模式识别重点

费政聪

作业详细解答，历年试卷及解答请点击

贝叶斯最小风险决策：

Condition risk

$$R(\alpha_i|\mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j) P(\omega_j|\mathbf{x})$$

Minimum risk decision (Bayes decision)

$$\arg\min_i R(\alpha_i \mid x)$$

最小错误率决策：

- Zero-one loss

$$\lambda(\alpha_i|\omega_j) = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases} \qquad i, j = 1, \ldots, c$$

$$\begin{aligned} R(\alpha_i|\mathbf{x}) &= \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j) P(\omega_j|\mathbf{x}) \\ &= \sum_{j \neq i} P(\omega_j|\mathbf{x}) \\ &= 1 - P(\omega_i|\mathbf{x}) \end{aligned}$$

- Minimum error decision: Maximum a posteriori (MAP)

$$\text{Decide } \omega_i \text{ if } P(\omega_i|\mathbf{x}) > P(\omega_j|\mathbf{x}) \qquad \text{for all } j \neq i$$

带拒识的决策：

- Formulation (Problem 13, Chapter 2)
  - C+1 classes

$$\lambda(\alpha_i \mid \omega_j) = \begin{cases} 0, & i = j \\ \lambda_s, & i \neq j \\ \lambda_r, & \text{reject} \end{cases} \qquad \lambda_r < \lambda_s$$

$$R(\alpha_i|\mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j) P(\omega_j|\mathbf{x})$$

$$\Rightarrow \quad R_i(\mathbf{x}) = \begin{cases} \lambda_s[1 - P(\omega_i \mid \mathbf{x})], & i = 1, \ldots, c \\ \lambda_r, & \text{reject} \end{cases}$$

$$\arg\min_i R_i(\mathbf{x}) = \begin{cases} \arg\max_i P(\omega_i \mid \mathbf{x}), & \text{if } \max_i P(\omega_i \mid \mathbf{x}) > 1 - \lambda_r / \lambda_s \\ \text{reject}, & \text{otherwise} \end{cases}$$

决策面：求的时候让两类决策函数相等。

一维高斯密度函数：

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$$

多维高斯密度函数：

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu)^t \Sigma^{-1}(\mathbf{x}-\mu)\right]$$

高斯密度判别函数：

- 判别函数 $g_i(\mathbf{x}) = \ln p(\mathbf{x}|\omega_i) + \ln P(\omega_i)$

$$p(\mathbf{x}\,|\,\omega_i) = \frac{1}{(2\pi)^{d/2}\,|\,\Sigma_i\,|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu_i)\Sigma_i^{-1}(\mathbf{x}-\mu_i)\right]$$

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\boldsymbol{\Sigma}_i| + \ln P(\omega_i)$$

三种情况讨论：

Case 1: $\Sigma_i=\sigma^2 I$

$$g_i(\mathbf{x}) = \mathbf{w}_i^t\mathbf{x} + w_{i0}$$

$$\mathbf{w}_i = \frac{1}{\sigma^2}\boldsymbol{\mu}_i \qquad w_{i0} = \frac{-1}{2\sigma^2}\boldsymbol{\mu}_i^t\boldsymbol{\mu}_i + \ln P(\omega_i)$$

决策函数：

决策面：

$$\mathbf{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_i + \boldsymbol{\mu}_j) - \frac{\sigma^2}{\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|^2}\ln\frac{P(\omega_i)}{P(\omega_j)}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)$$

位置移向先验概率小的类别

Case 2: $\Sigma_i= \Sigma$

线性判别函数！ $g_i(\mathbf{x}) = \mathbf{w}_i^t\mathbf{x} + w_{i0}$

$$\mathbf{w}_i = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i \qquad w_{i0} = -\frac{1}{2}\boldsymbol{\mu}_i^t\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i + \ln P(\omega_i)$$

– 二类决策面 $g_i(\mathbf{x}) = g_j(\mathbf{x})$

$$\Longrightarrow \quad \mathbf{w}^t(\mathbf{x}-\mathbf{x}_0) = 0 \qquad \mathbf{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)$$

$$\mathbf{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_i + \boldsymbol{\mu}_j) - \frac{\ln\,[P(\omega_i)/P(\omega_j)]}{(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^t\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)$$

