IN THE NAME OF ALLAH

SUPPORT VECTOR MACHINE (SVM)

Hossein Khosravi

Department of Electrical and Robotic Engineering, Shahrood University of Technology

Outline

- *A short preface*
- *A brief history of SVM*
- *Linear SVM*
  - *Linearly separable*
    - *A simple example*
  - *Linearly non-separable*
- *Nonlinear SVM (Kernel Function & Kernel trick)*
  - *A simple example*
- *SVM Applications*

Preface

- *"I was shocked to see a student's report on performance comparisons between support vector machines (SVMs) and fuzzy classifiers that we had developed with our best endeavors. Classification performance of our fuzzy classifiers was comparable, but in most cases inferior, to that of support vector machines."*

  "Professor Shigeo Abe", *"Kobe University, Kobe, Japan",* ***"Support Vector Machines for Pattern Classification",*** *"Springer-Verlag London Limited 2005".*

Introduction

- *Two **general approaches** for classification:*
  - ***parametric** approach:*

  *a **priori** knowledge of data distributions is assumed.*
  - ***nonparametric** approach:*

  no a priori knowledge is assumed.
- *Neural networks, fuzzy systems, and **support vector machines** are nonparametric classifiers.*
- *SVM is one of the supervised learning algorithms.*

Linear SVM:

Linearly Separable Case

Two-Class Classification Problem

- *Consider a two-class, linearly separable classification problem.*
- *Let $\{x_1, \ldots, x_n\}$ be our training data set.*
- *Define an **indicator** vector y:*

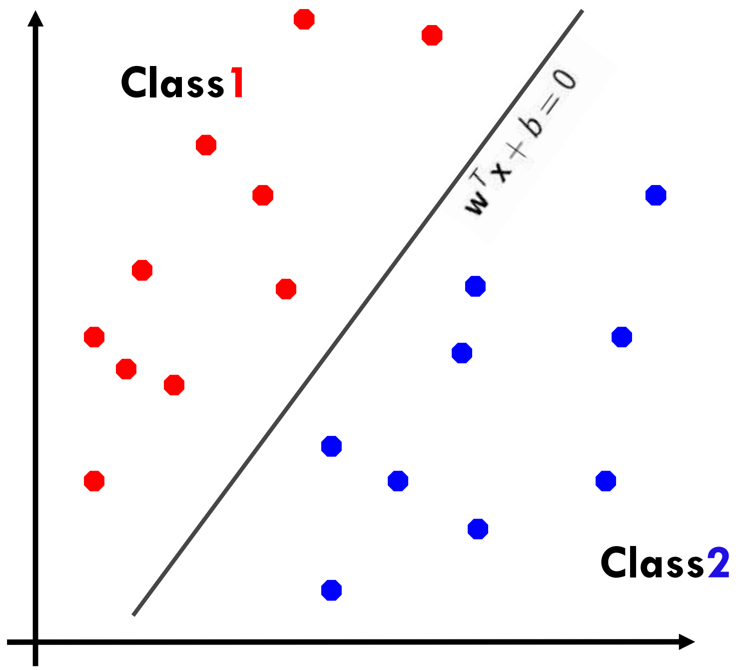
$$y_i = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ in class } 1 \\ -1 & \text{if } \mathbf{x}_i \text{ in class } 2 \end{cases}$$

- *There is a **hyperplane** which separates all data:*

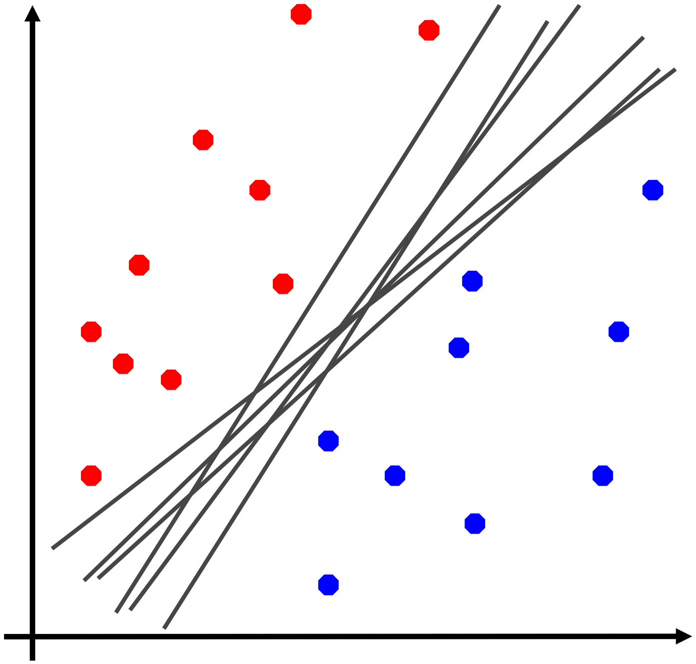
$$\mathbf{w}^T\mathbf{x} + b = 0$$

- ***Decision function**:*

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w}^T\mathbf{x} + b) \qquad \begin{matrix} (\mathbf{w}^T\mathbf{x}_i) + b > 0 & \text{if } y_i = 1 \\ (\mathbf{w}^T\mathbf{x}_i) + b < 0 & \text{if } y_i = -1 \end{matrix}$$



- ***Many possible choices*** *of w and b*
- *but there is* ***only one*** *that maximizes the margin. (the* ***optimal separating hyper plane****)*

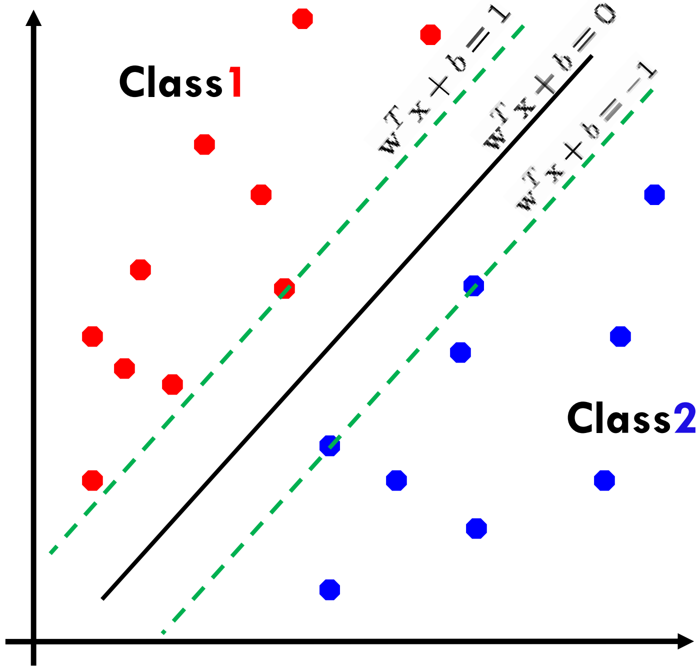


- *Because the training data are linearly separable, no training data satisfy*

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

- *Thus, to control separability, we consider the following inequalities:*

$$\mathbf{w}^T\mathbf{x}_i + b \begin{cases} \geq 1 & \text{for} \quad y_i = 1, \\ \leq -1 & \text{for} \quad y_i = -1 \end{cases}$$



