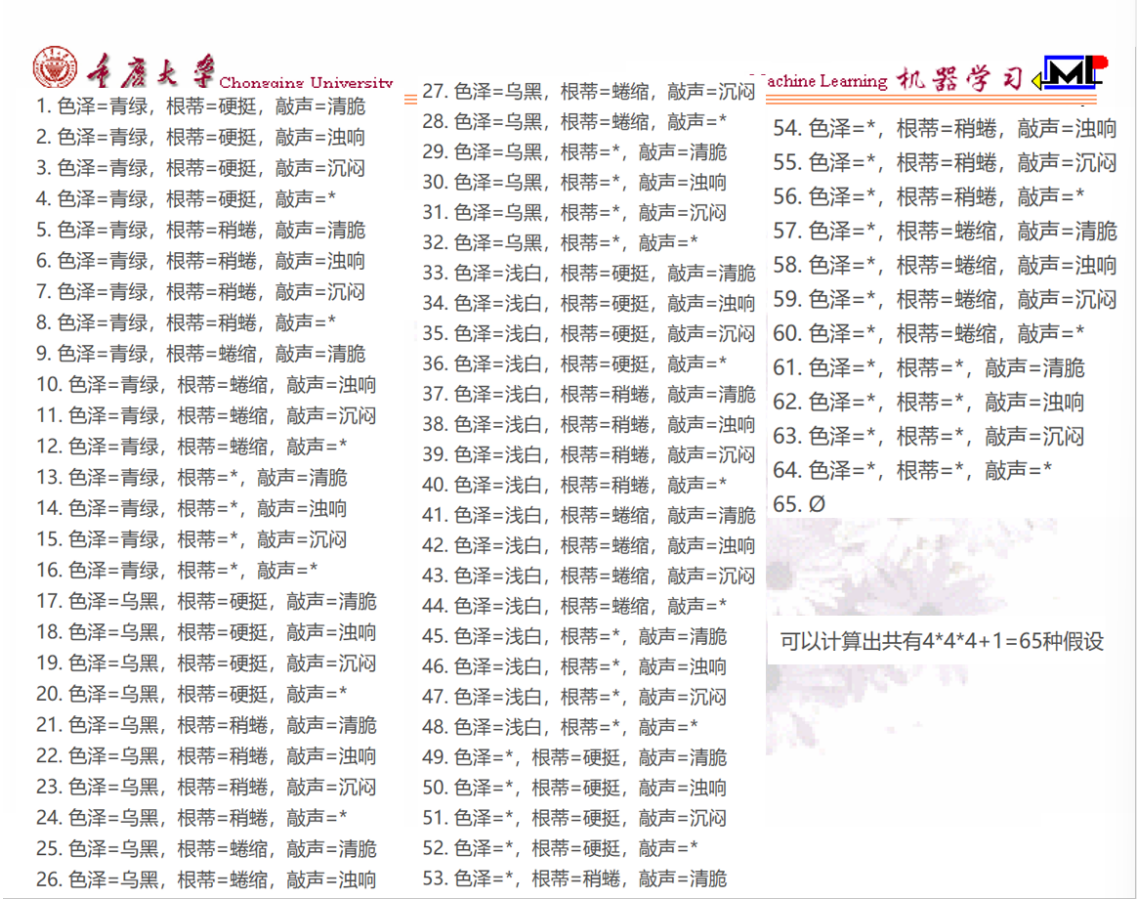
机器学习

第一章 绪论

1. 什么是机器学习：利用经验改善系统自身的性能计算机程序（算法）如何随着经验积累自动提高性能、系统自我改进的过程。
2. 三个特征：

   某类任务T：Task
   性能P：Performance
   经验E：Experience or Examples
3. 预测目标：
   1. 分类：离散值
   2. 回归：连续值
   3. 聚类：无标记信息
4. 有无标记信息：
   1. 监督学习：分类、回归
   2. 无监督学习：聚类
   3. 半监督学习或主动学习：两者结合
5. 泛化能力：模型适用于新样本的能力
6. 假设空间：把学习过程看成在所有假设构成的空间中进行搜索的过程，搜索目标是找到所有与训练集fit的假设,即能够将训练集中瓜正确判断的假设



