Introduction to
Machine Learning and Data Mining
(Học máy và Khai phá dữ liệu)

Khoat Than

School of Information and Communication Technology
Hanoi University of Science and Technology

2020

Contents

- Introduction to Machine Learning & Data Mining
- Unsupervised learning
- **Supervised learning**
  - **Support Vector Machines**
- Practical advice

Support Vector Machines (1)

- **Support Vector Machines (SVM)** (máy vectơ hỗ trợ) was proposed by Vapnik and his colleagues 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 $\mathbf{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_r, y_r)\}$ 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

Linear SVM

- SVM finds a hyperplane of the form:

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b \qquad \text{[Eq.1]}$$

  - $\mathbf{w}$ is the weight vector; b is a real number (bias).
  - $\langle \mathbf{w} \cdot \mathbf{x} \rangle$ and $\langle \mathbf{w}, \mathbf{x} \rangle$ denote the inner product of two vectors
- Such that for each x_i:

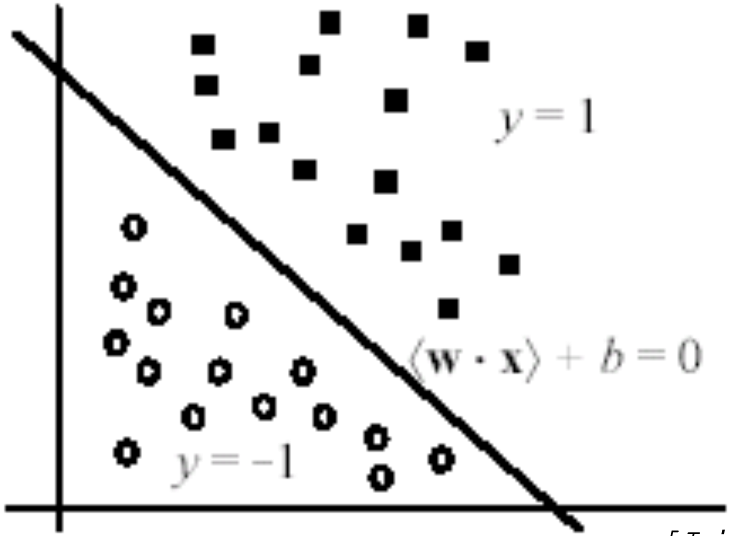
$$y_i = \begin{cases} 1 & if \quad \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 0 \\ -1 & if \quad \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b < 0 \end{cases} \qquad \text{[Eq.2]}$$

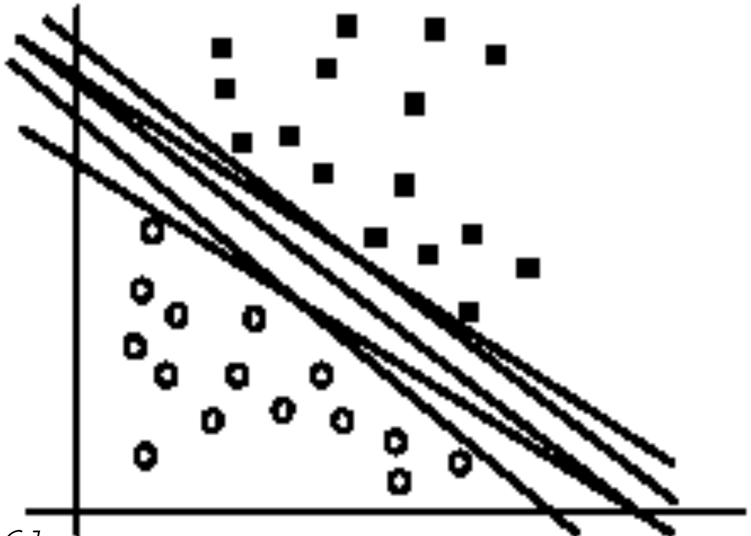
Separating hyperplane

- 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?