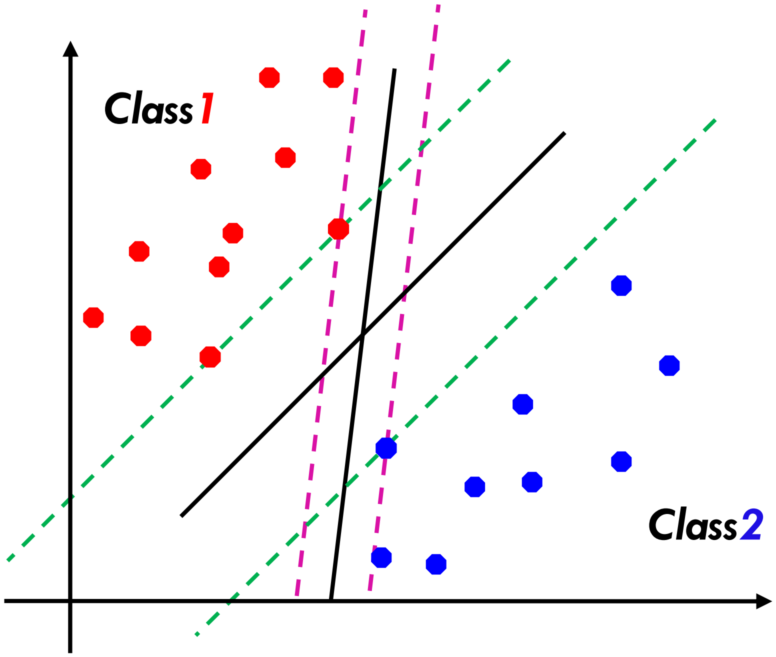
- *Here, 1 and −1 can be: constant a and −a.*
- *Equation is equivalent to:*

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad i = 1, \ldots, n$$

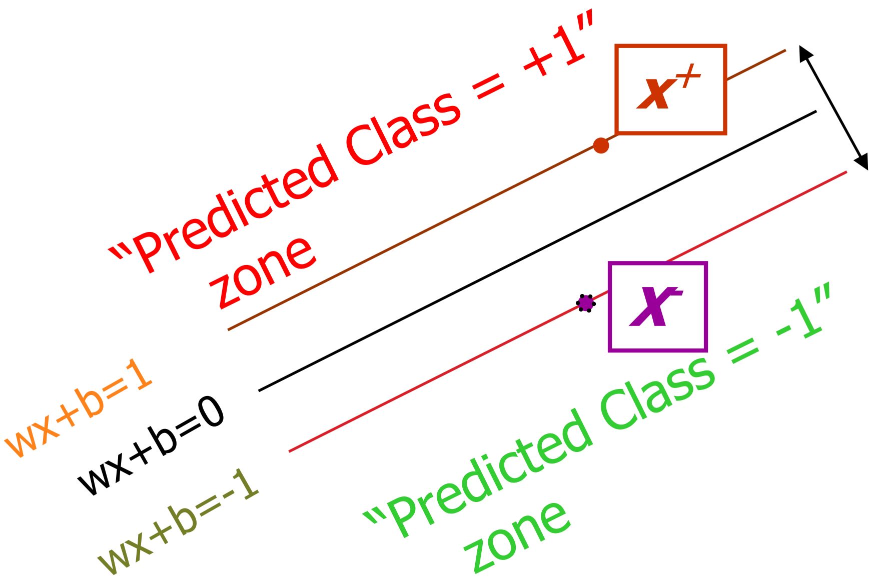
- *The generalization region for the decision function:*

$$D(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = c \quad \text{for} \quad -1 < c < 1$$

- *Thus there are an infinite number of decision functions, which are separating hyperplanes.*
- *the hyperplane with the maximum margin is called the* ***optimal separating hyperplane*****.**



Linear SVM Mathematically



M=Margin Width

What we know:

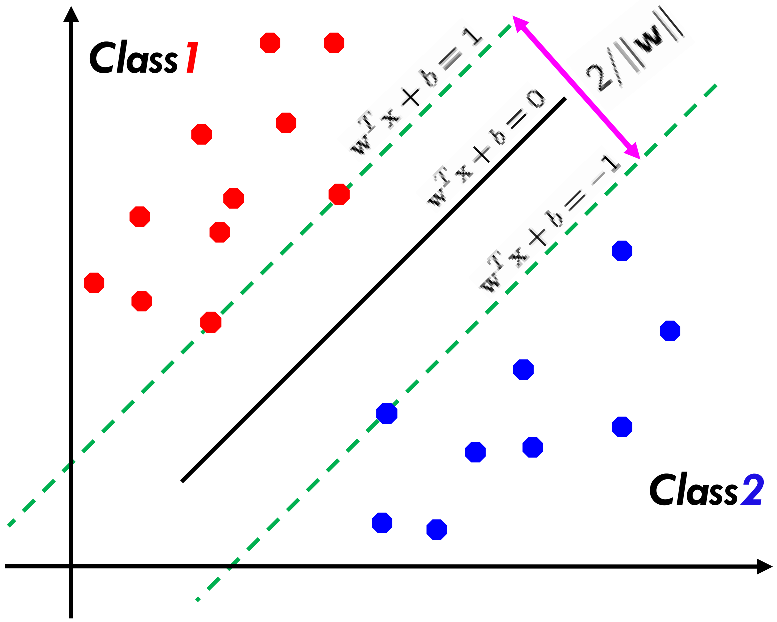
- $\mathbf{w} \cdot \mathbf{x}^+ + b = +1$
- $\mathbf{w} \cdot \mathbf{x}^- + b = -1$
- $\mathbf{w} \cdot (\mathbf{x}^+ - \mathbf{x}^-) = 2$

$$M = \frac{(x^+ - x^-) \cdot w}{|w|} = \frac{2}{|w|}$$

***See** Distance between two straight lines.pdf*

- *Distance between* $\mathbf{w}^T\mathbf{x} + b = 1$ *and* $\mathbf{w}^T\mathbf{x} + b = -1$ *is* $2/\|\mathbf{w}\|$

- *Maximizing the margin = minimizing* $\|\mathbf{w}\|$
- *Therefore, the optimal separating hyperplane can be obtained by the following* ***quadratic problem:***



$$\min_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{subject to} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad i = 1, \ldots, n$$

Because of the quadratic problem with the inequality constraints, the value of the objective function is ***unique*** *(there is* ***one global extremum*** *point).This is* ***one of the advantages*** *over multilayer* ***neural networks*** *with* ***numerous local minima.***

?