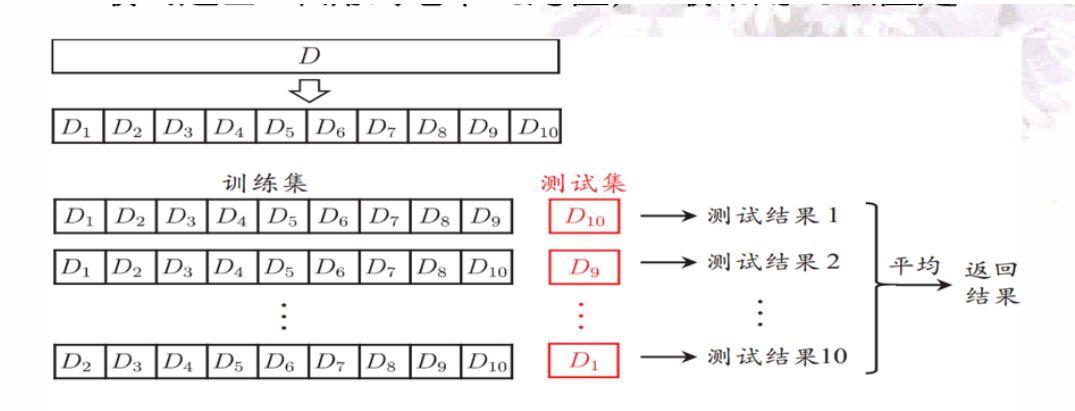
7. 版本空间：与训练数据相一致的假设集合

8. 归纳偏好：学习过程中对某种类型假设的偏好称作归纳偏好
9. “奥卡姆剃刀”：若有多个假设与观察一致，选最简单的那个
10. 没有免费的午餐定理（NoFreeLunch）：一个算法 a 如果在某些问题上比另一个算法b 好，必然存在另一些问题， b 比a 好
11. 发展历程：推理期（发现逻辑推理无法实现人工智能） -> 知识期（希望机器能够自己学习知识） -> 学习期（决策树、基于逻辑的学习、连接主义学习、统计学习）
12. 一般过程：
    1. 数据获取
    2. 特征工程
    3. 模型选择
    4. 模型训练
    5. 模型评估
    6. 超参数条件
    7. 预测

第二章 模型评估与选择

1. 错误率: 错分样本的占比
2. 误差：样本真实输出与预测输出之间的差异
    1. 训练(经验)误差：训练集上
    2. 测试误差：测试集
    3. 泛化误差：除训练集外所有样本
3. 过拟合：学习器把训练样本学习的“太好”，将训练样本本身的特点当做所有样本的一般性质，导致泛化性能下降

4. 欠拟合：对训练样本的一般性质尚未学好
5. 评估方法：
    1. 留出法：直接将数据集划分为两个互斥集合，训练/测试样本比例：2:1~4:1
    2. 交叉验证法：将数据集分层采样划分为k个大小相似的互斥子集，每次用k-1个子集的并集作为训练集，余下的子集作为测试集，最终返回k个测试结果的均值，k最常用的取值是10



    3. k折交叉验证通常随机使用不同的划分重复p次，最终的评估结果是这p次k折交叉验证结果的均值，例如常见的“10次10折交叉验证“
    4. 自助法：m次有放回重复抽样
6. 性能度量：
    1. 回归任务最常用的性能度量是“均方误差”

