HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Machine Learning

IT3190E

Lecture: Support vector machines

ONE LOVE. ONE FUTURE.

Contents

Support Vector Machines (1)

- **Support Vector Machines (SVM)** (máy vectơ hỗ trợ) was proposed by Vapnik and his colleages in 1970s. Then it became famous and popular in 1990s.
- Originally, SVM is a method for linear classification. It finds a hyperplane (also called *linear classifier*) to separate the two classes of data.
- For *non-linear classification* for which no hyperplane separates well the data, *kernel functions* (hàm nhân) will be used.
  - Kernel functions play the role to transform the data into another space, in which the data is linearly separable.
- Sometimes, we call linear SVM when no kernel function is used. (in fact, linear SVM uses a linear kernel)

Support Vector Machines (2)

- SVM has a strong theory that supports its performance.
- It can work well with very high dimensional problems.
- It is now one of the most popular and strong methods.
- For text categorization, linear SVM performs very well.

1. SVM: the linearly separable case

- Problem representation:
  - Training data **D** = {($\mathbf{x}_1$, y_1), ($\mathbf{x}_2$, y_2), …, ($\mathbf{x}_r$, y_r)} with *r* instances.
  - Each $\mathbf{x}_i$ is a vector in an n-dimensional space, e.g., $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in})^T$. Each dimension represents an attribute.
  - Bold characters denote vectors.
  - y_i is a class label in {-1; 1}. '1' is possitive class, '-1' is negative class.
- **Linear separability assumption:** there exists a hyperplane (of linear form) that well separates the two classes
(giả thuyết tồn tại một siêu phẳng mà phân tách 2 lớp được)

Linear SVM

- SVM finds a hyperplane of the form:

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b$$

  - $\mathbf{w}$ is the weight vector; b is a real number (bias). [Eq.1]
  - $\langle \mathbf{w} \cdot \mathbf{x} \rangle$ and $\langle \mathbf{w}, \mathbf{x} \rangle$ denote the inner product of two vectors (tích vô hướng của hai véctơ)

- Such that for each x_i:

$$y_i = \begin{cases} 1 & if \quad \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 0 \\ -1 & if \quad \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b < 0 \end{cases}$$

[Eq.2]

Separating hyperplane

- The hyperplane (H_0) which separates the possitive from negative class is of the form:

$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$$

- It is also known as the *decision boundary*/surface.
- But there might be infinitely many separating hyperplanes. Which one should we choose?

[Liu, 2006]

Hyperplane with max margin

- SVM selects the hyperplane with **max margin**.
  (SVM tìm siêu phẳng tách mà có lề lớn nhất)
- It is proven that *the max-margin hyperplane has minimal errors among all possible hyperplanes.*

[Liu, 2006]

Marginal hyperplanes

- Assume that the two classes in our data can be separated clearly by a hyperplane.
- Denote $(\mathbf{x}^+,1)$ in possitive class and $(\mathbf{x}^-,-1)$ in negative class which are *closest* to the separating hyperplane H_0 $(\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0)$
- We define two parallel *marginal hyperplanes* as follows:
  - H_+ crosses $\mathbf{x}^+$ and is parallel with H_0: $\langle \mathbf{w} \cdot \mathbf{x}^+ \rangle + b = 1$
  - H_- crosses $\mathbf{x}^-$ and is parallel with H_0: $\langle \mathbf{w} \cdot \mathbf{x}^- \rangle + b = -1$
  - No data point lies between these two marginal hyperplanes. And satisfying:

$$\begin{aligned} \langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b &\geq 1, && \text{if } y_i = 1 \\ \langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b &\leq -1, && \text{if } y_i = -1 \end{aligned} \qquad \text{[Eq.3]}$$

The margin (1)

- **Margin** (mức lề) is defined as the distance between the two marginal hyperplanes.
  - Denote d_+ the distance from H_0 to H_+.
  - Denote d_- the distance from H_0 to H_-.
  - $(d_+ + d_-)$ is the margin.
- Remember that the distance from a point $\mathbf{x_i}$ to the hyperplane H_0 ($\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$) is computed as:

$$\frac{|\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b|}{\|\mathbf{w}\|} \qquad \text{[Eq.4]}$$

  - Where:

$$\|\mathbf{w}\| = \sqrt{<\mathbf{w} \cdot \mathbf{w}>} = \sqrt{w_1^2 + w_1^2 + \ldots + w_1^2} \qquad \text{[Eq.5]}$$

The margin (2)

- So the distance d_+ from $\mathbf{x}^+$ to H_0 is

$$d_+ = \frac{|\langle \mathbf{w} \cdot \mathbf{x}^+ \rangle + b|}{\|\mathbf{w}\|} = \frac{|1|}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|} \qquad \text{[Eq.6]}$$

- Similarly, the distance d_- from $\mathbf{x}^-$ to H_0 is

$$d_- = \frac{|\langle \mathbf{w} \cdot \mathbf{x}^- \rangle + b|}{\|\mathbf{w}\|} = \frac{|-1|}{\|\mathbf{w}\|} = \frac{1}{\|\mathbf{w}\|} \qquad \text{[Eq.7]}$$

- As a result, the margin is:

$$margin = d_+ + d_- = \frac{2}{\|\mathbf{w}\|} \qquad \text{[Eq.8]}$$

SVM: learning with max margin (1)

- *SVM learns a classifier H_0 with a maximum margin*, i.e., the hyperplane that has the greatest margin among all possible hyperplanes.