以上两种情况都是线性判别函数，因为二次项系数相同，被消去。

Case 3: Σ_i= arbitrary

决策面是二次。

贝叶斯分类错误率：

$$\begin{aligned} P(error) &= P(\mathbf{x} \in \mathcal{R}_2, \omega_1) + P(\mathbf{x} \in \mathcal{R}_1, \omega_2) \\ &= P(\mathbf{x} \in \mathcal{R}_2|\omega_1)P(\omega_1) + P(\mathbf{x} \in \mathcal{R}_1|\omega_2)P(\omega_2) \\ &= \int_{\mathcal{R}_2} p(\mathbf{x}|\omega_1)P(\omega_1)\,d\mathbf{x} + \int_{\mathcal{R}_1} p(\mathbf{x}|\omega_2)P(\omega_2)\,d\mathbf{x}. \end{aligned}$$

高斯函数参数的最大似然估计：

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{k=1}^{n}(\mathbf{x}_k - \hat{\boldsymbol{\mu}})(\mathbf{x}_k - \hat{\boldsymbol{\mu}})^t$$

最大似然估计和贝叶斯估计的区别：

参数固定值；参数视为随机变量。

期望最大法求混合高斯：

1. Choose an initial set of parameters for Θ^{old}
2. Do
   E-step: Evaluate p(Z|X, Θ^{old})
   M-step: Update parameters
   $$\Theta^{new} = \arg\max_{\Theta} Q(\Theta, \Theta^{old})$$
   If convergence condition is not satisfied
   $$\Theta^{old} \leftarrow \Theta^{new}$$
3. End

隐马尔可夫的基本成分和含义：

$$\boldsymbol{\lambda} = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$$

状态转移矩阵，观测矩阵，初始状态

Parzen 窗和 K-NN 区别：

Parzen window: 固定局部区域体积V，k变化

k-nearest neighbor: 固定局部样本数k，V变化

窗函数估计概率密度：

– 以x为中心、体积为 $V_n = h_n^d$ 的局部区域内样本数

$$k_n = \sum_{i=1}^{n} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)$$

– 概率密度估计k_n/nV_n

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{V_n}\varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)$$

K-NN 估计后验概率：

– k_i NNs from class i $\quad k = \sum_{i=1}^{c} k_i$

$$p_n(\mathbf{x}, \omega_i) = \frac{k_i/n}{V}$$

$$P_n(\omega_i|\mathbf{x}) = \frac{p_n(\mathbf{x}, \omega_i)}{\sum_{j=1}^{c} p_n(\mathbf{x}, \omega_j)} = \frac{k_i}{k}$$

感知器准则基本思想：

- **感知准则函数－基本思想**
  – 考虑如下准则函数：
  $$J_p(\mathbf{a}) = \sum_{\mathbf{y}\in Y}(-\mathbf{a}^T\mathbf{y}),$$ 其中，Y为错分样本集合
  – 当 $\mathbf{y}$ 被错分时， $\mathbf{a}^T\mathbf{y} \le 0$，则 $-\mathbf{a}^T\mathbf{y} \ge 0$。因此 $J_p(\mathbf{a})$ 总是大于等于0。在可分情形下，当且仅当 Y 为空集时 $J_p(\mathbf{a})$ 将等于零，这时将不存在错分样本。
  – 因此，目标是最小化 $J_p(\mathbf{a})$: $\min_{\mathbf{a}} J_p(\mathbf{a})$

感知器准则基本算法：

Batch Perceptron—基本算法

1 begin initialize: **a**, η, certain θ(small value), k=0
2 do $k \leftarrow k$+1
3 $\mathbf{a} = \mathbf{a} + \eta_k \sum_{\mathbf{y} \in Y(k)} \mathbf{y}$ // $Y(k) = Y_k$
4 until $|\eta_k \sum \mathbf{y}| < \theta$, $\mathbf{y} \in Y_k$ // 一个较松的停止条件
5 return **a**
6 end

