Pattern Recognition

Lecture 5

Support Vector Machines

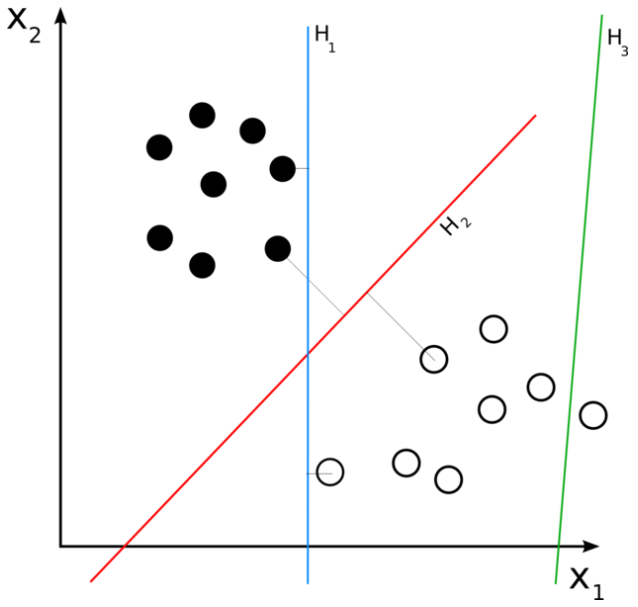
主要内容

- Hard Margin SVM
- Soft Margin SVM
- SMO 算法
- LS-SVM
- SVM类型算法的模型选择

Hard Margin SVM

- SVM的主要思想
  - 假设n个样本，$\{(x_1, y_1), \ldots, (x_n, y_n)\}$，其中$y_i \in \{-1, 1\}$ **线性可分**
  - 空间中存在无穷多个误差为零的判定面
  - 哪个判定面是最优的？

1. 训练误差为零的判定面
2. 分类间隔尽量的大



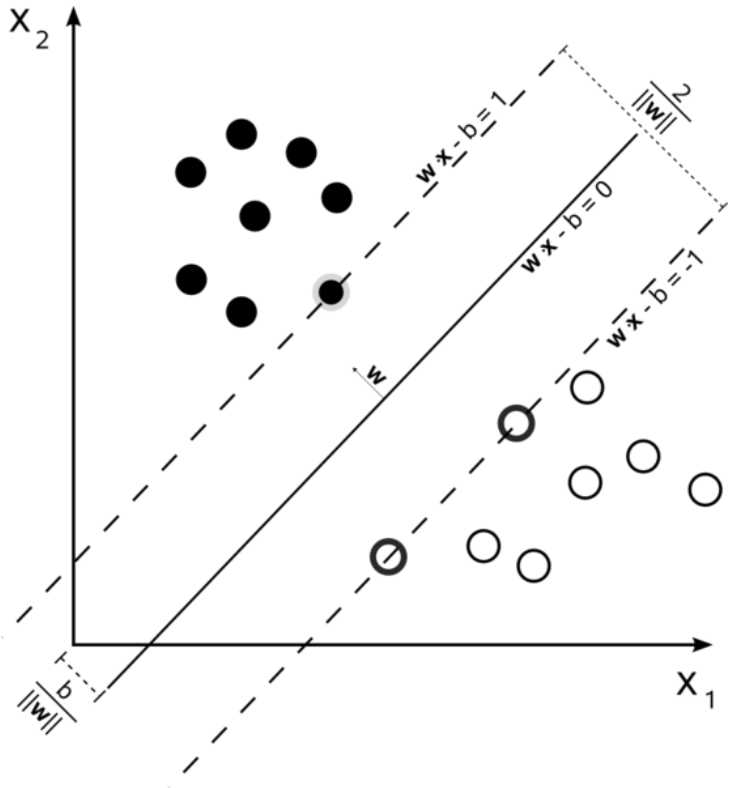
Hard Margin SVM

- 符号和定义
  - 判定函数形式： $g(\boldsymbol{x}) = \boldsymbol{w}^{\boldsymbol{t}}\boldsymbol{x} + b$
  - 因此，$y_i(w^t x_i + b) \geq m \quad \forall\, i$
  - 裕量（margin）：$\frac{2m}{||w||}$
  - 令m=1

- SVM问题的定义

凸优化问题

$$\min \frac{||\boldsymbol{w}||^2}{2}$$
$$\text{s.t.} \quad y_i(w^t x_i + b) \geq 1 \quad \forall\, i$$



