《机器学习》读书笔记

黄奕诚

目录

1 绪论

1.1 引言

- 机器学习致力于研究如何通过计算的手段，利用经验来改善系统自身的性能。
- 机器学习研究的主要内容：在计算机上从数据中产生“模型”的算法，即“学习算法”。

1.2 基本术语

- 数据集（data set）：一组记录的集合
- 示例（instance）/样本（sample）：每条记录，即关于一个事件或对象的描述
- 属性（attribute）/特征（feature）：反映事件或对象在某方面的表现或性质的事项
- 属性值（attribute value）：属性上的取值
- 属性空间（attribute space）/样本空间（sample space）/输入空间：属性张成的空间，记为 $\mathcal{X}$
- 特征向量（feature vector）：一个示例（在样本空间对应的坐标向量）
- 学习（learning）/训练（training）：从数据中学得模型的过程
- 训练数据（training data）：训练过程中使用的数据
- 训练样本（training sample）：训练数据中的每个样本
- 训练集（training set）：训练样本组成的集合
- 假设（hypothesis）：对应了关于数据的某种潜在规律的学得模型
- 真实（ground-truth）：潜在规律自身
- 学习器（learner）：学习算法在给定数据和参数空间上的实例化
- 标记（label）：关于示例结果的信息

- 样例（example）：拥有标记信息的示例
- 标记空间（label space）/输出空间：所有标记的集合，记为 $\mathcal{Y}$
- 分类（classification）：预测的是离散值的学习任务（二分类 $\mathcal{Y} = \{-1, +1\}$或$\{0, 1\}$；三分类 $|\mathcal{Y}| > 2$）
- 回归（regression）：预测的是连续值的学习任务（$\mathcal{Y} = \mathbb{R}$）
- 测试（testing）：使用学得模型进行预测的过程
- 测试样本（testing sample）：被预测的样本
- 无监督学习（unsupervised learning）：训练数据中没有标记信息的学习任务，代表是聚类（clustering）
- 监督学习（supervised learning）：训练数据中具有标记信息的学习任务，代表是分类和回归
- 泛化（generalization）能力：学得模型适用于新样本的能力

1.3 假设空间

- “从样例中学习”是一个归纳的过程。
- 可以把学习过程看作一个在所有假设组成的空间中进行搜索的过程，搜索目标是找到与训练集“匹配”（fit）的假设。
- 假设空间可以表示为一课属性值中通配符逐渐被具体数值取代的树。
- 可以用许多策略对假设空间进行搜索，如自顶向下（从一般到特殊）、自底向上（从特殊到一般）。
- 可能有多个假设与训练集一致，即存在着一个与训练集一致的“假设集合”，称之为“版本空间”（version space）。

1.4 归纳偏好

- 多个与训练集一致的假设所对应的模型在面临新样本时，可能产生不同的输出。而对于一个具体的学习算法而言，必须要产生一个模型。此时学习算法本身的偏好会起到关键的作用。

- 归纳偏好（inductive bias）：机器学习算法在学习过程中对某种类型假设的偏好。
- 奥卡姆剃刀（Occam's razor）：若有多个假设与观察一致，则选最简单的那个。【常用的、自然科学研究中最基本的原则】
- 设 f 为希望学习的真实目标函数，则基于训练数据 X 的算法 $\mathfrak{L}_a$ 在训练集之外的所有样本上的误差与学习算法无关，即

$$\sum_f E_{ote}(\mathfrak{L}_a|X,f) = \sum_f E_{ote}(\mathfrak{L}_b|X,f)$$

"没有免费的午餐"定理（NFL 定理）：所有学习算法的期望性相同。

1.5 发展历程

1. 二十世纪五十年代到七十年代初："推理期"——赋予机器逻辑推理能力
2. 二十世纪七十年代中期开始："知识期"
   a. 机械学习（信息存储与检索）
   b. 示教学习（从指令中学习）
   c. 类比学习（通过观察和发现学习）
   d. 归纳学习（从样例中学习）
      - 符号主义学习（决策树、基于逻辑的学习）
      - 连接主义学习（神经网络）
      - 统计学习（支持向量机、核方法）
      - 深度学习

1.6 应用现状

- 计算机科学诸多分支学科领域（如计算机视觉、自然语言处理）
- 交叉学科（如生物信息学）
- 数据挖掘（机器学习领域和数据库领域是数据挖掘的两大支撑）
- 人类日常生活（天气预报、搜索引擎、自动驾驶、政治选举等）
- 促进人们理解"人类如何学习"

2 模型评估与选择

2.1 经验误差与过拟合

- 设在 m 个样本中有 a 个样本分类错误，则错误率（error rate）为 $E = a/m$，精度（accuracy）为 $1 - a/m$。
- 误差（error）：学习器的实际预测输出与样本的真实输出之间的差异。训练误差（training error）/经验误差（empirical error）：学习器在训练集上的误差。泛化误差（generalization error）：学习器在新样本上的误差。想要使泛化误差最小，而新样本未知，所以努力使经验误差最小化。
- 过拟合（overfitting）：学习器将训练样本自身的一些特点当作为所有潜在样本都会具有的一般性质。【关键障碍、无法彻底避免】欠拟合（underfitting）：学习器对训练样本的一般性质尚未学好。【较容易克服】若 “P≠NP”，过拟合就不可避免。

2.2 评估方法

为了对学习器对泛化误差进行评估，需要使用一个测试集（testing set）来测试学习器对新样本的判别能力，然后以测试集上的测试误差（testing error）作为泛化误差的近似。

若当前只有一个包含 m 个样例的数据集

$$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$$

，则对其进行适当的处理，从中产生训练集 S 和测试集 T。

2.2.1 留出法

直接将数据集 D 划分为两个互斥的集合，其中一个作为训练集 S，另一个作为测试集 T，即 $D = S \cup T, S \cap T = \emptyset$，在 S 上训练出模型后，用 T 来评估其测试误差，作为对泛化误差的估计。

- 划分尽可能保持数据分布的一致性，例如在分类任务中至少要保持样本的类别比例相似（分层采样）。

- 一般采用若干次随机划分、重复进行实验评估后取平均值作为评估结果。
- S 和 D 大小权衡没有完美的解决方案，常见做法是 2/3∼4/5 的训练样本比例。

2.2.2 交叉验证法

将数据集 D 划分为 k 个大小相似的互斥子集，即

$$D = D_1 \cup D_2 \cup \cdots \cup D_k, D_i \cup D_j = \emptyset (i \neq j)$$

每个子集 D_i 都尽可能保持数据分布的一致性（分层抽样）。然后从中选取 $k-1$ 个子集为训练集，剩下一个子集为测试集。可进行 k 次训练和测试，最终返回 k 个测试结果的均值。也称为“k 折交叉验证”（k-fold cross validation）。

- k 最常用的取值是 10，常用的还有 5、20 等。
- 留一法（Leave-One-Out）不受随机样本划分的影响，评估结果比较准确，但计算开销大。

2.2.3 自助法

以自助采样法（bootstrap sampling）为基础，给定包含 m 个样本的数据集 D，每次随机从 D 中挑选一个样本，将其拷贝放入 D'，再将该样本放回初始数据集 D 中。这个过程重复执行 m 次后，得到了包含 m 个样本的数据集 D'。此时将 D' 用作训练集，$D \backslash D'$ 用作测试集。

- D 有约 36.8% 的样本未出现在采样数据集 D' 中。
- 亦称为“包外估计”（out-of-bag estimate）。
- 自助法在数据集较小、难以有效划分训练/测试集时很有用，且能从初始数据集中产生多个不同的训练集。
- 因为自助法产生的数据集改变了初始数据集的分布，会引入估计偏差。

2.2.4 调参与最终模型

- 常用的调参做法：对每个参数选定一个范围和变化步长，进行计算开销和性能估计之间的折中。
- 在模型选择完成后，学习算法和参数配置已选定，此时用数据集 D 重新训练模型，使用所有 m 个样本，得到最终提交给用户的模型。

2.3 性能度量

给定样例集

$$D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$$

其中 y_i 是示例 $\boldsymbol{x}_i$ 的真实标记。要评估学习器 f 的性能，即把学习器预测结果 $f(\boldsymbol{x})$ 与真实标记 y 进行比较。

回归任务中最常用的性能度量：“均方误差”（mean squared error）

$$E(f; D) = \frac{1}{m}\sum_{i=1}^{m}(f(\boldsymbol{x}_i) - y_i)^2$$

更一般地，对于数据分布 $\mathcal{D}$ 和概率密度函数 $p(\cdot)$，均方误差可描述为

$$E(f; \mathcal{D}) = \int_{x\sim\mathcal{D}}(f(\boldsymbol{x}) - y)^2 p(\boldsymbol{x})d\boldsymbol{x}$$

对于分类任务——

2.3.1 错误率与精度

- 分类错误率

$$E(f; D) = \frac{1}{m}\sum_{i=1}^{m}\mathbb{I}(f(\boldsymbol{x}_i) \neq y_i)$$

精度

$$\mathrm{acc}(f; D) = \frac{1}{m}\sum_{i=1}^{m}\mathbb{I}(f(\boldsymbol{x}_i) = y_i) = 1 - E(f; D)$$

- 对于数据分布 $\mathcal{D}$ 和概率密度函数 $p(\cdot)$，错误率

$$E(f;\mathcal{D}) = \int_{x\sim\mathcal{D}} \mathbb{I}(f(\boldsymbol{x}) \neq y)p(\boldsymbol{x})d\boldsymbol{x}$$

精度

$$\text{acc}(f;\mathcal{D}) = \int_{x\sim\mathcal{D}} \mathbb{I}(f(\boldsymbol{x}) = y)p(\boldsymbol{x})d\boldsymbol{x} = 1 - E(f;\mathcal{D})$$

2.3.2 查准率、查全率与 $F1$

真实情况	预测结果	
	正例	反例
正例	TP（真正例）	FN（假反例）
反例	FP（假正例）	TN（真反例）

查准率（precision）

$$P = \frac{TP}{TP + FP}$$

查全率（recall）

$$R = \frac{TP}{TP + FN}$$

- 平衡点（Break-Even Point，BEP）：$R = P$ 时的取值，数值越高可以认为学习器越优。

- $F1$ 度量

$$F1 = \frac{2 \times P \times R}{P + R} = \frac{2 \times TP}{\text{样例总数} + TP - TN}$$

实际上 $F1$ 是 R 和 P 的调和平均

$$\frac{1}{F1} = \frac{1}{2}(\frac{1}{P} + \frac{1}{R})$$

- F_β 度量：考虑 R 与 P 的不同偏好，设 β 为查全率 R 对查准率 P 的相对重要性，则

$$F_\beta = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R}$$

实际上 F_β 是加权调和平均

$$\frac{1}{F_\beta} = \frac{1}{1 + \beta^2}(\frac{1}{P} + \frac{\beta^2}{R})$$

- 宏 $F1$：在各混淆矩阵上分别计算出各自的 (P_i, R_i)，再计算平均值：

$$\text{macro}-P = \frac{1}{n}\sum_{i=1}^{n} P_i$$

$$\text{macro}-R = \frac{1}{n}\sum_{i=1}^{n} R_i$$

$$\text{macro}-F1 = \frac{2 \times \text{macro}-P \times \text{macro}-R}{\text{macro}-P + \text{macro}-R}$$

- 微 $F1$：先将各混淆矩阵的对应元素进行平均得到四个指标，再基于这些平均值计算 $F1$：

$$\text{micro}-P = \frac{\overline{TP}}{\overline{TP} + \overline{FP}}$$

$$\text{micro}-R = \frac{\overline{TP}}{\overline{TP} + \overline{FN}}$$

$$\text{micro}-F1 = \frac{2 \times \text{micro}-P \times \text{micro}-R}{\text{micro}-P + \text{micro}-R}$$

△ 混淆矩阵介绍：每一列代表了预测类别，每一列的总数表示预测为该类别的数据的数目；每一行代表了数据的真实归属类别，每一行的数据总数表示该类别的数据实例的数目。例如共有 150 个样本数据，预测为 1、2、3 类各 50 个，分类结束后得到的混淆矩阵为

		预测		
		类 1	类 2	类 3
实际	类 1	43	2	0
	类 2	5	45	1
	类 3	2	3	49

2.3.3 ROC 与 AUC

ROC 全称是“受试者工作特征”(Receiver Operating Characteristic)曲线。横轴为“假正例率”(FPR)，纵轴为“真正例率”(TPR)。

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{TN + FP}$$

- 现实任务中 ROC 曲线的绘制方法：给定 m^+ 个正例和 m^- 个反例，根据学习器预测结果对样例进行排序，然后把分类阈值设为最大，此时 FPR 和 TPR 都为 0. 在坐标 $(0, 0)$ 处标记一个点，然后将分类阈值依次设为每个样例的预测值。设当前一个标记点坐标为 (x, y)，若当前为真正例，则对应标记点坐标为 $(x, y + \frac{1}{m^+})$；若当前为假正例，则对应标记点坐标为 $(x + \frac{1}{m^-}, y)$，然后用线段连接相邻点即得。

- AUC (Area Under ROC Curve) 即为 ROC 曲线下各部分的面积之和。设 ROC 曲线是由坐标为 $\{(x_i, y_i)|1 \leq i \leq m\}$ 的点按序连接而成，则 AUC 可估算为

$$\text{AUC} = \frac{1}{2} \sum_{i=1}^{m-1} (x_{i+1} - x_i) \cdot (y_i + y_{i+1})$$

- 给定 m^+ 个正例和 m^- 个反例，令 D^+ 和 D^- 分别表示正、反例集合，则排序“损失”(loss) 定义为

$$\ell_{rank} = \frac{1}{m^+ m^-} \sum_{\boldsymbol{x}^+ \in D^+} \sum_{\boldsymbol{x}^- \in D^-} \left(\mathbb{I}\big(f(\boldsymbol{x}^+) < f(\boldsymbol{x}^-)\big) + \frac{1}{2} \mathbb{I}\big(f(\boldsymbol{x}^+) = f(\boldsymbol{x}^-)\big) \right)$$

它对应 ROC 曲线之上的面积，有

$$\text{AUC} = 1 - \ell_{rank}$$

2.3.4 代价敏感错误率与代价曲线

- 不同类型的错误可能造成不同损失，所以为错误赋予“非均等代价”(unequal cost)。

- 以二分类为例，可以设定一个“代价矩阵”，如下表所示。

真实类别	预测类别	
	第 0 类	第 1 类
第 0 类	0	$cost_{01}$
第 1 类	$cost_{10}$	0

- “代价敏感”（cost-sensitive）错误率

$$E(f;D;cost)=\frac{1}{m}\left(\sum_{\boldsymbol{x}_i\in D^+}\mathbb{I}\big(f(\boldsymbol{x}_i)\neq y_i\big)\times cost_{01}+\sum_{\boldsymbol{x}_i\in D^-}\mathbb{I}\big(f(\boldsymbol{x}_i)\neq y_i\big)\times cost_{10}\right)$$

- 在非均等代价下，“代价曲线”（cost curve）可以刻画期望总体代价。设 p 是样例为正例的概率。横轴为正例概率代价

$$P(+)cost=\frac{p\times cost_{01}}{p\times cost_{01}+(1-p)\times cost_{10}}$$

纵轴为取值为 $[0,1]$ 的归一化代价

$$cost_{norm}=\frac{\mathrm{FNR}\times p\times cost_{01}+\mathrm{FPR}\times(1-p)\times cost_{10}}{p\times cost_{01}+(1-p)\times cost_{10}}$$

- 代价曲线的绘制方法：将 ROC 曲线上的每一点转化为代价平面上的一条线段，取所有线段的下界，围成的面积即为所有条件下学习器的期望总体代价。

2.4 比较检验

本节默认以错误率 ϵ 为性能度量。

2.4.1 假设检验

- 设一个学习器的泛化错误率为 ϵ，在 m 个样本中的测试错误率为 $\hat{\epsilon}$，则其被测得测试错误率为 $\hat{\epsilon}$ 的概率为

$$P(\hat{\epsilon};\epsilon)=\binom{m}{\hat{\epsilon}\times m}\epsilon^{\hat{\epsilon}\times m}(1-\epsilon)^{m-\hat{\epsilon}\times m}$$