样本松弛算法：

Batch Relaxation with Margin

1 begin initialize: **a**, b η_0, k=0
2 do $k \leftarrow k$+1 (mod n)
3 Y_k={}, $j = 0$
4 do $j \leftarrow j + 1$
5 if $\mathbf{a}^T\mathbf{y}_j \le b$, then append $\mathbf{y}_j$ to Y_k //如果错分
6 until $j = n$
7 $\mathbf{a}_{k+1} = \mathbf{a}_k - \eta_k \sum_{\mathbf{y} \in Y} \left((\mathbf{a}^T\mathbf{y} - b) / \| \mathbf{y} \|^2 \right) \cdot \mathbf{y}$,
8 until Y_k={}
9 return **a**
10 end

最小误差（MSE）准则：

$$J_s(\mathbf{a}) = \| \mathbf{e} \|^2 = \| \mathbf{Ya} - \mathbf{b} \|^2 = \sum_{i=1}^{n} (\mathbf{a}^T\mathbf{y}_i - b_i)^2$$

$$\frac{\partial J_s(\mathbf{a})}{\partial \mathbf{a}} = \sum_{i=1}^{n} 2(\mathbf{a}^T\mathbf{y}_i - b_i)\mathbf{y}_i = 2\mathbf{Y}^T(\mathbf{Ya} - \mathbf{b})$$

Widrow-Hoff 算法：

Widrow-Hoff (Least mean squared) Approach

1 begin initialize: **a**, **b**, η, threshold θ, k=0
2 do $k \leftarrow k$+1 (mod n)
3 $\mathbf{a} = \mathbf{a} + \eta_k (b_k - (\mathbf{a}_k)^T \mathbf{y}^k) \mathbf{y}^k$
4 until $\| (b_k - (\mathbf{a}_k)^T \mathbf{y}^k) \mathbf{y}^k \| < \theta$
5 return **a**
6 end

三层网络 BP 推导（注意目标函数等会改变）：

$$\Delta w_{ih} = -\eta \sum_{k,j} \frac{\partial E}{\partial z_j^k} \frac{\partial z_j^k}{\partial net_j^k} \frac{\partial net_j^k}{\partial y_h^k} \frac{\partial y_h^k}{\partial net_h^k} \frac{\partial net_h^k}{\partial w_{ih}}$$

BP 算法：

Stochastic Backpropagation

1 begin initialize: n_H, **w**, η, criterion θ, k=0
2 do $k \leftarrow k+1 \pmod{n}$
3 $\mathbf{x}^k$, randomly chosen a sample (pattern)
4 $w_{hj} \leftarrow w_{hj} + \eta\delta_j^k y_h^k, \quad w_{ih} \leftarrow w_{ih} + \eta\delta_h^k x_i^k$
5 until $\| \nabla J(\mathbf{w}) \| < \theta$
6 return **w**
7 end

自组织映射基本原理：

- 通过自动寻找样本中的内在规律和本质属性，自组织、自适应地改变网络参数与结构。
- 其自组织功能是通过竞争学习来实现的。
- **竞争学习规则—Winner-Take-All（胜者为王）**
  - 网络的输出神经元之间相互竞争并期望被激活；
  - 在每一时刻只有一个输出神经元被激活；
  - 被激活的神经元称为竞争获胜神经元，其它神经元的状态被抑制。

自组织映射学习算法：

- S_1：网络初始化－通常采用随机初始化方法
- S_2：输入向量
- S_3：计算映射层的权重向量和输入向量的距离：

$$d_j = \sqrt{\sum_{i=1}^{d}(x_i - w_{ij})^2}$$

- S_4：选择与权重向量的距离最小的神经元**（确定胜者）**
  - 计算并选择使输入向量和权重向量的距离最小的神经元，将其作为胜出神经元 (j^*)，并给出其邻接神经元集合 $h(.,j^*)$。
- S_5：调整权重
  - 胜出神经元和其邻接神经元的权重，按下式更新：

$$\Delta w_{ij} = \eta h(j, j^*)(x_i - w_{ij})$$
$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}$$

- S_6：检查是否达到预先设定的要求。
  - 如达到要求则算法结束；否则返回S_2，进入下一轮学习。

卷积神经网络权重数目计算：

- 图像大小：200×200
- 第一隐含层滤波器（卷积核）大小：5×5
- 第一隐含层结点集群（图像）个数：4
- 第一隐含层图像大小：196 ×196
- 第一隐含层pooling窗口大小：2×2
- 第一隐含层pooling之后图像大小：98×98