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

    2. 对于分类任务,**错误率**和**精度**是最常用的两种性能度量

        错误率：分错样本占样本总数的比例
        精度：分对样本占样本总数的比率

    3. 查准率和查全率（统计真实标记和预测结果的组合可以得到“混淆矩阵”）

        查准率：所有预测为好瓜的瓜中确实为好瓜的比例

        查全率（召回率）：所有好瓜中被预测正确的比例

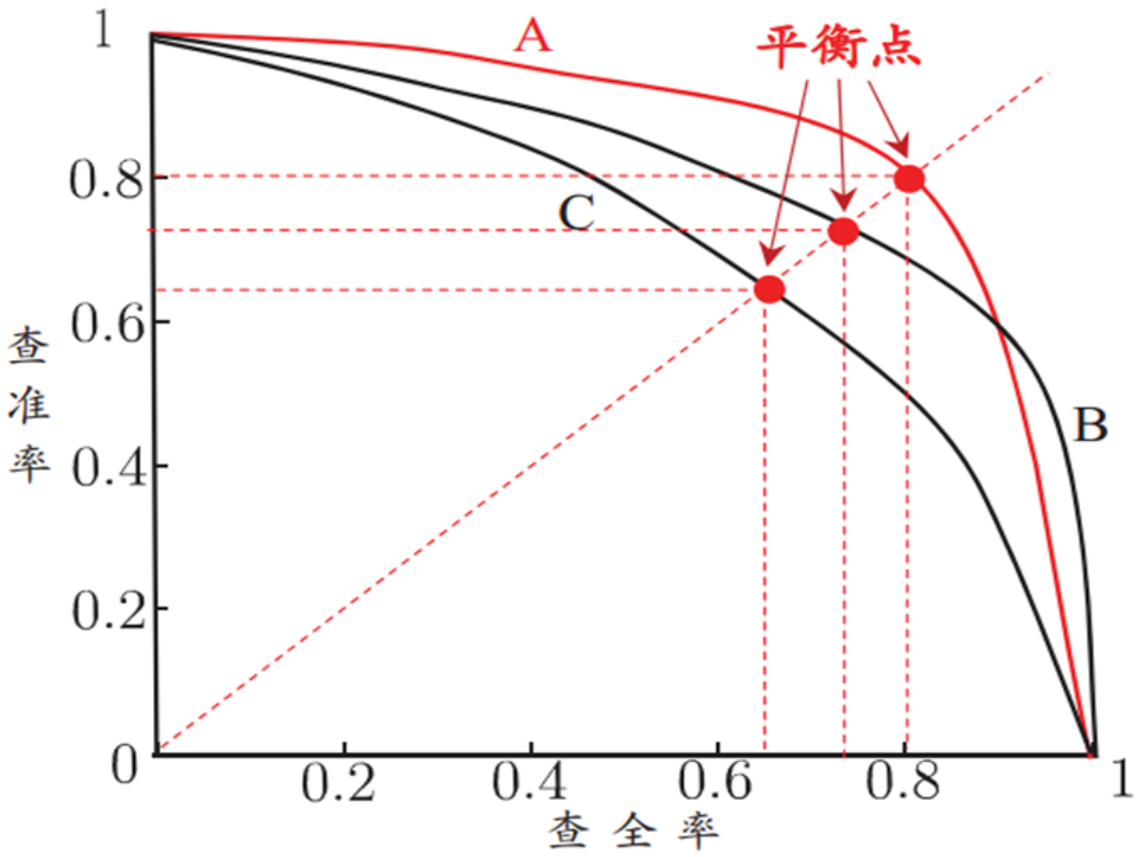
分类结果混淆矩阵

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

查全率 $R=\frac{TP}{TP+FN}$

7. PR曲线（Precision-Recall曲线）：遍历从0到1的不同阈值。对于每个阈值，将模型预测的概率大于该阈值的样本预测为正类，小于等于该阈值的样本预测为负类。计算在每个阈值下的精度（Precision）和召回率（Recall）



P-R曲线与平衡点示意图

平衡点是曲线上"查准率=查全率"时的取值，可用来用于度量P-R曲线有交叉的分类器性能高低

8. F1度量：

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

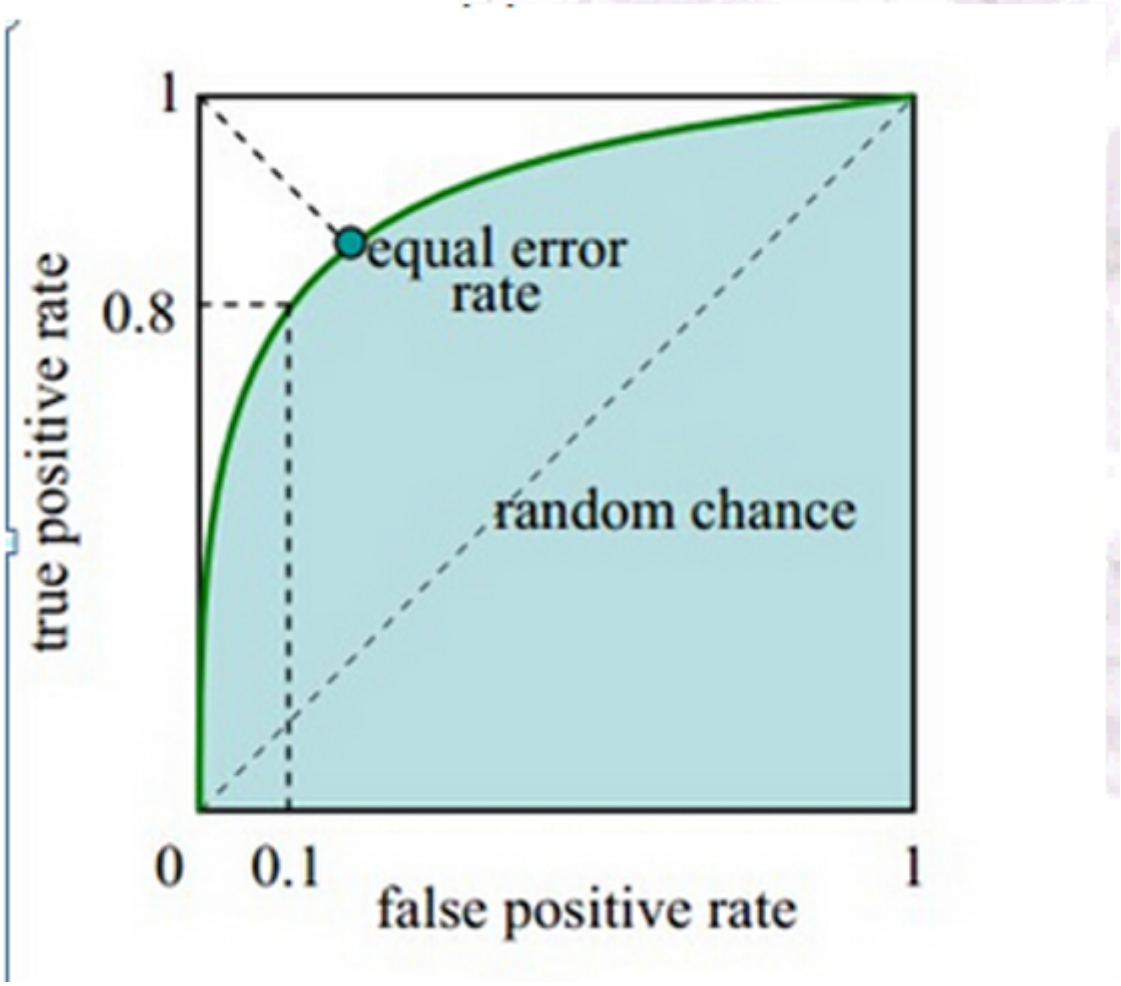
$$F_\beta = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R}$$

$\beta = 1$：标准F1

$\beta > 1$：偏重查全率(逃犯信息检索)

$\beta < 1$：偏重查准率(商品推荐系统)

9. ROC曲线：



理想目标：TPR=1，FPR=0，即图中(0,1)点，故ROC曲线越靠拢(0,1)点，越偏离45度对角线越好。

横轴FPR（真正例率）：FPR越大，预测正类中实际负类越多

纵轴TPR（假正例率）：TPR越大，预测正类中实际正类越多。

10. AUC：ROC曲线下的面积，AUC的取值范围在0.5和1之间，AUC作为数值可以直观的评价分类器的好坏，值越大越好

第三章 线性模型

1. 线性模型的一般形式：

$$f\left(\boldsymbol{x}\right)=w_1x_1+w_2x_2+\ldots+w_dx_d+b$$

2. 向量形式：

$$f\left(\boldsymbol{x}\right)=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$$

3. 线性回归的目的：学得一个线性模型以尽可能准确地预测实值输出标记
4. 最小二乘法：
    1. 在线性回归中，最小二乘法就是试图找到一条直线，使得所有样本到直线上的欧式距离之和最小

2.

$$(w^*, b^*) = \arg\min_{(w,b)} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$