- This learning principle can be formulated as the following quadratic optimization problem:
  - *Find **w** and b that maximize*

$$margin = \frac{2}{\|\mathbf{w}\|}$$

  - *and satisfy the below conditions for any training data **x_i**:*

$$\begin{cases} \langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b \geq 1, \text{if } \mathrm{y_i} = 1 \\ \langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b \leq -1, \text{if } \mathrm{y_i} = \text{-}1 \end{cases}$$

SVM: learning with max margin (2)

- Learning SVM is equivalent to the following minimization problem:
  - Minimize

$$\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} \qquad \text{[Eq.9]}$$

  - Conditioned on

$$\begin{cases} \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1, & \text{if } y_i = 1 \\ \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1, & \text{if } y_i = -1 \end{cases}$$

- Note, it can be reformulated as:
  - Minimize

$$\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} \qquad \text{[Eq.10]}$$

(P)

  - Conditioned on

$$y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad \forall i = 1..r$$

- This is a *constrained optimization problem*.

Constrained optimization (1)

- Consider the problem:

Minimize *f(**x**)* conditioned on *g(**x**)* = 0

- *Necessary condition:* a solution $\mathbf{x}_0$ will satisfy

$$\begin{cases} \left.\dfrac{\partial}{\partial \mathbf{x}}\big(f(\mathbf{x}) + \alpha g(\mathbf{x})\big)\right|_{\mathbf{x}=\mathbf{x}_0} = 0 \\ g(\mathbf{x}) = 0 \end{cases};$$

  - Where α is a *Lagrange multiplier*.

- In the cases of many constraints ($g_i(x)=0$ for i=1…r), a solution $\mathbf{x}_0$ will satisfy:

$$\begin{cases} \left.\dfrac{\partial}{\partial \mathbf{x}}\left(f(\mathbf{x}) + \sum_{i=1}^{r} \alpha_i g_i(\mathbf{x})\right)\right|_{\mathbf{x}=\mathbf{x}_0} = 0 \\ g_i(\mathbf{x}) = 0 \end{cases};$$

Constrained optimization (2)

- Consider the problem with inequality constraints:

Minimize $f(\boldsymbol{x})$ conditioned on $g_i(\boldsymbol{x}) \le 0$

- *Necessary condition:* a solution $\mathbf{x}_0$ will satisfy

$$\begin{cases} \dfrac{\partial}{\partial \mathbf{x}}\left(f(\mathbf{x}) + \sum_{i=1}^{r} \alpha_i g_i(\mathbf{x}) \right)\Bigg|_{\mathbf{x}=\mathbf{x}_0} = 0 ; \\ g_i(\mathbf{x}) \le 0 \end{cases}$$

  - Where $\alpha_i \ge 0$ is a Lagrange multiplier.
- $L = f(\mathbf{x}) + \sum_{i=1}^{r} \alpha_i g_i(\mathbf{x})$ is known as the Lagrange function.
  - $\mathbf{x}$ is called *primal variable* (biến gốc)
  - α is called *dual variable* (biến đối ngẫu)

SVM: learning with max margin (3)

- The Lagrange function for problem [Eq. 10] is

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{r} \alpha_i [y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) - 1] \quad \text{[Eq.11a]}$$

  - Where each $\alpha_i \geq 0$ is a Lagrange multiplier.

- Solving [Eq. 10] is equivalent to the following minimax problem:

$$\arg\min_{\mathbf{w},b} \max_{\alpha \geq 0} L(\mathbf{w}, b, \alpha) \quad \text{[Eq.11b]}$$

$$= \arg\min_{\mathbf{w},b} \max_{\alpha \geq 0} \left(\frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{r} \alpha_i [y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) - 1] \right)$$

SVM: learning with max margin (4)

- The *primal problem* [Eq. 10] can be derived by solving:

$$\max_{\alpha \geq 0} L(\mathbf{w}, b, \alpha) = \max_{\alpha \geq 0} \left(\frac{1}{2} \langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{r} \alpha_i [y_i (\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) - 1] \right)$$

- Its *dual problem* (đối ngẫu) can be derived by solving:

$$\min_{\mathbf{w}, b} L(\mathbf{w}, b, \alpha) = \min_{\mathbf{w}, b} \left(\frac{1}{2} \langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{r} \alpha_i [y_i (\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) - 1] \right)$$

- It is known that the optimal solution to [Eq. 10] will satisfy some conditions which is called the **Karush-Kuhn-Tucker** (KKT) conditions.

SVM: Karush-Kuhn-Tucker

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{r} \alpha_i y_i \mathbf{x_i} = 0$$ [Eq.12]

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{r} \alpha_i y_i = 0$$ [Eq.13]

$$y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) - 1 \geq 0, \ \forall \mathbf{x_i} \ (i = 1..r)$$ [Eq.14]

$$\alpha_i \geq 0$$ [Eq.15]

$$\alpha_i(y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) - 1) = 0$$ [Eq.16]

- The last equation [Eq. 16] comes from a nice result from the duality theory.
  - Note: any $\alpha_i > 0$ will imply that the associated point $\mathbf{x_i}$ lies in a boundary hyperplane (H_+ or H_-).
  - Such a boundary point is named as a ***support vector***.
  - A non-support vector will correspond to $\alpha_i = 0$.

