

# Slide09

## Haykin Chapter 6: Support-Vector Machines

CPSC 636-600

Instructor: Yoonsuck Choe

Spring 2012

Note: Part of this lecture drew material from Ricardo Gutierrez-Osuna's Pattern Analysis lectures.

1

### Optimal Hyperplane

For linearly separable patterns  $\{(\mathbf{x}_i, d_i)\}_{i=1}^N$  (with  $d_i \in \{+1, -1\}$ ):

- The separating hyperplane is  $\mathbf{w}^T \mathbf{x} + b = 0$ :

$$\mathbf{w}^T \mathbf{x} + b \geq 0 \text{ for } d_i = +1$$

$$\mathbf{w}^T \mathbf{x} + b < 0 \text{ for } d_i = -1$$

- Let  $\mathbf{w}_o$  be the optimal hyperplane and  $b_o$  the optimal bias.

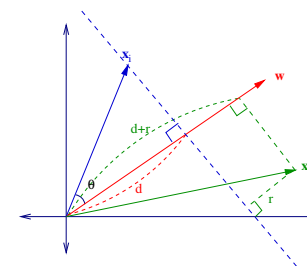
3

### Introduction

- Support vector machine is a *linear machine* with some very nice properties.
- The basic idea of SVM is to construct a separating hyperplane where the *margin of separation* between positive and negative examples are maximized.
- Principled derivation: structural risk minimization
  - error rate is bounded by: (1) training error-rate and (2) VC-dimension of the model.
  - SVM makes (1) become zero and minimizes (2).

2

### Distance to the Optimal Hyperplane

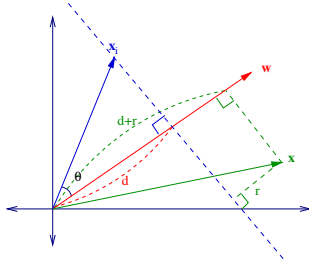


- From  $\mathbf{w}_o^T \mathbf{x} = -b_o$ , the distance from the origin to the hyperplane is calculated as

$$d = \|\mathbf{x}_i\| \cos(\mathbf{x}_i, \mathbf{w}_o) = \frac{-b_o}{\|\mathbf{w}_o\|}$$

4

## Distance to the Optimal Hyperplane (cont'd)



- The distance from an arbitrary point to the hyperplane can be calculated as:
  - When the point is in the positive area:

$$r = \|x\| \cos(\mathbf{x}, \mathbf{w}_o) - d = \frac{\mathbf{x}^T \mathbf{w}_o}{\|\mathbf{w}_o\|} + \frac{b_o}{\|\mathbf{w}_o\|} = \frac{\mathbf{x}^T \mathbf{w}_o + b_o}{\|\mathbf{w}_o\|}.$$

- When the point is in the negative area:

$$r = d - \|x\| \cos(\mathbf{x}, \mathbf{w}_o) = -\frac{\mathbf{x}^T \mathbf{w}_o}{\|\mathbf{w}_o\|} - \frac{b_o}{\|\mathbf{w}_o\|} = -\frac{\mathbf{x}^T \mathbf{w}_o + b_o}{\|\mathbf{w}_o\|}.$$

5

## Optimal Hyperplane and Support Vectors (cont'd)

- The optimal hyperplane is supposed to maximize the margin of separation  $\rho$ .
- With that requirement, we can write the conditions that  $\mathbf{w}_o$  and  $b_o$  must meet:

$$\mathbf{w}_o^T \mathbf{x} + b_o \geq +1 \quad \text{for } d_i = +1$$

$$\mathbf{w}_o^T \mathbf{x} + b_o \leq -1 \quad \text{for } d_i = -1$$

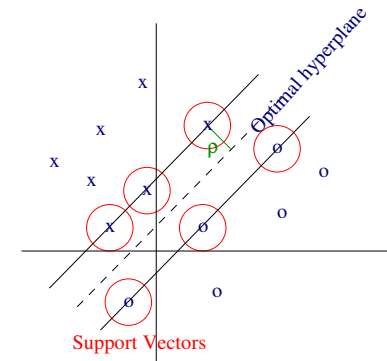
Note:  $\geq +1$  and  $\leq -1$ , and support vectors are those  $\mathbf{x}^{(s)}$  where equality holds (i.e.,  $\mathbf{w}_o^T \mathbf{x}^{(s)} + b_o = +1$  or  $-1$ ).

