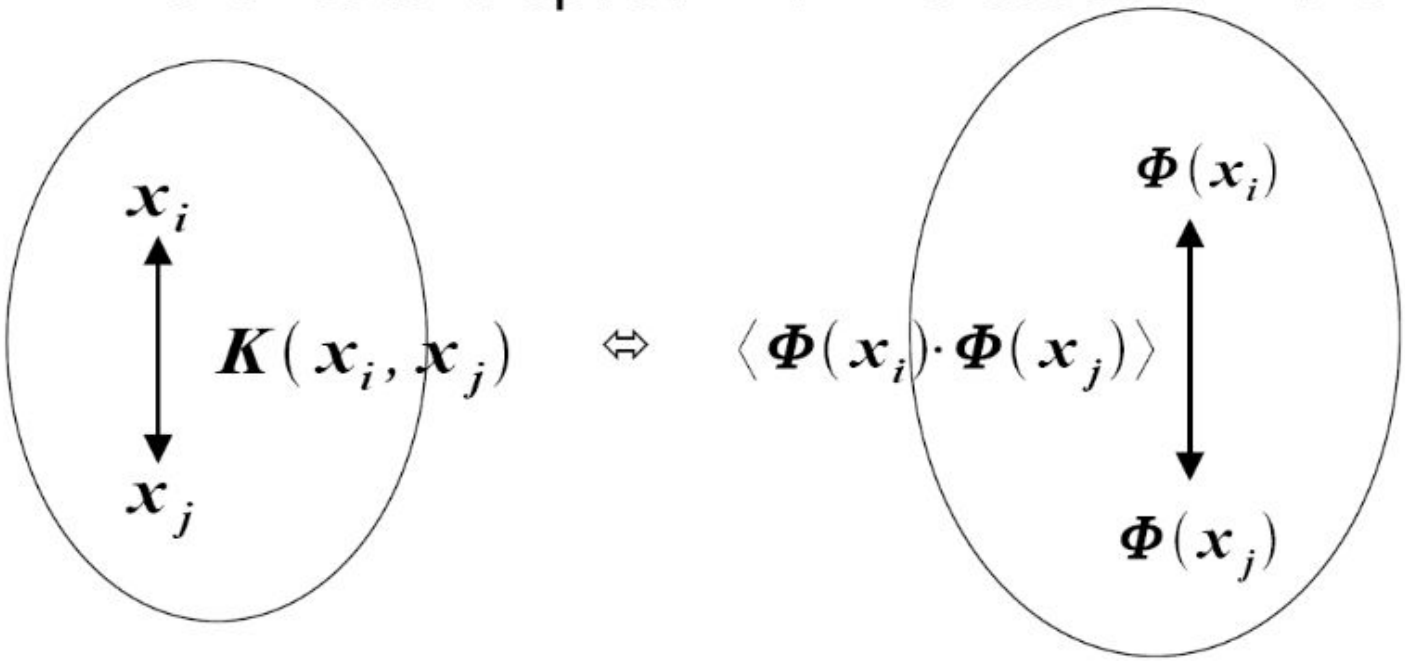
- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