SVM: learning with max margin (5)

- In general, the KKT conditions do not guarantee the optimality of the solution.
- Fortunately, due to the convexity of the primal problem [Eq.10], the *KKT conditions are both necessary and sufficient to assure the global optimality of the solution. It means a vector satisfying all KKT conditions provides the globally optimal classifier.*
  - Convex optimization is 'easy' in the sense that we always can find a good solution with a provable guarantee.
  - There are many algorithms in the literature, but most are iterative.
- In fact, problem [Eq.10] is pretty hard to derive an efficient algorithm. Therefore, its **dual problem** is more preferable.

SVM: the dual form (1)

- Remember that the dual counterpart of [Eq.10] is

$$\min_{\mathbf{w},b} L(\mathbf{w},b,\alpha) = \min_{\mathbf{w},b}\left(\frac{1}{2}\langle\mathbf{w}\cdot\mathbf{w}\rangle - \sum_{i=1}^{r}\alpha_i[y_i(\langle\mathbf{w}\cdot\mathbf{x}_i\rangle + b) - 1]\right)$$

- By taking the gradient of L(**w**,b,α) in variables (**w**,b) and zeroing it, we can find the following dual function:

$$L_D(\alpha) = \sum_{i=1}^{r}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{r}\alpha_i\alpha_j y_i y_j\langle\mathbf{x}_i\cdot\mathbf{x}_j\rangle \quad \text{[Eq.17]}$$

SVM: the dual form (2)

- Solving problem [Eq.10] is equivalent to solving its dual problem below:
  - Maximize $L_D(\boldsymbol{\alpha}) = \sum_{i=1}^{r} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{r} \alpha_i \alpha_j y_i y_j \langle \mathbf{x_i} \cdot \mathbf{x_j} \rangle$ [Eq.18]
  - Such that $\begin{cases} \sum_{i=1}^{r} \alpha_i y_i = 0 \\ \alpha_i \geq 0, \forall i = 1..r \end{cases}$ **(D)**
- The constraints in **(D)** is much simpler than those of the primal problem. Therefore deriving an efficient method to solve this problem might be easier.
  - However, existing algorithms for this problem are iterative and complicated. Therefore, we will not discuss any algorithm in detail !

SVM: the optimal classifier

- Once the dual problem is solved for $\boldsymbol{\alpha}$, we can recover the optimal solution to problem [Eq.10] by using the KKT.
- Let SV be the set of all support vectors
  - SV is a subset of the training data.
  - $\alpha_i > 0$ suggests that $\mathbf{x_i}$ is a support vector.
- We can compute **w*** by using [Eq.12]. So:

$$\mathbf{w}^* = \sum_{i=1}^{r} \alpha_i y_i \mathbf{x}_i = \sum_{\mathbf{x_i} \in SV} \alpha_i y_i \mathbf{x}_i; \qquad \text{(due to } \alpha_j = 0 \text{ for any } \mathbf{x_j} \text{ not in SV)}$$

- To find b*, we take an index k such that $\alpha_k > 0$:
  - It means $y_k(\langle \mathbf{w}^* \cdot \mathbf{x}_k \rangle + b^*) - 1 = 0$ due to [Eq.16].
  - Hence,
  - $b^* = y_k - \langle \mathbf{w}^* \cdot \mathbf{x}_k \rangle$

SVM: classifying new instances

- The decision boundary is

$$f(\mathbf{x}) = \langle \mathbf{w}^* \cdot \mathbf{x} \rangle + b^* = \sum_{\mathbf{x_i} \in SV} \alpha_i y_i \langle \mathbf{x_i} \cdot \mathbf{x} \rangle + b^* = 0 \qquad \text{[Eq.19]}$$

- For a new instance **z**, we compute:

$$sign(\langle \mathbf{w}^* \cdot \mathbf{z} \rangle + b^*) = sign\left(\sum_{\mathbf{x_i} \in SV} \alpha_i y_i \langle \mathbf{x_i} \cdot \mathbf{z} \rangle + b^* \right) \qquad \text{[Eq.20]}$$

  - If the result is 1, **z** will be assigned to the possitive class; otherwise **z** will be assigned to the negative class.

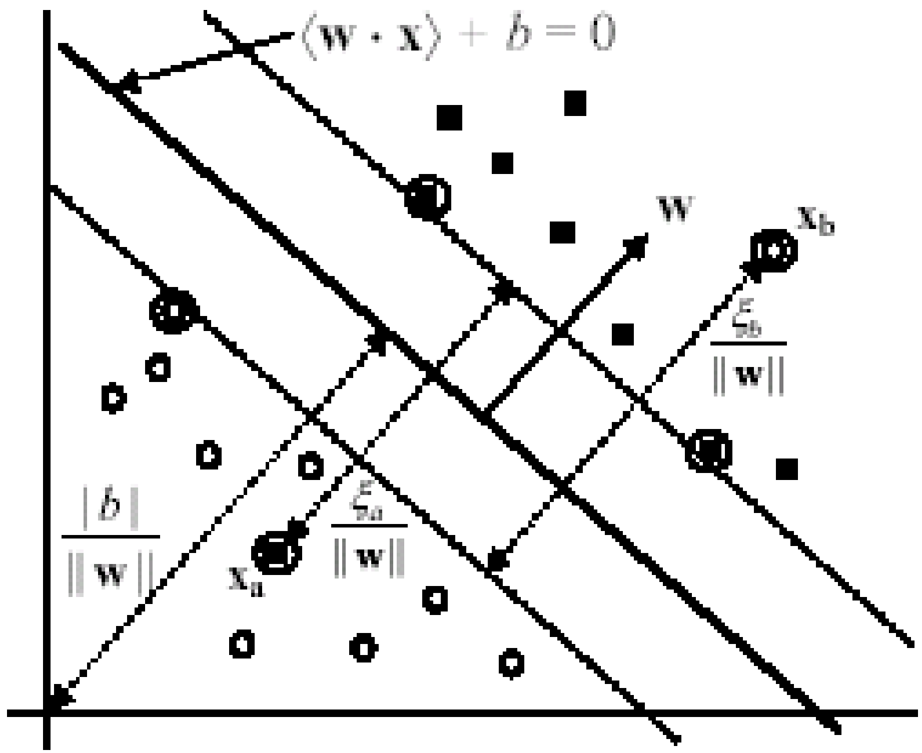
- Note that this classification principle
  - Just depends on the support vectors.
  - Just needs to compute some dot products.