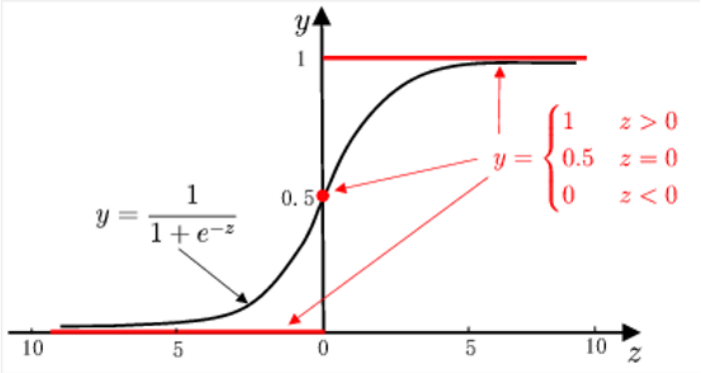
$$= \arg\min_{(w,b)} \sum_{i=1}^{m} (y_i - wx_i - b)^2$$

5. 对数几率回归（逻辑回归）

- **单调可微、任意阶可导**

$$y = \frac{1}{1 + e^{-z}}$$

单位阶跃函数与对数几率函数的比较



6. 极大似然法：最大化样本属于其真实标记的概率

7. 线性判别分析（LDA）：

   1. LDA思想：

      欲使同类样例的投影点尽可能接近，可以让同类样例投影点的协方差尽可能小

      欲使异类样例的投影点尽可能远离，可以让类中心之间的距离尽可能大

   2. **类内散度矩阵、类间散度矩阵**

   3. 最大化目标：

最大化目标

$$J = \frac{\left\| \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\mu}_0 - \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\mu}_1 \right\|_2^2}{\boldsymbol{w}^{\mathrm{T}} \boldsymbol{\Sigma}_0 \boldsymbol{w} + \boldsymbol{w}^{\mathrm{T}} \boldsymbol{\Sigma}_1 \boldsymbol{w}}$$

分子：类中心之间的距离

分母：同类样例投影点的协方差

$$= \frac{\boldsymbol{w}^{\mathrm{T}} (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1) (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^{\mathrm{T}} \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}} (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1) \boldsymbol{w}}$$

   4. 最大化目标重写，得到**广义瑞利商**

$$J = \frac{\boldsymbol{w}^{\mathrm{T}} \mathbf{S}_b \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}} \mathbf{S}_w \boldsymbol{w}}$$