- Since  $r = (\mathbf{w}_o^T \mathbf{x} + b_o) / \|\mathbf{w}_o\|$ ,

$$r = \begin{cases} 1/\|\mathbf{w}_o\| & \text{if } d = +1 \\ -1/\|\mathbf{w}_o\| & \text{if } d = -1 \end{cases}$$

7

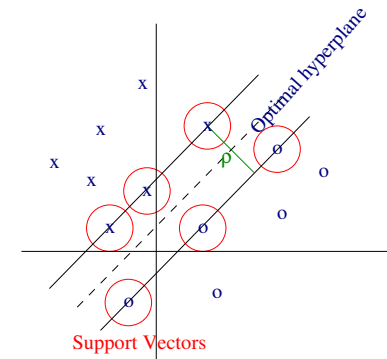
## Optimal Hyperplane and Support Vectors



- Support vectors:** input points closest to the separating hyperplane.
- Margin of separation  $\rho$ :** distance between the separating hyperplane and the closest input point.

6

## Optimal Hyperplane and Support Vectors (cont'd)



- Margin of separation *between two classes* is

$$\rho = 2r = \frac{2}{\|\mathbf{w}_o\|}.$$

- Thus, maximizing the margin of separation *between two classes* is equivalent to minimizing the Euclidean norm of the weight  $\mathbf{w}_o$ !

8

## Primal Problem: Constrained Optimization

For the training set  $\mathcal{T} = \{(\mathbf{x}_i, d_i)\}_{i=1}^N$  find  $\mathbf{w}$  and  $b$  such that

- they minimize a certain value ( $1/\rho$ ) while satisfying a constraint (all examples are correctly classified):
  - Constraint:  $d_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$  for  $i = 1, 2, \dots, N$ .
  - Cost function:  $\Phi(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w}$ .

This problem can be solved using the *method of Lagrange multipliers* (see next two slides).

9

## Lagrange Multipliers (cont'd)

Must find  $x, y, \alpha$  that minimizes

$F(x, y, \alpha) = (x - 2)^2 + (y - 2)^2 + \alpha(x^2 + y^2 - 1)$ . Set the partial derivatives to 0, and solve the system of equations.

$$\frac{\partial F}{\partial x} = 2(x - 2) + 2\alpha x = 0$$

$$\frac{\partial F}{\partial y} = 2(y - 2) + 2\alpha y = 0$$

$$\frac{\partial F}{\partial \alpha} = x^2 + y^2 - 1 = 0$$

Solve for  $x$  and  $y$  in the 1st and 2nd, and plug in those to the 3rd equation

$$x = y = \frac{2}{1 + \alpha}, \text{ so } \left(\frac{2}{1 + \alpha}\right)^2 + \left(\frac{2}{1 + \alpha}\right)^2 = 1$$

from which we get  $\alpha = 2\sqrt{2} - 1$ . Thus,  $(x, y) = (1/\sqrt{2}, 1/\sqrt{2})$ .

11

## Mathematical Aside: Lagrange Multipliers

Turn a constrained optimization problem into an unconstrained optimization problem by absorbing the constraints into the cost function, weighted by the *Lagrange multipliers*.

Example: Find closest point on the circle  $x^2 + y^2 = 1$  to the point  $(2, 3)$  (adapted from Ballard, *An Introduction to Natural Computation*, 1997, pp. 119–120).

