《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

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

- 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：

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

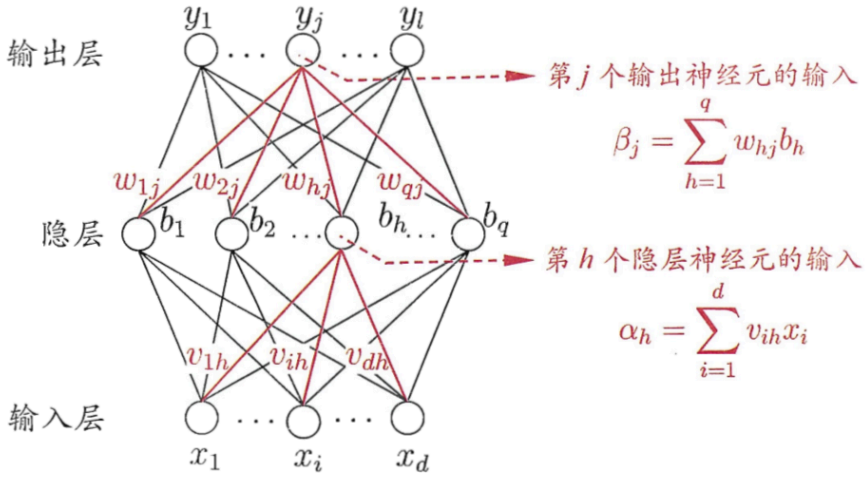
其中 η 为 $(0,1)$ 内的小正数，称为学习率。

- 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：

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

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 $\| \boldsymbol{w} \|^2$

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