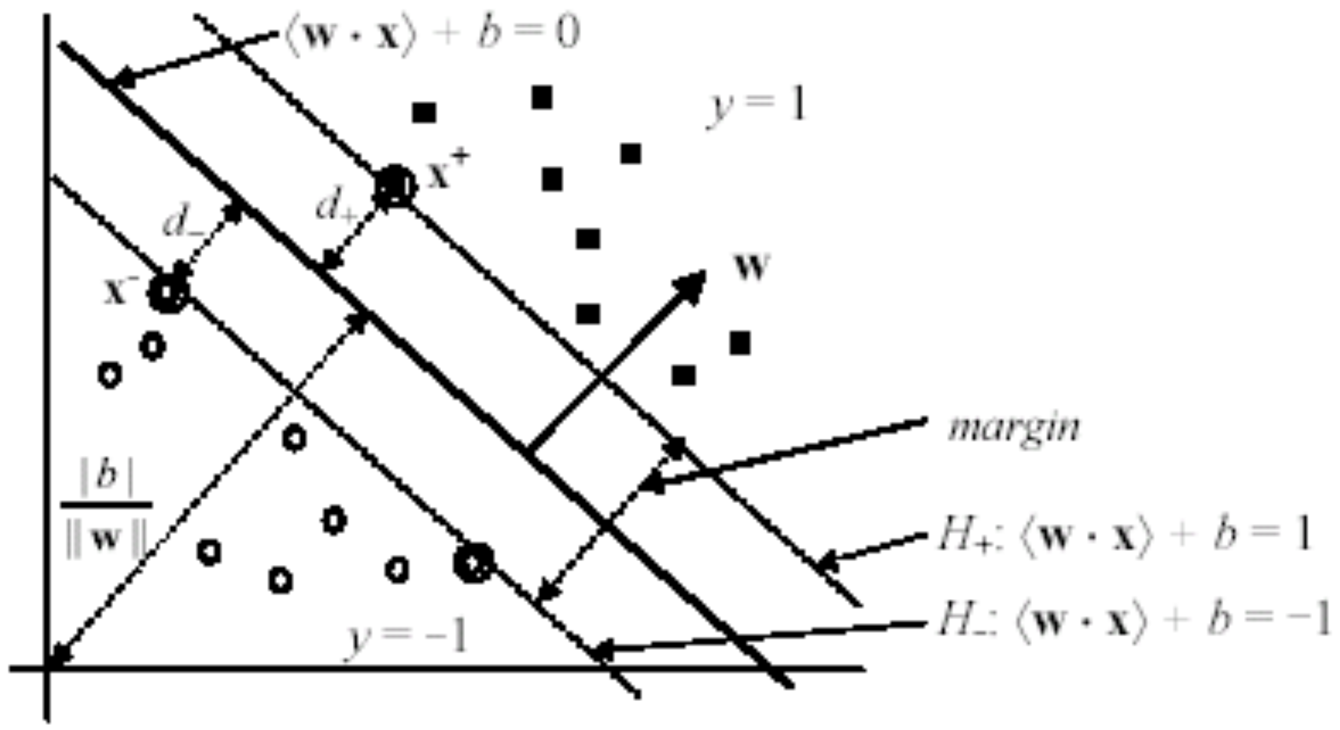




[Liu, 2006]

Hyperplane with max margin

- SVM selects the hyperplane with **max margin**.
- 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:

$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1, \quad \text{if } y_i = 1$$

$$\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1, \quad \text{if } y_i = -1 \qquad \text{[Eq.3]}$$

The margin (1)

- **Margin** 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

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

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

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

- As a result, the margin is:

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

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 the*

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

  - *And satisfy the below conditions for any training data $\mathbf{x_i}$:*

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

SVM: learning with max margin (2)

- Learning SVM is equivalent to the following minimization problem:
  - Minimize $$\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2}$$ [Eq.9]
  - Conditioned on $$\begin{cases} \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \geq 1, & if\ y_i = 1 \\ \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \leq -1, & if\ y_i = -1 \end{cases}$$
- Note, it can be reformulated as:
  - Minimize $$\frac{\langle \mathbf{w} \cdot \mathbf{w} \rangle}{2}$$ [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 $f(\mathbf{x})$ conditioned on $g(\mathbf{x}) = 0$

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

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

  - Where a is a *Lagrange multiplier*.

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

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

Constrained optimization (2)

- Consider the problem with inequality constraints:

  Minimize $f(\mathbf{x})$ conditioned on $g_i(\mathbf{x}) \leq 0$

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

$$\begin{cases} \dfrac{\partial}{\partial \mathbf{x}} \left(f(\mathbf{x}) + \sum_{i=1}^{r} \alpha_i g_i(\mathbf{x}) \right) \bigg|_{\mathbf{x}=\mathbf{x_0}} = 0 \\ g_i(\mathbf{x}) \leq 0 \end{cases};$$

  - Where $\alpha \geq 0$ 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*.
  - α is called *dual variable*.

SVM: learning with max margin (3)

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

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

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

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

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

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

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

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

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

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

$$\alpha_i \geq 0 \quad \text{[Eq.15]}$$

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

- 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 is 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:

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

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]

    (D)
  - Such that $$\begin{cases} \sum_{i=1}^{r} \alpha_i y_i = 0 \\ \alpha_i \geq 0, \forall i = 1..r \end{cases}$$
- The constraints in **(D)** is much more 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:

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

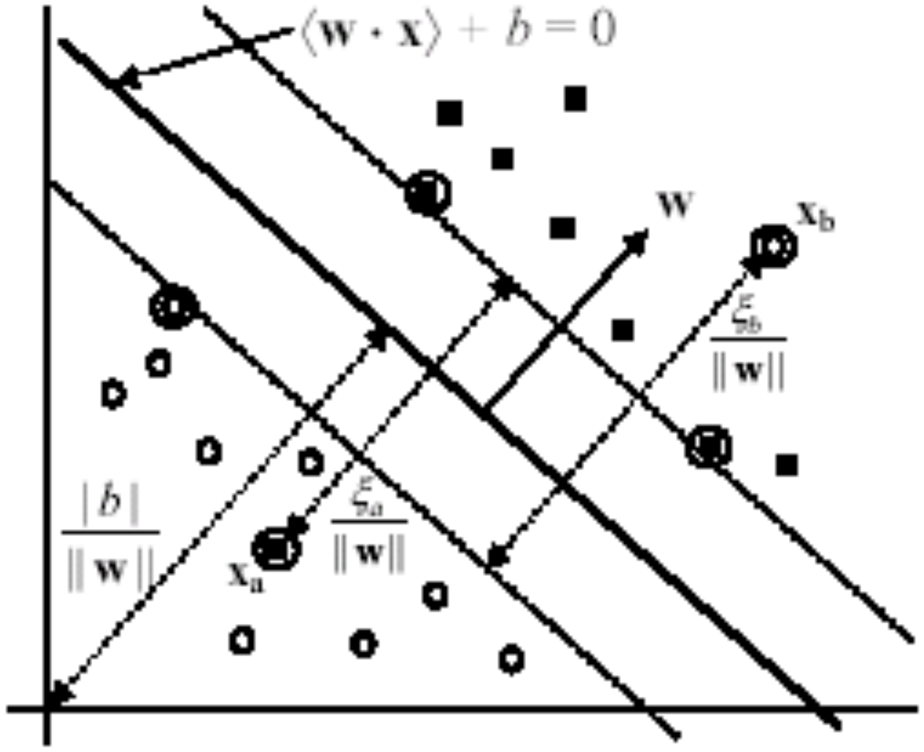
  - If the result is 1, **z** will be assigned to the possitive class; otherwise **z** will be assigned to the negative class.
- Note that this classification principle
  - Just depends on the support vectors.
  - Just needs to compute some dot products.

2. Soft-margin SVM

- *What if the two classes are not linearly separable?*
  - Linear separability is ideal in practice.
  - Data are often noisy or erronous, making two classes overlapping.
- 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):

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

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

  - 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:

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

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

Penalty on noises/errors

- We should enclose some information on noises/errors into the objective function when learning.
  - 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.
  - 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* $$\begin{cases} y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \quad \forall i = 1..r \\ \xi_i \geq 0, \quad \forall i = 1..r \end{cases}$$
- 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

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

- 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.

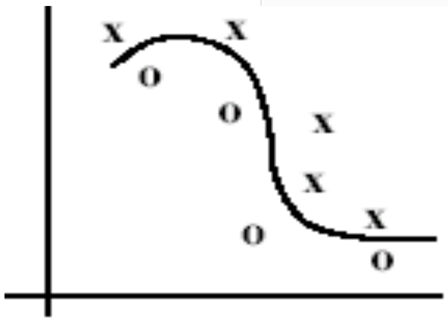
- *Note: it is important to choose a good value of C, since it significantly affects performance of SVM.*
  - We often use a validation set to choose a value for C.

Linear SVM: summary

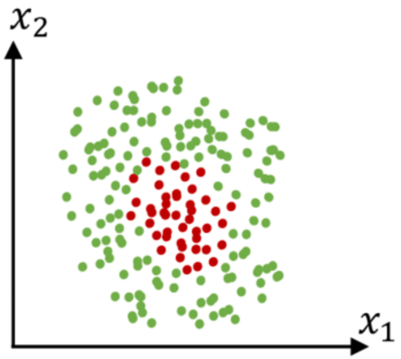
- Classification is based on a separating hyperplane.
- Such a hyperplane is represented as a combination of some support vectors.
- The determination of support vectors reduces to solve a quadratic programming problem.
- In the dual problem and the separating hyperplane, dot products can be used in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