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{subject to} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad i = 1, \ldots, n$$

How can we solve this problem?

Lagrangian Function

- *Suppose we want to:*
  - *minimize f(**x**)*
  - *subject to constrained g(**x**) ≥0*

$$\min_{\mathbf{w}, b} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{subject to} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad i = 1, \ldots, n$$

- *We define the unconstrained **Primal Lagrangian function:***

$$L(\mathrm{x}, \alpha) = f(\mathrm{x}) - \alpha g(\mathrm{x})$$

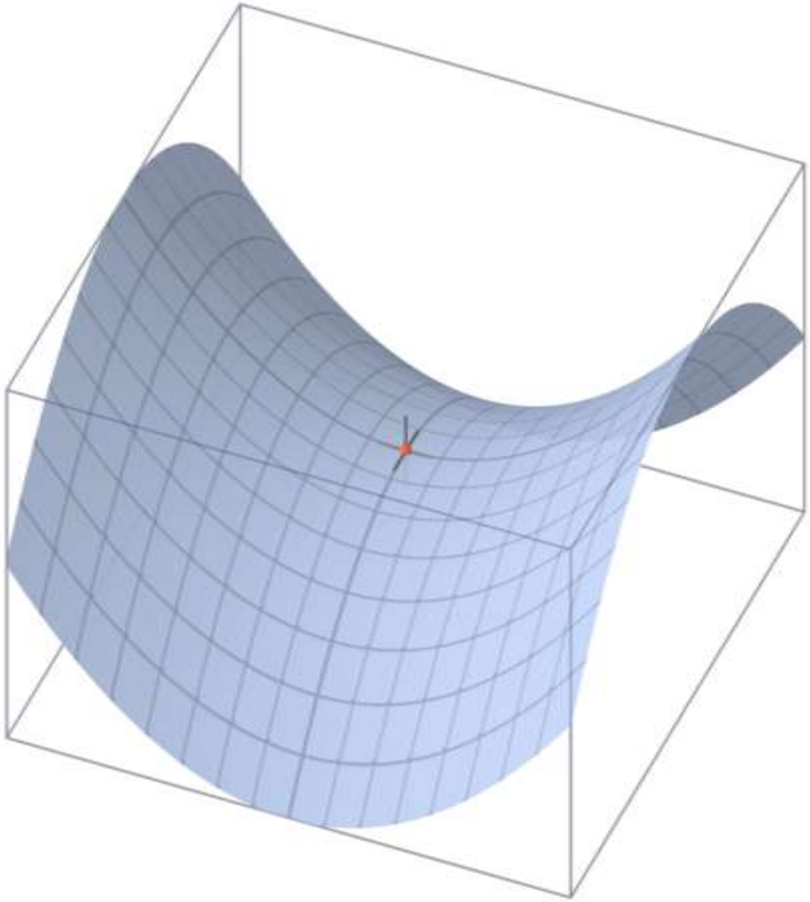
$$L_p(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{n} \alpha_i\{y_i[\mathbf{w}^T\mathbf{x}_i + b] - 1\}$$

  - *Where **α≥0** is the Lagrange multiplier.*

- *Then we find the **stationary (saddle)** point of **L** with respect to x and α (α≥0).*

Saddle Point

- A saddle point on the graph of $z=x^2-y^2$ (in red)



- ***Saddle Points***:

  Lagrangian L has to be minimized with respect to w and b, and maximized with respect to αi (αi ≥ 0):

$$\max_{\alpha}\left(\min_{\mathrm{w,b}} L(\mathbf{w},\mathrm{b},\boldsymbol{\alpha})\right)$$

- *It satisfies the following **Karush-Kuhn-Tucker (KKT)** conditions:*

$$\frac{\partial L_p}{\partial \mathbf{w}_o} = 0 \quad \mathbf{w}_o = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial L_p}{\partial b} = 0 \quad \sum_{i=1}^{n} \alpha_i y_i = 0.$$

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

Substituting these equations into a primal Lagrangian ***L(w, b, α)*** *, we change to the* ***dual Lagrangian L(α)***:

$$\max\ L_d(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t. } \alpha_i \geq 0, \quad i = 1, \ldots, n \quad \text{and}$$

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

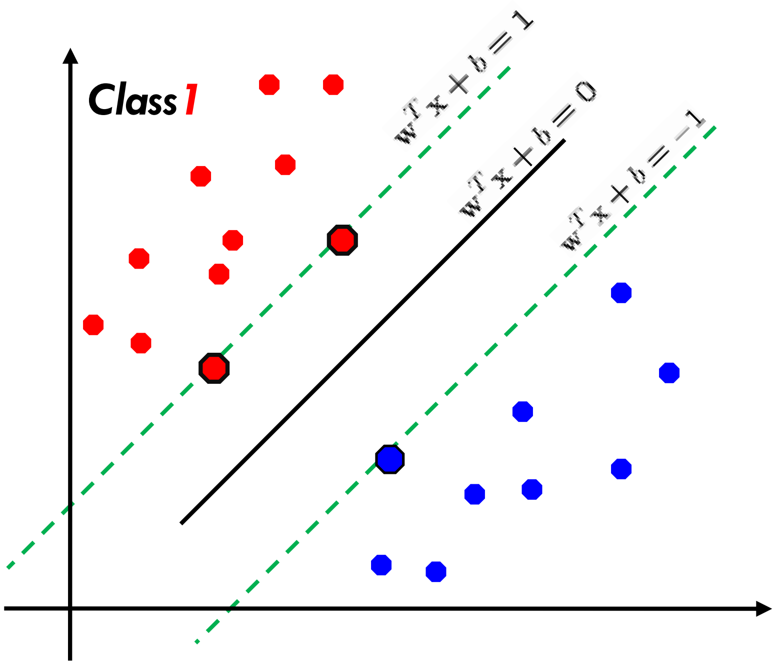
- We can find α_i *by training*
- *Data that are associated with* ***positive*** *α_i are* ***Support Vectors*** *for Classes 1 and 2.*
- *As before we had:*

$$\mathbf{w}_o = \sum_{i=1}^{n} \alpha_{oi} y_i \mathbf{x}_i$$

  - *αi≠0 only if Xi is a support vector.*

$$b = y_i - \mathbf{w}^T \mathbf{x}_i$$

  - *Where Xi is a support vector.*



- *It is better to take the average, among the support vectors :*

$$b_o = \frac{1}{N_{SV}} \sum_{s=1}^{N_{SV}} (y_s - \mathbf{x}_s^T \mathbf{w}_o) \quad s = 1, \ldots, N_{sv}$$