分子：类间散度

分母：类内散度

   5. 用法：

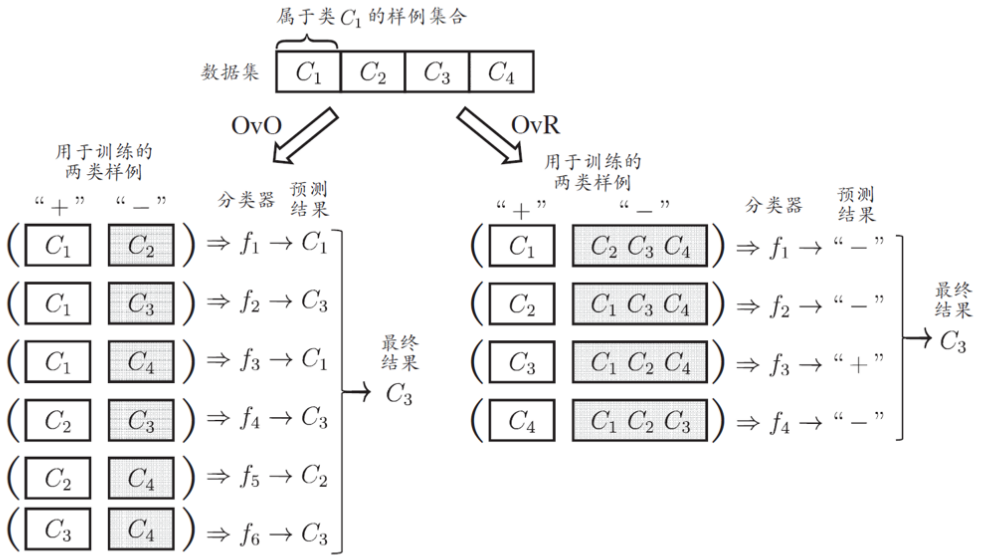
      1. 实际中LDA主要用于降维

      2. 一般来说，如果数据是有类别标签的，那么优先选择LDA去尝试降维；

3. 当然也可以使用PCA做很小幅度的降维去消去噪声，然后再使用LDA降维。

4. 如果没有类别标签，那么肯定PCA是最先考虑的一个选择。

8. 多分类学习：一对一（OvO）、一对其余（OvR）、多对多（MvM）



第四章 决策树

1. 决策树基于树结构来预测
2. 主要决策树的算法：ID3 + C4.5 + CART
    1. ID3：最早提出，使用信息增益，只能处理离散型属性
    2. C4.5：使用信息增益率，可以处理连续型属性
    3. CART：Gini指数，构造的是二叉树，可以用于回归
3. 基本流程：
    1. **特征选择**：特征选择是指从训练数据中众多的特征中选择一个特征作为当前节点的分裂标准，如何选择特征有着很多不同量化评估标准，从而衍生出不同的决策树算法。
    2. **决策树生成**：根据选择的特征评估标准，从上至下递归地生成子节点，直到数据集不可分则停止决策树停止生长。 树结构来说，递归结构是最容易理解的方式
    3. **剪枝**：决策树容易过拟合，一般来需要剪枝，缩小树结构规模、缓解过拟合。剪枝技术有预剪枝和后剪枝两种

Algorithm 1 决策树学习基本算法

输入:

- 训练集 $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$;
- 属性集 $A = \{a_1, \ldots, a_d\}$.

过程: 函数 TreeGenerate(D, A)

```
1: 生成结点 node;
2: if D 中样本全属于同一类别 C then
3:     将 node 标记为 C 类叶结点; return
4: end if
5: if A = ∅ OR D 中样本在 A 上取值相同 then
6:     将 node 标记叶结点，其类别标记为 D 中样本数最多的类; return
7: end if
8: 从 A 中选择最优划分属性 a_*;
9: for a_* 的每一个值 a_*^v do
10:    为 node 生成每一个分枝; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;
11:    if D_v 为空 then
12:        将分枝结点标记为叶结点, 其类别标记为 D 中样本最多的类; return
13:    else
14:        以 TreeGenerate(D_v, A − {a_*}) 为分枝结点
15:    end if
16: end for
```

输出: 以 node 为根结点的一棵决策树

4. 经典的属性划分方法：**信息增益、信息增益率、基尼指数**
5. **熵**：表示随机变量不确定性的度量

$$l(x_i) = -log_2 p(x_i)$$

- **其中p(xi)是选择该分类的概率。**

6. 存在问题：信息增益对可取值数目较多的属性有所偏好
7. 信息增益率：信息增益/固有值，会导致属性的重要性随着固有值的增大而减小
8. 存在问题：信息增益率对可取值数目较少的属性有所偏好
9. **基尼指数**：反映了从 中随机抽取两个样本，其类别标记不一致的概率
10. 基尼指数越小，数据集的纯度越高，选择那个使划分后基尼指数最小的属性作为最优划分属性
11. 剪枝：对付“过拟合”
12. 预剪枝 + 后剪枝
13. 判断决策树泛化性能是否提升的方法：**留出法**预留一部分数据用作“验证集”以进行性能评估
14. 预剪枝：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记为叶结点，其类别标记为训练样例数最多的类别，**缺点**：欠拟合风险
15. 后剪枝：后自底向上地对非叶结点进行考察，若将该结点对应的子树替换为叶结点能带来决策树泛化性能提升，则将该子树替换为叶结点，缺点：训练时间开销大

16. 连续值处理：**连续属性离散化**（二分法）：将连续属性的取值从小到大排序，基于划分点t分为两个子集，采用离散属性值方法，考察这些划分点，选取最优的划分点。
17. 缺失值处理：若样本 在划分属性上的取值未知，则将x**同时划入所有子结点**，且样本权值在与属性值 对应的子结点中调整为 (直观来看，相当于让同一个样本以不同概率划入不同的子结点中去)
18. 多变量决策树：非叶节点不再是仅对某个属性,而是对属性的线性组合