2. Soft-margin SVM

- *What if the two classes are not linearly separable?* *(Trường hợp 2 lớp không thể phân tách tuyến tính thì sao?)*
  - Linear separability is ideal in practice.
  - Data are often noisy or erronous, making two classes overlapping (nhiễu/lỗi có thể làm 2 lớp giao nhau)
- In the case of linear separability:
  - *Minimize* $\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2}$
  - *Conditioned on* $y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad \forall i = 1..r$
- In the cases of **noises** or **overlapping**, those constraints may never meet simutaneously.
  - It means we cannot solve for **w*** and b*.

Example of inseparability

- Noisy points $\mathbf{x}_a$ and $\mathbf{x}_b$ are mis-labeled.



Relaxing the constraints

- To work with noises/errors, we need to relax the constraints about margin by using some slack variables ξ_i (≥ 0):
  (Ta sẽ mở rộng ràng buộc về lề bằng cách thêm biến bù)

$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1 - \xi_i \qquad \text{if } y_i = 1$$

$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1 + \xi_i \qquad \text{if } y_i = -1$$

  - For a noisy/erronous point $\mathbf{x}_i$, we have: $\xi_i > 1$
  - Otherwise $\xi_i = 0$.

- Therefore, we have the following conditions for the cases of nonlinear separability:

$$y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i \qquad \text{for all } i = 1 \ldots r$$

$$\xi_i \geq 0 \qquad \text{for all } i = 1 \ldots r$$

Penalty on noises/errors

- We should enclose some information on noises/errors into the objective function when learning
(ta nên đính thêm thông tin về nhiễu/lỗi vào hàm mục tiêu)
  - Otherwise, the resulting classifier easily overfits the data.
- A penalty term will be used so that learning is to minimize

$$\frac{\langle \boldsymbol{W}, \boldsymbol{W} \rangle}{2} + C \sum_{i=1}^{r} \xi_i^k$$

  - Where C (>0) is the penalty constant (hằng số phạt).
  - The greater C, the heavier the penalty on noises/errors.
- $k = 1$ is often used in practice, due to simplicity for solving the optimization problem.

The new optimization problem

- *Minimize* $$\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C\sum_{i=1}^{r} \xi_i$$ [Eq.21]
  - *Conditioned on* $$\begin{cases} y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, & \forall i = 1..r \\ \xi_i \geq 0, & \forall i = 1..r \end{cases}$$
- This problem is called **Soft-margin SVM**.
- It is equivalent to minimize the following function

$$\left[\frac{1}{r}\sum_{i=1}^{r} \max(0, 1 - y_i(\langle \boldsymbol{w} \cdot \boldsymbol{x}_i \rangle + b))\right] + \lambda \|\boldsymbol{w}\|_2^2$$

  - $\max(0, 1 - y_i(\langle \boldsymbol{w} \cdot \boldsymbol{x}_i \rangle + b))$ is called *Hinge loss*
  - Some popular losses: squared error, cross entropy, hinge
  - $\lambda > 0$ is a constant

The new optimization problem

- Its Lagrange function is

$$L = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle + C\sum_{i=1}^{r}\xi_i - \sum_{i=1}^{r}\alpha_i[y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) - 1 + \xi_i] - \sum_{i=1}^{r}\mu_i\xi_i$$

[Eq.22]

  - Where α_i (≥ 0) and μ_i (≥ 0) are Lagrange multipliers.

Karush-Kuhn-Tucker conditions (1)

$$\frac{\partial L_P}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{r} \alpha_i y_i \mathbf{x_i} = 0$$ [Eq.23]

$$\frac{\partial L_P}{\partial b} = -\sum_{i=1}^{r} \alpha_i y_i = 0$$ [Eq.24]

$$\frac{\partial L_P}{\partial \xi_i} = C - \alpha_i - \mu_i = 0, \quad \forall i = 1..r$$ [Eq.25]

Karush-Kuhn-Tucker conditions (2)

$$y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) - 1 + \xi_i \geq 0, \quad \forall i = 1..r$$ [Eq.26]

$$\xi_i \geq 0$$ [Eq.27]

$$\alpha_i \geq 0$$ [Eq.28]

$$\mu_i \geq 0$$ [Eq.29]

$$\alpha_i(y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) - 1 + \xi_i) = 0$$ [Eq.30]

$$\mu_i \xi_i = 0$$ [Eq.31]

The dual problem

- *Maximize* $$L_D(\boldsymbol{\alpha}) = \sum_{i=1}^{r} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{r} \alpha_i \alpha_j y_i y_j \langle \mathbf{x_i} \cdot \mathbf{x_j} \rangle$$
  - *Such that* $$\begin{cases} \sum_{i=1}^{r} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C, \quad \forall i = 1..r \end{cases}$$ [Eq.32]