它在 $\epsilon=\hat{\epsilon}$ 时最大。

- 二项检验：假设 $\epsilon \leq \epsilon_0$，则在 $1-\alpha$ 的概率内所能观测到的最大错误率为

$$\sum_{i=\epsilon_0 \times m+1}^{m} \binom{m}{i} \epsilon^i (1-\epsilon)^{m-i} < \alpha$$

$$\bar{\epsilon} = \max\epsilon$$

若测试错误率 $\hat{\epsilon}$ 小于临界值 $\bar{\epsilon}$，则能以 $1-\alpha$ 的置信度认为学习器的泛化错误率不大于 ϵ_0，否则假设被拒绝。

- t 检验：若得到了 k 个测试错误率 $\hat{\epsilon}_1, \hat{\epsilon}_2, \ldots, \hat{\epsilon}_k$，则平均测试错误率 μ 和方差 σ^2 为

$$\mu = \frac{1}{k} \sum_{i=1}^{k} \hat{\epsilon}_i$$

$$\sigma^2 = \frac{1}{k-1} \sum_{i=1}^{k} (\hat{\epsilon}_i - \mu)^2$$

它们可看作泛化错误率 ϵ_0 的独立采样，则变量

$$\tau_t = \frac{\sqrt{k}(\mu - \epsilon_0)}{\sigma}$$

服从自由度为 $k-1$ 的 t 分布。若 $|\mu - \epsilon_0|$ 位于 $[t_{-\alpha/2}, t_{\alpha/2}]$ 内，则接受假设 $\mu = \epsilon_0$，否则拒绝该假设。

2.4.2 交叉验证 t 检验

- k 折交叉验证“成对 t 检验”：对每对结果求差 $\Delta_i = \epsilon_i^A - \epsilon_i^B$，若两个学习器性能相同，则差值均值为 0。做 t 检验，在显著度 α 下，若

$$\tau_t = \left| \frac{\sqrt{k}\mu}{\sigma} \right| < t_{\alpha/2, k-1}$$

则接受假设。其中 $t_{\alpha/2,k-1}$ 指自由度为 $k-1$ 的 t 分布上尾部累积分布为 $\alpha/2$ 的临界值。

- 考虑到交叉验证法等实验估计方法，不同轮次的训练集会有一定程度的重叠，导致测试错误率并不独立。故可采用 5×2 交叉验证法（5 次 2 折交叉验证）。每次 2 折交叉验证之前随机将数据打乱，使得 5 次交

叉验证中的数据划分不重复。设 Δ_i^k 表示第 i 次第 k 上的差值。

$$\mu = 0.5(\Delta_1^1 + \Delta_1^2)$$

$$\sigma_i^2 = \big(\Delta_i^1 - \frac{\Delta_i^1 + \Delta_i^2}{2}\big)^2 + \big(\Delta_i^2 - \frac{\Delta_i^1 + \Delta_i^2}{2}\big)^2$$

变量

$$\tau_t = \frac{\mu}{\sqrt{0.2\sum_{i=1}^{5}\sigma_i^2}}$$

服从自由度为 5 的 t 分布，其双边检验的临界值为 $t_{\alpha/2,5}$。

2.4.3 McNemar 检验

对于二分类问题，可统计两个学习器 A 和 B 的分类结果样本数差别，列出“列联表”（contingency table）

算法 B	算法 A	
	正确	错误
正确	e_{00}	e_{01}
错误	e_{10}	e_{11}

假设两学习器性能相同，则 $e_{01} = e_{10}$，于是 $|e_{01} - e_{10}|$ 服从正态分布，变量

$$\tau_{\chi^2} = \frac{(|e_{01} - e_{10}| - 1)^2}{e_{01} + e_{10}}$$

服从自由度为 1 的 χ^2 分布，若其小于临界值 χ_α^2 则接受假设，否则拒绝假设，较小者性能更优。

2.4.4 Friedman 检验与 Nemenyi 后续检验

- 在多个数据集上比较算法。
- 算法排序：使用留出法或交叉验证法得到每个算法在每个数据集上的测试结果，然后在每个数据集上根据测试性能由好到坏排序，序值从 1 递增，若相同则平分序值。
- “原始 Friedman 检验”：假定在 N 个数据集上比较 k 个算法，令 r_i 表示第 i 个算法的平均序值，暂不考虑平分序值，则 r_i 均值为 $(k+1)/2$，

方差为 $(k^2-1)/12$。变量

$$\tau_{\chi^2}=\frac{k-1}{k}\cdot\frac{12N}{k^2-1}\sum_{i=1}^{k}\big(r_i-\frac{k+1}{2}\big)^2=\frac{12N}{k(k+1)}\bigg(\sum_{i=1}^{k}r_i^2-\frac{k(k+1)^2}{4}\bigg)$$

当 k 和 N 都较大时服从自由度为 $k-1$ 的 χ^2 分布。

- Friedman 检验：变量

$$\tau_F=\frac{(N-1)\tau_{\chi^2}}{N(k-1)-\tau_{\chi^2}}$$

服从自由度为 $k-1$ 和 $(k-1)(N-1)$ 的 F 分布。

- Nemenyi 检验：若“所有算法的性能相同”这一假设被拒绝，此时计算出平均序值差别的临界值域

$$CD=q_\alpha\sqrt{\frac{k(k+1)}{6N}}$$

若某两个算法的平均序值之差超出了 CD，则以相应的置信度拒绝“这两个算法性能相同”这一假设。

- Friedman 检验图：横轴为平均序值，纵轴为各个算法，对每个算法以一个圆点表示平均序值，以圆点为中心的横线段表示临界值域的大小。若两个算法的横线段有交叠，则说明它们没有显著区别，否则可以进行显著比较。

2.5 偏差与方差

- 偏差-方差分解：对学习算法的期望泛化错误率进行拆解。
- 学习算法的期望预测

$$\bar{f}(\boldsymbol{x})=\mathbb{E}_D[f(\boldsymbol{x};D)]$$

- 使用样本数相同的不同训练集产生的方差

$$var(\boldsymbol{x})=\mathbb{E}\big[(f(\boldsymbol{x};D)-\bar{f}(\boldsymbol{x})^2\big]$$

- 噪声

$$\varepsilon^2 = \mathbb{E}_D\big[(y_D - y)^2\big]$$

- 偏差（期望输出与真实标记的差别）

$$bias^2(\boldsymbol{x}) = \big(\bar{f}(\boldsymbol{x}) - y\big)^2$$

- 假定 $\mathbb{E}_D[y_D - y] = 0$，则可通过多项式展开得到

$$E(f;D) = \mathbb{E}_D\big[(f(\boldsymbol{x};D) - y_D)^2\big] = bias^2(\boldsymbol{x}) + var(\boldsymbol{x}) + \varepsilon^2$$

  泛化误差可分解为偏差、方差与噪声之和。

- 偏差：学习算法本身的拟合能力；方差：数据的充分性；噪声：学习问题本身的难度

- 偏差-方差窘境（bias-variance dilemma）：训练不足时偏差主导，训练加深时方差主导，训练充足时容易发生过拟合。

3 线性模型

3.1 基本形式

示例 $\boldsymbol{x} = (x_1; x_2; \ldots; x_d)$，其中 x_i 是 $\boldsymbol{x}$ 在第 i 个属性上的取值，则线性模型

$$f(\boldsymbol{x}) = w_1x_1 + w_2x_2 + \cdots + w_dx_d + b = \boldsymbol{w}^T\boldsymbol{x} + b$$

3.2 线性回归

- 一元线性回归的目标

$$f(x_i) = wx_i + b, \text{ 使得} f(x_i) \simeq y_i$$

  其中

$$(w^*, b^*) = \arg_{(w,b)}\min \sum_{i=1}^{m}(y_i - wx_i - b)^2$$

利用“最小二乘法”的最小二乘“参数估计”将 $E_{(w,b)} = \sum_{i=1}^{m}(y_i - wx_i - b)^2$ 对 w 和 b 分别求导并令为零即可得到 w, b 最优解的闭式解

$$w = \frac{\sum_{i=1}^{m} y_i(x_i - \bar{x})}{\sum_{i=1}^{m} x_i^2 - \frac{1}{m}\left(\sum_{i=1}^{m} x_i\right)^2} \qquad b = \frac{1}{m}\sum_{i=1}^{m}(y_i - wx_i)$$

- 多元线性回归的目标

$$f(\boldsymbol{x}_i) = \boldsymbol{w}^T\boldsymbol{x}_i + b, \text{ 使得} f(\boldsymbol{x}_i) \simeq y_i$$

设 $\hat{\boldsymbol{w}} = (\boldsymbol{w}; b)$，将数据集 D 表示为一个 $m \times (d+1)$ 大小的矩阵 $\boldsymbol{X}$

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1 \\ x_{21} & x_{22} & \cdots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{md} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1^T & 1 \\ \boldsymbol{x}_2^T & 1 \\ \vdots & \vdots \\ \boldsymbol{x}_m^T & 1 \end{pmatrix}$$

其中

$$\hat{\boldsymbol{w}}^* = \arg_{\hat{\boldsymbol{w}}}\min(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{w}})^T(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{w}})$$

令 $E_{\hat{\boldsymbol{w}}} = (\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{w}})^T(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{w}})$, 对 $\hat{\boldsymbol{w}}$ 求导并令为零即 $2\boldsymbol{X}^T(\boldsymbol{X}\hat{\boldsymbol{w}} - \boldsymbol{y}) = 0$ 即可。若 $\boldsymbol{X}^T\boldsymbol{X}$ 正定（满秩），则求出 $\hat{\boldsymbol{w}}^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$，此时

$$f(\hat{\boldsymbol{x}}_i) = \hat{\boldsymbol{x}}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

若 $\boldsymbol{X}^T\boldsymbol{X}$ 不满秩，则可能有多解，通过引入正则化由归纳偏好决定。

- 广义线性模型：考虑单调可微函数 $g(\cdot)$

$$y = g^{-1}(\boldsymbol{w}^T\boldsymbol{x} + b)$$

其中 $g(\cdot)$ 称为“联系函数”。

3.3 对数几率回归

- 单位阶跃函数：对于二分类问题将线性回归模型的实值转化为 0/1 值。其中预测值为临界值零可任意判别

$$y = \begin{cases} 0, & z < 0; \\ 0.5, & z = 0; \\ 1, & z > 0; \end{cases}$$

- 对数几率函数

$$y = \frac{1}{1 + e^{-(\boldsymbol{w}^T \boldsymbol{x} + b)}}$$

可化为

$$\ln \frac{y}{1-y} = \boldsymbol{w}^T \boldsymbol{x} + b$$

其中 y 可视为 $\boldsymbol{x}$ 作为正例的可能性，$1-y$ 可视为其作为反例的可能性。

- 对数几率回归的 $\boldsymbol{w}, b$ 估计方法：极大似然法。【TODO：细节待学完 7.2 节极大似然法及梯度下降法再补充】

3.4 线性判别分析

- 思想：设法将样例投影到一条直线上，使得同类样例的投影尽可能接近、异类样例的投影尽可能远离。对于新样本根据其投影到这条直线的投影点位置进行分类。

△ L2 范数（例如欧氏距离）

$$\| x \|_2 = \sqrt{\sum_{i=1}^{k} |x_i|^2}$$

- 目的：使同类样例投影点的协方差（$\boldsymbol{w}^T \Sigma_0 \boldsymbol{w} + \boldsymbol{w}^T \Sigma_1 \boldsymbol{w}$）尽可能小，使类中心之间的距离（$\| \boldsymbol{w}^T \boldsymbol{\mu}_0 - \boldsymbol{w}^T \boldsymbol{\mu}_1 \|_2^2$）尽可能大。

- 类内散度矩阵

$$\boldsymbol{S}_w = \Sigma_0 + \Sigma_1 = \sum_{\boldsymbol{x}\in X_0} (\boldsymbol{x}-\boldsymbol{\mu}_0)(\boldsymbol{x}-\boldsymbol{\mu}_0)^T + \sum_{\boldsymbol{x}\in X_1} (\boldsymbol{x}-\boldsymbol{\mu}_1)(\boldsymbol{x}-\boldsymbol{\mu}_1)^T$$

- 类间散度矩阵

$$\boldsymbol{S}_b = (\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)^T$$

- 欲最大化目标（$\boldsymbol{S}_b$ 与 $\boldsymbol{S}_w$ 的广义 Rayleigh 商）

$$J = \frac{\| \boldsymbol{w}^T\boldsymbol{\mu}_0 - \boldsymbol{w}^T\boldsymbol{\mu}_1 \|_2^2}{\boldsymbol{w}^T\Sigma_0\boldsymbol{w} + \boldsymbol{w}^T\Sigma_1\boldsymbol{w}} = \frac{\boldsymbol{w}^T(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)^T\boldsymbol{w}}{\boldsymbol{w}^T(\Sigma_0+\Sigma_1)\boldsymbol{w}} = \frac{\boldsymbol{w}^T\boldsymbol{S}_b\boldsymbol{w}}{\boldsymbol{w}^T\boldsymbol{S}_w\boldsymbol{w}}$$

- 确定 $\boldsymbol{w}$ 的方法：J 中的解与 $\boldsymbol{w}$ 的长度无关，故令 $\boldsymbol{w}^T\boldsymbol{S}_w\boldsymbol{w}=1$，转化为：已知 $\boldsymbol{w}^T\boldsymbol{S}_w\boldsymbol{w}=1$ 求 $-\boldsymbol{w}\boldsymbol{S}_b\boldsymbol{w}$ 的最小值。由拉格朗日乘子法，即 $\boldsymbol{S}_b\boldsymbol{w} = \lambda\boldsymbol{S}_w\boldsymbol{w}$。又 $\boldsymbol{S}_b\boldsymbol{w}$ 方向恒为 $\boldsymbol{\mu}_0-\boldsymbol{\mu}_1$，令 $\boldsymbol{S}_b\boldsymbol{w} = \lambda(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)$，得

$$\boldsymbol{w} = \boldsymbol{S}_w^{-1}(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)$$

对 $\boldsymbol{S}_w$ 进行奇异值分解 $\boldsymbol{S}_w = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$，由 $\boldsymbol{S}_w^{-1} = \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^{-1}$ 得到 $\boldsymbol{w}$。

> △ 上例用拉格朗日乘子法的计算细节：目标函数为 $f(\boldsymbol{x}) = -\boldsymbol{w}\boldsymbol{S}_b\boldsymbol{w}$，约束方程 $g(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{S}_w\boldsymbol{w} - 1 = 0$。等价于由方程 $g(\boldsymbol{x})=0$ 确定的 $d-1$ 维曲面上寻找能使 $f(\boldsymbol{x})$ 最小化的点，满足
>
> (1) 约束曲面上任意点 $\boldsymbol{x}$ 的梯度 $\nabla g(\boldsymbol{x})$ 正交于约束曲面
>
> (2) 在最优点 $\boldsymbol{x}^*$，$f(\boldsymbol{x})$ 在该点的梯度 $\nabla f(\boldsymbol{x}^*)$ 正交于约束曲面
>
> 于是在最优点 $\boldsymbol{x}^*$，梯度 $\nabla g(\boldsymbol{x})$ 和 $\nabla f(\boldsymbol{x})$ 的方向必相同或相反，即存在 $\lambda\neq 0$ 使得
>
> $$\nabla f(\boldsymbol{x}^*) + \lambda\nabla g(\boldsymbol{x}^*) = 0$$
>
> 即
>
> $$\boldsymbol{S}_b\boldsymbol{w} = \lambda\boldsymbol{S}_w\boldsymbol{w}$$

- LDA 推广到多分类任务。假设存在 N 个类，且第 i 类示例数为 m_i。

设 $\boldsymbol{\mu}$ 是所有示例的均值向量。全局散度矩阵

$$\boldsymbol{S}_t = \boldsymbol{S}_b + \boldsymbol{S}_w = \sum_{i=1}^{m} (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T$$

类内散度矩阵

$$\boldsymbol{S}_w = \sum_{i=1}^{N} \boldsymbol{S}_{w_i} = \sum_{i=1}^{N} \sum_{\boldsymbol{x} \in X_i} (\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^T$$

类间散度矩阵

$$\boldsymbol{S}_b = \boldsymbol{S}_t - \boldsymbol{S}_w = \sum_{i=1}^{N} m_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T$$

常用实现的优化目标

$$\max_{\boldsymbol{W}} \frac{\mathrm{tr}(\boldsymbol{W}^T \boldsymbol{S}_b \boldsymbol{W})}{\mathrm{tr}(\boldsymbol{W}^T \boldsymbol{S}_w \boldsymbol{W})}$$