第五章 神经网络

1. 对于非线性的分类问题，在特征量较少的情况，可以用多项式类型的Logistic回归或其它算法来处理
2. 关键不是神经元，而是连接线，每个连接线对应一个不同的权重，需要训练得到。
3. 一个神经网络的训练算法，就是让权重的值调整到最佳
4. **感知机**（单层神经网络）：
    1. 两个层次：输入层和输出层，输入层只做输入不做计算，输出层对前一层的输入进行计算并输出
    2. 输出层是M-P神经元
    3. 学习

    感知机学习规则

    $$\min_{\boldsymbol{w}} L(\boldsymbol{w}) = \min_{\boldsymbol{w}} \sum_{\boldsymbol{x}_i \in M} (y_i - y_i)\boldsymbol{w}\ \boldsymbol{x}_i$$

    对训练样例 $(\boldsymbol{x}, y)$，当前感知机的输出为 $\hat{y}$，则感知机权重调整规则为：

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

    其中 $\eta \in (0,1)$ 称为学习率

    若感知机对训练样例$(\boldsymbol{x}, y)$预测正确，则感知机不发生变化；否则根据错误程度进行权重的调整.

5. 激活函数：sigmoid
6. 多层神经网络：虽然层数保持不变，但是第二个神经网络的参数数量却是第一个神经网络的接近两倍之多，从而带来了更好的**表示能力**。表示能力是多层神经网络的一个重要性质。
7. M-P神经元模型：
    1. 输入：来自其他n个神经元传递过来的输入信号
    2. 处理：输入信号通过带权重的连接进行传递, 神经元接受到总输入值将与神经元的阈值进行比较
    3. 输出：通过激活函数的处理以得到输出
8. 理想的激活函数是**阶跃函数**，但是具有不连续、不光滑的性质
9. 若两类模式线性可分, 则感知机的学习过程一定会收敛，非线性可分问题：**多层感知机**
10. 误差逆传播算法（BP算法）：是最成功的训练多层前馈神经网络的学习算法。
    1. 向前计算
    2. 梯度下降：调整参数θ使得代价函数J(θ)取得最小值的最基本方法之一
    3. BP算法基于梯度下降策略, 以目标的负梯度方向对参数进行调整
    4. 学习率控制着算法每一轮迭代中的更新步长, 若太长则让容易震荡, 太小则收敛速度又会过慢

BP 学习算法

输入： 训练集 $D=\{(\boldsymbol{x}_k,\boldsymbol{y}_k)\}_{k=1}^{m}$;
学习率 η.

过程：

```
在(0,1)范围内随机初始化网络中所有连接权和阈值
repeat
    for all (x_k, y_k) ∈ D do
        根据当前参数和式(5.3) 计算当前样本的输出 ŷ_k;
        根据式(5.10) 计算输出层神经元的梯度项 g_j;
        根据式(5.15) 计算隐层神经元的梯度项 e_h;
        根据式(5.11)-(5.14) 更新连接权 w_hj, v_ih 与阈值 θ_j, γ_h
    end for
until 达到停止条件
```

输出： 连接权与阈值确定的多层前馈神经网络

5. 标准BP算法：每次针对单个训练样例更新权值与阈值，参数更新频繁, 不同样例可能抵消, 需要多次迭代.
6. 累计BP算法：其优化的目标是最小化整个训练集上的累计误差，读取整个训练集一遍才对参数进行更新, 参数更新频率较低.

11. 多层前馈网络表示能力：只需要一个包含足够多神经元的隐层, 多层前馈神经网络就能以任意精度逼近任意复杂度的连续函数
12. 局限：神经网络由于强大的表示能力, 经常遭遇**过拟合**
13. 如何设置隐层神经元的个数仍然是个未决问题. 实际应用中通常使用"试错法"调整
14. **缓解过拟合的策略**：早停、正则化
15. **全局最小**和**局部极小**：参数空间梯度为零的点, 只要误差函数值小于邻点的误差函数值, 就是局部极小点
16. 跳出局部最小的策略：**模拟退火技术、随机梯度下降、遗传算法**
17. 其他常见的神经网络：RBF、ART、SOM
18. 深度学习：增加隐层神经元的数目（**模型宽度**），增加隐层数目（**模型深度**） 预训练 +微调
19. 卷积神经网络（CNN）：卷积层 + 采样层 + 连接层
20. 在CNN中通常将 sigmoid 激活函数替换为修正的线性函数Relu
21. CNN 可以用BP进行训练, 但在训练中, 无论是卷积层还是采样层, 每一组神经元都是用相同的连接权, 从而大幅减少了需要训练的参数数目