Formulation Summary

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad i = 1, \ldots, n$

$$\max \ L_d(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t. } \alpha_i \geq 0, \quad i = 1, \ldots, n \quad \text{and}$$

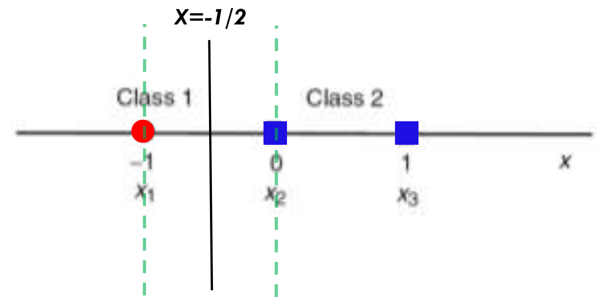
$$\sum_{i=1}^{n} \alpha_i y_i = 0.$$

$$\mathbf{w}_o = \sum_{i=1}^{n} \alpha_{oi} y_i \mathbf{x}_i$$

$$b_o = \frac{1}{N_{SV}} \sum_{s=1}^{N_{SV}} (y_s - \mathbf{x}_s^T \mathbf{w}_o) \quad s = 1, \ldots, N_{sv}$$

$$d(\mathbf{x}) = \mathbf{w_0}^T \mathbf{x} + b_0$$

Example



- *Example: Consider a very simple linearly separable one-dimensional case:*
  - ***X1=-1 , X2=0 , X3=1***

$$\text{max: } L_d(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 - 0.5(\alpha_1 + \alpha_3)^2$$

$$s.t \begin{cases} \alpha_1 - \alpha_2 - \alpha_3 = 0 \\ \alpha_i \geq 0 \quad for \quad i=1,2,3 \end{cases}$$

$$\max\ L_d(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t. } \alpha_i \geq 0, \quad i = 1, \ldots, n \quad \text{and}$$

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

$$\alpha_2 = \alpha_1 - \alpha_3 \quad \Rightarrow \quad \text{max: } L_d(\alpha) = 2\alpha_1 - 0.5(\alpha_1 + \alpha_3)^2$$

- Because *α1≥0 and α3≥0, L(α)is maximized when α3=0* ***(X3 is not a support vector)***:

$$\text{max: } L_d(\alpha) = 2\alpha_1 - 0.5\alpha_1^2 \quad \Rightarrow \quad \alpha_1 = 2 \ , \ \alpha_3 = 0 \ , \ \alpha_2 = 2$$

- Therefore ***X1 , X2 are Support Vectors.***

$$\mathbf{w} = \sum_{i=1}^{3} \alpha_i y_i \mathbf{x}_i = -2$$

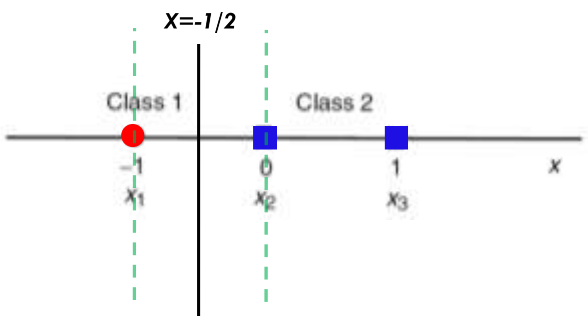
$$b = \frac{1}{2} \sum_{i=1}^{2} (y_i - \mathbf{w}^t \mathbf{x}_i) = -1$$

$$\mathbf{w}_o = \sum_{i=1}^{n} \alpha_{oi} y_i \mathbf{x}_i$$

$$b_o = \frac{1}{N_{SV}} \sum_{s=1}^{N_{SV}} (y_s - \mathbf{x}_s^T \mathbf{w}_o) \quad s = 1, \ldots, N_{sv}$$

- *Decision boundary:*

$$\mathbf{w}^T \mathbf{x} + b = 0 \;\Rightarrow\; -2x - 1 = 0 \;\Rightarrow\; x = -\frac{1}{2}$$



Lagrangian Matrix Form

- *Dual Lagrangian can be rewritten into* ***matrix format*** *as:*

$$\max \; L_d(\alpha) = -0.5\alpha^t YRY\alpha + f^t\alpha$$
$$s.t \; \begin{cases} \alpha_i \geq 0 & i=1,\dots,n \\ y^t\alpha = 0 \end{cases}$$

$$\max \; L_d(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j$$
$$\text{s.t. } \alpha_i \geq 0, \quad i = 1, \dots, n \quad \text{and}$$
$$\sum_{i=1}^{n} \alpha_i y_i = 0.$$

$$Y = \begin{pmatrix} y_1 & & 0 \\ & \ddots & \\ 0 & & y_n \end{pmatrix}_{n\times n}, \quad R_{n\times n} \Rightarrow R_{ij} = \mathrm{x}^t_{i_{(1\times d)}} \mathrm{x}_{j_{(d\times 1)}}$$

$$, \quad \alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \dots \\ \alpha_n \end{bmatrix}_{n\times 1}, \quad f = \begin{bmatrix} 1 \\ 1 \\ \dots \\ 1 \end{bmatrix}_{n\times 1}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}_{n\times 1}$$

MATLAB Function

- Function **`z=quadprog(H,f,[],[],a,K,Kl,Ku)`** in MATLAB solves the problem:

$$\min \ \frac{1}{2} z^t H z + f^t z \qquad \text{s.t} \begin{cases} az = K \\ K_l \le z \le K_u \end{cases}$$

$\Longleftrightarrow$

$$\max \ L_d(\alpha) = -0.5\alpha^t YRY\alpha + f^t\alpha \qquad \text{s.t} \begin{cases} y^t\alpha = 0 \\ \alpha_i \ge 0 \quad i=1,\dots,n \end{cases}$$

- $z = \alpha$
- $H = YRY$
- $f = -1$
- $a = yt$
- $K = 0$
- $Kl = 0$ and $Ku = C$