Hard Margin SVM

- 如何求解？
  - 转化为拉格朗日乘子待定问题（问题1）

$$\min_{\mathbf{w}} \max_{\boldsymbol{\alpha}} \left\{ \frac{||\boldsymbol{w}||^2}{2} - \sum_{i=1}^{n} \alpha_i [y_i(\boldsymbol{w}^t \boldsymbol{x}_i + b) - 1] \right\}$$
$$\text{subject to } \alpha_i \geq 0, \forall\, i$$

二次规划问题

  - 可以看到，所有$y_i(\boldsymbol{w}^t \boldsymbol{x}_i + b) > 1$的$\boldsymbol{x}_i$对应的$\alpha_i$都为0
  - 所有$y_i(\boldsymbol{w}^t \boldsymbol{x}_i + b) = 1$的$\boldsymbol{x}_i$称为**支持向量**（support vector）

Hard Margin SVM

- 由 *Karush–Kuhn–Tucker* 条件，可知解的形式如下：

$$\boldsymbol{w^t} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x_i}$$

- 把问题1写成其对偶形式，可转化成问题2

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \boldsymbol{x_i^t x_j}$$

$$\text{subject to} \sum_{i=1}^{n} \alpha_i y_i = 0; \alpha_i \geq 0 \; \forall i$$

Hard Margin SVM

- 非线性SVM
  - Kernel trick

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \phi(\boldsymbol{x_i})^t \phi(\boldsymbol{x_j})$$

$$\text{subject to } \sum_{i=1}^{n} \alpha_i y_i = 0; \alpha_i \geq 0 \; \forall i$$

  - 通过一个适当的非线性映射$\phi(\boldsymbol{x})$，将数据由原始特征空间映射到一个新特征空间，然后在新空间中寻求最优（线性）判定面
  - 注意：不需要知道$\phi(\boldsymbol{x})$，而只须知道内积计算公式

Hard Margin SVM

- 如何选择非线性映射函数$\boldsymbol{\phi(x)}$
- 令 $\boldsymbol{\phi(x_i)^t\phi(x_j)}$ 为一个非线性核函数(kernel function)，对应变换空间被称为**核空间**(kernel space)
- 常用核函数
  - RBF

$$K(x_i, x_j) = \boldsymbol{\phi(x_i)^t\phi(x_j)} = \exp(-\sigma^2 \left|\left|\boldsymbol{x_i - x_j}\right|\right|^2)$$

  - Polynomial

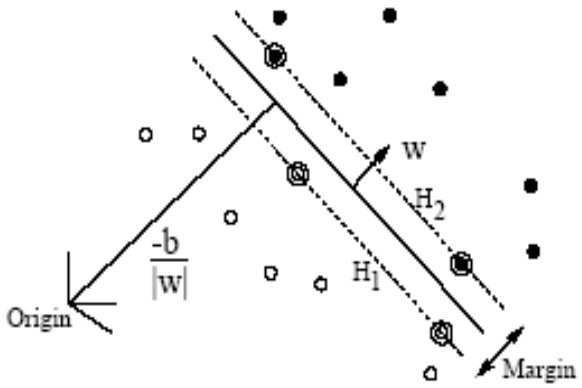
$$K(x_i, x_j) = \boldsymbol{\phi(x_i)^t\phi(x_j)} = \left(\boldsymbol{x_i^t x_j} - b\right)^p$$

  - Sigmoid

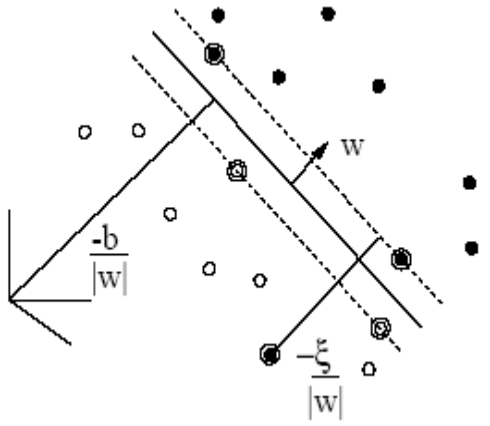
$$K(x_i, x_j) = \boldsymbol{\phi(x_i)^t\phi(x_j)} = \tanh(\boldsymbol{x_i^t x_j} - b)$$