- 第二层滤波器（卷积核）大小：3×3
- 第二隐含层结点集群(图像）个数：7
- 第二隐含层滤波器总数：28 ← (也可以理解为7个3×3 ×4的三维滤波器)
- 第二隐含层图像大小：96 ×96
- 第二隐含层pooling窗口大小：2×2
- 第二隐含层pooling之后图像大小：48×48

K 均值聚类的假设：

- 各类出现的先验概率均相等；
- 每个均本点以概率为1属于一个类（后验概率0-1近似）；

K 均值聚类算法：

K-Means Clustering—Algorithm 1

1 begin initialization n, c, $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$,…, $\boldsymbol{\mu}_c$.
2 do classify n samples according to nearest $\boldsymbol{\mu}_i$
3 re-compute $\boldsymbol{\mu}_i$
4 until no change in $\boldsymbol{\mu}_i$
5 return $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$,…, $\boldsymbol{\mu}_c$

高斯混合密度角度理解 K 均值聚类：

$$P(\omega_i|x_k,\hat{\mu}) = \begin{cases} 1, & x_k \in \omega_i \\ 0, & x_k \notin \omega_i \end{cases} \tag{1}$$

$$\hat{P}(\omega_i) = \frac{n_i}{n}, \tag{2}$$

$$\hat{\mu}_i = \frac{1}{n_i}\sum_{k=1}^{n_i} x_k^{(i)}, \tag{3}$$

$$\hat{\Sigma}_i = \frac{1}{n_i}\sum_{k=1}^{n_i}(x_k^i - \hat{\mu}_i)(x_k^i - \hat{\mu}_i)^T \tag{4}$$

此时样本属于哪一类需要计算 $\|x_k - \hat{\mu}_i\|^2$ 来判断。因此需要通过迭代来得到 c 个高斯成分的均值。测试过程中，以这些均值作为 c 个类 (簇) 的类中心，计算每个样本点到类中心的欧氏距离，将样本点归入到距离最近的类。从而完成 K-均值聚类的计算工作。

常用的聚类准则：

均方误差准则，散度准则，行列式准则

分级聚类和系统树：

- **例子3：请按最小距离准则对如下6个样本进行分级聚类：**

$$\mathbf{x}_1 = (0, 3, 1, 2, 0)$$
$$\mathbf{x}_2 = (1, 3, 0, 1, 0)$$
$$\mathbf{x}_3 = (3, 3, 0, 0, 1)$$
$$\mathbf{x}_4 = (1, 1, 0, 2, 0)$$
$$\mathbf{x}_5 = (3, 2, 1, 2, 1)$$
$$\mathbf{x}_6 = (4, 1, 1, 1, 0)$$

解：将每个样本单独看成一类，计算各点对之间的距离见下表：

	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$	$\mathbf{x}_6$
$\mathbf{x}_1$	0					
$\mathbf{x}_2$	$\sqrt{3}$	0				
$\mathbf{x}_3$	$\sqrt{15}$	$\sqrt{6}$	0			
$\mathbf{x}_4$	$\sqrt{6}$	$\sqrt{5}$	$\sqrt{13}$	0		
$\mathbf{x}_5$	$\sqrt{11}$	$\sqrt{8}$	$\sqrt{6}$	$\sqrt{7}$	0	
$\mathbf{x}_6$	$\sqrt{21}$	$\sqrt{14}$	$\sqrt{8}$	$\sqrt{11}$	$\sqrt{4}$	0

基于上述矩阵，根据最小距离准则，应将 $\mathbf{x}_1$和$\mathbf{x}_2$合并为一类，得到 $G_1=\{\mathbf{x}_1,\mathbf{x}_2\}$，$\{\mathbf{x}_3\}$，$\{\mathbf{x}_4\}$，$\{\mathbf{x}_5\}$，$\{\mathbf{x}_6\}$。按最小距离原则重新计算各类之间的距离，见下表：

	G_1	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$	$\mathbf{x}_6$
G_1	0				
$\mathbf{x}_3$	$\sqrt{6}$	0			
$\mathbf{x}_4$	$\sqrt{5}$	$\sqrt{13}$	0		
$\mathbf{x}_5$	$\sqrt{8}$	$\sqrt{6}$	$\sqrt{7}$	0	
$\mathbf{x}_6$	$\sqrt{14}$	$\sqrt{8}$	$\sqrt{11}$	$\sqrt{4}$	0