- Note that neither ξ nor μ_i appears in the dual problem.
- This problem is almost similar with that [Eq.18] in the case of linearly separable classification.
- The only difference is the constraint: $\alpha_i \le C$

Soft-margin SVM: the optimal classifier

- Once the dual problem is solved for α, we can recover the optimal solution to problem [Eq.21].
- Let SV be the set of all *support/noisy vectors*
  - SV is a subset of the training data.
  - $\alpha_i > 0$ suggests that $\mathbf{x}_i$ is a support/noisy vector.
- We can compute **w*** by using [Eq.12]. So:

$$\mathbf{w}^* = \sum_{i=1}^{r} \alpha_i y_i \mathbf{x}_i = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \mathbf{x}_i; \qquad \text{(due to } \alpha_j = 0 \text{ for any } \mathbf{x}_j \text{ not in SV)}$$

- To find b*, we take an index k such that $C > \alpha_k > 0$:
  - It means $\xi_k = 0$ due to [Eq.25] and [Eq.31];
  - And $y_k(\langle \mathbf{w}^* \cdot \mathbf{x}_k \rangle + b^*) - 1 = 0$ due to [Eq.30].
  - Hence, $b^* = y_k - \langle \mathbf{w}^* \cdot \mathbf{x}_k \rangle$

Some notes

- From [Eq.25-31] we conclude that

$$\boxed{\begin{array}{llll} \text{If} \quad \alpha_i = 0 & \text{then} & y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) \geq 1, & \text{and} \ \xi_i = 0 \\ \text{If} \quad 0 < \alpha_i < C & \text{then} & y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) = 1, & \text{and} \ \xi_i = 0 \\ \text{If} \quad \alpha_i = C & \text{then} & y_i(\langle \mathbf{w} \cdot \mathbf{x_i} \rangle + b) < 1, & \text{and} \ \xi_i > 0 \end{array}}$$

- The classifier can be expressed as a *linear combination* of few training points.
  - Most training points lie outside the margin area: $\alpha_i = 0$
  - The support vectors lie in the marginal hyperplanes: $0 < \alpha_i < C$
  - The noisy/erronous points will associate with $\alpha_i = C$
- Hence the optimal classifier is a very sparse *combination* of the training data.

Soft-margin SVM: classifying new instances

- The decision boundary is

$$f(\mathbf{x}) = \langle \mathbf{w}^* \cdot \mathbf{x} \rangle + b^* = \sum_{\mathbf{x_i} \in SV} \alpha_i y_i \langle \mathbf{x_i} \cdot \mathbf{x} \rangle + b^* = 0 \qquad \text{[Eq.19]}$$

- For a new instance **z**, we compute:

$$sign(\langle \mathbf{w}^* \cdot \mathbf{z} \rangle + b^*) = sign\left(\sum_{\mathbf{x_i} \in SV} \alpha_i y_i \langle \mathbf{x_i} \cdot \mathbf{z} \rangle + b^* \right) \qquad \text{[Eq.20]}$$

  - If the result is 1, **z** will be assigned to the possitive class; otherwise **z** will be assigned to the negative class.

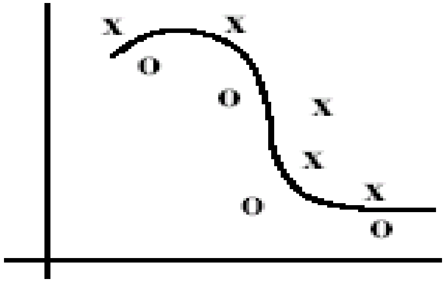
- *Note: it is important to choose a good value of C, since it significantly affects performance of SVM.*
  - We often use a validation set to choose a value for C.

Linear SVM: summary

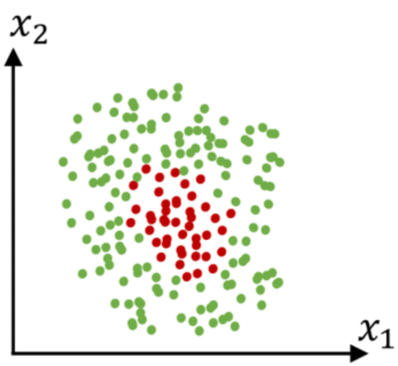
- Classification is based on a separating hyperplane.
- Such a hyperplane is represented as a combination of some support vectors.
- The determination of support vectors reduces to solve a quadratic programming problem.
- In the dual problem and the separating hyperplane, dot products can be used in place of the original training data.
  - This is the door for us to learn a nonlinear classifier.

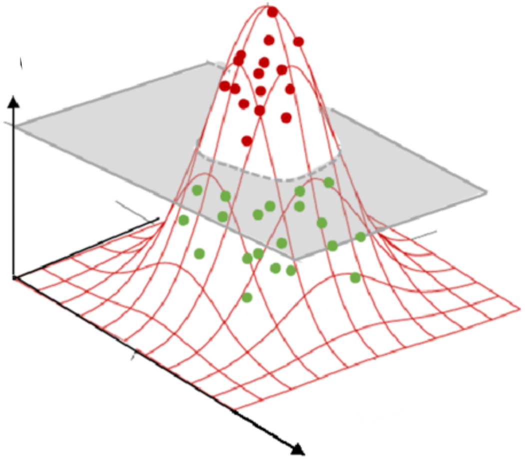
3. Non-linear SVM

- Consider the case in which our data are not linearly separable
  - This may often happen in practice