可以通过广义特征值 $\boldsymbol{S}_b \boldsymbol{W} = \lambda \boldsymbol{S}_w \boldsymbol{W}$ 求解，$\boldsymbol{W}$ 的闭式解为 $\boldsymbol{S}_w^{-1} \boldsymbol{S}_b$ 的 d' 个最大非零广义特征值对应的特征向量组成的矩阵，有 $d' \leq N - 1$，实现了降维。

3.5 多分类学习

- 基本思路：将多分类任务拆为若干个二分类任务求解，最经典的拆分策略有三种。
- 一对一（OvO）：将 N 个类别两两配对，产生 $N(N-1)/2$ 个二分类任务。最终把预测得最多的类别作为分类结果。
- 一对其余（OvR）：每次将一个类的样例作为正例，所有其它类的样例作为反例，产生 N 个分类任务。最终若仅有一个分类器预测为正类，则对应的类别标记为最终分类结果，否则考虑各分类器的置信区间，选择置信度最大的类别标记作为分类结果。
- 多对多（MvM）：每次将若干个类作为正类，若干个其它类作为反类，可用纠错输出码（ECOC）技术。
- ECOC 工作过程：

(1) 编码（类别划分）：对 N 个类别做 M 次划分，每次划分将一部分类别作为正类、一部分类别作为反类，产生 M 个分类器。

(2) 解码（距离比较）：用 M 个分类器对测试样本进行预测，这些预测标记组成一个编码，计算其与各个类别各自编码的距离，返回距离最小的类别作为分类结果。

常用的编码矩阵为二元码和三元码（有停用类）。

- 任何两个类别之间的编码距离越远，纠错能力越强，而码长的增加会增大确定最优编码的难度。

3.6 类别不平衡问题

- 类别不平衡（class-imbalance）：分类任务中不同类别的训练样例数目差别很大。以下为类别不平衡学习的策略。
- 再缩放（rescaling）：假设“训练集是真实样本总体的无偏采样”，令

$$\frac{y'}{1-y'} = \frac{y}{1-y} \times \frac{m^-}{m^+}$$

也是代价敏感学习，其中的 m^-/m^+ 可用 $cost^+/cost^-$ 代替。

- 欠采样（undersampling）/下采样（downsampling）：去除一些正例（反例）使得正、反例数目接近。可用 EasyEnsemble 算法将反例划分为若干个集合供不同学习器使用，全局上不会丢失重要信息。
- 过采样（oversampling）/上采样（upsampling）：增加一些正例（反例）使得正、反例数目接近。可用 SMOTE 算法对训练集中的正例进行插值。
- 阈值移动（threshold-moving）：基于原始训练集学习，在用训练好的分类器进行预测时将“再缩放”中的式子嵌入到决策过程中。

4 决策树

4.1 基本流程

- 决策树的性质：包含一个根结点、若干个内部结点、若干个叶结点，叶结点对应于决策结果，其他每个结点对应于一个属性测试，根结点包含样本全集。从根结点到每个叶结点的路径对应了一个判定测试序列。
- 决策树的目的：产生一棵泛化能力强的决策树。
- 决策树的基本流程（分治策略）

Algorithm 1 TreeGenerate(D, A)

Input: 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
属性集 $A = \{a_1, a_2, \ldots, a_d\}$
Output: 以 node 为根结点的一棵决策树

生成结点 node;
if D 中样本全属于同一类别 C **then**
  将 node 标记为 C 类叶结点;
  return
end if
if $A = \emptyset$ **or** D 中样本在 A 上取值相同 **then**
  将 node 标记为叶结点，其类别标记为 D 中样本数最多的类;
  return
end if
从 A 中选择最优划分属性 a_*;
for a_* 的每一个值 a_*^v **do**
  为 node 生成一个分支;
  令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
  if D_v 为空 **then**
    将分支结点标记为叶结点，其类别标记为 D 中样本数最多的类;
    return
  else
    以 TreeGenerate($D_v, A \backslash \{a_*\}$) 为分支结点
  end if
end for

4.2 划分选择

目的：使决策树的分支结点所包含的样本尽可能属于同一类别，结点的纯度越来越高。

4.2.1 信息增益

- ID3 决策树算法以信息增益为准则选择划分属性。
- 信息熵（设 D 中共有 $\mathcal{Y}$ 类样本，第 k 类样本所占的比例为 p_k，值越小则 D 的纯度越高）

$$\mathrm{Ent}(D) = -\sum_{k=1}^{|\mathcal{Y}|} p_k \log_2 p_k$$

- 设离散属性 a 的取值集合为 $\{a^1, a^2, \ldots, a_V\}$，用 a 对样本集合 D 进行划分，产生 V 个分支结点，第 v 个分支结点包含了 D 中所有在属性 a 上取值为 a^v 的样本，记为 D^v，于是信息增益（值越大则纯度提升越大）：

$$\mathrm{Gain}(D, a) = \mathrm{Ent}(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} \mathrm{Ent}(D^v)$$

- 最终选择的属性

$$a_* = \arg_{a \in A} \max \quad \mathrm{Gain}(D, a)$$

- 缺点：对可取值数目较多对属性有所偏好。

4.2.2 增益率

- C4.5 决策树算法以增益率为准则选择划分属性。
- 增益率

$$\mathrm{Gain_ratio}(D, a) = \frac{\mathrm{Gain}(D, a)}{\mathrm{IV}(a)}$$

其中 $\mathrm{IV}(a)$ 为属性 a 的“固有值”，可能取值数目越多则通常越大。

$$\mathrm{IV}(a) = -\sum_{v=1}^{V} \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|}$$

- 缺点：对可取值数目较少的属性有所偏好，所以选择候选划分属性时使用了一个启发式算法：先从候选划分属性中找出信息增益高于平均水平的属性，再从中选择增益率最高的。

4.2.3 基尼指数

- CART 决策树算法以基尼指数（Gini index）为准则选择划分属性。
- 基尼值（度量数据集 D 的纯度），反映了从 D 中随机抽取两个样本，其类别标记不一致的概率，值越小则纯度越高。

$$\mathrm{Gini}(D) = \sum_{k=1}^{|\mathcal{Y}|} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{|\mathcal{Y}|} p_k^2$$

- 属性 a 的基尼指数

$$\mathrm{Gini_ndex}(D, a) = \sum_{v=1}^{V} \frac{|D^v|}{|D|} \mathrm{Gini}(D^v)$$

- 最终选择属性

$$a_* = \arg_{a \in A} \min \quad \mathrm{Gini\_index}(D, a)$$

4.3 剪枝处理

- 目的：去掉一些分支来降低过拟合的风险。
- 基本策略之“预剪枝”（prepruning）：在决策树生成过程中对每个结点在划分前先进行估计，若当前结点的划分不能带来决策树泛化性能提升，则停止划分并将当前结点标记为叶结点。
- 基本策略之“后剪枝”（postpruning）：先从训练集生成一棵完整的决策树，然后自底向上对非叶结点考察，若将该结点的子树替换为叶结点能提升决策树的泛化性能，则将该子树替换为叶结点。
- 泛化性能评估方法用 2.2 节的性能评估方法即可，简单地，可以使用留出法并计算错误率（精度）。

4.3.1 预剪枝

- 降低了过拟合的风险。
- 显著减少决策树训练和测试时间开销。
- 可能因为本身的贪心禁止了那些后续划分使泛化性能显著提升的分支的展开，带来欠拟合的风险。

4.3.2 后剪枝

- 通常比预剪枝决策树保留了更多的分支。
- 欠拟合风险很小，泛化性能往往优于预剪枝决策树。
- 训练时间比未剪枝或预剪枝决策树大得多。

4.4 连续与缺失值

4.4.1 连续值处理

- 思路：连续属性离散化。
- 最简单的策略：二分法（C4.5 决策树算法中采用）。
- 二分法过程：给定样本集 D 和连续属性 a，假定 a 在 D 上出现了 n 个不同的取值，将它们从小到大排序，记为 $\{a^1, a^2, \ldots, a^n\}$，基于划分点 t 划分为子集 D_t^- 和 D_t^+，前者包含在属性 a 上不大于 t 的样本。我们把每个区间 $[a^i, a^{i+1})$ 的中位点 $\frac{a^i+a^{i+1}}{2}$ 作为候选划分点。
- 改造后的信息增益

$$\mathrm{Gain}(D,a) = \max_{t\in T_a} \mathrm{Gain}(D,a,t) = \max_{t\in T_a} \big(\mathrm{Ent}(D) - \sum_{\lambda\in\{-,+\}} \frac{|D_t^\lambda|}{|D|}\mathrm{Ent}(D_t^\lambda)\big)$$

- 与离散属性的区别：若当前结点划分属性为连续属性，该属性还可以作为其后代结点的划分属性。

4.4.2 缺失值处理

- 策略：利用某个属性 a 上无缺失值样本的信息增益扩放到所有样本的信息增益。(C4.5 决策树算法中采用)
- 划分属性的选择过程：$\tilde{D}$ 表示 D 中在属性 a 上没有缺失值的样本子集；$\tilde{D}^v$ 表示 $\tilde{D}$ 中在属性 a 上取值为 a^v 的样本子集 $(1 \le v \le V)$；$\tilde{D}_k$ 表示 $\tilde{D}$ 中属于第 k 类的样本子集 $(1 \le k \le |\mathcal{Y}|)$。为每个样本 $\boldsymbol{x}$ 赋予一个权重 $w_{\boldsymbol{x}}$

  无缺失值样本的比例：$\rho = \dfrac{\sum_{\boldsymbol{x}\in\tilde{D}} w_{\boldsymbol{x}}}{\sum_{\boldsymbol{x}\in D} w_{\boldsymbol{x}}}$

  无缺失值样本中第 k 类的比例：$\tilde{p}_k = \dfrac{\sum_{\boldsymbol{x}\in\tilde{D}_k} w_{\boldsymbol{x}}}{\sum_{\boldsymbol{x}\in\tilde{D}} w_{\boldsymbol{x}}}$

  无缺失值样本中在属性 a 上取值为 a^v 的样本的比例：$\tilde{r}_v = \dfrac{\sum_{\boldsymbol{x}\in\tilde{D}^v} w_{\boldsymbol{x}}}{\sum_{\boldsymbol{x}\in\tilde{D}} w_{\boldsymbol{x}}}$

  信息增益的推广式

$$\mathrm{Gain}(D,a) = \rho \times \mathrm{Gain}(\tilde{D},a) = \rho \times \left(\mathrm{Ent}(\tilde{D}) - \sum_{v=1}^{V} \tilde{r}_v \mathrm{Ent}(\tilde{D}^v)\right)$$

$$\mathrm{Ent}(\tilde{D}) = -\sum_{k=1}^{|\mathcal{Y}|} \tilde{p}_k \log_2 \tilde{p}_k$$

- 已经划分属性进行样本划分的过程：
  - 若样本 $\boldsymbol{x}$ 在 a 上的取值已知，则将其划入与其取值对应的子结点，样本权值在子结点中保持为 $w_{\boldsymbol{x}}$
  - 若样本 $\boldsymbol{x}$ 在 a 上的取值未知，则将其划入所有子结点，样本权值在与属性值 a^v 对应的子结点中调整为 $\tilde{r}_v \cdot w_{\boldsymbol{x}}$。

4.5 多变量决策树

- 决策树所形成的分类边界的特点：轴平行。每一段划分都直接对应某个属性取值。当真实分类边界比较复杂时决策树也会相当复杂。
- 多变量决策树：非叶结点是一个形如 $\sum_{i=1}^{d} w_i a_i = t$ 的线性分类器。在学习过程中试图建立一个合适的线性分类器。实现了斜划分。

5 神经网络

5.1 神经元模型

- 神经网络的定义：神经网络是由具有适应性的简单单元组成的广泛并行互连的网络，它的组织能够模拟生物神经系统对真实世界物体作出对交互反应。
- 最基本的成分：神经元模型
- M-P 神经元模型：神经元接收到来自 n 个其他神经元的输入信号 x_i，它们各通过带权重 w_i 的连接进行传递，神经元将总输入值 $\sum_{i=1}^{n} w_i x_i$ 与神经元的阈值 θ 进行比较，得到 $\sum_{i=1}^{n} w_i x_i - \theta$，然后通过“激活函数”$f$ 处理以产生神经元的输出：

$$y = f\big(\sum_{i=1}^{n} w_i x_i - \theta\big)$$

- 激活函数的模型
  - 阶跃函数（不连续、不光滑）：

$$\mathrm{sgn}(x) = \begin{cases} 1, & x \geq 0; \\ 0, & x < 0 \end{cases}$$

  - Sigmoid 函数（将可能在较大范围内变化的输入值挤压到 $(0,1)$ 输出值范围内）：

$$\mathrm{sigmoid}(x) = \frac{1}{1+e^{-x}}$$

5.2 感知机与多层网络

1. 感知机（Perceptron）：两层神经元，输入层接受外界输入信号后传递给输出层，输出层是 M-P 神经元，也称为“阈值逻辑单元”
   - 感知机实现逻辑“与”：$w_1 = w_2 = 1, \theta = 0$；逻辑“或”：$w_1 = w_2 = 1, \theta = 0.5$；逻辑“非”：$w_1 = -0.6, w_2 = 0, \theta = -0.5$。
   - 将阈值看作固定输入为-1.0 的“哑结点”所对应的权重 w_{n+1}。统一为权重的学习。对训练样例 $(\boldsymbol{x}, y)$，若当前感知机的输出为 $\hat{y}$，则其

权重调整如下：

$$w_i \leftarrow w_i + \Delta w_i \qquad \Delta w_i = \eta(y - \hat{y})x_i$$

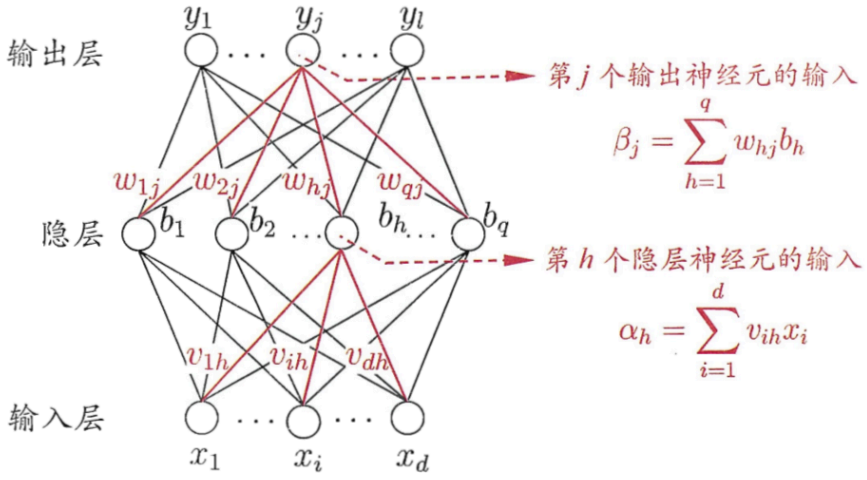
其中 η 为 $(0,1)$ 内的小正数，称为学习率。

- 感知机只能求解线性可分问题（存在一个线性超平面区分两类模式），对于非线性可分问题会发生振荡（fluctuation）。

2. 多层前馈神经网络：每层神经元与下一层全互连，不存在同层连接和跨层连接。输入层神经元仅接收输入，隐层与输出层包含功能神经元。

3. 神经网络的学习过程：根据训练数据来调整神经元之间的连接权以及每个功能神经元的阈值。

5.3 误差逆传播算法



- 任意参数 v 的更新估计式：$v \leftarrow v + \Delta v$
- 目标：最小化训练集 D 上的累积误差

$$E = \frac{1}{m}\sum_{k=1}^{m} E_k$$

$$E_k = \frac{1}{2}\sum_{j=1}^{l}(\hat{y}_j^k - y_j^k)^2$$

- 参数调整原则：梯度下降。例如 $\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{hj}}$
- 推导方法：参数影响的逐级传递，例如

$$\frac{\partial E_k}{\partial w_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial w_{hj}}$$

  Sigmoid 函数的性质：$f'(x) = f(x)(1 - f(x))$

- 各个参数的调整值：

$$\Delta w_{hj} = \eta g_j b_h$$

$$\Delta \theta_j = -\eta g_j$$

$$\Delta v_{ih} = \eta e_h x_i$$

$$\Delta \gamma_h = -\eta e_h$$

  其中

$$g_j = \hat{y}_j^k (1 - \hat{y}_j^k)(y_j^k - \hat{y}_j^k)$$

$$e_h = b_h (1 - b_h) \sum_{j=1}^{l} w_{hj} g_j$$

- BP 算法执行过程：

  1. 输入：训练集 $D = \{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^m$ 和学习率 η
  2. 在 $(0, 1)$ 范围内随机初始化网络中所有连接权和阈值
  3. 重复计算 $\hat{\boldsymbol{y}}_k, g_j, e_h$ 并更新四个参数直到达到停止条件（如训练误差已达到一个很小的值）
  4. 输出：连接权与阈值确定的多层前馈神经网络