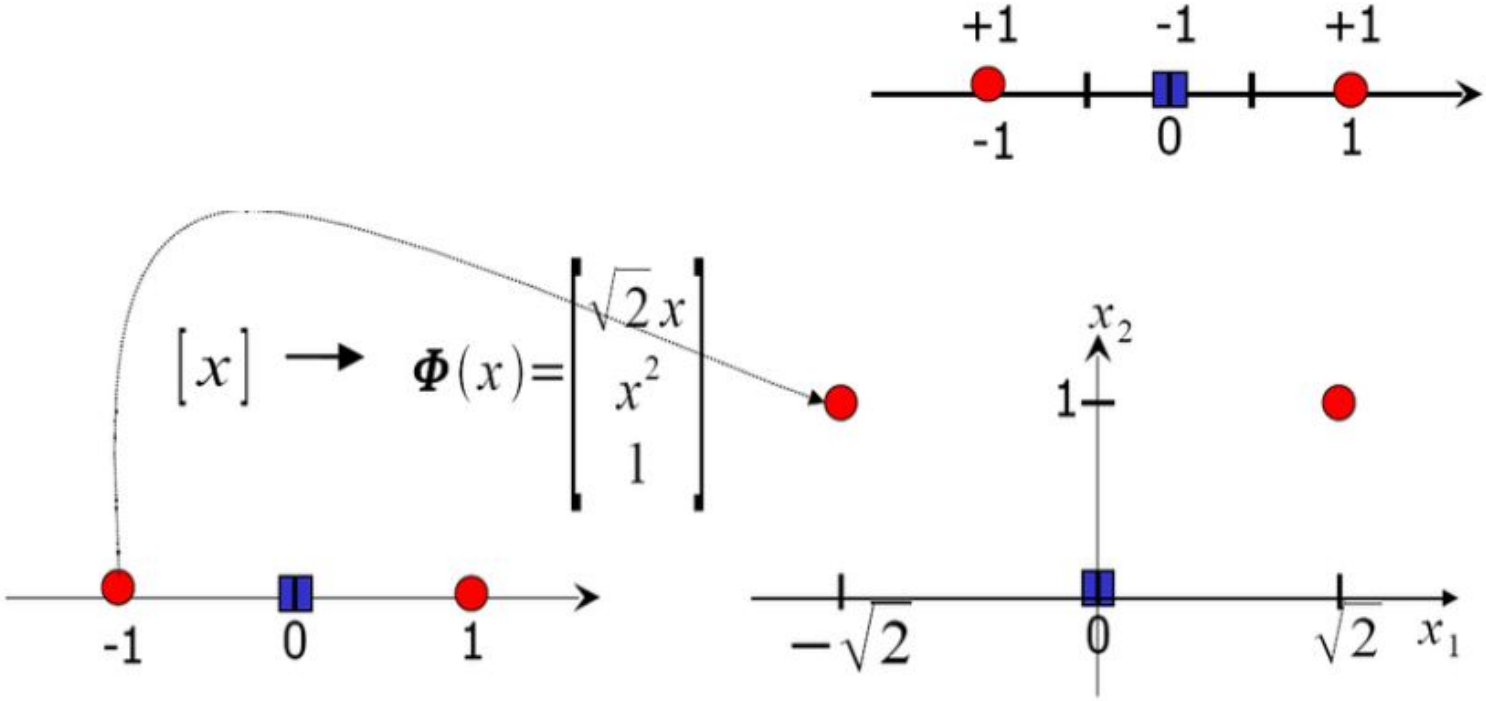
Kernel Trick

- Kernel function
  - in the original space
- Inner product
  - In the feature space with increased dimension

"similarity measure"
between 2 data samples



Kernel Trick



$$K(x_i, x_j) = (x_i x_j + 1)^2$$

$$\langle \Phi(x_i) \cdot \Phi(x_j) \rangle = 2x_i x_j + x_i^2 x_j^2 + 1 = (x_i x_j + 1)^2 = K(x_i, x_j)$$

References

- Bishop, C. Neural Networks for Pattern Recognition, 2005.
- Hastie, Trevor, et al. "*The elements of statistical learning: data mining, inference and prediction*." The Mathematical Intelligencer 27.2, 2005
- Murphy, Kevin P. *Machine learning: a probabilistic perspective*. MIT press, 2012.
- Bishop, C. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc, 2006

Appendix

Soft Margin SVM

Again, the solution for the optimization problem involves constructing the Lagrangian function

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\varepsilon_i - \sum_{i=1}^{N} a_i[t_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) - 1 + \varepsilon_i] - \sum_{i=1}^{N}\mu_i\varepsilon_i$$

where a_i and μ_i are Lagrange multipliers.

adopting the same procedure of the SVM hard margin, we can obtain the dual formulation

$$\begin{aligned} \max_{\mathbf{a}} \quad & \tilde{L}(\mathbf{a}) = \sum_{i=1}^{N} a_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j t_i t_j \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) \\ \text{s.t.} \quad & 0 \le a_i \le C \quad \forall\, i = 1, \ldots, N \\ & \sum_{j=1}^{N} a_i t_i = 0 \end{aligned}$$

which is identical to the separable case, except that Lagrange multipliers (a_i) are limited by C. This is know as *box constraints.*

Soft Margin SVM

Again, training patterns for which $a_i \geq 0$ constitute the support vectors and hence must satisfy

$$t_i[\mathbf{w}^T\phi(\mathbf{x}_i) + b] = 1 - \varepsilon_i$$

- support vectors whose $a_i < C$ are located on the margin.

- support vectors whose $a_i = C$ can lie inside the margin and can either be correctly classified if $\varepsilon_i \leq 1$ or misclassified if $\varepsilon_i > 1$.

Geometry of Linear Decision Boundary

1. decision boundary: $y(\mathbf{x}) = 0$

- corresponds to a (n-1)-dimensional hyperplane in $\mathbb{R}^n$

- Ex:. if $n = 2$ then $y(\mathbf{x}) = 0$ is a straight line.

2. the vector $\mathbf{w}$ is normal to any vector lying in the hyperplane $y(\mathbf{x}) = 0$, so that $\mathbf{w}$ determines the orientation of the decision boundary.

- let $\mathbf{x}^a$ and $\mathbf{x}^b$ be two points on the hyperplane, then

$$y(\mathbf{x}^a) = y(\mathbf{x}^b) = 0$$

$$\mathbf{w}^T(\mathbf{x}^a - \mathbf{x}^b) = 0$$

Geometry of Linear Decision Boundary

3. The normal distance of a vector $\mathbf{x}$ to the hyperplane is given by

$$r = \frac{y(\mathbf{x})}{||\mathbf{w}||}$$

- such that $\mathbf{x}$ can be described as $\mathbf{x} = \mathbf{x}^p + r\frac{\mathbf{w}}{||\mathbf{w}||}$

where $\mathbf{x}^p$ is the normal projection of $\mathbf{x}$ over the hiperplane.