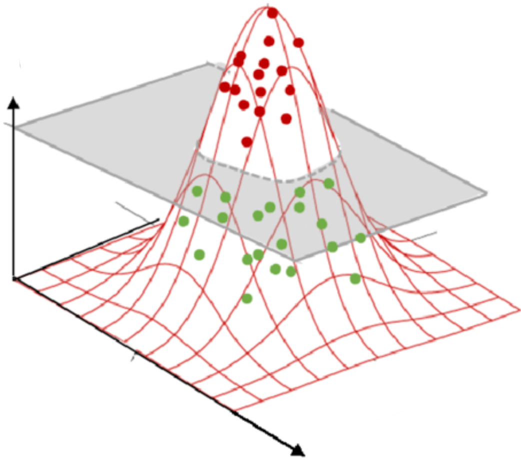
3. Non-linear SVM

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



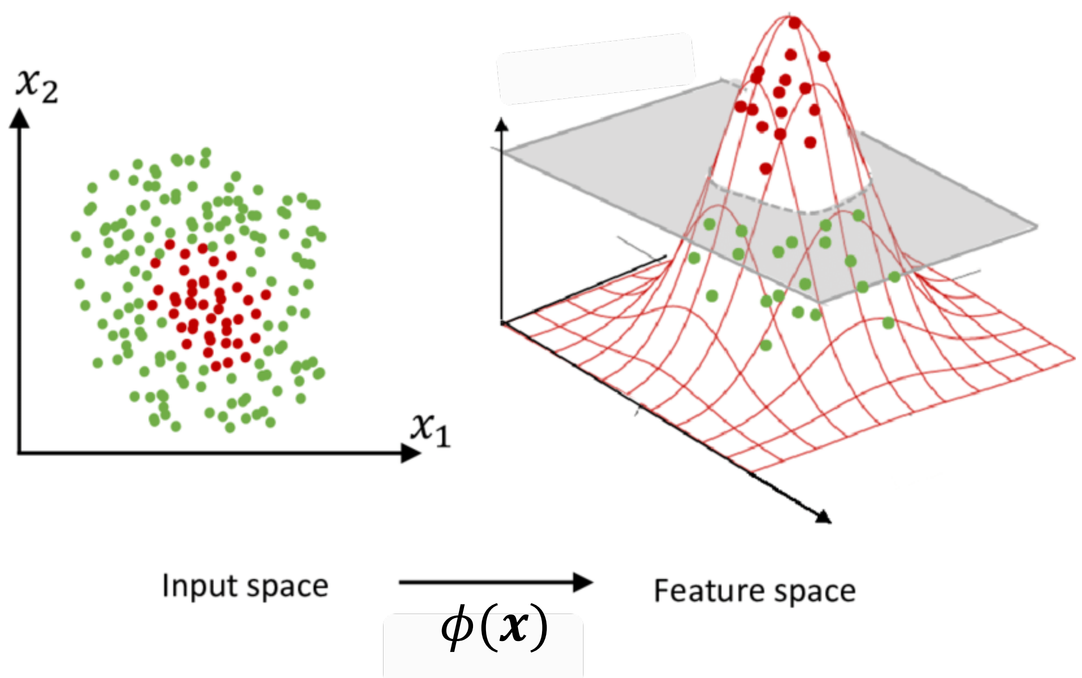
- How about using a non-linear function?
- **Idea of Non-linear SVM:**
  - **Step 1:** transform the input into another space, which often has *higher dimensions*, so that the projection of data is linearly separable
  - **Step 2:** use linear SVM in the new space





Non-linear SVM

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



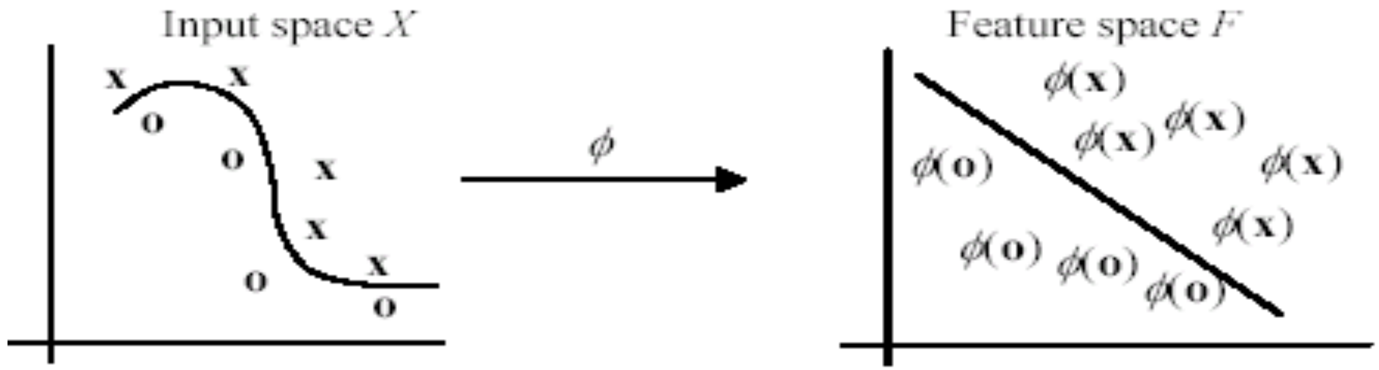
Non-linear SVM: transformation

- 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

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

- 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

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

Such that

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

- **The dual problem:**

Maximize

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

Such that

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

- **Classifier:**

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

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

  - 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

- 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.