- 累积误差逆传播（ABP）算法：省去 BP 算法对每个样例都更新参数都步骤，而是读取整个训练集一遍后才更新参数，直接针对累积误差最小化。

- 缓解过拟合的策略

  – 早停：将数据分成训练集和验证集，若训练集误差降低而验证集误差升高，则停止训练。

– 正则化：在误差目标函数中增加一个用于描述网络复杂度的部分。令 $\lambda \in (0,1)$，如

$$E = \lambda \frac{1}{m} \sum_{k=1}^{m} E_k + (1-\lambda) \sum_i w_i^2$$

5.4 全局最小与局部极小

- 神经网络训练过程：在参数空间中寻找一组最优参数使得 E 最小。
- 局部极小：对 $\boldsymbol{w}^*$ 和 θ^*，若存在 $\epsilon > 0$ 使得

$$\forall (\boldsymbol{w};\theta) \in \{(\boldsymbol{w};\theta) | \parallel (\boldsymbol{w};\theta) - (\boldsymbol{w}^*;\theta^*) \parallel \leq \epsilon\}$$

都有 $E(\boldsymbol{w};\theta) \geq E(\boldsymbol{w}^*;\theta^*)$，则 $(\boldsymbol{w}^*;\theta^*)$ 为局部极小解。

- 全局最小：对参数空间中任意 $(\boldsymbol{w};\theta)$ 都有 $E(\boldsymbol{w};\theta) \geq E(\boldsymbol{w}^*;\theta^*)$，则 $(\boldsymbol{w}^*;\theta^*)$ 为全局最小解。
- 使用最为广泛的参数寻优方法：梯度下降法（容易找到局部极小）。
- 跳出局部极小，进一步接近全局最小的策略：
  - 以多组不同参数值初始化多个神经网络（从多个不同的初始点开始搜索）；
  - 使用“模拟退火”技术：每一步有一定概率接受比当前解更差的结果，且接受次优解概率随时间推移而逐渐降低；
  - 使用随机梯度下降（即使陷入局部极小点，计算出来的梯度可能不为零）；
  - 遗传算法等其它启发式算法。

5.5 其他常见神经网络

5.5.1 RBF 网络

- RBF 网络（径向基网络）：一种单隐层前馈神经网络。
- 隐层激活函数：径向基函数；输出层是对隐层输出的线性组合。

- 设 q 为隐层神经元个数，$\boldsymbol{c}_i$ 和 w_i 是第 i 个隐层神经元所对应的中心和权重。径向基函数 $\boldsymbol{\rho}(\boldsymbol{x}, \boldsymbol{c}_i)$ 通常定义为样本 $\boldsymbol{x}$ 到数据中心 $\boldsymbol{c}_i$ 之间欧氏距离的单调函数。常用的高斯径向基函数

$$\rho(\boldsymbol{x}, \boldsymbol{c}_i) = e^{-\beta_i \|\boldsymbol{x}-\boldsymbol{c}_i\|^2}$$

- 训练过程：通过随机采样、聚类等方法确定神经元中心 $\boldsymbol{c}_i$→ 利用 BP 算法确定参数 w_i 和 β_i

5.5.2 ART 网络

- ART 网络（自适应谐振网络）：一种竞争学习型无监督神经网络。
- 构成：比较层（接收输入样本，传递给识别层）、识别层（每个神经元对应一个模式类，数目也在训练过程中动态增长）、识别阈值、重置模块。
- 接收比较层的输入 → 识别层神经元相互竞争（向量距离）→ 获胜神经元抑制其他识别层神经元的激活 → 输入向量与获胜神经元的代表向量相似度大于识别阈值？则当前输入样本被归为该代表向量所属类别并更新连接权：重置模块在识别层增设新神经元，代表向量即为当前输入向量。
- 识别阈值高 → 输入样本分类数目多且精细；识别阈值低 → 输入样本分类数目少且粗略。
- 缓解了“可塑性-稳定性窘境”，可进行增量学习或在线学习。

5.5.3 SOM 网络

- SOM 网络（自组织映射网络）：一种竞争学习型无监督神经网络。
- 特点：将高维输入数据映射到低维空间，同时保持输入数据在高维空间的拓扑结构。获胜神经元决定了输入向量在低维中的位置。
- 目标：为每个输出层神经元找到合适的权向量，达到拓扑结构的保持。
- 训练过程：接收一个训练样本 → 每个输出层神经元计算其与自身权向量的距离 → 距离最近者获胜（最佳匹配单元）→ 最佳匹配单元及

其邻近神经元的权向量被调整使得它们与当前输入样本的距离缩小 → 不断迭代直到收敛

5.5.4 级联相关网络

- 自适应网络的代表：网络结构也是学习目标之一。
- 级联：建立层次连接的层级结构。从仅有输入输出层增加新的隐层神经元。
- 相关：通过最大化新神经元的输出与网络误差之间的相关性来训练相关的参数。
- 特点：无需设置网络层数、隐层神经元数目，训练速度快，数据较小时易陷入过拟合。

5.5.5 Elman 网络

- 最常用的递归神经网络 (RNN) 之一：可出现环形，有信息反馈，网络在 t 时刻的输出状态与 t 时刻的输入及 $t-1$ 时刻的网络状态都有关。
- 结构与多层前馈网络相似，隐层神经元的输出反馈给所有隐层神经元。
- 隐层神经元常采用 Sigmoid 激活函数，网格训练常用推广的 BP 算法。

5.5.6 Bolzmann 机

- 定义了“能量”，能量最小化时网络达理想状态。
- 神经元全部为布尔型，神经元两两全连接。设状态向量 $\boldsymbol{s} \in \{0,1\}^n$ 表示 n 个神经元的状态。则 $\boldsymbol{s}$ 对应的 Boltzmann 机能量为

$$E(\boldsymbol{s}) = -\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{ij}s_i s_j - \sum_{i=1}^{n}\theta_i s_i$$

- 若神经元以任意不依赖于输入值顺序进行更新，网络最终达到 Boltzmann 分布（平衡态），此时状态向量 $\boldsymbol{s}$ 出现的概率

$$P(\boldsymbol{s}) = \frac{e^{-E(\boldsymbol{s})}}{\sum_{\boldsymbol{t}} e^{-E(\boldsymbol{t})}}$$

- 训练过程：将每个训练样本视为一个训练向量，使其出现的概率尽可能大。常用受限 Boltzmann 机（仅保留显层与隐层的连接）。
- 受限 Boltzmann 机的训练方法：对比散度算法。设显、隐层神经元个数分别为 d, q，状态向量分别为 $\boldsymbol{v}, \boldsymbol{h}$

$$P(\boldsymbol{v}|\boldsymbol{h}) = \prod_{i=1}^{d} P(v_i|\boldsymbol{h}) \quad P(\boldsymbol{h}|\boldsymbol{v}) = \prod_{j=1}^{q} P(h_j|\boldsymbol{v})$$

- 受限 Bolzmann 的工作过程：对每个训练样本 $\boldsymbol{v}$，先计算 $P(\boldsymbol{h}|\boldsymbol{v})$，然后分布采样得到 $\boldsymbol{h}$，再计算 $P(\boldsymbol{v}'|\boldsymbol{h})$，从 $\boldsymbol{v}'$ 产生 $\boldsymbol{h}$。连接权的更新公式为

$$\Delta w = \eta(\boldsymbol{v}\boldsymbol{h}^T - \boldsymbol{v}'\boldsymbol{h}'^T)$$

5.6 深度学习

- 典型的深度学习模型：多隐层神经网络
- 多隐层网络训练的手段：无监督逐层训练（预训练 + 微调）。将大量参数分组，对每组找到局部较优，联系起来进行全局寻优，节省训练开销。
- 卷积神经网络(CNN)：待本书主要内容学完后，对于 CNN 方面将着重学习：CS231n Convolutional Neural Networks for Visual Recognition

6 支持向量机

6.1 间隔与支持向量

- 划分超平面：

$$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b = 0$$

其中 $\boldsymbol{w} = (w_1; w_2; \ldots; w_d)$ 为法向量，决定方向；b 为位移项，决定超平面与原点之间的距离。

- 样本空间中任意点 $\boldsymbol{x}$ 到超平面 $(\boldsymbol{x}, b)$ 的距离

$$r = \frac{|\boldsymbol{w}^{\mathrm{T}} + b|}{\| \boldsymbol{w} \|}$$

- 若超平面 $(\boldsymbol{w}, b)$ 能将训练样本正确分类，则对于任意 $(\boldsymbol{x}_i, y_i) \in D$ 有

$$\begin{cases} \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b \geq +1, & y_i = +1; \\ \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b \leq -1, & y_i = -1. \end{cases}$$

- 支持向量：满足 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b = \pm 1, y_i = \pm 1$ 的距离超平面最近的几个训练样本点。

- 间隔：两个异类支持向量到超平面的距离之和：

$$\gamma = \frac{2}{\| \boldsymbol{w} \|}$$

- 支持向量机（SVM）的基本型：最大化间隔，即最小化 $\| \boldsymbol{w} \|^2$

$$\begin{aligned} \min_{\boldsymbol{w},b} \quad & \frac{1}{2} \| \boldsymbol{w} \|^2 \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geq 1, i = 1, 2, \ldots, m. \end{aligned}$$

6.2 对偶问题

求解 6.1 节基本型的一个高效方法：

1. 用拉格朗日乘子法得到对偶问题，即对基本型对每条约束添加拉格朗日乘子 $a_i \geq 0$，得到

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \| \boldsymbol{w} \|^2 + \sum_{i=1}^{m} \alpha_i (1 - y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b))$$

分别令对 $\boldsymbol{w}$ 和 b 的偏导数为零再代入消去 $\boldsymbol{w}, b$ 可得到对偶问题如下：

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}^{\mathrm{T}} \boldsymbol{x}_j \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0 \\ & \alpha_i \geq 0, \quad i = 1, 2, \ldots, m. \end{aligned}$$

这样在解出 $\boldsymbol{\alpha}$ 后，求出 $\boldsymbol{w}$ 和 b 就可以得到模型 $f(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{x} + b$

2. 上述过程需满足 KKT 条件，即要求

$$\begin{cases} \alpha_i \geq 0; \\ y_i f(\boldsymbol{x}_i) - 1 \geq 0; \\ \alpha_i (y_i f(\boldsymbol{x}_i) - 1) = 0. \end{cases}$$

> △ KKT 条件对于上述结论的具体分析过程：对于不等式约束 $g(\boldsymbol{x}_i) = 1 - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) = 1 - y_i f(\boldsymbol{x}_i) \leq 0$，对于 $g(\boldsymbol{x}_i) < 0$ 的情形，可通过 $\nabla f(\boldsymbol{x}_i) = 0$ 来获得最优点，等价于在 $L(\boldsymbol{w}, b, \alpha_i)$ 中对 α_i 置零然后对 $\nabla_{\boldsymbol{w},b} L(\boldsymbol{x}_i, \alpha_i)$ 置零；对于 $g(\boldsymbol{x}_i) = 0$ 的情形，类似于拉格朗日乘子法中等式约束的分析，但此时 $\nabla f(\boldsymbol{x}_i^*)$ 的方向必与 $\nabla g(\boldsymbol{x}_i^*)$ 相反。综上有 $\alpha_i g(\boldsymbol{x}_i) = 0$。由此可转化为在如下约束下最小化拉格朗日函数的问题：
>
> $$\begin{cases} 1 - y_i f(\boldsymbol{x}_i) \leq 0; \\ \alpha_i \geq 0; \\ \alpha_i (1 - y_i f(\boldsymbol{x}_i)) = 0. \end{cases}$$

由此可以总结：若 $\alpha_i = 0$，则该样本不会出现在求和中，没有影响；若 $y_i f(\boldsymbol{x}_i) = 1$，则对应的样本点在最大间隔边界上，是支持向量。因此在支持向量机训练完成后，最终模型仅与支持向量有关。

3. 求解第一步最后的对偶问题的方法：二次规划算法（问题规模正比于训练样本数，开销太大）；SMO 算法。SMO 不断执行以下两个步骤直至收敛：

   - 选取一对需更新的变量 α_i 和 α_j；
   - 固定 α_i 和 α_j 以外的参数，求解对偶模型中获得更新后的 α_i 和 α_j。

   选择参数的启发式方法：使选取的两变量所对应样本之间的间隔最大。（对它们进行更新一般会带给目标函数值更大的变化）仅考虑 α_i, α_j 时，因为有

$$\alpha_i y_i + \alpha_j y_j = -\sum_{k \neq i,j} \alpha_k y_k, \quad \alpha_i \geq 0, \alpha_j \geq 0$$

   可以消去 α_j 得到单变量二次规划问题，仅有的约束是 $\alpha_i \geq 0$。

4. 确定偏移项 b 的方法: 对于所有支持向量 $(\boldsymbol{x}_s, y_s), s \in S$ 都有 $y_s f(\boldsymbol{x}_s) = 1$ 即

$$y_s\left(\sum_{i\in S}\alpha_i y_i \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_s + b\right) = 1$$

现实任务中一般使用所有支持向量求解的平均值:

$$b = \frac{1}{|S|}\sum_{s\in S}\left(1/y_s - \sum_{i\in S}\alpha_i y_i \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_S\right)$$

6.3 核函数

- 在现实任务中，原始样本空间可能不存在一个能正确划分两类样本的超平面，此时可将样本从原始空间映射到一个更高维的特征空间，使得样本在这个特征空间内线性可分。若原始空间是有限维，则必定存在这样一个高维特征空间。

- 设 $\phi(\boldsymbol{x})$ 表示将 $\boldsymbol{x}$ 映射后的特征向量。于是在特征空间中划分超平面所对应的模型可表示为 $f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}) + b$。此时对偶问题约束条件不变，目标变为

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$$

直接在高维空间计算 $\phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$ 很难，因此定义“核函数”:

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)\rangle = \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}_j)$$

由此“支持向量展式”为:

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}) + b = \sum_{i=1}^{m}\alpha_i y_i \phi(\boldsymbol{x}_i)^{\mathrm{T}}\phi(\boldsymbol{x}) + b = \sum_{i=1}^{m}\alpha_i y_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

- 核函数的定理如下:

Theorem 1. 令 $\mathcal{X}$ 为输入空间，$\kappa(\cdot,\cdot)$ 是定义在 $\mathcal{X}\times\mathcal{X}$ 上的对称函数，则 κ 是核函数当且仅当对于任意数据 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$，“核

矩阵 $\mathbf{K}$ 总是半定的。其中

$$\mathbf{K}=\begin{bmatrix}\kappa(\boldsymbol{x}_1,\boldsymbol{x}_2) & \cdots & \kappa(\boldsymbol{x},\boldsymbol{x}_j) & \cdots & \kappa(\boldsymbol{x},\boldsymbol{x}_m)\\ \vdots & \ddots & \vdots & \ddots & \vdots\\ \kappa(\boldsymbol{x}_i,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_i,\boldsymbol{x}_j) & \cdots & \kappa(\boldsymbol{x}_i,\boldsymbol{x}_m)\\ \vdots & \ddots & \vdots & \ddots & \vdots\\ \kappa(\boldsymbol{x}_m,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_m,\boldsymbol{x}_j) & \cdots & \kappa(\boldsymbol{x}_m,\boldsymbol{x}_m)\end{bmatrix}$$

任何一个核函数都隐式地定义了一个“再生核希尔伯特空间”。

△ 半正定矩阵的定义

- 广义：设 $\boldsymbol{A}$ 是 n 阶方阵，如果对任何非零向量 $\boldsymbol{X}$，都有 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}\geq 0$，则称 A 为半正定矩阵。
- 狭义（常用）：设 $\boldsymbol{A}$ 为实对称矩阵，若对于每个非零实向量 $\boldsymbol{X}$，都有 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}\geq 0$，则称 $\boldsymbol{A}$ 为半正定矩阵，称 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{X}$ 为半正定二次型。

△ 再生核希尔伯特空间等空间的概念可见文章：From Zero to Reproducing Kernel Hilbert Spaces in Twelve Pages or Less。这玩意有点硬核 OTZ

- 支持向量机的最大变数：核函数选择。可以使用常用的核函数并通过函数组合得到。

名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j)^d$	$d\geq 1$ 为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2}{2\sigma^2})$	$\sigma>0$ 为高斯核的带宽
拉普拉斯核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|}{\sigma})$	$\sigma>0$
Sigmoid 核	$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\tanh(\beta\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j+\theta)$	tanh 为双曲正切函数，$\beta>0,\theta<0$

若 κ_1,κ_2 为核函数，则线性组合（任意正数 γ_1,γ_2）$\gamma_1\kappa_1+\gamma_2\kappa_2$、直积 $\kappa_1\otimes\kappa_2(\boldsymbol{x},\boldsymbol{z})=\kappa_1(\boldsymbol{x},\boldsymbol{z})\kappa_2(\boldsymbol{x},\boldsymbol{z})$ 也是核函数。对于任意函数 $g(x)$，$\kappa(\boldsymbol{x},\boldsymbol{z})=g(\boldsymbol{x})\kappa_1(\boldsymbol{x},\boldsymbol{z})g(\boldsymbol{z})$ 也是核函数。

6.4 软间隔与正则化

- 在现实任务中很难确定合适的核函数 → 允许支持向量机在一些样本上出错 → 加入“软间隔”概念（区别于“硬间隔”）↔ 允许某些样本不满足约束 $y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geq 1$

- 优化目标可改为

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C\sum_{i=1}^{m} \ell\big(y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) - 1\big)$$

  其中 $C > 0$ 为常数，ℓ 是损失函数，一般为“0/1 损失函数”（负则输出 1，否则输出 0），也可以是替代损失函数，如

  - hinge 损失：$\ell_{hinge}(z) = \max(0, 1 - z)$；
  - 指数损失：$\ell_{exp}(z) = \exp(-z)$；
  - 对率损失：$\ell_{log}(z) = \log(1 + \exp(-z))$.