- Minimize  $F(x, y) = (x - 2)^2 + (y - 3)^2$  subject to the constraint  $x^2 + y^2 - 1 = 0$ .
- Absorb the constraint into the cost function, after multiplying the *Lagrange multiplier*  $\alpha$ :

$$F(x, y, \alpha) = (x - 2)^2 + (y - 3)^2 + \alpha(x^2 + y^2 - 1).$$

10

## Primal Problem: Constrained Optimization (cont'd)

Putting the constrained optimization problem into the Lagrangian form, we get (utilizing the Kuhn-Tucker theorem)

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

- From  $\frac{\partial J(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = 0$ :

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

- From  $\frac{\partial J(\mathbf{w}, b, \alpha)}{\partial b} = 0$ :

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

12

## Primal Problem: Constrained Optimization (cont'd)

- Note that when the optimal solution is reached, the following condition must hold (Karush-Kuhn-Tucker complementary condition)

$$\alpha_i [d_i(\mathbf{w}^T \mathbf{x}_i + b) - 1] = 0$$

for all  $i = 1, 2, \dots, N$ .

- Thus, *non-zero*  $\alpha_i$ s can be attained only when  $[d_i(\mathbf{w}^T \mathbf{x}_i + b) - 1] = 0$ , i.e., when the  $\alpha_i$  is associated with a support vector  $\mathbf{x}^{(s)}$ !
- Other conditions include  $\alpha_i \geq 0$ .

13

## Dual Problem

- Given the training sample  $\{(\mathbf{x}_i, d_i)\}_{i=1}^N$ , find the Lagrange multipliers  $\{\alpha_i\}_{i=1}^N$  that maximize the objective function:

$$Q(\alpha) = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^N \alpha_i$$

subject to the constraints

- $-\sum_{i=1}^N \alpha_i d_i = 0$
- $\alpha_i \geq 0$  for all  $i = 1, 2, \dots, N$ .
- The problem is stated entirely in terms of the training data  $(\mathbf{x}_i, d_i)$ , and the dot products  $\mathbf{x}_i^T \mathbf{x}_j$  play a key role.

15

## Primal Problem: Constrained Optimization (cont'd)

- Plugging in  $\mathbf{w} = \sum_{i=1}^N \alpha_i d_i \mathbf{x}_i$  and  $\sum_{i=1}^N \alpha_i d_i = 0$  back into  $J(\mathbf{w}, b, \alpha)$ , we get the **dual problem**.

$$\begin{aligned} J(\mathbf{w}, b, \alpha) &= \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_{i=1}^N \alpha_i [d_i(\mathbf{w}^T \mathbf{x}_i + b) - 1] \\ &= \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_{i=1}^N \alpha_i d_i \mathbf{w}^T \mathbf{x}_i \\ &\quad - b \sum_{i=1}^N \alpha_i d_i + \sum_{i=1}^N \alpha_i \\ &\quad \left\{ \text{noting } \mathbf{w}^T \mathbf{w} = \sum_{i=1}^N \alpha_i d_i \mathbf{w}^T \mathbf{x}_i \right. \\ &\quad \left. \text{and from } \sum_{i=1}^N \alpha_i d_i = 0 \right\} \\ &= -\frac{1}{2} \sum_{i=1}^N \alpha_i d_i \mathbf{w}^T \mathbf{x}_i + \sum_{i=1}^N \alpha_i \\ &= -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^N \alpha_i \\ &= Q(\alpha). \end{aligned}$$

- So,  $J(\mathbf{w}, b, \alpha) = Q(\alpha)$  ( $\alpha_i \geq 0$ ).
- This results in the **dual problem** (next slide).

14

## Solution to the Optimization Problem

Once all the optimal Lagrange multipliers  $\alpha_{o,i}$  are found,  $\mathbf{w}_o$  and  $b_o$  can be found as follows:

$$\mathbf{w}_o = \sum_{i=1}^N \alpha_{o,i} d_i \mathbf{x}_i$$

and from  $\mathbf{w}_o^T \mathbf{x}_i + b_o = d_i$  when  $\mathbf{x}_i$  is a support vector:

$$b_o = d^{(s)} - \mathbf{w}_o^T \mathbf{x}^{(s)}$$

Note: calculation of final estimated function does not need any explicit calculation of  $\mathbf{w}_o$  since they can be calculated from the dot product between the input vectors!

$$\mathbf{w}_o^T \mathbf{x} = \sum_{i=1}^N \alpha_{o,i} d_i \mathbf{x}_i^T \mathbf{x}$$

16

## Margin of Separation in SVM and VC Dimension

Statistical learning theory shows that it is desirable to reduce both the error (empirical risk) and the VC dimension of the classifier.