Soft Margin SVM

- 实际问题很少线性可分。虽然理论上可通过非线性映射得到线性可分的数据，但如何获得这样的映射，且避免过拟合，是一个问题。
- 更实际的策略是允许一定误差



Hard Margin（硬间隔）



Soft Margin（软间隔）

Soft Margin SVM

- Soft margin SVM的基本形式

$$\min \frac{||w||}{2} + C\sum_i \varepsilon_i$$
$$s.t.\, y_i(w^t x_i - b) \geq 1 - \varepsilon_i, \varepsilon_i \geq 0, \forall\, i$$

- 转化成拉格朗日乘子待定问题（问题1）

$$\min_{\boldsymbol{w}} \max_{\boldsymbol{\alpha},\boldsymbol{\mu}} \left\{ \frac{||w||^2}{2} + C\sum_i \varepsilon_i - \sum_i \alpha_i [y_i(w^t x_i - b) - 1 + \varepsilon_i] - \sum_i \mu_i \varepsilon_i \right\}$$
$$s.t.\, \alpha_i \geq 0, \varepsilon_i \geq 0\ \forall i$$

Soft Margin SVM

- 同样，根据KKT条件可以把上述问题写为

$$\max_{\alpha}\left\{\sum_{i}\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \phi(x_i)^t\phi(x_j)\right\}$$
$$\text{s.t.}\sum_{i}\alpha_i y_i=0, 0\le\alpha_i\le C, \forall i$$

- α_i取值的意义

$$\alpha_i=0\Longrightarrow y_i g(x_i)>1$$
$$0<\alpha_i<C\Longrightarrow y_i g(x_i)>1$$
$$\alpha_i=C\Longrightarrow y_i g(x_i)<1$$

SMO 算法

- 优点
  - 可保证解的全局最优性，不存在陷入局部极小解的问题。
  - 分类器复杂度由支撑向量的个数，而非特征空间（或核空间）的维数决定，因此较少因维数灾难发生过拟合现象。
- 缺点
  - 需要求解二次规划问题，其规模与训练模式量成正比，因此计算复杂度高，且存储开销大，不适用于需进行在线学习/训练的大规模分类问题。

SMO 算法

- 无论Hard Margin或Soft Margin SVM，均可用经典的二次规划方法求解，但同时求解*n*个拉格朗日乘子涉及很多次迭代，计算开销太大，所以一般采用Sequential Minimal Optimization（SMO）算法（J. Platt, 1999）

- **基本思路**：每次只更新两个乘子，迭代获得最终解。

SMO 算法

- 假定在某一次迭代中，需要更新x_1, x_2对应的拉格朗日乘子α_1, α_2
- 这个小规模的二次规划问题写为

$$L_s = \max_{\alpha}\{(\alpha_1+\alpha_2)+\sum_{i=3}^{n}\alpha_i-\frac{1}{2}\left\|\alpha_1 y_1\phi(\boldsymbol{x_1})+\alpha_1 y_1\phi(\boldsymbol{x_1})+\sum_{i=3}^{n}\alpha_i y_i\phi(\boldsymbol{x_i})\right\|^2\}$$

$$s.t.\,\alpha_1 y_1+\alpha_2 y_2=-\sum_{i=3}^{n}\alpha_i y_i\,, 0<\alpha_i<C, \forall i$$

SMO 算法

- **算法框架：**
  for *i*=1:*iter*
    a. **根据预先设定的规则**，从所有样本中选出两个
    b. **保持其他拉格朗日乘子不变**，更新所选样本对应的拉格朗日乘子
  end

- **优点：**只有两个变量的二次规划问题存在解析解。
- **关键技术细节：**
  1. 在每次迭代中，怎样更新乘子？
  2. 怎么选择每次迭代需要更新的乘子？