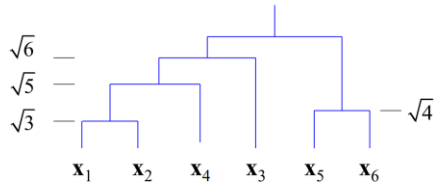
基于上述矩阵，根据最小距离准则，应将 $\mathbf{x}_5$和$\mathbf{x}_6$合并为一类，得到 $G_1=\{\mathbf{x}_1,\mathbf{x}_2\}$，$\{\mathbf{x}_3\}$，$\{\mathbf{x}_4\}$，$G_2=\{\mathbf{x}_5,\mathbf{x}_6\}$。按最小距离原则重新计算各类之间的距离，见下表：

	G_1	$\mathbf{x}_3$	$\mathbf{x}_4$	G_2
G_1	0			
$\mathbf{x}_3$	$\sqrt{6}$	0		
$\mathbf{x}_4$	$\sqrt{5}$	$\sqrt{13}$	0	
G_2	$\sqrt{8}$	$\sqrt{6}$	$\sqrt{7}$	0

基于上述矩阵，根据最小距离准则，应将 G_1和$\mathbf{x}_3$合并为一类，得到 $G_3=G_1\cup\{\mathbf{x}_4\}$，$\{\mathbf{x}_3\}$，$G_2=\{\mathbf{x}_5,\mathbf{x}_6\}$。按最小距离原则重新计算各类之间的距离，见下表：

	G_3	$\mathbf{x}_3$	G_2
G_3	0		
$\mathbf{x}_3$	$\sqrt{6}$	0	
G_2	$\sqrt{7}$	$\sqrt{6}$	0

基于上述矩阵，根据最小距离准则，应将 G_3和$\mathbf{x}_3$合并为一类，当然也可以将 G_2 和 $\mathbf{x}_3$合并为一类。如选择前者，得到 $G_4=G_3\cup\{\mathbf{x}_3\}$，$G_2=\{\mathbf{x}_5,\mathbf{x}_6\}$。最后，$G_3$和$G_2$合并为一类，这样所有数据将被合并在一起。



系统树图

谱聚类基本流程：

- 利用点对之间的相似性，构建亲和度矩阵；
- 构建拉普拉斯矩阵；
- 求解拉普拉斯矩阵最小的特征值对应的特征向量（通常舍弃零特征所对应的分量全相等的特征向量）；
- 由这些特征向量构成样本点的新特征，采用K-means等聚类方法完成最后的聚类。

常见的集成算法：

- **Bagging**
- **Random subspace （随机子空间）**
- **Boosting/adaboost**
- **随机森林（讲了决策树之后才能展开）**

Adaboost 基本思想：

加法、前向分步，指数损失。

- **关于第一个问题：**
  - Adaboost的做法是：**提高那些被前一轮弱分类器分错的样本的权重，降低已经被正确分类的样本的权重。**错分的样本将在下一轮弱分类器中得到更多关注。于是分类问题被一系列弱分类器**“分而治之”**。
- **关于第二个问题：**
  - 关于弱分类器的组合，Adaboost的做法是：采用加权(多数)表决的方法。具体地，加大分类错误率较小的弱分类器的权重，使其在表决中起更大的作用。

Adaboost 基本流程：

- (1) 初始化训练数据的权值分布

$$D_1=\{w_{11}, w_{12}, \ldots, w_{1n}\},\ w_{1i}=1/n,\ i=1,\ldots,n$$

- (2) 对 $m=1, 2, \ldots, M$
- (2a) 使用具有权值分布 D_m 的训练数据，学习基本分类器

$$G_m(\mathbf{x}):\ \boldsymbol{X}\rightarrow\{-1,+1\}$$

- (2b) 计算 $G_m(\mathbf{x})$ 在训练数据集上的分类错误率**(加权)**：

$$e_m=P(G_m(\mathbf{x}_i)\neq y_i)=\sum_{i=1}^{n} w_{mi}\ I(G_m(\mathbf{x}_i)\neq y_i)$$