- Vapnik (1995, 1998) showed: Let  $D$  be the diameter of the smallest ball containing all input vectors  $\mathbf{x}_i$ . The set of optimal hyperplanes defined by  $\mathbf{w}_o^T \mathbf{x} + b_o = 0$  has a VC dimension  $h$  bounded from above as

$$h \leq \min \left\{ \left\lceil \frac{D^2}{\rho^2} \right\rceil, m_0 \right\} + 1$$

where  $\lceil \cdot \rceil$  is the ceiling,  $\rho$  the margin of separation equal to  $2/\|\mathbf{w}_o\|$ , and  $m_0$  the dimensionality of the input space.

- The implication is that the VC dimension can be controlled independently of  $m_0$ , by choosing an appropriate (large)  $\rho$ !

17

## Soft-Margin Classification (cont'd)

- We want to find a separating hyperplane that minimizes:

$$\Phi(\xi) = \sum_{i=1}^N I(\xi_i - 1)$$

where  $I(\xi) = 0$  if  $\xi \leq 0$  and 1 otherwise.

- Solving the above is NP-complete, so we instead solve an approximation:

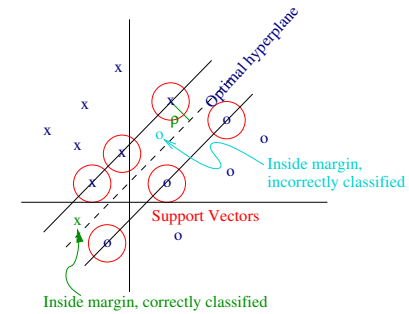
$$\Phi(\xi) = \sum_{i=1}^N \xi_i$$

- Furthermore, the weight vector can be factored in:

$$\Phi(\mathbf{x}, \xi) = \underbrace{\frac{1}{2} \mathbf{w}^T \mathbf{w}}_{\text{Controls VC dim}} + C \underbrace{\sum_{i=1}^N \xi_i}_{\text{Controls error}}$$

with a control parameter  $C$ . 19

## Soft-Margin Classification



- Some problems can violate the condition:

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

- We can introduce a new set of variables  $\{\xi_i\}_{i=1}^N$ :

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

where  $\xi_i$  is called the *slack variable*.

18

## Soft-Margin Classification: Solution

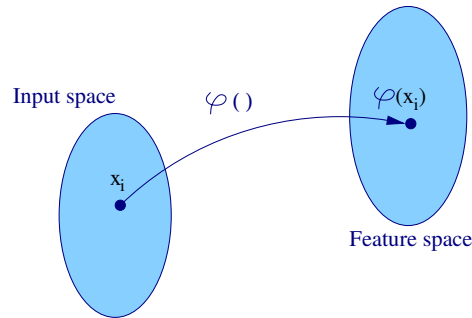
- Following a similar route involving Lagrange multipliers, and a more restrictive condition of  $0 \leq \alpha_i \leq C$ , we get the solution:

$$\mathbf{w}_o = \sum_{i=1}^{N_s} \alpha_{o,i} d_i \mathbf{x}_i$$

$$b_o = d_i(1 - \xi_i) - \mathbf{w}_o^T \mathbf{x}_i$$

20

## Nonlinear SVM



- Nonlinear mapping of an input vector to a high-dimensional *feature space* (exploit Cover's theorem)
- Construction of an optimal hyperplane for separating the features identified in the above step.