SMO 算法

- 更新拉格朗日乘子α_1, α_2
  - 步骤1：计算上下界L和H
    - $L = \max(0, \alpha_2^{old} - \alpha_1^{old}), H = \min(C, C + \alpha_2^{old} - \alpha_1^{old}), if\ y_1 \neq y_2$
    - $L = \max(0, \alpha_2^{old} + \alpha_1^{old} - C), H = \min(C, \alpha_2^{old} + \alpha_1^{old}), if\ y_1 = y_2$
  - 步骤2：计算L_s的二阶导数
    - $\eta = 2\boldsymbol{\phi}(\boldsymbol{x_1})^t\boldsymbol{\phi}(\boldsymbol{x_2}) - \boldsymbol{\phi}(\boldsymbol{x_1})^t\boldsymbol{\phi}(\boldsymbol{x_1}) - \boldsymbol{\phi}(\boldsymbol{x_2})^t\boldsymbol{\phi}(\boldsymbol{x_2})$
  - 步骤3：更新L_s

$$\alpha_2^{new} = \alpha_2^{old} - \frac{y_2(e_1 - e_2)}{\eta}$$

$$e_i = g^{old}(\boldsymbol{x_i}) - y_i$$

SMO 算法

- 步骤4：计算变量α_2

$$\alpha^{temp} = \begin{cases} H, & \text{if}\alpha_2^{new} \geq H \\ \alpha_2^{new}, & \text{if } L \leq \alpha_2^{new} \leq H \\ L, & \text{if } \alpha_2^{new} \leq L \end{cases}$$

- 步骤5：更新α_1

$$\alpha_1^{new} = \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha^{temp})$$

SMO 算法

- 选择需要更新的乘子
  - 两个基本原则
    - 使乘子能满足KKT条件
    - 对一个满足KKT条件的乘子进行更新，应能最大限度增大目标函数的值（类似于梯度下降）
  - 步骤1：先“扫描”所有乘子，把第一个违反KKT条件的作为更新对象，令为α_2
  - 步骤2：在所有不违反KKT条件的乘子中，选择使$|e_1 - e_2|$最大的α_1
  - 由于所有乘子可分为三种情况，当还存在违反KKT条件的乘子时，在扫描时可忽略在0和C之间的乘子，以加速扫描进程。

SMO 算法

- 一个实际的例子

Training Set Size	SMO Time (CPU sec)	PCG Time (CPU sec)	Non-Bound SVs	Bound SVs	SMO Iterations	PCG Iterations
2477	26.3	64.9	439	43	10838	1888
3470	44.1	110.4	544	66	13975	2270
4912	83.6	372.5	616	90	18978	5460
7366	156.7	545.4	914	125	27492	5274
9888	248.1	907.6	1118	172	29751	5972
17188	581.0	3317.9	1780	316	42026	9413
24692	1214.0	6659.7	2300	419	55499	14412
49749	3863.5	23877.6	3720	764	93358	24235

Table 12.6 SMO and PCG Chunking for a Gaussian SVM on the Web data set.

LS-SVM

- LS-SVM是经典SVM的变体，Suykens和Vandewalle1999年提出

$$\min_{w}\left\{\frac{||w||^2}{2}+C\sum_{i}||\varepsilon_i||^2\right\}$$

$$s.t.\, y_i(w^t x_i+b)=1-\varepsilon_i, \forall i$$

- 与SVM的关键不同是约束条件由不等式变为了等式
- 对应的优化问题：

$$\min_{\mathbf{w,b,e}}\left\{\frac{1}{2}\sum_{i,j}\alpha_i,\alpha_j K(\boldsymbol{x_i},\boldsymbol{x_j})+C\sum_{i}\left[1-y_i\sum_{j}\alpha_j K\left(\boldsymbol{x_i},\boldsymbol{x_j}\right)-y_i b\right]^2\right\}$$

$$s.t.\, y_i\left(\sum_{j}\alpha_j K\left(\boldsymbol{x_i},\boldsymbol{x_j}\right)+b\right)=1-\varepsilon_i, \forall i$$

LS-SVM

- 不等式变为等式后，拉格朗日乘子可以通过解线性系统得到

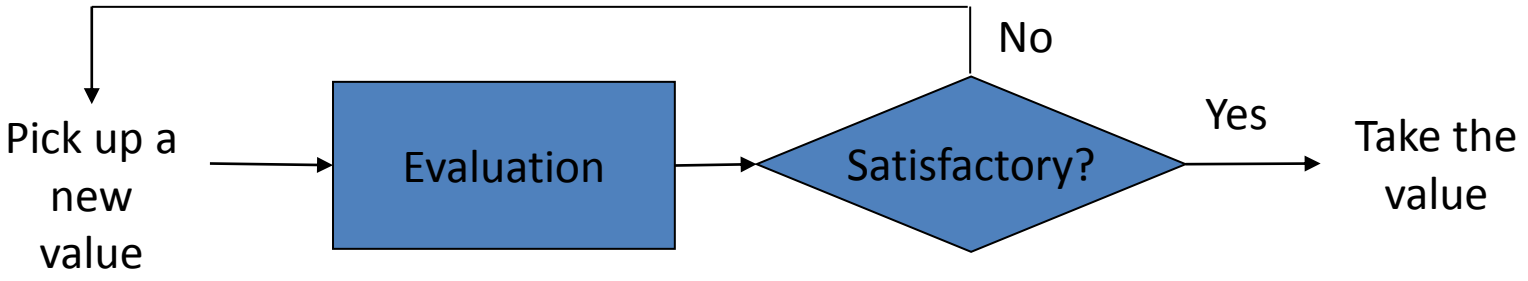
$$\begin{bmatrix} \Omega + C^{-1}\mathrm{I}_n & Y \\ Y^t & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{1_n} \\ 0 \end{bmatrix}$$