  后续分析以 hinge 损失为例。

- 引入松弛变量 $\xi_i \geq 0$，则得到“软间隔支持向量机”：

$$\begin{aligned} \min_{\boldsymbol{w},b,\xi_i} \quad & \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C\sum_{i=1}^{m} \xi_i \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^{\mathrm{T}} + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, 2, \ldots, m. \end{aligned}$$

  仿照 6.2 节的步骤，先用拉格朗日乘子法得到拉格朗日函数（拉格朗日乘子为 $\alpha_i \geq 0, \mu_i \geq 0$）：

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}, \boldsymbol{\mu}) = \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C\sum_{i=1}^{m} \xi_i + \sum_{i=1}^{m} \alpha_i(1 - \xi_i - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b)) - \sum_{i=1}^{m} \mu_i\xi_i$$

  分别对 $\boldsymbol{w}, b, \xi_i$ 偏导令为零可得 $\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i$、$0 = \sum_{i=1}^{m} \alpha_i y_i$、$C = \alpha_i + \mu_i$。代入得到对偶问题

$$\max_{\boldsymbol{\alpha}} \quad \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i y_i = 0$$
$$0 \leq \alpha_i \leq C, i = 1, 2, \ldots, m.$$

对此 KKT 条件要求:

$$\begin{cases} \alpha_i \geq 0, \quad \mu_i \geq 0, \\ y_i f(\boldsymbol{x}_i) - 1 + \xi_i \geq 0, \\ \alpha_i (y_i f(\boldsymbol{x}_i) - 1 + \xi_i) = 0, \\ \xi_i \geq 0, \quad \mu_i \xi_i = 0 \end{cases}$$

若 $\alpha_i = 0$ 则该样本无影响，若 $\alpha_i > 0$ 则有 $y_i f(\boldsymbol{x}_i) = 1 - \xi_i$，该样本是支持向量。若 $\alpha_i < C$，则 $\mu_i > 0$，故 $\xi_i = 0$，该样本位于最大间隔边界上。若 $\alpha_i = C$ 则有 $\mu_i = 0$，此时若 $\xi_i \leq 1$ 则该样本落在最大间隔内部，否则该样本被错误分类。软间隔支持向量机的最终模型仍然仅与支持向量有关。

- 共性：优化目标中，第一项用来描述划分超平面的间隔大小，另一项用于表述训练集上的误差。一般形式为

$$\min_f \Omega(f) + C \sum_{i=1}^{m} \ell(f(\boldsymbol{x}_i), y_i)$$

其中

  - 该形式可称为“正则化”问题。
  - $\Omega(f)$：结构风险。描述模型 f 的某些性质，为引入领域知识和用户意图提供途径，并且可以削减假设空间，降低过拟合风险。它也被称为正则化项。
  - $\sum_{i=1}^{m} \ell(f(\boldsymbol{x}_i), y_i)$：经验风险。描述模型与训练数据的契合程度。
  - C：用于对二者进行折中。被称为正则化常数。

6.5 支持向量回归

- 对于回归问题，支持向量回归（SVR）假设能够容忍 $f(\boldsymbol{x})$ 与 y 之间最多有 ϵ 的偏差，即差值大于 ϵ 时才计算损失 $\rightarrow$ 以 $f(\boldsymbol{x})$ 为中心，构

建一个宽度为 2ϵ 的间隔带。落入此带的训练样本被认为是预测正确。

- SVR 问题的目标:

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C \sum_{i=1}^{m} \ell_\epsilon(f(\boldsymbol{x}_i) - y_i)$$

其中 C 为正则化常数，ℓ_ϵ 是 ϵ-不敏感损失函数:

$$\ell_\epsilon(z) = \begin{cases} 0, & \text{if} \quad |z| \leq \epsilon, \\ |z| - \epsilon, & \text{otherwise.} \end{cases}$$

- 求解过程: 引入两个松弛变量 → 引入四个拉格朗日乘子 → 得到对偶问题 → 加入 KKT 条件 → 表出 SVR 解的形式、$\boldsymbol{w}$、b

6.6 核方法

- 不论是 SVM 还是 SVR，学得的模型总能表示成核函数 $\kappa(\boldsymbol{x}, \boldsymbol{x}_i)$ 的线性组合。

- "表示定理":

Theorem 2. 令 $\mathbb{H}$ 为核函数 κ 对应的再生核希尔伯特空间，$\parallel h \parallel_{\mathbb{H}}$ 表示 $\mathbb{H}$ 空间中关于 h 的范数，对于任意单调递增函数 $\Omega : [0, \infty] \mapsto \mathbb{R}$ 和任意非负损失函数 $\ell : \mathbb{R}^m \mapsto [0, \infty]$，优化问题

$$\min_{h \in \mathbb{H}} \quad F(h) = \Omega(\parallel h \parallel_{\mathbb{H}}) + \ell\big(h(\boldsymbol{x}_1), h(\boldsymbol{x}_2), \ldots, h(\boldsymbol{x}_m)\big)$$

的解总可写为

$$h^*(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i)$$

- 利用核方法将线性学习器拓展为非线性学习器。以线性判别分析拓展为核线性判别分析（KLDA）为例:

1. 假设可通过某种映射 $\phi : \mathcal{X} \mapsto \mathbb{F}$ 将样本映射到一个特征空间 $\mathbb{F}$，然后在 $\mathbb{F}$ 中执行线性判别分析，以求得

$$h(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}} \phi(\boldsymbol{x})$$

2. KLDA 学习目标为

$$\max_{\boldsymbol{w}} J(\boldsymbol{w}) = \frac{\boldsymbol{w}^{\mathrm{T}} \boldsymbol{S}_b^{\phi} \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}} \boldsymbol{S}_w^{\phi} \boldsymbol{w}}$$

设 $i \in \{0,1\}$，均值及两个散度矩阵分别为

$$\boldsymbol{\mu}_i^{\phi} = \frac{1}{m_i} \sum_{\boldsymbol{x} \in X_i} \phi(\boldsymbol{x})$$

$$\boldsymbol{S}_b^{\phi} = (\boldsymbol{\mu}_1^{\phi} - \boldsymbol{\mu}_0^{\phi})(\boldsymbol{\mu}_1^{\phi} - \boldsymbol{\mu}_0^{\phi})^{\mathrm{T}}$$

$$\boldsymbol{S}_w^{\phi} = \sum_{i=0}^{1} \sum_{\boldsymbol{x} \in X_i} (\phi(\boldsymbol{x}) - \boldsymbol{\mu}_i^{\phi})(\phi(\boldsymbol{x}) - \boldsymbol{\mu}_i^{\phi})^{\mathrm{T}}$$

3. 此时用核函数 $\kappa(\boldsymbol{x}, \boldsymbol{x}_i) = \phi(\boldsymbol{x}_i)^{\mathrm{T}} \phi(\boldsymbol{x})$ 来隐式地表达映射 ϕ 和特征空间 $\mathbb{F}$。对于表示定理中的优化问题，把 $J(\boldsymbol{w})$ 作为损失函数 ℓ，令 $\Omega \equiv 0$，由此函数 $h(\boldsymbol{x})$ 可写为

$$h(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i)$$

结合步骤 1 中的式子，可推出

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i \phi(\boldsymbol{x}_i)$$

4. 令 $\mathbf{K}$ 为核函数 κ 所对应的核矩阵。令 $\mathbf{1}_i \in \{1,0\}^{m \times 1}$ 为第 i 类样本的指示向量。

$$\hat{\boldsymbol{\mu}}_0 = \frac{1}{m_0} \mathbf{K} \mathbf{1}_0 \qquad \hat{\boldsymbol{\mu}}_1 = \frac{1}{m_1} \mathbf{K} \mathbf{1}_1$$

$$\mathbf{M} = (\hat{\boldsymbol{\mu}}_0 - \hat{\boldsymbol{\mu}}_1)(\hat{\boldsymbol{\mu}}_0 - \hat{\boldsymbol{\mu}}_1)^{\mathrm{T}}$$

$$\mathbf{N} = \mathbf{K}\mathbf{K}^{\mathrm{T}} - \sum_{i=0}^{1} m_i \hat{\boldsymbol{\mu}}_i \hat{\boldsymbol{\mu}}_i^{\mathrm{T}}$$

于是学习目标等价于

$$\max_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{M}\boldsymbol{\alpha}}{\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{N}\boldsymbol{\alpha}}$$

5. 用 LDA 方法求解得到 $\boldsymbol{\alpha}$，再由 $h(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i)$ 得到投影函数 $h(\boldsymbol{x})$。

- 一个推荐的 SVM 开源库：LIBSVM

7 贝叶斯分类器

7.1 贝叶斯决策论

- 贝叶斯决策论：基于所有与分类任务相关的概率和误判损失来选择最优的类别标记。
- 符号说明：c_i 表示被分类为第 i 类，即 $c_i \in \{c_1, c_2, \ldots, c_N\} = \mathcal{Y}$；$\lambda_{ij}$ 表示将一个真实标记为 c_j 的样本误分类为 c_i 所产生的损失。$P(c_i \mid \boldsymbol{x})$ 是一种后验概率，表示将样本 $\boldsymbol{x}$ 分类为 c_i 所产生的期望损失。对应的条件风险为 $R(c_i \mid \boldsymbol{x}) = \sum_{i=1}^{N} \lambda_{ij} P(c_j \mid \boldsymbol{x})$
- 目标：寻找一个判定准则 $h : \mathcal{X} \mapsto \mathcal{Y}$ 以最小化总体风险

$$R(h) = \mathbb{E}_{\boldsymbol{x}}[R(h(\boldsymbol{x}) \mid \boldsymbol{x})]$$

- 贝叶斯判定准则：为最小化总体风险，只需在每个样本上选择那个能使条件风险 $R(c \mid \boldsymbol{x})$ 最小的类别标记，即

$$h^*(\boldsymbol{x}) = \arg_{c \in \mathcal{Y}} \min R(c \mid \boldsymbol{x})$$

此时 h^* 被称为贝叶斯最优分类器，与之对应的总体风险 $R(h^*)$ 称为贝叶斯风险。

- 若目标是最小化分类错误率，则 $\lambda_{ij} = \begin{cases} 0, & \text{if} \quad i = j; \\ 1, & \text{otherwise} \end{cases}$，此时

$$R(c \mid \boldsymbol{x}) = 1 - P(c \mid \boldsymbol{x}) \quad h^*(\boldsymbol{x}) = \arg_{c \in \mathcal{Y}} \max P(c \mid \boldsymbol{x})$$

- 获得后验概率 $P(c \mid \boldsymbol{x})$ 的方法：
  - 判别式模型：直接建模 $P(c \mid \boldsymbol{x})$ 来预测 c（决策树、BP 神经网络、支持向量机）
  - 生成式模型：先对联合概率分布 $P(\boldsymbol{x}, c)$ 建模，然后获得 $P(c \mid \boldsymbol{x})$。此处
    $$P(c \mid \boldsymbol{x}) = \frac{P(c)P(\boldsymbol{x} \mid c)}{P(\boldsymbol{x})}$$
    * $P(c)$：类先验概率。表达了样本空间中各类样本所占的比例。当训练集包含充足的独立同分布样本时，可通过各样本出现的概率进行估计。
    * $P(\boldsymbol{x} \mid c)$：类条件概率。求法见下一节。
    * $P(\boldsymbol{x})$：证据因子。与标记无关，可用于归一化。

7.2 极大似然估计

- 极大似然估计在概率论与数理统计课上学过。这里假设 $P(\boldsymbol{x} \mid c)$ 具有确定的形式并且被参数向量 $\boldsymbol{\theta}_c$ 唯一确定。目的是利用训练集 D 估计参数 $\boldsymbol{\theta}_c$。
$$P(D_c \mid \boldsymbol{\theta}_c) = \prod_{\boldsymbol{x} \in D_c} P(\boldsymbol{x} \mid \boldsymbol{\theta}_c)$$
$$LL(\boldsymbol{\theta}_c) = \log P(D_c \mid \boldsymbol{\theta}_c) = \sum_{\boldsymbol{x} \in D_c} \log P(\boldsymbol{x} \mid \boldsymbol{\theta}_c)$$
$$\hat{\boldsymbol{\theta}}_c = \arg_{\boldsymbol{\theta}_c} \max LL(\boldsymbol{\theta}_c)$$

7.3 朴素贝叶斯分类器

- 属性条件独立性假设：对已知类别，假设所有属性相互独立。
- 基于属性条件独立性假设的类后验概率、贝叶斯判定准则：
$$P(c \mid \boldsymbol{x}) = \frac{P(c)P(\boldsymbol{x} \mid c)}{P(\boldsymbol{x})} = \frac{P(c)}{P(\boldsymbol{x})} \prod_{i=1}^{d} P(x_i \mid c)$$
$$h_{nb}(\boldsymbol{x}) = \arg_{c \in \mathcal{Y}} \max P(c) \prod_{i=1}^{d} P(x_i \mid c)$$

- 类先验概率：

$$P(c) = \frac{|D_c|}{|D|}$$

- 离散属性的条件概率（D_{c,x_i} 指 D_c 中在第 i 个属性上取值为 x_i 的样本组成的集合）：

$$P(x_i \mid c) = \frac{|D_{c,x_i}|}{|D_c|}$$

连续属性的条件概率密度函数（假设 $p(x_i \mid c) \sim \mathcal{N}(\mu_{c,i}, \sigma_{c,i}^2)$）：

$$p(x_i \mid c) = \frac{1}{\sqrt{2\pi}\sigma_{c,i}} \exp\left(- \frac{(x_i - \mu_{c,i})^2}{2\sigma_{c,i}^2} \right)$$

- 拉普拉斯修正：为了避免其它属性携带的信息被训练集中未出现的属性值抹去（导致概率估值为零）。设 N 为训练集 D 中可能的类别数，N_i 表示第 i 个属性可能的取值数。可修正如下：

$$\hat{P}(c) = \frac{|D_c| + 1}{|D| + N}$$

$$\hat{P}(x_i \mid c) = \frac{|D_{c,x_i}| + 1}{|D_c| + N_i}$$

7.4 半朴素贝叶斯分类器

- 基本思想：适当考虑一部分属性间的相互依赖信息，既不需进行完全联合概率计算，又不至于彻底忽略了比较强的属性依赖关系。

- 独依赖估计（ODE）：假设每个属性在类别之外最多仅依赖于一个其他属性。

$$P(c \mid \boldsymbol{x}) \propto P(c) \prod_{i=1}^{d} P(x_i \mid c, pa_i)$$

其中 pa_i 为属性 x_i 所依赖的属性，称为 x_i 的父属性。问题的关键转化为：如何确定每个属性的父属性？

  - SPODE：假设所有属性都依赖于同一个属性（超父）。通过交叉验证等模型选择方法确定超父属性。
  - TAN：基于最大带权重生成树算法。先计算任意两个属性间的“条