21

## Inner-Product Kernel (cont'd)

- The inner product  $\varphi^T(\mathbf{x})\varphi(\mathbf{x}_i)$  is between two vectors in the feature space.
- The calculation of this inner product can be simplified by use of an *inner-product kernel*  $K(\mathbf{x}, \mathbf{x}_i)$ :

$$K(\mathbf{x}, \mathbf{x}_i) = \varphi^T(\mathbf{x})\varphi(\mathbf{x}_i) = \sum_{j=0}^{m_1} \varphi_j(\mathbf{x})\varphi_j(\mathbf{x}_i)$$

where  $m_1$  is the dimension of the feature space. (Note:

$$K(\mathbf{x}, \mathbf{x}_i) = K(\mathbf{x}_i, \mathbf{x}).)$$

- So, the optimal hyperplane becomes:

$$\sum_{i=1}^N \alpha_i d_i K(\mathbf{x}, \mathbf{x}_i) = 0$$

23

## Inner-Product Kernel

- Input  $\mathbf{x}$  is mapped to  $\varphi(\mathbf{x})$ .
- With the weight  $\mathbf{w}$  (including the bias  $b$ ), the decision surface in the feature space becomes (assume  $\varphi_0(\mathbf{x}) = 1$ ):

$$\mathbf{w}^T \varphi(\mathbf{x}) = 0$$

- Using the steps in linear SVM, we get

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

- Combining the above two, we get the decision surface

$$\sum_{i=1}^N \alpha_i d_i \varphi^T(\mathbf{x}_i) \varphi(\mathbf{x}) = 0.$$

22

## Inner-Product Kernel (cont'd)

- **Mercer's theorem** states that  $K(\mathbf{x}, \mathbf{x}_i)$  that follow certain conditions (continuous, symmetric, positive semi-definite) can be expressed in terms of an inner-product in a nonlinearly mapped feature space.
- Kernel function  $K(\mathbf{x}, \mathbf{x}_i)$  allows us to calculate the inner product  $\varphi^T(\mathbf{x})\varphi(\mathbf{x}_i)$  in the mapped feature space without any explicit calculation of the mapping function  $\varphi(\cdot)$ .

24

## Examples of Kernel Functions

- Linear:  $K(\mathbf{x}, \mathbf{x}_i) = \mathbf{x}^T \mathbf{x}_i$ .
- Polynomial:  $K(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^T \mathbf{x}_i + 1)^p$ .
- RBF:  $K(\mathbf{x}, \mathbf{x}_i) = \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x} - \mathbf{x}_i\|^2\right)$ .
- Two-layer perceptron:  $K(\mathbf{x}, \mathbf{x}_i) = \tanh(\beta_0 \mathbf{x}^T \mathbf{x}_i + \beta_1)$   
(for some  $\beta_0$  and  $\beta_1$ ).

25

## Nonlinear SVM: Solution

- The solution is basically the same as the linear case, where  $\mathbf{x}^T \mathbf{x}_i$  is replaced with  $K(\mathbf{x}, \mathbf{x}_i)$ , and an additional constraint that  $\alpha \leq C$  is added.

27

## Kernel Example

- Expanding

$$K(\mathbf{x}, \mathbf{x}_i) = (1 + \mathbf{x}^T \mathbf{x}_i)^2$$

with  $\mathbf{x} = [x_1, x_2]^T$ ,  $\mathbf{x}_i = [x_{i1}, x_{i2}]^T$ ,

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}_i) &= 1 + x_1^2 x_{i1}^2 + 2x_1 x_2 x_{i1} x_{i2} \\ &\quad + x_2^2 x_{i2}^2 + 2x_1 x_{i1} + 2x_2 x_{i2} \\ &= [1, x_1^2, \sqrt{2}x_1 x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2] \\ &\quad [1, x_{i1}^2, \sqrt{2}x_{i1} x_{i2}, x_{i2}^2, \sqrt{2}x_{i1}, \sqrt{2}x_{i2}]^T \\ &= \boldsymbol{\varphi}(\mathbf{x})^T \boldsymbol{\varphi}(\mathbf{x}_i), \end{aligned}$$

where  $\boldsymbol{\varphi}(\mathbf{x}) = [1, x_1^2, \sqrt{2}x_1 x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2]^T$ .

26

## Nonlinear SVM Summary

Project input to high-dimensional space to turn the problem into a linearly separable problem.

Issues with a projection to higher dimensional feature space:

- **Statistical problem:** Danger of invoking curse of dimensionality and higher chance of overfitting
  - Use large margins to reduce VC dimension
- **Computational problem:** computational overhead for calculating the mapping  $\boldsymbol{\varphi}(\cdot)$ :
  - Solve by using the kernel trick.

28