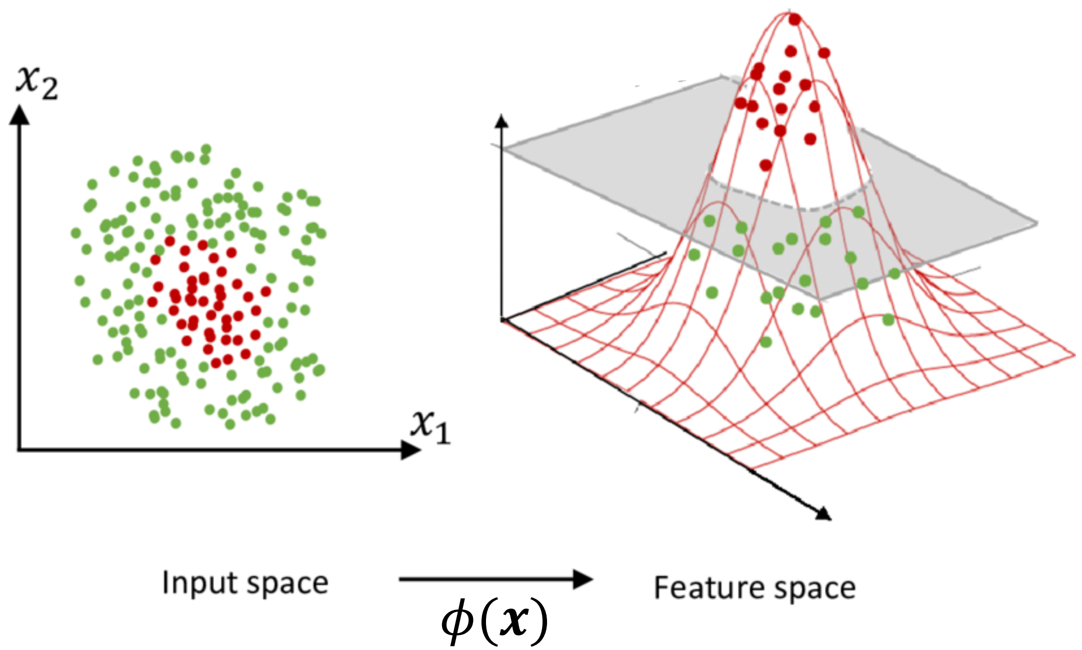
- How about using a non-linear function?
- **Idea of Non-linear SVM:**
  - **Step 1:** transform the input into another space, which often has *higher dimensions*, so that the projection of data is linearly separable
  - **Step 2:** use linear SVM in the new space





Non-linear SVM

- **Input space:** initial representation of data
- **Feature space:** the new space after the transformation



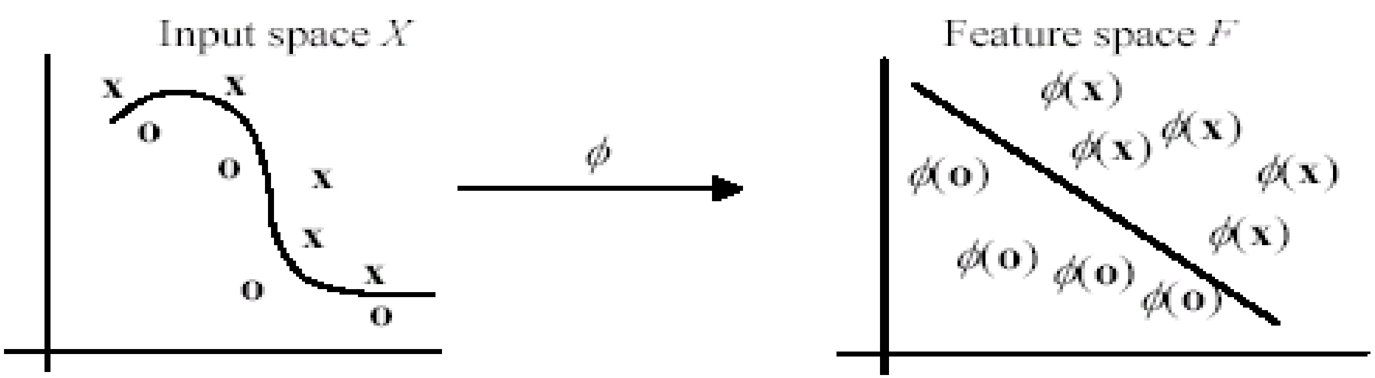
Non-linear SVM: transformation

- Our idea is to map the input **x** to a new representation, using a non-linear mapping

$$\phi:\ X \longrightarrow F$$
$$\boldsymbol{x} \longmapsto \phi(\boldsymbol{x})$$

- In the feature space, the original training data $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_r, y_r)\}$ are represented by

$$\{(\phi(\mathbf{x}_1), y_1), (\phi(\mathbf{x}_2), y_2), \ldots, (\phi(\mathbf{x}_r), y_r)\}$$



Non-linear SVM: transformation

- Consider the input space to be 2-dimensional, and we choose the following map

$$\phi: \ X \longrightarrow F$$
$$(x_1, x_2) \longmapsto (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

- So instance **x** = (2, 3) will have the representation in the feature space as

$$\phi(\mathbf{x}) = (4, 9, 8.49)$$

Non-linear SVM: learning & prediction

- **Training problem:**

Minimize
$$L_P = \frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2} + C\sum_{i=1}^{r} \xi_i \quad \text{[Eq.34]}$$