$$\Omega_{ij} = y_i y_j K(x_i, x_j)$$

- 解线性系统远易于解二次规划问题。但相应的，绝大部分拉格朗日乘子都将不为零。也就是说，LS-SVM分类器的测试时间将长于SVM分类器

SVM类型算法的模型选择

- **问题定义：**SVM类型的算法需要预设一些参数，例如核函数的参数（kernel parameter）、正则化参数（regularization parameter）等。它们一般又被称为hyper-parameter。
- 给定一种求解算法，这些参数的选取直接决定最终解的质量。这又是一个优化问题，且一般无解析解。
- 如何选取：试错



SVM类型算法的模型选择

- 把参数调整视为一个优化问题，则该问题的解可以表达为一个向量（parameter vector）的形式，该向量每一个元素对应算法的一个参数。
- **求解套路：**
  1. 定义一种**目标函数**，来描述函数的泛化性能，如验证误差、交叉验证误差等
  2. 利用现有的搜索算法求解，如
     a. 网格搜索（难以解决3维以上的问题）
     b. 梯度下降（要求核函数可导）
     c. 演化算法（维度方面不及梯度下降，但可处理不可导的核函数）

SVM类型算法的模型选择

- **一种简单的基于演化算法的参数调整方法：**

For i=1:iter

1. 随机生成多个参数向量（解）
2. 在目标函数上验证解的质量
3. 根据解的质量由好到坏进行排序。取出其中较好的一部分（例如，一半）解，在这些解的每一个元素上加上一个随机数，从而得到一些新解
4. 把新解和老解比较，取出最好的一部分，作为下一次迭代的初始解

end

SVM类型算法的模型选择

- 简单的性能比较

Dataset	Grid-G	Grad-G	Evo-G	Grad-FSG	Evo-FSG
Banana	10.79 ± 0.61	10.61 ± 0.67	10.72 ± 0.65	**10.59 ± 0.55**	**10.59 ± 0.61**
B-Cancer	26.77 ± 4.52	27.17 ± 4.63	26.31 ± 4.36	26.99 ± 4.52	**26.28 ± 4.47**
Diabetis	23.49 ± 1.81	23.47 ± 1.81	23.74 ± 1.86	**23.38 ± 1.77**	23.60 ± 1.82
Heart	16.41 ± 3.55	16.70 ± 3.51	16.75 ± 3.60	16.61 ± 3.87	**16.33 ± 3.68**
Image	2.84 ± 0.67	2.87 ± 0.67	2.84 ± 0.55	2.79 ± 0.62	**2.05 ± 0.53**
Splice	10.89 ± 0.72	**10.80 ± 0.63**	10.88 ± 0.71	10.84 ± 0.58	10.93 ± 0.70
Thyroid	5.11 ± 2.27	5.03 ± 1.89	5.09 ± 2.19	4.83 ± 2.07	**4.55 ± 2.19**
Waveform	10.01 ± 0.56	10.08 ± 0.56	9.90 ± 0.53	9.85 ± 0.50	**9.70 ± 0.52**

Table 2: Comparisons of model selection approaches: Grid search with standard Gaussian kernel (Grid-G), Gradient-based method with standard Gaussian kernel (Grad-G) and feature-scaling Gaussian kernel (Grad-FSG), Evolutionary-based method with standard Gaussian kernel (Evo-G) and feature-scaling kernel (Evo-FSG).

SVM 工具包

- LIBSVM： 国立台湾大学林智仁（Chih-Jen Lin）

http://www.csie.ntu.edu.tw/~cjlin/libsvm/index.html