Linear SVM

Linearly Non-Separable Case

Linearly Non-Separable

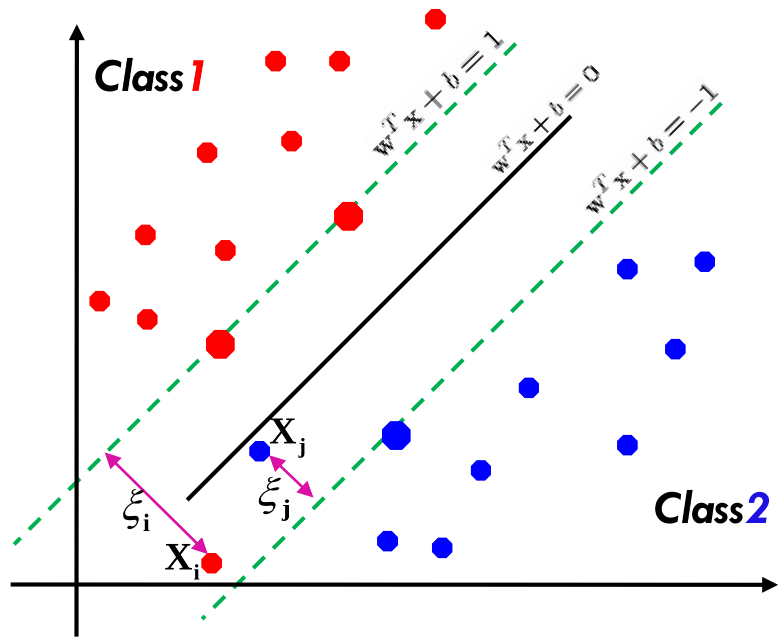
- What if the problem is **not separable** in feature space?
- We allow "**Error**" in classification. (**ξi ≥ 0**)
- So the separating hyperplane must satisfy:

$$\min \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i$$

$$\text{s.t. } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, n,$$

$$\xi_i \geq 0.$$

- maximization of the margin and minimization of the errors and is determined by user.



- *Introducing the nonnegative Lagrange multipliers αi and βi,* ***Primal Lagrangian*** *function is:*

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

- *As befor, the problem is solved by the* ***saddle point*** *of the Lagrange functional (Lagrangian):*

$$\max_{\boldsymbol{\alpha},\beta}\left(\min_{\mathbf{w},\mathrm{b}.\xi} L(\mathbf{w},\mathrm{b},\boldsymbol{\alpha},\xi,\beta)\right)$$

- *For the optimal solution, the following **KKT conditions** are satisfied:*

$$\frac{\partial L}{\partial \mathbf{w}_o} = 0, \text{ i.e., } \quad \mathbf{w}_o = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial L}{\partial b_o} = 0, \text{ i.e., } \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

$$\frac{\partial L}{\partial \xi_{io}} = 0, \text{ i.e., } \quad \alpha_i + \beta_i = C$$

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

- *Substituting these equations into a primal variables Lagrangian L(**w, b, ξ,α,β**) , we change to the **dual** variables Lagrangian L(α):*

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

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

Subject to $\quad 0 \le \alpha_i \le C$

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

- *The solution to this maximization problem is identical to the separable case except for a modification of the bounds of the Lagrange multipliers.*
- ξ_i approximates the number of misclassified samples.
- ξ_i are "slack variables" in optimization
- Note that ξ_i=0 if there is no error for $\mathbf{x}_i$
- ξ_i is an upper bound of the number of errors
- *The penalty parameter C, which is now the upper bound on αi, is determined by the user.*

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

Subject to $0 \le \alpha_i \le C$

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

MATLAB Example

svm_iris.m

SVM
Non-Linear Case

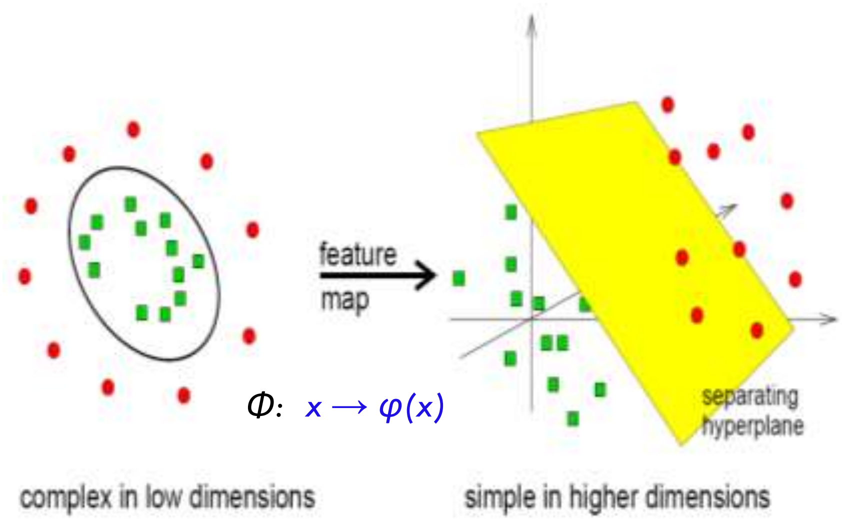
Non-Linear SVM

- *What if the training set is **not linearly separable**?*
- *The input space can be **mapped to higher-dimensional feature space (Φ),** where the training set is separable.*
- *The solution for the **linear** case:*

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

- *For the **nonlinear** classifier (in the hilbert space):*

$$\boldsymbol{w} = \sum_{i=1}^{N} y_i \alpha_i \varphi(\boldsymbol{x}_i)$$



Input Space ***Feature Space (Hilbert space)***

Non-Linear SVM

- *Dual Lagrange function:*

$$L_d(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_j$$

- *Introducing the* ***Kernel Function*** *we'll have:*

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

  - *Change all inner products to kernel functions*

Table 2.2. Popular Admissible Kernels

Kernel Functions	Type of Classifier
$K(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^T \mathbf{x}_i)$	Linear, dot product, kernel, CPD[a]
$K(\mathbf{x}, \mathbf{x}_i) = [(\mathbf{x}^T \mathbf{x}_i) + 1]^d$	Complete polynomial of degree d, PD[b]
$K(\mathbf{x}, \mathbf{x}_i) = \exp(-[\Vert\mathbf{x} - \mathbf{x}_i\Vert^2]/2\sigma^2)$	Gaussian RBF, PD[b]
$K(\mathbf{x}, \mathbf{x}_i) = \tanh[(\mathbf{x}^T \mathbf{x}_i) + b]^*$	Multilayer perceptron, CPD
$K(\mathbf{x}, \mathbf{x}_i) = 1/\sqrt{\Vert\mathbf{x} - \mathbf{x}_i\Vert^2 + \beta}$	Inverse multiquadric function, PD

[a] Conditionally positive definite [b] Positive definite
[*] only for certain values of b

- *Dual Lagrangian problem:*

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

$$\text{s.t. } 0 \le \alpha_i \le C \quad i = 1, \ldots, n. \text{ and}$$

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

Kernel Trick

φ(xi)?

$$\boldsymbol{w} = \sum_{i=1}^{N} y_i \alpha_i \varphi(\boldsymbol{x}_i)$$

Kernel Trick

Solution to linear problem:

Φ: $x \rightarrow \varphi(x)$

Solution to nonlinear problem:

$$\mathbf{w}_o = \sum_{i=1}^{n} \alpha_{oi} y_i \mathbf{x}_i$$

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i)$$

$$b = y_i - \mathbf{w}^T \mathbf{x}_i$$

$$b_o = \frac{1}{N_{SV}} \sum_{s=1}^{N_{SV}} (y_s - \mathbf{x}_s^T \mathbf{w}_o) \quad s = 1, \ldots, N_{sv}$$

$$b = y_i - \mathbf{w}^T \varphi(\mathbf{x}_i)$$

$$b = y_i - \sum_{i,j=1}^{N_{SV}} \alpha_i y_i K(x_i, x_j)$$

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

$$d(\mathbf{x}) = \mathbf{w}^T \varphi(\mathbf{x}) + b$$

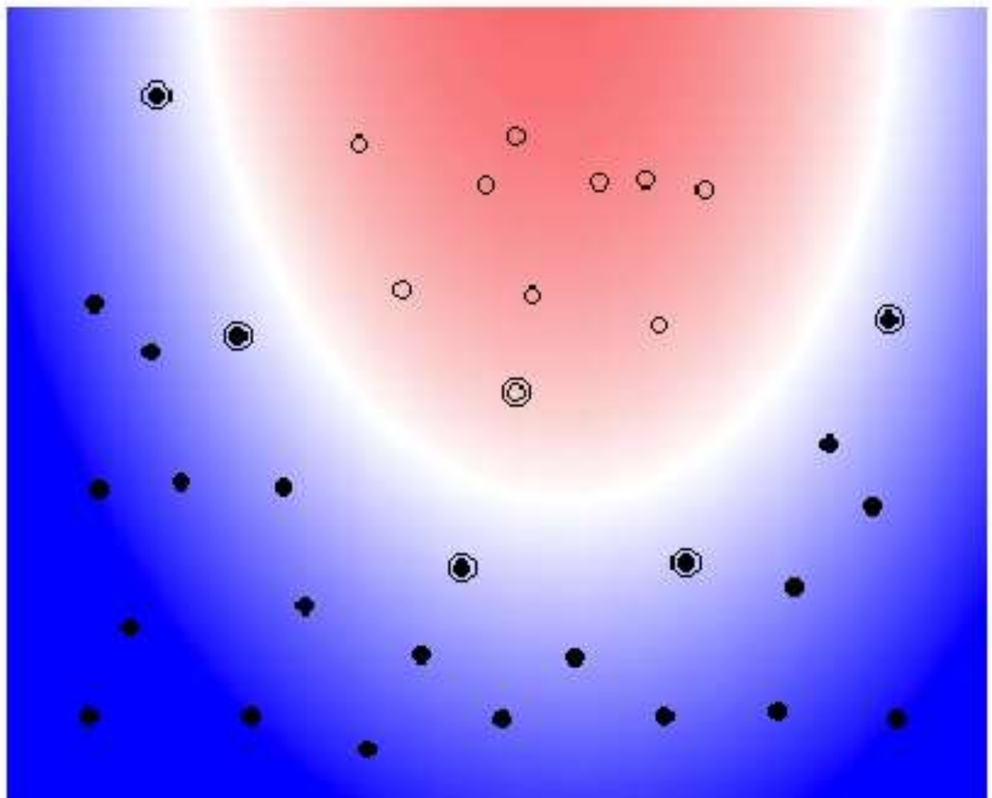
$$d(\mathbf{x}) = \sum_{i=1}^{N_{SV}} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b$$

Nonlinear Kernel (I)

Example: SVM with Polynomial of Degree 2

Kernel: $K(\vec{x}_i, \vec{x}_j) = [\vec{x}_i \cdot \vec{x}_j + 1]^2$

plot by Bell SVM applet

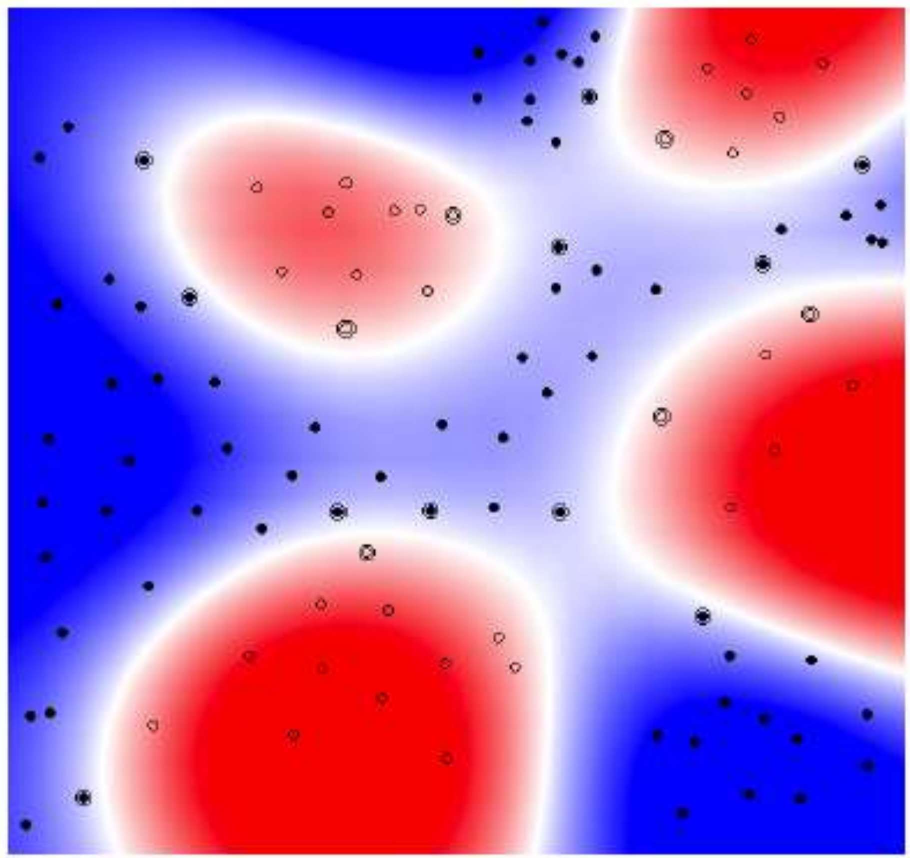


Nonlinear Kernel (II)

Example: SVM with RBF-Kernel

Kernel: $K(\vec{x}_i, \vec{x}_j) = \exp(-|\vec{x}_i - \vec{x}_j|^2 / \sigma^2)$

plot by Bell SVM applet



Matrix Format

- *Matrix format for nonlinear dual program:*

$$\max\ L_d(\alpha) = -0.5\alpha^t Y K Y \alpha + f^t \alpha$$

$$s.t \quad \begin{cases} y^t\alpha = 0 \\ 0 \le \alpha_i \le C \quad i=1,\dots,n \end{cases}$$