真值函数

- (2c) 计算 $G_m(\mathbf{x})$ 的**贡献系数**：

$$\alpha_m=\frac{1}{2}\ln\frac{1-e_m}{e_m}$$

- (2d) 更新训练数据集的权重分布：

$$D_{m+1}=\{w_{m+1,1}, w_{m+1,2}, \ldots, w_{m+1,n}\}$$

- (3) 构建基本分类器的线性组合：

$$f(\mathbf{x})=\sum_{m=1}^{M}\alpha_m G_m(\mathbf{x})$$

对于两类分类问题，得到最终的分类器：

$$G(\mathbf{x})=sign\big(f(\mathbf{x})\big)=sign\left(\sum_{m=1}^{M}\alpha_m G_m(\mathbf{x})\right)$$

SVM 基本形式：

Hard margin：

$$\arg\max_{\mathbf{w},b} \arg\min_{\mathbf{x}_i \in D} \frac{|b+\mathbf{x}_i \cdot \mathbf{w}|}{\sqrt{\sum_{i=1}^{d} w_i^2}}$$
$$\text{subject to } \forall \mathbf{x}_i \in D : y_i\left(\mathbf{x}_i \cdot \mathbf{w}+b\right) \geq 0$$

$$\arg\min_{\mathbf{w},b} \sum_{i=1}^{d} w_i^2$$
$$\text{subject to } \forall \mathbf{x}_i \in D : y_i\left(\mathbf{x}_i \cdot \mathbf{w}+b\right) \geq 1$$

Soft margin：

$$\{\vec{w}^*, b^*\} = \min_{\vec{w},b} \sum_{i=1}^{d} w_i^2 + c\sum_{j=1}^{N} \varepsilon_j$$
$$y_1\left(\vec{w}\cdot\vec{x}_1+b\right) \geq 1-\varepsilon_1, \varepsilon_1 \geq 0$$
$$y_2\left(\vec{w}\cdot\vec{x}_2+b\right) \geq 1-\varepsilon_2, \varepsilon_2 \geq 0$$
$$\ldots$$
$$y_N\left(\vec{w}\cdot\vec{x}_N+b\right) \geq 1-\varepsilon_N, \varepsilon_N \geq 0$$

Hard margin dual:

$$\max.\ W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1,j=1}^{n} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$
$$\text{subject to } \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

Soft margin dual:

$$\max.\ W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1,j=1}^{n} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$
$$\text{subject to } C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

Hinge loss 的含义：分类正确对没有影响，分类错误根据距离体现错误程度。

数据点位置和参数 a 的关系：

$$\begin{cases} \alpha_i = 0 & \Rightarrow y_i f(x_i) \geq 1 \Rightarrow \textit{Samples outside the boundary} \\ 0 < \alpha_i < C & \Rightarrow y_i f(x_i) = 1 \Rightarrow \textit{Samples on the boundary} \\ \alpha_i = C & \Rightarrow y_i f(x_i) \leq 1 \Rightarrow \textit{Samples within the boundary} \end{cases}$$

Kernel 的基本思想：

General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:

Kernel trick 的基本思想：

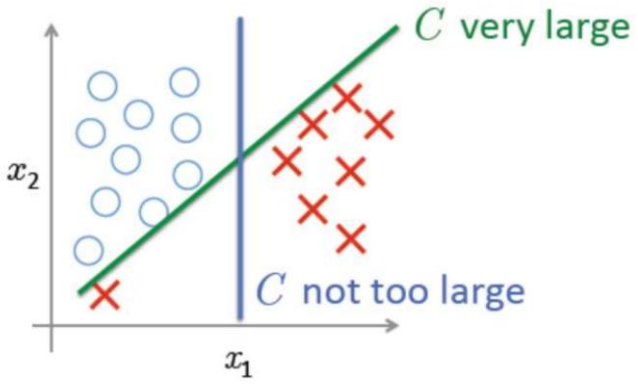
Since data is only represented as dot products, we need not do the mapping explicitly.

SVM 的缺点：

- Training (Testing) is quite slow compared to ANN
  - Because of Constrained Quadratic Programming
- Essentially a binary classifier
  - However, there are some tricks to evade this.
- Very sensitive to noise. Why?
  - A few off data points can completely throw off the algorithm
- Biggest Drawback: The choice of Kernel function.
  - There is no "set-in-stone" theory for choosing a kernel function for any given problem (still in research...)
  - Once a kernel function is chosen, there is only ONE modifiable parameter, the error penalty *C*.

