THE STUDY OF KERNEL METHODS

WANG Wei
CS&E, CUHK

Outline

- Kernel functions
- Structure risk minimization
- Multiple kernel learning

Learning and Similarity

- Training set $(X_1, y_1), (X_2, y_2), \ldots (X_m, y_m) \in R^n \times Y$
- Generalization: Giving an unknown sample x to predict a suitable label y
- (X, y) should be similar to one of the classes
- How to calculate the similarity?

Similarity measurement

Similarity measurement:

- length of x: $\|x\| = \sqrt{(x \cdot x)}$ $\|x'\| = \sqrt{(x' \cdot x')}$
- distance of x and x':

$$\|x - x'\| = \sqrt{((x - x') \cdot (x - x'))} = \sqrt{(x \cdot x) - 2(x \cdot x') + (x' \cdot x')}$$

- cosine similarity: $\cos\beta = \frac{(x \cdot x')}{\|x\| \cdot \|x'\|} = \frac{(x \cdot x')}{\sqrt{(x \cdot x) \cdot (x' \cdot x')}}$

Dot product determines the similarity!

$$(x, x') = \sum_{i=1}^{n} x_i x_i'$$

Similarity Vs. Kernel

- Dot product is not sufficient
  - Input space is not a dot product space
  - More general similarity measurement by applying a map.

$$\Phi : \mathcal{X} \rightarrow \mathcal{H}$$

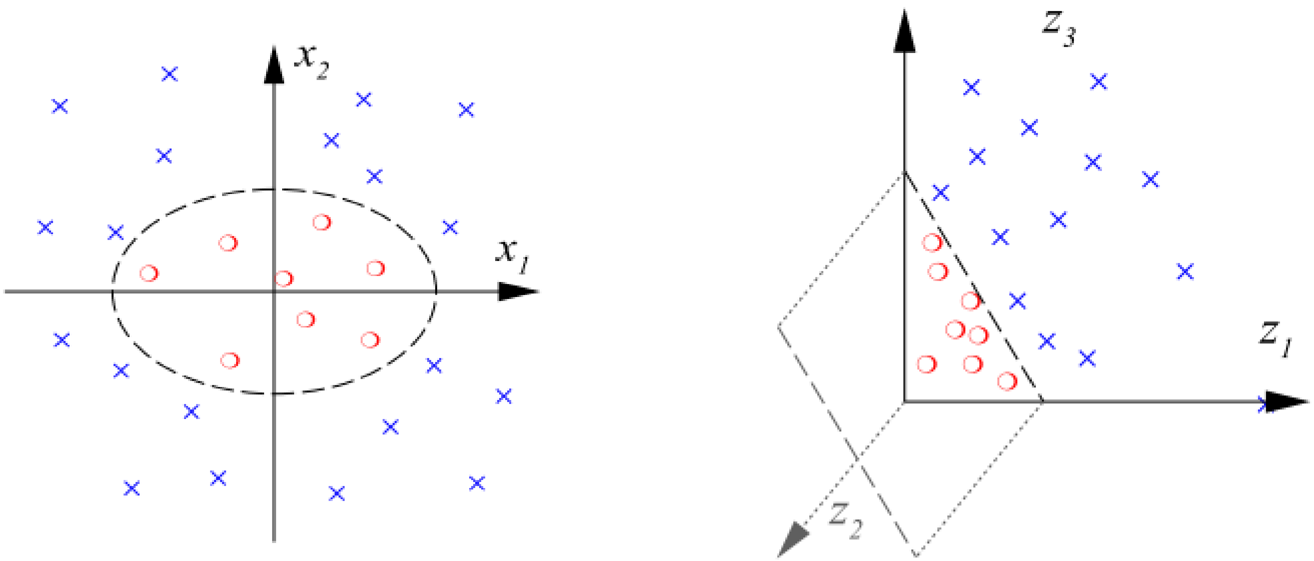
- $\mathcal{H}$ is called feature space or Hilbert space.