件互信息”:

$$I(x_i, x_j \mid y) = \sum_{x_i, x_j; c \in \mathcal{Y}} P(x_i, x_j \mid c) \log \frac{P(x_i, x_j \mid c)}{P(x_i \mid c) P(x_j \mid c)}$$

然后以属性为结点构建完全图，任意两个结点之间边的权重设为 $I(x_i, x_j \mid y)$，构建它的最大带权重生成树，挑选根变量，将边置为有向。最后加入类别结点 y，增加从 y 到每个属性的有向边。

– AODE：一种基于集成学习机制的独依赖分类器。尝试将每个属性作为超父来构建 SPODE，将具有足够训练数据支撑的 SPODE 集成起来作为最终结果。

$$P(c \mid \boldsymbol{x}) \propto \sum_{\substack{i=1 \\ |D_{x_i}| \geq m'}}^{d} P(c, x_i) \prod_{j=1}^{d} P(x_j \mid c, x_i)$$

其中 D_{x_i} 是在第 i 个属性上取值为 x_i 的样本的集合。m' 为阈值常数。此处

$$\hat{P}(c, x_i) = \frac{|D_{c,x_i}| + 1}{|D| + N \times N_i}$$

$$\hat{P}(x_j \mid c, x_i) = \frac{|D_{c,x_i,x_j}| + 1}{|D_{c,x_i}| + N_j}$$

- 若将属性 pa_i 替换成包含 k 个属性的集合 $\boldsymbol{pa}_i$，从而将 ODE 拓展为 kDE。随着 k 的增加，准确估计概率 $P(x_i \mid y, \boldsymbol{pa}_i)$ 所需样本数量将以指数级增加。

7.5 贝叶斯网

也称为“信念网”(belief network)。借助有向无环图 (DAG) 来刻画属性之间的依赖关系。使用条件概率表 (CPT) 来描述属性的联合概率分布。一个贝叶斯网 B 可以由结构 G 和参数 Θ 构成。$B = \langle G, \Theta \rangle$。假设属性 x_i 在 G 的父结点集为 π_i，则 Θ 包含了每个属性的条件概率表 $\theta_{x_i|\pi_i} = P_B(x_i \mid \pi_i)$。

7.5.1 结构

- 给定父节点集，假设每个属性与它的非后裔属性独立，则 $B = \langle G, \Theta \rangle$ 将属性 $x_1, x_2, \ldots, x_d$ 的联合概率分布定义为

$$P_B(x_1, x_2, \ldots, x_d) = \prod_{i=1}^{d} P_B(x_i \mid \pi_i) = \prod_{i=1}^{d} \theta_{x_i|\pi_i}$$

- 三个变量的典型依赖关系——

  1. 同父结构：$x_1 \to x_3, x_1 \to x_4$。有 $x_3 \perp x_4 \mid x_1$，而 $x_3 \not\perp\!\!\!\perp x_4$ 不成立。（当 x_1 完全未知时的边界独立性）
  2. V 型结构：$x_1 \to x_4, x_2 \to x_4$。有 $x_1 \not\perp\!\!\!\perp x_2$，而 $x_1 \perp x_2 \mid x_4$ 不成立。
  3. 顺序结构：$z \to x, x \to y$。有 $y \perp z \mid x$，而 $y \not\perp\!\!\!\perp z$ 不成立。

- 使用“有向分离”（D-separation）将有向图转化为无向图，分析变量间的条件独立性。

  1. 找出有向图中的所有 V 型结构，在 V 型结构的两个父结点之间加上一条无向边。
  2. 将所有有向边改为无向边。

  产生的无向图称为“道德图”（moral graph）。令父结点相连的过程称为“道德化”（moralization）。

- 基于道德图的条件独立性判断方法：假定道德图中有变量 x, y 和变量集合 $\boldsymbol{z} = \{z_i\}$，若变量 x 和 y 能在图上被 $\boldsymbol{z}$ 分开，即从道德图中将 $\boldsymbol{z}$ 去除后，x 和 y 分属两个连通分支，则称变量 x 和 y 被 $\boldsymbol{z}$ 有向分离，$x \perp y \mid \boldsymbol{z}$ 成立。

7.5.2 学习

- 贝叶斯网的首要任务：根据训练数据集来找出结构最“恰当”的贝叶斯网。

- 常用方法：评分搜索。定义评分函数，以此评估贝叶斯网与训练数据的契合程度，以此寻找结构最优的贝叶斯网。

- 学习目标：找到综合编码长度（包括描述网络和编码数据）最短的贝叶斯网。即“最小描述长度”（MDL）准则。
- 给定训练集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$，贝叶斯 $B = \langle G, \Theta \rangle$ 在 D 上的评分函数可写为

$$s(B \mid D) = f(\theta)|B| - LL(B \mid D)$$

  其中 $|B|$ 是贝叶斯网的参数个数，$f(\theta)$ 表示描述每个参数 θ 所需的字节数，

$$LL(B \mid D) = \sum_{i=1}^{m} \log P_B(\boldsymbol{x}_i)$$

  是贝叶斯网的对数似然。

  - 若 $f(\theta) = 1$，得到 AIC（Akaike Information Criterion），即 $AIC(B \mid D) = |B| - LL(B \mid D)$。
  - 若 $f(\theta) = \frac{1}{2}\log m$，得到 BIC（Bayesian Information Criterion），即 $BIC(B \mid D) = \frac{\log m}{2}|B| - LL(B \mid D)$。
  - 若 $f(\theta) = 0$，则学习任务退化为极大似然估计。
- 若 G 固定，则 $f(\theta)|B|$ 为常数，问题等价于对 Θ 进行极大似然估计。这里每个参数 $\theta_{x_i|\pi_i}$ 能直接在训练数据 D 上通过经验估计获得：

$$\theta_{x_i|\theta_i} = \hat{P}_D(x_i \mid \pi_i)$$

  其中 $\hat{P}_D(\cdot)$ 是 D 上的经验分布。
- 求解方法
  - 从所有可能的网络结构空间搜索最优贝叶斯网结构是 NP 难问题。
  - 贪心法（近似解）：从某个网络结构出发，每次调整一条边，直到评分函数值不再降低为止。
  - 添加约束（近似解）：比如将网络结构限定为树形结构。

7.5.3 推断

- 推断：通过已知变量观测值来推测待查询变量的过程。证据：已知变量观测值。

- 直接精确计算后验概率：NP-hard。
- 近似推断方法：吉布斯采样(Gibbs sampling)。设 $\boldsymbol{Q}=\{Q_1,Q_2,\ldots,Q_n\}$ 表示待查询变量，$\boldsymbol{E}=\{E_1,E_2,\ldots,E_k\}$ 为证据变量，其取值为 $\boldsymbol{e}=\{e_1,e_2,\ldots,e_k\}$，目标是计算后验概率 $P(\boldsymbol{Q}=\boldsymbol{q}\mid\boldsymbol{E}=\boldsymbol{e})$，其中 $\boldsymbol{q}=\{q_1,q_2,\ldots,q_n\}$ 是待查询变量的一组取值。吉布斯采样就是在贝叶斯网所有变量的联合状态空间与证据 $\boldsymbol{E}=\boldsymbol{e}$ 一致的子空间中进行“随机漫步”。

Algorithm 2 吉布斯采样算法

Input: 贝叶斯网 $B=\langle G,\Theta\rangle$，采样次数 T，证据变量 $\boldsymbol{E}$ 及其取值 $\boldsymbol{e}$，待查询变量 $\boldsymbol{Q}$ 及其 $\boldsymbol{q}$。
Output: $P(\boldsymbol{Q}=\boldsymbol{q}\mid\boldsymbol{E}=\boldsymbol{e})\simeq\frac{n_q}{T}$

$n_q=0$
$\boldsymbol{q}^0=$ 对$\boldsymbol{Q}$随机赋初值
for $t=1,2,\ldots,T$ **do**
  for $Q_i\in\boldsymbol{Q}$ **do**
    $\boldsymbol{Z}=\boldsymbol{E}\cup\boldsymbol{Q}\backslash\{Q_i\}$
    $\boldsymbol{z}=\boldsymbol{e}\cup\boldsymbol{q}^{t-1}\backslash\{q_i^{t-1}\}$
    根据 B 计算分布 $P_B(Q_i\mid\boldsymbol{Z}=\boldsymbol{z})$
    $q_i^t=$ 根据$P_B(Q_i\mid\boldsymbol{Z}=\boldsymbol{z})$采样所获$Q_i$取值
    $\boldsymbol{q}^t=$ 将$\boldsymbol{q}^{t-1}$中的q_i^{t-1}用q_i^t替换
  end for
  if $\boldsymbol{q}^t=\boldsymbol{q}$ **then**
    $n_q=n_q+1$
  end if
end for

7.6 EM 算法

- 样本中可能出现隐变量（未观测的、变量值未知的变量）。设已观测变量集为 $\boldsymbol{X}$、隐变量集为 $\boldsymbol{Z}$、模型参数为 $\boldsymbol{\Theta}$。通过对 $\boldsymbol{Z}$ 计算期望，最大化已观测数据的对数边际似然。

$$LL(\Theta\mid\boldsymbol{X})=\ln P(\boldsymbol{X}\mid\Theta)=\ln\sum_{\boldsymbol{Z}}P(\boldsymbol{X},\boldsymbol{Z}\mid\Theta)$$

- EM 算法（期望最大值算法）：若参数 Θ 已知，则可根据训练数据推

断出最优隐变量 $\boldsymbol{Z}$ 的值；反之若 $\boldsymbol{Z}$ 已知，则可对参数 Θ 已知做极大似然估计。

- 原型：
  1. 基于 Θ^t 推断隐变量 $\boldsymbol{Z}$ 的期望，记为 $\boldsymbol{Z}^t$；
  2. 基于已观测变量 $\boldsymbol{X}$ 和 $\boldsymbol{Z}^t$ 对参数 Θ 做极大似然估计，记为 Θ^{t+1}。
  3. 重复以上两步直至收敛。
- 不取 $\boldsymbol{Z}$ 的期望，而是基于 Θ^t 计算隐变量 $\boldsymbol{Z}$ 的概率分布 $P(\boldsymbol{Z} \mid \boldsymbol{X}, \Theta^t)$。
  1. E 步——以当前参数 Θ^t 推断隐变量分布 $P(\boldsymbol{Z}|\boldsymbol{X}, \Theta^t)$，并计算对数似然 $LL(\Theta \mid \boldsymbol{X}, \boldsymbol{Z})$ 关于 $\boldsymbol{Z}$ 的期望

$$Q(\Theta \mid \Theta^t) = \mathbb{E}_{\boldsymbol{Z}|\boldsymbol{X},\Theta^t} LL(\Theta \mid \boldsymbol{X}, \boldsymbol{Z})$$

  2. M 步——寻找参数最大化期望似然

$$\Theta^{t+1} = \arg_{\Theta} \max Q(\Theta \mid \Theta^t)$$

8 集成学习

8.1 个体与集成

- 集成学习（多分类器系统）：通过构建并结合多个学习器来完成学习任务。
- 同质集成：集成中只包含同种类型的个体学习器。此时个体学习器可称为“基学习器”。
- 异质集成：集成中包含不同类型的个体学习器。此时个体学习器可称为“组件学习器”。
- 个体学习器的准确性和多样性存在冲突。研究核心：产生并结合“好而不同”的个体学习器。
- 集成学习方法
  - 个体学习器间强依赖，必须串行生成 →Boosting

– 个体学习器间不强依赖，可同时生成 →Bagging 和 Random Forest

8.2 Boosting

- Boosting：一族可将弱学习器提升为强学习器的算法。先从初始训练集训练出一个基学习器，根据其表现调整训练样本分布，更多关注做错的训练样本，基于此训练下一个基学习器。如此直至基学习器数目达到事先指定的值 T，最终将 T 个基学习器加权结合。

- AdaBoost 算法(标准型□只适用于二分类任务)基于“加性模型 $H(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x})$，最小化指数损失函数

$$\ell_{\exp}(H \mid \mathcal{D}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H(\boldsymbol{x})}]$$

Algorithm 3 AdaBoost 算法

Input: 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$，基学习算法 $\mathfrak{L}$，训练轮数 T

Output: $H(\boldsymbol{x}) = \operatorname{sign}\big(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x})\big)$

$\mathcal{D}_1(\boldsymbol{x}) = 1/m$
for $t = 1, 2, \ldots, T$ **do**
  $h_t = \mathfrak{L}(D, \mathcal{D}_t)$
  $\epsilon_t = P_{\boldsymbol{x}\sim\mathcal{D}_t}(h_t(\boldsymbol{x}) \neq f(\boldsymbol{x}))$
  if $\epsilon_t > 0.5$ **then**
    break
  end if
  $\alpha_t = \frac{1}{2}\ln(\frac{1-\epsilon_t}{\epsilon_t})$
  $\mathcal{D}_{t+1}(\boldsymbol{x}) = \frac{\mathcal{D}_t(\boldsymbol{x})\exp(-\alpha_t f(\boldsymbol{x})h_t(\boldsymbol{x}))}{Z_t}$
end for

- 使用“指数损失函数”的合理性：求指数损失函数对 $H(\boldsymbol{x})$ 的偏导并令零可得到

$$H(\boldsymbol{x}) = \frac{1}{2}\ln\frac{P(f(x) = 1 \mid \boldsymbol{x})}{P(f(x) = -1 \mid \boldsymbol{x})}$$

由此 $\operatorname{sign}(H(\boldsymbol{x})) = \arg_{y\in\{-1,1\}} \max P(f(x) = y \mid \boldsymbol{x})$ 达到了贝叶斯最

优错误率。说明指数损失函数是分类任务原本 0/1 损失函数的一致的替代损失函数，且连续可微。

- 分类器权重更新公式的推导：使 $\alpha_t h_t$ 最小化指数损失函数：

$$\ell_{\exp}(\alpha_t h_t \mid \mathcal{D}_t) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}\left[e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}\right] = e^{-\alpha_t}(1-\epsilon_t) + e^{\alpha_t}\epsilon_t$$

对 α_t 求偏导并令零，可得到权重更新公式

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

- 样本分布更新公式的推导：

1. 理想的下一轮基学习器 h_t 能纠正 H_{t-1} 的全部错误，即最小化

$$\ell_{\exp}(H_{t-1} + h_t \mid \mathcal{D}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}e^{-f(\boldsymbol{x})h_t(\boldsymbol{x})}]$$

2. 使用泰勒展式，可将 $e^{-f(\boldsymbol{x})h_t(\boldsymbol{x})}$ 近似为 $1-f(\boldsymbol{x})h_t(\boldsymbol{x})+\frac{f^2(\boldsymbol{x})h_t^2(\boldsymbol{x})}{2} = \frac{3}{2} - f(\boldsymbol{x})h_t(\boldsymbol{x})$
3. 理想的基学习器

$$\begin{aligned} h_t(\boldsymbol{x}) &= \arg_h \min \ell_{\exp}(H_{t-1} + h \mid \mathcal{D}) \\ &= \arg_h \max \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\frac{e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]} f(\boldsymbol{x})h(\boldsymbol{x})\right] \end{aligned}$$

其中 $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]$ 是一个常数。

4. 令

$$\mathcal{D}_t(\boldsymbol{x}) = \frac{\mathcal{D}(\boldsymbol{x})e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}]}$$

等价于令

$$h_t(\boldsymbol{x}) = \arg_h \max \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_t}[f(\boldsymbol{x})h(\boldsymbol{x})]$$

5.

$$\begin{aligned} \mathcal{D}_{t+1}(\boldsymbol{x}) &= \frac{\mathcal{D}(\boldsymbol{x})e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}]} \\ &= \frac{\mathcal{D}(\boldsymbol{x})e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}]} \end{aligned}$$

$$=\mathcal{D}_t(\boldsymbol{x})\cdot e^{-f(\boldsymbol{x})\alpha_t h_t(\boldsymbol{x})}\frac{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_{t-1}(\boldsymbol{x})}\right]}{\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[e^{-f(\boldsymbol{x})H_t(\boldsymbol{x})}\right]}$$

- 重赋权法：在训练的每一轮中，根据样本分布为每个训练样本重新赋予一个权重。在训练的每一轮检查当前生成的基学习器是否满足基本条件，一旦不满足，则抛弃当前基学习器，学习过程停止。
- 重采样法：在训练的每一轮中，根据样本分布对训练集重新进行采样，再据此对基学习器进行训练。可以不抛弃不满足条件的当前基学习器，使学习过程维持到预设的 T 轮完成。
- 偏差-方差分解角度：关注降低偏差