Such that
$$\begin{cases} y_i\left(\langle \mathbf{w} \cdot \phi(\mathbf{x_i})\rangle + b\right) \geq 1 - \xi_i, & \forall i = 1..r \\ \xi_i \geq 0, & \forall i = 1..r \end{cases}$$

- **The dual problem:**

Maximize
$$L_D = \sum_{i=1}^{r} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{r} \alpha_i \alpha_j y_i y_j \langle \phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j})\rangle \quad \text{[Eq.35]}$$

Such that
$$\begin{cases} \sum_{i=1}^{r} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C, & \forall i = 1..r \end{cases}$$

- **Classifier:**

$$f(\mathbf{z}) = \langle \boldsymbol{w}^*, \phi(\mathbf{z})\rangle + b^* = \sum_{\boldsymbol{x}_i \in SV} \alpha_i y_i \langle \phi(\boldsymbol{x}_i), \phi(\mathbf{z})\rangle + b^* \quad \text{[Eq.36]}$$

Non-linear SVM: difficulties

- How to find the mapping?
  - An intractable problem
- The curse of dimensionality
  - As the dimensionality increases, the volume of the space increases so fast that the available data become sparse.
  - This sparsity is problematic.
  - Increasing the dimensionality will require significantly more training data.

Dữ liệu dù thu thập được lớn đến đâu thì cũng là **quá nhỏ** so với không gian của chúng

Non-linear SVM: Kernel functions

- An explicit form of a tranformation is not necessary
- **The dual problem:**

Maximize
$$L_D = \sum_{i=1}^{r} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{r} \alpha_i \alpha_j y_i y_j \langle \phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j}) \rangle$$

Such that
$$\begin{cases} \sum_{i=1}^{r} \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C, \quad \forall i = 1..r \end{cases}$$

- **Classifier:** $f(\boldsymbol{z}) = \langle \boldsymbol{w}^*, \phi(\boldsymbol{z}) \rangle + b^* = \sum_{\boldsymbol{x}_i \in SV} \alpha_i y_i \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{z}) \rangle + b^*$
- Both require only the inner product $\langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$
- **Kernel trick:** Nonlinear SVM can be used by replacing those inner products by evaluations of some ***kernel function***

$$K(\mathbf{x},\mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle \qquad \text{[Eq.37]}$$

Kernel functions: example

- Polynomial

$$K(\boldsymbol{x}, \boldsymbol{z}) = \langle \boldsymbol{x}, \boldsymbol{z} \rangle^d$$

- Consider the polynomial with degree d=2. For any vectors **x**=(x_1,x_2) and **z**=(z_1,z_2)

$$\begin{aligned} \langle \boldsymbol{x}, \boldsymbol{z} \rangle^2 &= (x_1 z_1 + x_2 z_2)^2 \\ &= x_1^2 z_1^2 + 2 x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\ &= \left\langle \left(x_1^2, x_2^2, \sqrt{2} x_1 x_2\right), \left(z_1^2, z_2^2, \sqrt{2} z_1 z_2\right) \right\rangle \\ &= \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{z}) \rangle = K(\boldsymbol{x}, \boldsymbol{z}) \end{aligned}$$

  - Where $\phi(\boldsymbol{x}) = \left(x_1^2, x_2^2, \sqrt{2} x_1 x_2\right)$.

- Therefore the polynomial is the product of two vectors $\phi(\boldsymbol{x})$ and $\phi(\boldsymbol{z})$.

Kernel functions: popular choices

- Polynomial

$$K(\mathbf{x},\mathbf{z}) = \left(\langle \mathbf{x} \cdot \mathbf{z} \rangle + \theta\right)^d; \quad \text{trong đó}: \ \theta \in R, d \in N$$

- Gaussian radial basis function (RBF)

$$K(\mathbf{x},\mathbf{z}) = e^{-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{2\sigma}}; \quad \text{trong đó}: \ \sigma > 0$$

- Sigmoid

$$K(\mathbf{x},\mathbf{z}) = \tanh(\beta\langle \mathbf{x} \cdot \mathbf{z} \rangle - \lambda) = \frac{1}{1+e^{-(\beta\langle \mathbf{x} \cdot \mathbf{z} \rangle - \lambda)}}; \quad \text{trong đó}: \ \beta, \lambda \in R$$

- What conditions ensure a kernel function?

  Mercer's theorem

SVM: summary

- SVM works with real-value attributes
  - Any nominal attribute need to be transformed into a real one
- The learning formulation of SVM focuses on 2 classes
  - How about a classification problem with > 2 classes?
  - One-vs-the-rest, one-vs-one: a multiclass problem can be solved by reducing to many different problems with 2 classes
- The decision function is simple, but may be hard to interpret
  - It is more serious if we use some kernel functions

SVM: some packages

- LibSVM:
  - http://www.csie.ntu.edu.tw/~cjlin/libsvm/
- Linear SVM for large datasets:
  - http://www.csie.ntu.edu.tw/~cjlin/liblinear/
  - http://www.cs.cornell.edu/people/tj/svm_light/svm_perf.html
- Scikit-learn in python:
  - http://scikit-learn.org/stable/modules/svm.html
- SVM*light*:
  - http://www.cs.cornell.edu/people/tj/svm_light/index.html

References

- B. Liu. Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data. Springer, 2006.
- C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery, 2(2): 121-167, 1998.
- Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." *Machine learning* 20.3 (1995): 273-297.