  Define a similarity measure from the dot product in $\mathcal{H}$

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle$$

Kernel trick

- Using a linear classifier algorithm to solve a non-linear problem by mapping the original non-linear observations into a higher-dimensional space



Kernel trick

- Nonlinear mapping:

$$\begin{aligned} \Phi : \mathcal{X} &\rightarrow \mathcal{H} \\ \mathbf{x} &\mapsto \Phi(\mathbf{x}) \end{aligned}$$

$$(\Phi(\mathbf{x}_1), y_1), \ldots, (\Phi(\mathbf{x}_n), y_n)$$

- The dot product can be computed in $\mathcal{H}$, without explicitly using or even knowing the mapping Φ.

Kernel trick

- Examples of common kernels:

Polynomial $k(x, x') = (\langle x, x' \rangle + c)^d$

Gaussian $k(x, x') = \exp(-\|x - x'\|^2/(2\sigma^2))$

Sigmoidal $\tanh(\kappa(\mathbf{x} \cdot \mathbf{y}) + \theta)$

- Any algorithm that only depends on the dot product can benefit from the kernel trick.
- Think of kernel as a nonlinear similarity measurement.

Structural Risk Minimization (SRM)

- $$R[f] \leq R_{emp}[f] + \sqrt{\frac{h\left(\ln \frac{2n}{h} + 1\right) - \ln(\delta/4)}{n}}$$

  Expected error

  Training error

  Complexity term

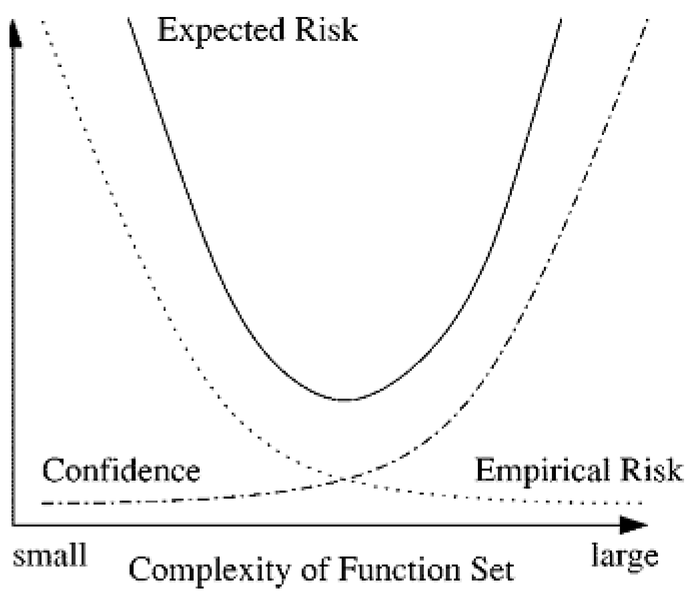
  $$complexity \propto \Phi\left(\frac{h}{n}\right)$$

- Training error reflects the accuracy of training set.

  $$R_{emp}[f] = \frac{1}{n}\sum_{i=1}^{\ell}(f(\mathbf{x}_i), y_i).$$

- A "simple" function that explains most of the data is preferable to a complex one (Occam's razor) .

Structural Risk Minimization (SRM)



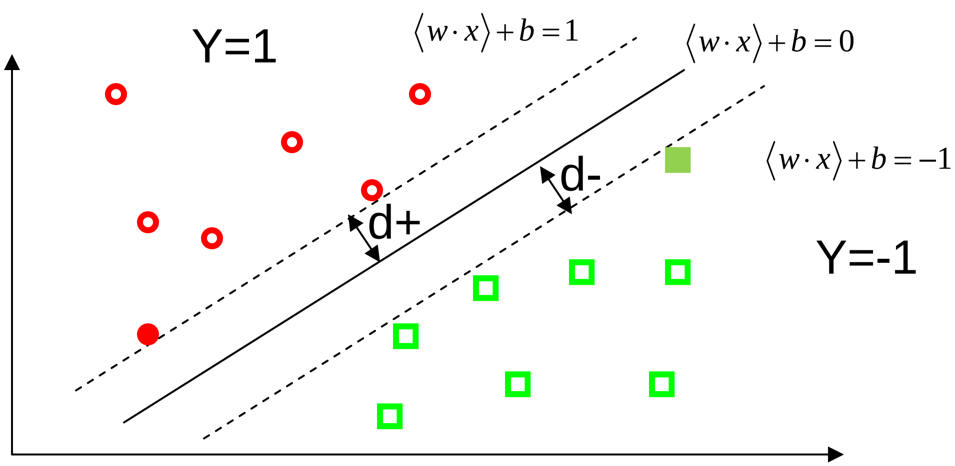
Find the best tradeoff between empirical error and complexity