参数 C 对模型的影响：

一般 C 取 0.1，模型的泛化能力较强。



信息增益计算：

定义 5.2（信息增益） 特征 A 对训练数据集 D 的信息增益 $g(D,A)$，定义为集合 D 的经验熵 $H(D)$ 与特征 A 给定条件下 D 的经验条件熵 $H(D|A)$ 之差，即

$$g(D,A)=H(D)-H(D|A) \tag{5.6}$$

一般地，熵 $H(Y)$ 与条件熵 $H(Y|X)$ 之差称为互信息（mutual information）. 决策树学习中的信息增益等价于训练数据集中类与特征的互信息.

C4.5 和 ID3 的区别：

- 用信息增益率来选择属性，克服了用信息增益选择属性时偏向选择取值多的属性的不足
- 在树构造过程中进行剪枝
- 能够完成对连续属性的离散化处理
- 能够对不完整数据进行处理

CART 基本思想：

决策树的生成就是递归地构建二叉决策树的过程. 对回归树用平方误差最小化准则，对分类树用基尼指数（Gini index）最小化准则，进行特征选择，生成二叉树.

剪枝的基本方法：

Decision Tree Pruning

- Prune while building tree (stopping early)
- Prune after building tree (post pruning)

剪枝算法的流程：

Post-Pruning (Reduced-Error Pruning)

- Grow decision tree to its entirety
- Split data into training and validation set
- Do until further pruning is harmful:
  - Evaluate impact on validation set of pruning each possible node (plus those below it)
  - Greedily remove the one that most improves validation set accuracy
- Produces smallest version of most accurate subtree.

随机森林的基本思想：

Random Forest，顾名思义，Random就是随机抽取，Forest就是说这里不止一棵树，而由一群决策树组成的一片森林，连起来就是用随机抽取的方法训练出一群决策树来完成分类任务。

RF用了两次随机抽取，一次是对训练样本的随机抽取；另一次是对变量（特征）的随机抽取。这主要是为了解决样本数量有限的问题。

PCA 算法基本流程：

- Compute the mean of the data

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

- Compute the sample covariance matrix (using the mean subtracted data)

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top$$

- Do the eigenvalue decomposition of the $D \times D$ matrix $\mathbf{S}$
- Take the top K eigenvectors (corresponding to the top K eigenvalues)
- Call these $\mathbf{u}_1, \ldots, \mathbf{u}_K$ (s.t. $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_{K-1} \geq \lambda_K$)
- $\mathbf{U} = [\mathbf{u}_1\ \mathbf{u}_2\ \ldots\ \mathbf{u}_K]$ is the projection matrix of size $D \times K$
- Projection of each example $\mathbf{x}_n$ is computed as $\mathbf{z}_n = \mathbf{U}^\top \mathbf{x}_n$
  - $\mathbf{z}_n$ is a $K \times 1$ vector (also called the embedding of $\mathbf{x}_n$)

LDA 学习的准则和优化结果：

$$FDR = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}$$

$$g(x) = (\mu_1 - \mu_2)^T S_w^{-1} x + w_0$$

LDA 的假设：

- ✓ 每一个类是单模态高斯分布 → 多模态LDA
- ✓ 每一个类的协方差矩阵都相同 → 异方差LDA
- ✓ 降维维数不能超过C-1
- ✓ Class separation problem

四种线性降维区别：

- **PCA** (unsupervised)
  - optimal reconstruction subspace
  - RBM, AutoEncoder, PCANet, NMF ...
- **CCA** (paired data)
  - optimal correlation subspace
- **LDA** (supervised)
  - optimal classification subspace under assumptions
- **ICA** (unsupervised)
  - independent subspace for signal separation

举例非线性降维并说明基本思想：

- Nonlinearize a linear dimensionality reduction method. E.g.:
  - **Kernel PCA (nonlinear PCA)**
- Using manifold based methods. E.g.:
  - **Locally Linear Embedding (LLE)**
  - **Isomap**

降维和特征选择的区别：

- Dimensionality reduction
  - All original features are used
  - The transformed features are linear combinations of the original features
- Feature selection
  - Only a subset of the original features are selected