Exercises

- **Question 1: Hyperplane Definition** What is the equation of a hyperplane in D-dimensional space?
- A) w^T x = 1
- B) w^T x + b = 0
- C) ||w|| = 1
- D) y = sign(w^T x)

Exercises

- **Question 2: Distance from a Point to Hyperplane** The distance of a point x_n from a hyperplane w^T x + b = 0 is given by:
- A) w^T x_n + b
- B) (w^T x_n + b) / ||w||^2
- C) |w^T x_n + b| / ||w||
- D) ||w|| × (w^T x_n + b)

Exercises

- **Question 3: Prediction Rule** For hyperplane-based binary classification, the prediction rule is:
- A) y* = w^T x* + b
- B) y* = sign(w^T x* + b)
- C) y* = ||w^T x* + b||
- D) y* = max(0, w^T x* + b)

Exercises

- **Question 4: 0-1 Loss Properties** The 0-1 loss function for classification is:
- A) Convex and differentiable
- B) Non-convex and differentiable
- C) Convex and non-differentiable
- D) Non-convex, non-differentiable, and NP-Hard to optimize

Exercises

- **Question 5: Perceptron Loss Function** The Perceptron loss function is:
- A) max{0, -yw^T x}
- B) max{0, 1 - yw^T x}
- C) log(1 + exp(-yw^T x))
- D) (y - ŷ)^2

Exercises

- **Question 6: Hinge Loss** The Hinge loss function used in SVMs is:
- A) max{0, -yw^T x}
- B) max{0, 1 - yw^T x}
- C) log(1 + exp(-yw^T x))
- D) |y - ŷ|

Exercises

- **Question 7: Perceptron Learning Characteristics** When using SGD with Perceptron loss (with k=0), the weight update occurs:
- A) After every iteration
- B) Only when the model makes a mistake
- C) When the margin is less than 1
- D) At fixed time intervals

Exercises

- **Question 8: Perceptron Gradient** For Perceptron loss with k=0, if the model makes a mistake on example (x_n, y_n), the gradient is:
- A) 0
- B) -y_n x_n
- C) x_n
- D) y_n

Exercises

- **Question 9: Perceptron Limitation** A key limitation of the Perceptron algorithm is:
- A) It cannot learn non-linear boundaries
- B) It doesn't guarantee any margin around the hyperplane
- C) It cannot handle multi-class problems
- D) It is too slow to train

Exercises

- **Question 10: Hard-Margin SVM Objective** In Hard-Margin SVM, maximizing the margin $2\gamma = 2/||w||$ is equivalent to:
- A) Maximizing $||w||$
- B) Minimizing $||w||$
- C) Minimizing $||w||^2$ or $||w||^2/2$
- D) Maximizing $||w||^2$

Exercises

- **Question 11: Hard-Margin SVM Constraint** The constraint in Hard-Margin SVM is:
- A) $y_n(w^T x_n + b) \geq 0$
- B) $y_n(w^T x_n + b) \geq 1$
- C) $y_n(w^T x_n + b) \leq 1$
- D) $|w^T x_n + b| \geq 1$

Exercises

- **Question 12: Hard-Margin SVM Assumption** Hard-Margin SVM assumes:
- A) Data is linearly separable with no points in the margin
- B) Data may have some noise
- C) Some misclassifications are allowed
- D) All training examples are support vectors

Exercises

- **Question 13: Solving Hard-Margin SVM** The Hard-Margin SVM optimization problem is solved using:
- A) Gradient descent only
- B) Lagrange's method with Lagrange multipliers
- C) Newton's method
- D) Random search

Exercises

- **Question 14: Soft-Margin SVM Slack Variables** In Soft-Margin SVM, the slack variable ξ_n represents:
- A) The margin width
- B) The extent of violation of the margin constraint
- C) The number of support vectors
- D) The learning rate

Exercises

- **Question 15: Slack Variable Interpretation** When slack variable $\xi_n > 1$, it means:
- A) The point is correctly classified with large margin
- B) The point is within the margin region
- C) The point is totally on the wrong side of the hyperplane
- D) The point is a support vector

Exercises

- **Question 16: Soft-Margin Constraint** The soft-margin constraint for all training examples is:
- A) y_n(w^T x_n + b) ≥ 1
- B) y_n(w^T x_n + b) ≥ 1 - ξ_n
- C) y_n(w^T x_n + b) ≤ 1 - ξ_n
- D) y_n(w^T x_n + b) = 1 - ξ_n

Exercises

- **Question 17: Hyperparameter C in Soft-Margin SVM** The hyperparameter C in Soft-Margin SVM controls:
- A) The learning rate
- B) The number of iterations
- C) The trade-off between large margin and small training error
- D) The dimensionality of the feature space

Exercises

- **Question 18: Large C Value Effect** When C is very large in Soft-Margin SVM:
- A) Large margin, large training error (bad)
- B) Small training error but also small margin (bad)
- C) Balanced trade-off
- D) No effect on the solution

Exercises

- **Question 19: Loss Functions Comparison** Which loss function is convex and differentiable?
- A) 0-1 Loss
- B) Perceptron Loss
- C) Hinge Loss
- D) Logistic Loss

Exercises

- **Question 20: SVM vs Perceptron** The key advantage of SVM over Perceptron is:
- A) SVM is faster to train
- B) SVM learns the maximum margin hyperplane
- C) SVM uses simpler mathematics
- D) SVM doesn't require hyperparameter tuning

Exercises

- What is the main difference between SVM and KNN?
- How many support vectors are there in the worst case? Why?
- The meaning of the constant C in SVM? Compare the role of C in SVM with that of λ in Ridge regression.

HUST

hust.edu.vn fb.com/dhbkhn

THANK YOU !