- *Matrix format for linear dual program:*

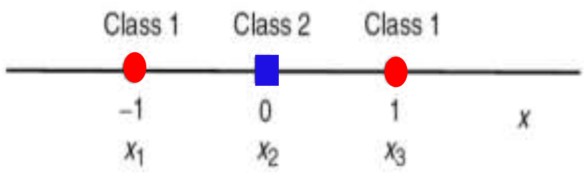
$$\max\ L_d(\alpha) = -0.5\alpha^t Y R Y \alpha + f^t \alpha$$

$$s.t \quad \begin{cases} y^t\alpha = 0 \\ 0 \le \alpha_i \le C \quad i=1,\dots,n \end{cases}$$

$$Y = \begin{pmatrix} y_1 & & 0 \\ & \ddots & \\ 0 & & y_n \end{pmatrix}_{n\times n} \ , \ K_{n\times n} \Rightarrow K_{ij} = K(x_i, x_j) \ , \ \alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \dots \\ \alpha_n \end{bmatrix}_{n\times 1} \ , \ f = \begin{bmatrix} 1 \\ 1 \\ \dots \\ 1 \end{bmatrix}_{n\times 1} \ , \ y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}_{n\times 1}$$

Example

- Example: Consider a very simple nonlinearly separable one-dimensional case with:
  - $K(x_i, x_j) = (x_i^T x_j + 1)^2$
  - $C=2$



$$\max\ L_d(\alpha) = -0.5\alpha^t YKY\alpha + f^t\alpha$$

$$= -(2\alpha_1^2 + \frac{1}{2}\alpha_2^2 + 2\alpha_3^2 - \alpha_2(\alpha_1 + \alpha_3)) + \alpha_1 + \alpha_2 + \alpha_3$$

$$s.t \quad \begin{cases} 0 \le \alpha_i \le 2 \quad for \quad i=1,2,3 \\ \alpha_1 - \alpha_2 + \alpha_3 = 0 \end{cases}$$

$$\max\ L_d(\alpha) = -0.5\alpha^t YKY\alpha + f^t\alpha$$

$$s.t \quad \begin{cases} y^t\alpha = 0 \\ 0 \le \alpha_i \le C \qquad i=1,\ldots,n \end{cases}$$

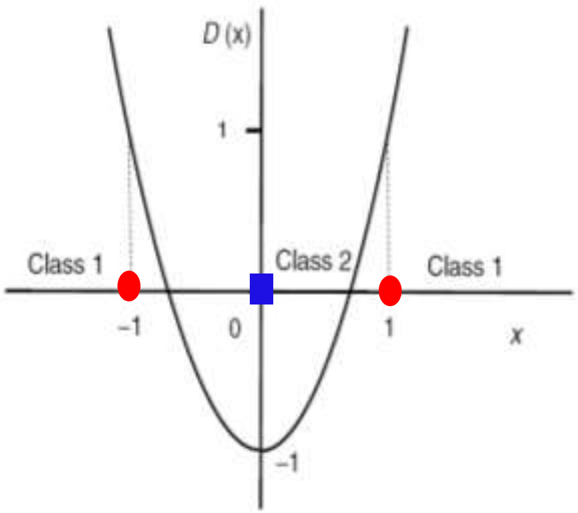
$$\Rightarrow \quad \alpha_1 = 1, \quad \alpha_2 = 2, \quad \alpha_3 = 1 \quad \Rightarrow \text{all are Support Vectors}$$

$$b = \text{mean}\left\{ y_i - \sum_{i,j=1}^{3} \alpha_i y_i K(x_i, x_j) \right\} = -1$$

$$d(x) = \mathbf{w}^T \varphi(x) + b = \sum_{i=1}^{3} y_i \alpha_i K(x, x_i) + b$$

$$= (x-1)^2 + (x+1)^2 - 3$$

$$= 2x^2 - 1$$



Summary: Steps for Classification

- Prepare the pattern matrix.
- Select the kernel function to use.
- Select the parameter of the kernel function and the value of C.
  - You can use the values suggested by the SVM software, or you can set apart a validation set to determine the values of the parameter.
- Execute the training algorithm and obtain the α_i .
- Unseen data can be classified using the α_i and the support vectors.

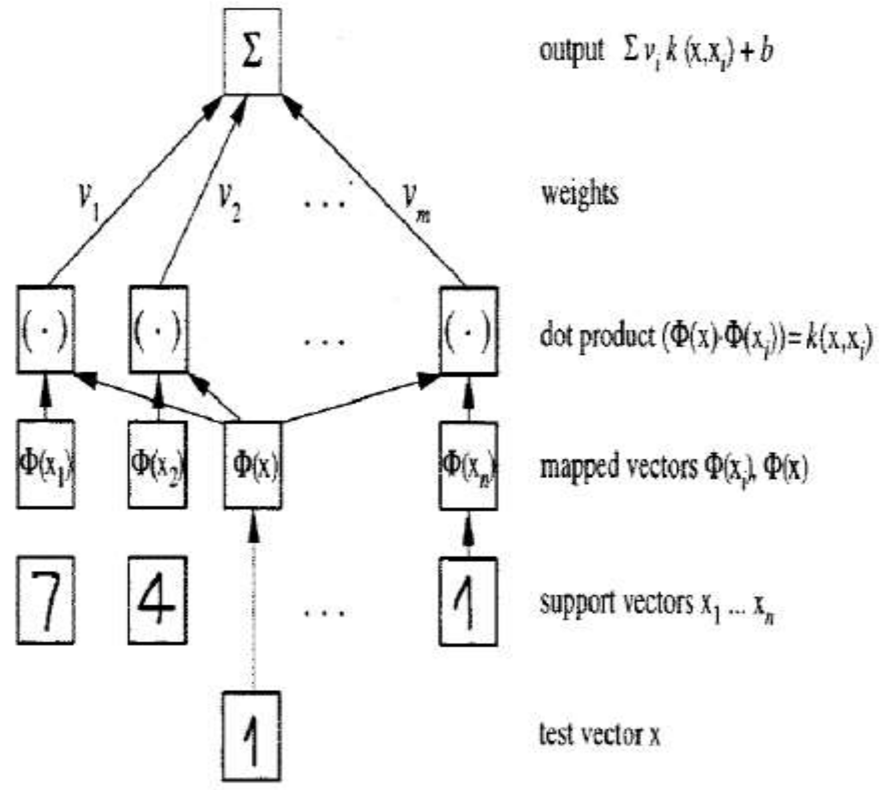
Strengths and Weaknesses of SVM

- Strengths
  - Training is relatively easy
    - No local optimal, unlike in neural networks
  - It scales relatively well to high dimensional data
  - Tradeoff between classifier complexity and error can be controlled explicitly
- Weaknesses
  - Need to choose a "good" kernel function.

SVM Applications

Handwriting Recognition

- *60,000 training examples, and10,000 test examples, 28x28.*
- *Linear SVM has around 8.5% test error.*
- Polynomial SVM has around 1% test error.