- We cannot obtain the expected risk itself, we will minimized the bound.
- keep the empirical risk zero, while minimizing the complexity term.

Structural Risk Minimization

$$h \leq \Lambda^2 R^2 + 1 \quad \text{and} \quad \|\mathbf{w}\|_2 \leq \Lambda \Longrightarrow h \propto f\left(w^2\right)$$

where R is the radius of the smallest ball around the training data, R is fixed for a given data set.



Margin is the minimal distance of a sample to the decision surface

$$d_+ = d_- = \frac{1}{\|w\|}$$

Structural Risk Minimization

- Minimize the training error $\longrightarrow$ $y_i \cdot [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1$
- Minimize the complexity term $\longrightarrow$ minimize $\frac{1}{\|w\|^2}$

  $\longrightarrow$ maximize the margin
- The original problem:

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

$$subject\ \ to\ \ y_i\left(\left(W \cdot X_i\right)+b\right) \geq 1$$

Lagrange function

- Introduce a Lagrange multiplier $\alpha_i \geq 0$
- Lagrange:

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

- At the deviation, we have

$$\frac{\partial}{\partial b} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0, \quad \frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0,$$

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

and $\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i.$

⟹ Substitute both into L to get the *dual problem*

The Support Vector expansion

- Karush-Kuhn-Tucker conditions:

$$\frac{\partial L}{\partial w_j} = w_j - \sum_{i=1}^{n} y_i \alpha_i x_{ij} = 0 \qquad \frac{\partial L}{\partial b} = -\sum_{i=1}^{n} y_i \alpha_i = 0$$

$$y_i \left(\langle w \cdot x \rangle + b \right) \geq 1$$

$$\alpha_i \geq 0$$

$$\alpha_i \left(y_i \left(\langle w \cdot x_i \rangle + b \right) - 1 \right) = 0$$

$y_i \cdot [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] > 1 \Longrightarrow \alpha_i = 0 \Longrightarrow \mathbf{x}_i$ irrelevant

$y_i \cdot [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] = 1$ (on the margin) $\Longrightarrow \mathbf{x}_i$ Support Vector

Dual problem

- Dual: maximize $W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$

  subject to $\alpha_i \geq 0, \quad i = 1, \ldots, m, \quad \text{and} \quad \sum_{i=1}^{m} \alpha_i y_i = 0$

- By solving the dual optimization problem, one obtains the coefficients α_i and W can be solved by the value of it.
- The solution is determined by the examples on the margin

$$\begin{aligned} f(\mathbf{x}) &= \operatorname{sgn}(\langle \mathbf{x}, \mathbf{w} \rangle + b) \\ &= \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b\right) \end{aligned}$$

Kernel expressions

- Original problem:

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

$$\text{subject to } y_i((\mathbf{w}\cdot\Phi(\mathbf{x}_i))+b)\geq 1, \qquad i=1,\ldots,n.$$

- Dual problem:

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j \,\mathrm{k}(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{subject to} \quad \alpha_i \geq 0, i=1,\ldots,n,$$

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

Soft Margin SVMs

- C-SVM:

  for $C > 0$ minimize $\tau(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{m} \xi_i$

  subject to $y_i \cdot (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$ (margin $2/\|\mathbf{w}\|$)
- ξ_i is slack variable, which is used to relax the hard margin constraint.
- C determines the tradeoff between the empirical risk and the complexity term.