8.3 Bagging 与随机森林

采样思路：希望基学习器有较大差异 & 每个基学习器使用的训练数据不能太小 → 使用相互有交叠的采样子集

8.3.1 Bagging

- 采样方法：自助采样法
- 输出结合方法：投票法（分类）、平均法（回归）
- 算法描述（$\mathcal{D}_{bs}$ 是自助采样产生的样本分布）

Algorithm 4 Bagging 算法

Input: 训练集 $D=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\ldots,(\boldsymbol{x}_m,y_m)\}$，基学习算法 $\mathfrak{L}$，训练轮数 T

Output: $H(\boldsymbol{x})=\arg_{y\in\mathcal{Y}}\max\sum_{t=1}^{T}\mathbb{I}(h_t(\boldsymbol{x})=y)$

1: **for** $t=1,2,\ldots,T$ **do**
2: $\quad h_t=\mathfrak{L}(D,\mathcal{D}_{bs})$
3: **end for**

- 计算复杂度：$T(O(m)+O(s))$。高效。
- 包外估计：因自助采样，有约 36.8% 的样本可用作验证集。令 D_t 表示 h_t 实际使用的训练样本集，$H^{oob}(\boldsymbol{x})$ 表示对样本 $\boldsymbol{x}$ 的包外预测。于

是

$$H^{oob}(\boldsymbol{x}) = \arg_{y\in\mathcal{Y}} \max \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y) \cdot \mathbb{I}(\boldsymbol{x} \notin D_t)$$

$$\epsilon^{oob} = \frac{1}{|D|} \sum_{(\boldsymbol{x},y)\in D} \mathbb{I}(H^{oob}(\boldsymbol{x}) \neq y)$$

- 其他用途：决策树 → 包外样本辅助剪枝或估计各结点的后验概率以辅助对零训练样本结点的处理；神经网络 → 使用包外样本辅助早期停止以减小过拟合风险。
- 偏差-方差分解角度：关注降低方差。

8.3.2 随机森林

- 以决策树为基学习器
- 在 Bagging 集成的基础上，引入随机属性选择，对基决策树的每个结点，从该结点的属性集合中随机选择一个包含 k 个属性的子集，然后从这个子集中选择一个最优属性进行划分。推荐 $k = \log_2 d$
- 简单、易实现、开销小、被誉为“代表集成学习技术水平的方法”。训练效率常优于 Bagging。

8.4 结合策略

学习器结合的三个好处：

- 统计方面：减少因假设空间很大导致泛化性能不佳的风险
- 计算方面：降低陷入糟糕局部极小点的风险
- 表示方面：可使假设空间有所扩大，以包含真实假设

8.4.1 平均法

- 是数值型输出 $h_i(\boldsymbol{x}) \in \mathbb{R}$ 的最常见结合策略。
- 简单平均法

$$H(\boldsymbol{x}) = \frac{1}{T} \sum_{i=1}^{T} h_i(\boldsymbol{x})$$

- 加权平均法

$$H(\boldsymbol{x}) = \sum_{i=1}^{T} w_i h_i(\boldsymbol{x})$$

- 一般而言，在个体学习器性能相差较大时宜使用加权平均法，而在个体学习器性能相近时宜使用简单平均法。

8.4.2 投票法

- 针对分类任务。类别标记集合为 $\{c_1, c_2, \ldots, c_N\}$。将 h_i 在样本 $\boldsymbol{x}$ 上的预测输出表示为一个 N 维向量 $(h_i^1(\boldsymbol{x}); h_i^2(\boldsymbol{x}); \ldots; h_i^N(\boldsymbol{x}))$，其中 $h_i^j(\boldsymbol{x})$ 是 h_i 在类别标记 c_j 上的输出。

- 投票方法

  - 绝对多数投票法（若某标记得票过半数则预测为该标记，否则拒绝预测）

$$H(\boldsymbol{x}) = \begin{cases} c_j, & \text{if} \quad \sum_{i=1}^{T} h_i^j(\boldsymbol{x}) > 0.5 \sum_{k=1}^{N} \sum_{i=1}^{T} h_i^k(\boldsymbol{x}); \\ \text{reject}, & \text{otherwise.} \end{cases}$$

  - 相对多数投票法（预测为得票最多的标记，若同时有多个标记或最高票，则随机选取一个）

$$H(\boldsymbol{x}) = c_{\arg_j \max \sum_{i=1}^{T} h_i^j(\boldsymbol{x})}$$

  - 加权投票法（通常 $w_i \geq 0, \sum_{i=1}^{T} w_i = 1$）

$$H(\boldsymbol{x}) = c_{\arg_j \max \sum_{i=1}^{T} w_i h_i^j(\boldsymbol{x})}$$

- 输出值 $h_i^j(\boldsymbol{x})$ 的类型分类

  - 类标记（硬投票）：$h_i^j(\boldsymbol{x}) \in \{0, 1\}$。若 h_i 将样本 $\boldsymbol{x}$ 预测为类型 c_j 则取值为 1，否则为 0。
  - 类概率（软投票）：$h_i^j(\boldsymbol{x}) \in [0, 1]$。相当于对后验概率 $P(c_j \mid \boldsymbol{x})$ 的一个估计。

8.4.3 学习法

- 策略：通过另一个学习器来进行结合。个体学习器 → 初级学习器；用于结合的学习器 → 次级学习器/元学习器
- Stacking 算法描述

Algorithm 5 Stacking 算法

Input: 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$，初级学习算法 $\mathfrak{L}_1, \mathfrak{L}_2, \ldots, \mathfrak{L}_T$，次级学习算法 $\mathfrak{L}$。

Output: $H(\boldsymbol{x}) = h'(h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_T(\boldsymbol{x}))$

```
for t = 1, 2, ..., T do
    h_t = 𝔏_t(D);
end for
D' = ∅;
for i = 1, 2, ..., m do
    for t = 1, 2, ..., T do
        z_it = h_t(x_i);
    end for
    D' = D' ∪ ((z_i1, z_i2, ..., z_iT), y_i);
end for
h' = 𝔏(D');
```

- 一般采用交叉验证法或留一法。
- 研究表明，将初级学习器的输出类概率作为次级学习器的输入属性，用多响应线性回归（MLR）作为次级学习算法效果较好。

8.5 多样性

8.5.1 误差-分歧分解

- 对示例 $\boldsymbol{x}$，学习器 h_i 的“分歧”（ambiguity）：

$$A(h_i \mid \boldsymbol{x}) = \big(h_i(\boldsymbol{x}) - H(\boldsymbol{x})\big)^2$$

- 集成的“分歧”（反映个体学习器的多样性）：

$$\overline{A}(h \mid \boldsymbol{x}) = \sum_{i=1}^{T} w_i A(h_i \mid \boldsymbol{x}) = \sum_{i=1}^{T} w_i \big(h_i(\boldsymbol{x}) - H(\boldsymbol{x})\big)^2$$

- 个体学习器 h_i 和集成 H 的平方误差

$$E(h_i \mid \boldsymbol{x}) = \big(f(\boldsymbol{x}) - h_i(\boldsymbol{x})\big)^2$$

$$E(H \mid \boldsymbol{x}) = \big(f(\boldsymbol{x}) - H(\boldsymbol{x})\big)^2$$

- 对于全样本，个体的泛化误差、分歧项为

$$E_i = \int E(h_i \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}$$

$$A_i = \int A(h_i \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}$$

- 对于全样本，集成的泛化误差、个体学习器泛化误差的加权均值、个体学习器的加权分分歧值为

$$E = \int E(H \mid \boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}$$

$$\overline{E} = \sum_{i=1}^{T} w_i E_i$$

$$\overline{A} = \sum_{i=1}^{T} w_i A_i$$

- 误差-分歧分解（个体学习器准确性越高、多样性越大，则集成越好）

$$E = \overline{E} - \overline{A}$$

8.5.2 多样性度量

以如下二分类任务为例（$a+b+c+d=m$）

	$h_i=+1$	$h_i=-1$
$h_j=+1$	a	c
$h_j=-1$	b	d

- 不合度量（disagreement measure）：值越大多样性越大

$$dis_{ij} = \frac{b+c}{m}$$

- 相关系数（correlation coefficient）：无关 0，正相关正，负相关负

$$\rho_{ij} = \frac{ad - bc}{\sqrt{(a+b)(a+c)(c+d)(b+d)}}$$

- Q-统计量：与相关系数符号相同，且 $|Q_{ij}| \geq |\rho_{ij}|$

$$Q_{ij} = \frac{ad - bc}{ad + bc}$$

- κ-统计量：若两个分类器在 D 上完全一致则 $\kappa = 1$，若仅是偶然达成一致，则 $\kappa = 0$。

$$\kappa = \frac{p_1 - p_2}{1 - p_2}$$

$$p_1 = \frac{a+d}{m}$$

$$p_2 = \frac{(a+b)(a+c) + (c+d)(b+d)}{m^2}$$

8.5.3 多样性增强

- 数据样本扰动：基于采样法。不稳定基学习器（样本扰动敏感）：决策树、神经网络；稳定基学习器：线性学习器、支持向量机、朴素贝叶斯、k 近邻学习器。

- 输入属性扰动：

Algorithm 6 随机子空间算法

Input: 训练集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$，基学习算法 $\mathfrak{L}$，基学习器数 T，子空间属性数 d'

Output: $H(\boldsymbol{x}) = \arg_{y \in \mathcal{Y}} \max \sum_{t=1}^{T} \mathbb{I}(h_t(\mathrm{Map}_{\mathcal{F}_t}(\boldsymbol{x})) = y)$

1: **for** $t = 1, 2, \ldots, T$ **do**
2: $\quad \mathcal{F}_t = \mathrm{RS}(D, d')$
3: $\quad D_t = \mathrm{Map}_{\mathcal{F}_t}(D)$
4: $\quad h_t = \mathfrak{L}(D_t)$
5: **end for**

- 输出表示扰动：对输出表示进行操纵

  - 翻转法：随机改变一些训练样本的标记

  - 输出调制法：将分类输出转化为回归输出后建构个体学习器
  - ECOC 法：利用纠错输出码将多分类任务拆解为一系列二分类任务

- 算法参数扰动：随机设置不同的参数。对参数较少的算法，可通过将其学习过程中某些环节用其他类似方式代替。

9 聚类

9.1 聚类任务

假定样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ 包含 m 个无标记样本，聚类算法将 D 划分为 k 个不相交的簇 $\{C_l \mid l = 1, 2, \ldots, k\}$，其中 $C_{l'} \cap_{l' \neq l} C_l =$ 且 $D = \cup_{l=1}^{k} C_l$。用 $\lambda_j \in \{1, 2, \ldots, k\}$ 表示样本 $\boldsymbol{x}_j$ 的“簇标记”，即 $\boldsymbol{x}_j \in C_{\lambda_j}$。于是聚类结果可以用簇标记向量 $\boldsymbol{\lambda} = (\lambda_1; \lambda_2; \ldots; \lambda_m)$ 表示。

9.2 性能度量

- 聚类性能度量分类：外部指标（与某个参考模型进行比较）；内部指标（直接考察聚类结果）
- 设带星号的为参考模型对应的变量。考虑两两配对，定义

$$
\begin{aligned}
a &= |SS|, & SS &= \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid \lambda_i = \lambda_j, \lambda_i^* = \lambda_j^*, i < j\} \\
b &= |SD|, & SD &= \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid \lambda_i = \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\} \\
c &= |DS|, & DS &= \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid \lambda_i \neq \lambda_j, \lambda_i^* = \lambda_j^*, i < j\} \\
d &= |DD|, & DD &= \{(\boldsymbol{x}_i, \boldsymbol{x}_j) \mid \lambda_i \neq \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\}
\end{aligned}
$$

  其中 $a + b + c + d = m(m-1)/2$

- 外部指标（以下三者均在 $[0, 1]$ 区间，越大越好）
  - Jaccard 系数（JC）

$$\mathrm{JC} = \frac{a}{a+b+c}$$

  - FM 指数（FMI）

$$\mathrm{FMI} = \sqrt{\frac{a}{a+b} \cdot \frac{a}{a+c}}$$

- Rand 指数（RI）

$$\mathrm{RI} = \frac{2(a+d)}{m(m-1)}$$

- 各种距离函数
  - 簇 C 内样本间平均距离

$$\mathrm{avg}(C) = \frac{2}{|C|(|C|-1)} \sum_{1 \le i < j \le |C|} \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

  - 簇 C 内样本间最远距离

$$\mathrm{diam}(C) = \max_{1 \le i < j \le |C|} \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

  - 簇 C_i 与簇 C_j 最近样本间距离

$$d_{\min}(C_i, C_j) = \min_{\boldsymbol{x}_i \in C_i, \boldsymbol{x}_j \in C_j} \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

  - 簇 C_i 与簇 C_j 中心点间距离

$$d_{\mathrm{cen}}(C_i, C_j) = \mathrm{dist}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)$$

    其中

$$\boldsymbol{\mu} = \frac{1}{|C|} \sum_{1 \le i \le |C|} \boldsymbol{x}_i$$

- 内部指标（DBI 越小越好，DI 越大越好）
  - DB 指数（DBI）

$$\mathrm{DBI} = \frac{1}{k} \sum_{i=1}^{k} \max_{j \ne i} \left(\frac{\mathrm{avg}(C_i) + \mathrm{avg}(C_j)}{d_{\mathrm{cen}}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)} \right)$$

  - Dunn 指数（DI）

$$\mathrm{DI} = \min_{1 \le i \le k} \left\{ \min_{j \ne i} \left(\frac{d_{\min}(C_i, C_j)}{\max_{1 \le l \le k} \mathrm{diam}(C_l)} \right) \right\}$$

9.3 距离计算

- 函数 $\text{dist}(\cdot,\cdot)$ 若是一个距离度量，需满足四个基本性质：
  1. 非负性：$\text{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)\geq 0$
  2. 同一性：$\text{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)=0$ 当且仅当 $\boldsymbol{x}_i=\boldsymbol{x}_j$
  3. 对称性：$\text{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)=\text{dist}(\boldsymbol{x}_j,\boldsymbol{x}_i)$
  4. 直递性：$\text{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)\leq \text{dist}(\boldsymbol{x}_i,\boldsymbol{x}_k)+\text{dist}(\boldsymbol{x}_k.\boldsymbol{x}_j)$
- 给定样本 $\boldsymbol{x}_i=(x_{i1};x_{i2};\ldots;x_{in})$ 和 $\boldsymbol{x}_j=(x_{j1};x_{j2};\ldots;x_{jn})$。闵可夫斯基距离（Minkowski distance）【可用于有序属性】：

$$\text{dist}_{\text{mk}}(\boldsymbol{x}_i,\boldsymbol{x}_j)=\left(\sum_{u=1}^{n}|x_{iu}-x_{ju}|^p\right)^{\frac{1}{p}}$$

  $p=2$ 时即为欧式距离，$p=1$ 时即为曼哈顿距离

$$\text{dist}_{\text{ed}}(\boldsymbol{x}_i,\boldsymbol{x}_j)=||\boldsymbol{x}_i-\boldsymbol{x}_j||_2=\sqrt{\sum_{u=1}^{n}|x_{iu}-x_{ju}|^2}$$

$$\text{dist}_{\text{man}}(\boldsymbol{x}_i,\boldsymbol{x}_j)=||\boldsymbol{x}_i-\boldsymbol{x}_j||_1=\sum_{u=1}^{n}|x_{iu}-x_{ju}|$$

- VMD 距离【可用于无序属性】：$m_{u,a,i}$ 表示在第 i 个样本簇中在属性 u 上取值为 a 的样本数。

$$\text{VDM}_p(a,b)=\sum_{i=1}^{k}\left|\frac{m_{u,a,i}}{m_{u,a}}-\frac{m_{u,b,i}}{m_{u,b}}\right|^p$$

- 结合两种距离，设有 n_c 个有序属性、$n-n_c$ 个无序属性

$$\text{MinkovDM}_p(\boldsymbol{x}_i,\boldsymbol{x}_j)=\left(\sum_{u=1}^{n_c}|x_{iu}-x_{ju}|^p+\sum_{u=n_c+1}^{n}\text{VDM}_p(x_{iu},x_{ju})\right)^{\frac{1}{p}}$$