بازشناسی گفتار از موزیک در پخش رادیویی به روش ماشین بردار پشتیبان

محمد مهدی همایون پور[۱]، سید حسین خاتون آبادی[۲]
آزمایشگاه سیستمهای هوشمند صوتی - گفتاری
دانشگاه صنعتی امیرکبیر(پلی تکنیک تهران) - دانشکده مهندسی کامپیوتر و فناوری اطلاعات
homayoun@ce.aut.ac.ir

- برای **یادگیری** ماشین بردار پشتیبان نمونه های آموزشی بصورت **۲۰ یا ۶۰** فایل صوتی ۱ ثانیه ای و ۲ ثانیه ای برای هرکدام از نمونه های گفتار و موزیک در نظر گرفته شده است.
- همچنین نمونه های **آزمایشی** بصورت **۱۵۰** فایل صوتی ۱ ثانیه ای و ۲ثانیه ای انتخاب شده اند . ویژگیهای استخراج شده به **شش** صورت LPCC ، MFCC، LPCCهمراه با مشتق اول آن، MFCCهمراه با مشتق اول آن ، مشتق اول LPCC و مشتق اول MFCC می باشند. آزمایشات با ضریبهای ریسک متفاوتی انجام شد که بهترین جوابها با انتخاب **C = ۲۰۰** بدست آمد.

جدول ۱- نتایج آزمایشات برحسب درصد صحت بازشناسی گفتار از موزیک
با استفاده از تابع هسته گوسی با ضریب g = .05 و ضریب ریسک C = 200

RBF (g=.05) C=200	LPCC	MFCC	LPCC & Delta LPCC	MFCC & Delta MFCC	Delta LPCC	Delta MFCC
Train: 20s *(20 files*1s)* Test: *1s*	۸۲,۰	۸۹,۳	۸۴,۰	۹۰,۷	۸۶,۷	۹۴,۷
Train: 60s *(60 files*1s)* Test: *1s*	۸۸,۷	۹۱,۳	۹۰,۷	۹۲,۰	۸۸,۷	۹۷,۳
Train: 40s *(20 files*2s)* Test: *1s*	۸۲,۰	۸۷,۳	۸۲,۷	۸۹,۳	۹۱,۳	۹۳,۳
Train: 120s *(60 files*2s)* Test: *1s*	۸۹,۳	۹۱,۳	۹۱,۳	۹۱,۳	۹۱,۳	۹۵,۳
Train: 40s *(20 files*2s)* Test: *2s*	۸۲,۷	۹۰,۰	۸۳,۳	۹۰,۰	۸۸,۷	۹۶,۷
Train: 120s *(60 files*2s)* Test: *2s*	۹۰,۷	۹۳,۳	۹۲,۰	۹۳,۳	۸۹,۳	۹۸,۷

Other applications

- *Face Detection*
- *Face Recognition*
- *Text region Detection*
- *3D object recognition*
- *Antenna array processing*
- *….*

References

1. N. Cristianini and J. Shawe-Taylor, ***An Introduction To Support Vector Machines (and other kernel-based learning methods)***, Cambridge University Press, UK, 2000.
2. Professor Dr Shigeo Abe, ***Support Vector Machines for Pattern Classification*** ,Kobe University, Kobe, Japan
3. Manel Martı´nez-Ramo´ n , ***Support Vector Machines for Antenna Array Processing and Electromagnetics***, Universidad Carlos III de Madrid, Spain, Morgan & Claypool ,USA,2006.
4. Te-Ming Huang and Vojislav Kecman , ***Kernel Based Algorithms for Mining Huge Data Sets***, Faculty of Engineering The University of Auckland, Springer-Verlag Berlin Heidelberg 2006.
5. Professor Lipo Wang, ***Support Vector Machines: Theory and Applications***, Nanyang Technological University, School of Electrial & Electronic Engineering, Springer Berlin Heidelberg New York, 2005.
6. Ingo Steinwart, ***Support Vector Machines***, Los Alamos National Laboratory, information Sciences Group (CCS-3), Springer Science+Business Media, LLC, 2008 .

7. Martin Law ,***A Simple Introduction to Support Vector Machines***, Lecture for CSE 802, Department of Computer Science and Engineering, Michigan State University, 2009.
8. Christorher J.C. Burges, ***A tutorial on support vactor machine for pattern regocnition***, bell laboratories*Data Mining and Knowledge Discovery, Kluwer Academic Publishers,*1998.
9. Andrew Ng , ***Support Vector Machines***, CS229 Lecture notes.
10. Dr Martin Brown, ***Lectures 11&12: Generalization, Regularization and Support Vector Machines***, http://www.eee.manchester.ac.uk/intranet/pg/coursematerial/
11. Dr. Toby P. Breckon**, *Support Vector Machines*,** School of Engineering Cranfield University, UK, 2007.
12. Florian Markowetz, ***Classication by Nearest Shrunken Centroids and Support Vector Machines*** , Max Planck Institute for Molecular Genetics, Dept. Computational Molecular Biology , Computational Diagnostics Group, Berlin , 2004.

13. Chiwoo Park , ***Support Vector Machine***, Advanced Metrology Lab., Texas A&M University, 2008.
14. j.p.lewis , ***A Short SVM (Support Vector Machine) Tutorial***, CGIT Lab / IMSC U. Southern California, 2004.
15. Johan Suykens***, Engineering Kernel Machines***, K.U. Leuven, ESAT-SCD/SISTA, Belgium, 2006.
16. Rong Qu , ***Interest Rate Forecasting: Support Vector Machines vs. Neural Networks ,*** School of Computer Science, The University of Nottingham , rxq@cs.nott.ac.uk.
17. Geoff Gordon, ***Support Vector Machines and Kernel Methods***, ggordon@cs.cmu.edu, 2004.

18. Chih-Jen Lin, ***Support Vector Machines***, Department of Computer Science, National Taiwan University, Talk at Machine Learning Summer School 2006.
19. Andrew W. Moore**, *Support Vector Machines* ,** Associate Professor School of Computer Science, Carnegie Mellon University, 2001.
20. Pai-Hsuen Chen, Chih-Jen Lin, and Bernhard Sch¨olkopf*,* ***A Tutorial on Support Vector Machines ,*** Department of Computer Science and Information Engineering National Taiwan University, Max Planck Institute for Biological Cybernetics, T¨ubingen, Germany.
21. Jason Weston, ***Support Vector Machine (and Statistical Learning Theory)*** Tutorial, NEC Labs America 4 IndependenceWay, Princeton, USA. jasonw@nec-labs.com
22. Alex J. Smola*†and Bernhard Sch¨olkopf‡,* ***A Tutorial on Support Vector Regression****∗,* September 30, 2003.
23. Steve R. Gunn, ***Support Vector Machines for Classification and Regression***, Faculty of Engineering, Science and Mathematics, School of Electronics and Computer Science, UNIVERSITY OF SOUTHAMPTON ,1998.