Soft Margin SVMs

- Dual problem:

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \, \mathrm{k}(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{subject to} \quad 0 \le \alpha_i \le C, \, i = 1, \ldots, n,$$

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

- KKT conditions:

$$\begin{aligned}
\alpha_i = 0 \quad &\Rightarrow \quad y_i f(\mathbf{x}_i) \ge 1 \quad \text{and} \quad \xi_i = 0 \\
0 < \alpha_i < C \quad &\Rightarrow \quad y_i f(\mathbf{x}_i) = 1 \quad \text{and} \quad \xi_i = 0 \\
\alpha_i = C \quad &\Rightarrow \quad y_i f(\mathbf{x}_i) \le 1 \quad \text{and} \quad \xi_i \ge 0.
\end{aligned}$$

Only when x_i is on the margin or inside the margin area, the corresponding α_i is nonzero.

Multiple Kernel Learning

- Using multiple kernels can improve performance

$$K(x,x') = \sum_{m=1}^{M} d_m K_m(x,x') \ , \quad \text{with } d_m \geq 0 \ , \ \sum_{m=1}^{M} d_m = 1$$

- K_m can simply be classical kernels with different parameters.

Algorithm for SimpleMKL

- Primal problem:

$$\begin{aligned} \min_{\{f_m\},b,\xi,d} \quad & \frac{1}{2}\sum_m \frac{1}{d_m}\|f_m\|^2_{\mathcal{H}_m} + C\sum_i \xi_i \\ \text{s.t.} \quad & y_i \sum_m f_m(x_i) + y_i b \geq 1 - \xi_i \quad \forall i \\ & \xi_i \geq 0 \quad \forall i \\ & \sum_m d_m = 1 \ , \quad d_m \geq 0 \quad \forall m \ , \end{aligned}$$

- Optimization Problem:

$$\min_d J(d) \quad \text{such that} \quad \sum_{m=1}^{M} d_m = 1, \ d_m \geq 0$$

$$J(d) = \begin{cases} \min_{\{f\},b,\xi} & \frac{1}{2}\sum_m \frac{1}{d_m}\|f_m\|^2_{\mathcal{H}_m} + C\sum_i \xi_i \quad \forall i \\ \text{s.t.} & y_i \sum_m f_m(x_i) + y_i b \geq 1 - \xi_i \\ & \xi_i \geq 0 \quad \forall i \ . \end{cases}$$

Algorithm for SimpleMKL

Algorithm 1 SimpleMKL algorithm

set $d_m = \frac{1}{M}$ for $m = 1, \ldots, M$
while stopping criterion not met **do**
  compute $J(d)$ by using an SVM solver with $K = \sum_m d_m K_m$
  compute $\frac{\partial J}{\partial d_m}$ for $m = 1, \ldots, M$ and descent direction D (14).
  set $\mu = \underset{m}{\operatorname{argmax}}\, d_m$, $J^\dagger = 0$, $d^\dagger = d$, $D^\dagger = D$
  while $J^\dagger < J(d)$ **do** {descent direction update}
    $d = d^\dagger$, $D = D^\dagger$
    $\nu = \underset{\{m|D_m<0\}}{\operatorname{argmin}} -d_m/D_m$, $\gamma_{\max} = -d_\nu/D_\nu$
    $d^\dagger = d + \gamma_{\max} D$, $D^\dagger_\mu = D_\mu - D_\nu$, $D^\dagger_\nu = 0$
    compute $J^\dagger$ by using an SVM solver with $K = \sum_m d^\dagger_m K_m$
  end while
  line search along D for $\gamma \in [0, \gamma_{\max}]$ {calls an SVM solver for each γ trial value}
  $d \leftarrow d + \gamma D$
end while

Check whether object value decreases or not

The maximum admissible step size

Special Acknowledgement

Haiqin