- 加权距离（以加权闵可夫斯基距离为例）

$$\text{dist}_{\text{wmk}}(\boldsymbol{x}_i,\boldsymbol{x}_j)=(w_i\cdot|x_{i1}-x_{j1}|^p+\cdots+w_n\cdot|x_{in}-x_{jn}|^p)^{\frac{1}{p}}$$

9.4 原型聚类

- 原型：样本空间中具有代表性的点
- 原型聚类算法：先对原型进行初始化，再对原型进行迭代更新求解。

9.4.1 k-均值算法

- 目标：最小化平方误差 $E = \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in C_i} ||\boldsymbol{x} - \boldsymbol{\mu}_i||_2^2$ (NP-hard)。故用贪心策略近似求解。
- 算法描述

Algorithm 7 k-均值算法

Input: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$、聚类簇数 k.
Output: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

从 D 中随机选择 k 个样本作为初始均值向量 $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k\}$;
repeat
  令 $C_i = \emptyset \quad (1 \leq i \leq k)$;
  for $j = 1, 2, \ldots, m$ **do**
    计算样本 $\boldsymbol{x}_j$ 与各均值向量 $\boldsymbol{\mu}_i \quad (1 \leq i \leq k)$ 的距离：

$$d_{ji} = ||\boldsymbol{x}_j - \boldsymbol{\mu}_i||_2$$

    根据距离最近的均值向量确定 $\boldsymbol{x}_j$ 的簇标记：

$$\lambda_j = \arg \min_{i \in \{1,2,\ldots,k\}} d_{ji}$$

    将样本 $\boldsymbol{x}_j$ 划入相应的簇：$C_{\lambda_j} = C_{\lambda_j} \cup \{\boldsymbol{x}_j\}$;
  end for
  for $i = 1, 2, \ldots, k$ **do**
    计算新均值向量：$\boldsymbol{\mu}'_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$;
    if $\boldsymbol{\mu}'_i \neq \boldsymbol{\mu}_i$ **then**
      将当前均值向量 $\boldsymbol{\mu}_i$ 更新为 $\boldsymbol{\mu}'_i$;
    else
      保持当前均值向量不变;
    end if
  end for
until 当前均值向量均未更新

9.4.2 学习向量量化

- LVQ 假设数据样本带有类别标记，利用这些监督信息辅助聚类。
- 目标：学得一组 n 维原型向量 $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_q\}$，每个原型向量代表一个聚类簇，簇标记 $t_i \in \mathcal{Y}$
- 算法描述

Algorithm 8 学习向量量化算法

Input: 样本集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$、原型向量个数 q、各原型向量预设的类别标记 $\{t_1, t_2, \ldots, t_q\}$、学习率 $\eta \in (0, 1)$

Output: 原型向量 $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_q\}$

初始化一组原型向量 $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_q\}$;
repeat
  从样本集 D 随机选取样本 $(\boldsymbol{x}_j, y_j)$;
  计算样本 $\boldsymbol{x}_j$ 与 $\boldsymbol{p}_i (1 \leq i \leq q)$ 的距离：$d_{ji} = ||\boldsymbol{x}_j - \boldsymbol{p}_i||_2$;
  找出与 $\boldsymbol{x}_j$ 距离最近的原型向量 p_{i^*}，$i^* = \arg\min_{i \in \{1,2,\ldots,q\}} d_{ji}$;
  if $y_j = t_{i^*}$ **then**
    $\boldsymbol{p}' = \boldsymbol{p}_{i^*} + \eta \cdot (\boldsymbol{x}_j - \boldsymbol{p}_{i^*})$;
  else
    $\boldsymbol{p}' = \boldsymbol{p}_{i^*} - \eta \cdot (\boldsymbol{x}_j - \boldsymbol{p}_{i^*})$;
  end if
  将原型向量 $\boldsymbol{p}_{i^*}$ 更新为 $\boldsymbol{p}'$;
until 满足停止条件

- 学得原型向量后，对任意样本，它被划入与其距离最近的原型向量所代表的簇中。形成了对样本空间的 Voronoi 划分。

9.4.3 高斯混合聚类

- 高斯分布、EM 算法、极大似然法和聚类的结合
- 服从高斯分布的概率密度函数

$$p(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$$

- 高斯混合分布

$$p_{\mathcal{M}}(\boldsymbol{x}) = \sum_{i=1}^{k} \alpha_i \cdot p(\boldsymbol{x} \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$$

- 生成样本 $\boldsymbol{x}_j$ 的每一个高斯混合成分的后验分布（记为 γ_{ji}）

$$p_{\mathcal{M}}(z_j = i \mid \boldsymbol{x}_j) = \frac{P(z_j = i) \cdot p_{\mathcal{M}}(\boldsymbol{x}_j \mid z_j = i)}{p_{\mathcal{M}}(\boldsymbol{x}_j)} = \frac{\alpha_i \cdot p(\boldsymbol{x}_j \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{l=1}^{k} \alpha_l \cdot p(\boldsymbol{x}_j \mid \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}$$

- 算法描述

Algorithm 9 高斯混合聚类算法

Input: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$、高斯混合成分个数 k
Output: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

初始化高斯混合分布的模型参数 $\{(\alpha_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \mid 1 \le i \le k\}$;
repeat
  for $j = 1, 2, \ldots, m$ **do**
    计算 $\boldsymbol{x}_j$ 由各混合成分生成的后验概率，即 $\gamma_{ji} = p_{\mathcal{M}}(z_j = i \mid \boldsymbol{x}_j)(1 \le i \le k)$;
  end for
  for $i = 1, 2, \ldots, k$ **do**
    计算新均值向量：$\boldsymbol{\mu}_i' = \frac{\sum_{j=1}^{m} \gamma_{ji} \boldsymbol{x}_j}{\sum_{j=1}^{m} \gamma_{ji}}$;
    计算新协方差矩阵：$\boldsymbol{\Sigma}_i' = \frac{\sum_{j=1}^{m} \gamma_{ji} (\boldsymbol{x}_j - \boldsymbol{\mu}_i')(\boldsymbol{x}_j - \boldsymbol{\mu}_i')^T}{\sum_{j=1}^{m} \gamma_{ji}}$;
    计算新混合系数：$\alpha_i' = \frac{\sum_{j=1}^{m} \gamma_{ji}}{m}$;
  end for
  将模型参数 $\{(\alpha_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \mid 1 \le i \le k\}$ 更新为 $\{(\alpha_i', \boldsymbol{\mu}_i', \boldsymbol{\Sigma}_i') \mid 1 \le i \le k\}$;
until 满足停止条件
$C_i = \emptyset (1 \le i \le k)$;
for $j = 1, 2, \ldots, m$ **do**
  根据 $\lambda_j = \arg\max_{i \in \{1,2,\ldots,k\}} \gamma_{ji}$ 确定 $\boldsymbol{x}_j$ 的簇标记 λ_j;
  将 $\boldsymbol{x}_j$ 划入相应的簇：$C_{\lambda_j} = C_{\lambda_j} \cup \{\boldsymbol{x}_j\}$;
end for

9.5 密度聚类

- 密度聚类算法从样本密度的角度考察样本之间的可连接性，基于可连接样本不断扩展聚类簇以获得最终的聚类结果。
- DBSCAN 算法基于一组邻域参数 $(\epsilon, MinPts)$ 来刻画样本分布的紧密程度
  - ϵ– 邻域：$N_\epsilon(\boldsymbol{x}_j) = \{\boldsymbol{x}_i \in D \mid \text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) \le \epsilon\}$

- 核心对象：$|N_\epsilon(\boldsymbol{x}_j)| \geq MinPts \rightarrow \boldsymbol{x}_j$ 是一个核心对象
- $\boldsymbol{x}_j$ 位于 $\boldsymbol{x}_i$ 的 $\epsilon-$ 邻域中 $\cap\, \boldsymbol{x}_i$ 是核心对象 $\rightarrow \boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度直达
- 对 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 若存在样本序列 $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n$，其中 $\boldsymbol{p}_1 = \boldsymbol{x}_i, \boldsymbol{p}_n = \boldsymbol{x}_j$，且 $\boldsymbol{p}_{i+1}$ 由 $\boldsymbol{p}_i$ 密度直达 $\rightarrow \boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度可达
- 对 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$，若存在 $\boldsymbol{x}_k$ 使得 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 均由 $\boldsymbol{x}_k$ 密度可达 $\rightarrow \boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 密度相连
- 簇：由密度可达关系导出的最大的密度相连样本集合，满足“连接性”（簇中任意两个样本密度相连）和“最大性”（簇中任意一个样本密度可达的样本都在该簇中）

- 算法描述

Algorithm 10 DBSCAN 算法

Input: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$、邻域参数 $(\epsilon, MinPts)$
Output: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

初始化核心对象集合：$\Omega = \emptyset$;
for $j = 1, 2, \ldots, m$ **do**
  确定样本 $\boldsymbol{x}_j$ 的 ϵ-邻域 $N_\epsilon(\boldsymbol{x}_j)$;
  if $|N_\epsilon(\boldsymbol{x}_j)| \geq MinPts$ **then**
    将样本 $\boldsymbol{x}_j$ 加入核心对象集合：$\Omega = \Omega \cup \{\boldsymbol{x}_j\}$;
  end if
end for
初始化聚类簇数和未访问样本集合：$k = 0, \quad \Gamma = D$;
while $\Omega \neq \emptyset$ **do**
  记录当前未访问样本集合：$\Gamma_{\text{old}} = \Gamma$;
  随机选取一个核心对象 $\boldsymbol{o} \in \Omega$，初始化队列 $Q = \langle \boldsymbol{o} \rangle$;
  $\Gamma = \Gamma \backslash \{\boldsymbol{o}\}$;
  while $Q \neq \emptyset$ **do**
    取出队列 Q 中的某个样本 $\boldsymbol{q}$;
    if $|N_\epsilon(\boldsymbol{q})| \geq MinPts$ **then**
      令 $\Delta = N_\epsilon(\boldsymbol{q}) \cap \Gamma$;
      将 Δ 中的样本加入队列 Q;
      $\Gamma = \Gamma \backslash \Delta$;
    end if
  end while
  $k = k + 1$，生成聚类簇 $C_k = \Gamma_{\text{old}} \backslash \Gamma$;
  $\Omega = \Omega \backslash C_k$;
end while

9.6 层次聚类

- AGNES 算法：自底向上层次聚类。先将每个样本视作一个初始聚类簇，再不断合并距离最近的两个聚类簇，达到预设的聚类簇个数。
- 聚类簇之间的距离计算

$$
\begin{aligned}
&\text{最小距离：}\ d_{\min}(C_i, C_j) = \min_{\boldsymbol{x}\in C_i, \boldsymbol{z}\in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z}) \\
&\text{最大距离：}\ d_{\max}(C_i, C_j) = \max_{\boldsymbol{x}\in C_i, \boldsymbol{z}\in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z}) \\
&\text{平均距离：}\ d_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{\boldsymbol{x}\in C_i} \sum_{\boldsymbol{z}\in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z})
\end{aligned}
$$

- 算法描述

Algorithm 11 AGNES 算法

Input: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$、簇距离度量函数 d、聚类簇数 k.
Output: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

for $j = 1, 2, \ldots, m$ **do**
  $C_j = \{\boldsymbol{x}_j\}$;
end for
for $i = 1, 2, \ldots, m$ **do**
  for $j = i + 1, \ldots, m$ **do**
    $M(j, i) = M(i, j) = d(C_i, C_j)$;
  end for
end for
设置当前聚类簇个数：$q = m$;
while $q > k$ **do**
  找出距离最近的两个聚类簇 C_{i^*} 和 C_{j^*};
  合并 C_{i^*} 和 C_{j^*}：$C_{i^*} = C_{i^*} \cup C_{j^*}$;
  for $j = j^* + 1, j^* + 2, \ldots, q$ **do**
    将聚类簇 C_j 重编号为 C_{j-1};
  end for
  删除距离矩阵 M 的第 j^* 行与第 j^* 列;
  for $j = 1, 2, \ldots, q - 1$ **do**
    $M(j, i^*) = M(i^*, j) = d(C_{i^*}, C_j)$;
  end for
  $q = q - 1$;
end while

10 降维与度量学习

10.1 k 近邻学习

- 机制：给定测试样本，基于某种距离度量找出训练集中与其最靠近的 k 个训练样本，并基于它们采用投票法或平均法进行预测。
- 是一种“懒惰学习”（训练时间开销为零，收到测试样本后才进行处理，相对于“急切学习”）。
- 最近邻分类器的泛化错误率不超过贝叶斯最优分类器的两倍。

10.2 低维嵌入

- 密采样：任意测试样本 $\boldsymbol{x}$ 附近任意小的 δ 距离范围内总能找到一个训练样本。若属性维数过多，满足密采样条件所需的样本过于巨大，导致数据样本稀疏、距离计算困难，引起“维数灾难”。
- 降维：通过某种数学变换将原始高维属性空间转变为一个低维子空间。
- 多维缩放（MDS）：设 m 个样本在原始空间的距离矩阵为 $\mathbf{D} \in \mathbb{R}^{m\times m}$，其中第 i 行第 i 列为样本 $\boldsymbol{x}_i$ 到 $\boldsymbol{x}_j$ 的距离。目标是获得样本在 d' 维空间的表示 $\mathbf{Z} \in \mathbb{R}^{d'\times m}, d' \leq d$，且任意两个样本在 d' 维空间中的欧氏距离等于原始空间中的距离，即 $||\boldsymbol{z}_i - \boldsymbol{z}_j|| = dist_{ij}$。

Algorithm 12 MDS 算法

Input: 距离矩阵 $\mathbf{D} \in \mathbb{R}^{m\times m}$，其元素 $dist_{ij}$ 为样本 $\boldsymbol{x}_i$ 到 $\boldsymbol{x}_j$ 的距离、低维空间维数 d'

Output: 矩阵 $\tilde{\mathbf{V}}\tilde{\mathbf{\Lambda}}^{1/2} \in \mathbb{R}^{m\times d'}$，每行是一个样本的低维坐标

1: 计算 $dist_{i\cdot}^2 = \frac{1}{m}\sum_{j=1}^{m} dist_{ij}^2$ 及 $dist_{\cdot j}^2, dist_{\cdot\cdot}^2$;
2: 根据 $b_{ij} = -\frac{1}{2}(dist_{ij}^2 - dist_{i\cdot}^2 - dist_{\cdot j}^2 + dist(\cdot\cdot)^2)$ 计算矩阵 $\mathbf{B}$;
3: 对矩阵 $\mathbf{B}$ 做特征值分解;
4: 取 $\tilde{\mathbf{\Lambda}}$ 为 d' 个最大特征值所构成的对角矩阵，$\tilde{\mathbf{V}}$ 为相应的特征向量矩阵;

- 线性降维方法：给定 d 维空间的样本 $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m) \in \mathbb{R}^{d\times m}$，变换之后得到 $d' \leq d$ 维空间中的样本

$$\mathbf{Z} = \mathbf{W}^T\mathbf{X}$$

其中 $\mathbf{W} \in \mathbb{R}^{d \times d'}$ 是变换矩阵，$\mathbf{Z} \in \mathbb{R}^{d' \times m}$ 是样本在新空间中的表达。

10.3 主成分分析

- 对于正交属性空间中的样本点，若存在一个超平面，满足“最近重构性”（样本点到这个超平面的距离都足够近）、“最大可分性”（样本点在这个超平面上的投影能尽可能分开），可分别从这两点推导出主成分分析。
- 设投影变换后得到的新坐标系为 $W = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'})$，则 PCA 的优化目标为

$$\begin{aligned} \max_{\mathbf{W}} \quad & \operatorname{tr}(\mathbf{W}^T \mathbf{X}\mathbf{X}^T \mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^T \mathbf{W} = \mathbf{W} \end{aligned}$$

- 算法描述

Algorithm 13 PCA 算法

Input: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$、低维空间维数 d'
Output: 投影矩阵 $\mathbf{W}^* = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'})$

1: 对所有样本进行中心化：$\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i - \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{x}_i$;
2: 计算样本的协方差矩阵：$\mathbf{X}\mathbf{X}^T$;
3: 对协方差矩阵 $\mathbf{X}\mathbf{X}^T$ 做特征值分解;
4: 取最大的 d' 个特征值所对应的特征向量 $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'}$;

- d' 值可由用户指定，也可以通过 kNN+ 交叉验证自动取值，也可取使得 $\sum_{i=1}^{d'} \lambda_i / \sum_{i=1}^{d} \lambda_i \geq t$ 的最小 d' 值。
- 降维（丢弃 $d - d'$ 个特征值的特征向量）的意义：增大采样密度、减小噪声影响。