第六章 支持向量机

1. SVM属于有监督学习算法
2. 既可以分类也可以回归
3. **支持向量**：位于最大间隔上的样本点
4. **最优分离超平面**：尽可能远离所有类别的数据点
5. 最大化超平面之间的距离：有约束的最小化优化问题 -> 拉格朗日乘子法
6. **拉格朗日对偶**：将有约束的原始目标函数转化为无约束新构造的拉格朗日目标函数
7. **SMO序列最小化算法**：SMO算法是将大优化问题分解为多个小优化问题来求解。这些小优化问题往往很容易求解，并且对它们进行顺序求解的结果与将它们作为整体来求解的结果完全一致的。在结果完全相同的同时，SMO算法的求解时间短很多。

8. SVM处理的数据可以分为三类：
    1. 线性可分，通过硬间隔最大化，学习线性分类器
    2. 近似线性可分，通过软间隔最大化，学习线性分类器
    3. 线性不可分，通过核函数以及软间隔最大化，学习非线性分类器
9. 硬间隔对应于线性可分数据集，可以将所有样本正确分类，也正因为如此，受噪声样本影响很大，不推荐。
10. 软间隔对应于通常情况下的数据集（近似线性可分或线性不可分），允许一些超平面附近的样本被错误分类，从而提升了泛化性能。**松弛变量**调节错分率。
11. **核函数**：将数据映射到高维空间
    1. 事实上大部分时候数据并不是线性可分的，这个时候满足这样条件的超平面就根本不存在。对于非线性的情况，SVM 的处理方法是选择一个核函数 κ(·,·)，通过将数据映射到高维空间，最终在高维特征空间中构造出最优分离超平面，从而把平面上本身不好分的非线性数据分开。来解决在原始空间中线性不可分的问题。
    2. 基本想法：不显式地设计核映射, 而是设计核函数
    3. 只要一个对称函数所对应的核矩阵半正定, 则它就能作为核函数来使用.
    4. 常用核函数：线性核、多项式核、高斯核、拉普阿斯核、sigmoid核
12. 正则化的回归分析可以避免过拟合
13. 支持向量回归： 允许模型输出和实际输出间存在 2*e 的偏差.落入中间2*e间隔带的样本不计算损失, 从而使得模型获得稀疏性
14. **核方法**：无论是支持向量机还是支持向量回归, 学得的模型总可以表示成核函数的线性组合
15. 核LDA：先将样本映射到高维特征空间, 然后在此特征空间中做线性判别分析

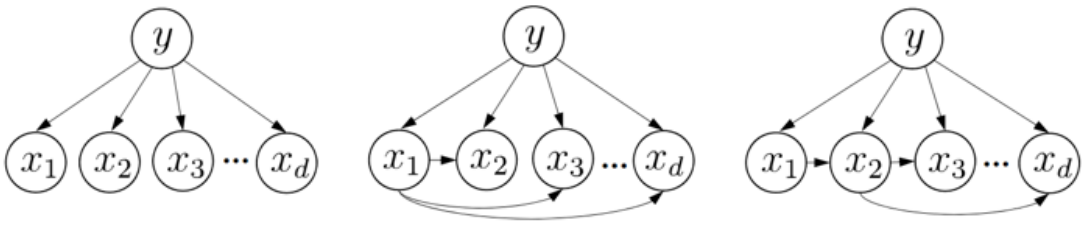
第七章 贝叶斯分类器

1. 条件概率：后验概率 = 先验概率 * 调整因子
2. **贝叶斯决策论**：贝叶斯决策考虑如何基于这些概率和误判损失来选择最优的类别标记
3. **条件风险**：基于后验概率可获得将样本分类为所产生的期望损失（expected loss），即在样本上的“条件风险”
4. **极大似然估计**：估计类条件概率的常用策略：先假定其具有某种确定的概率分布形式，再基于训练样本对概率分布参数估计
5. 主要困难：类条件概率是所有属性上的联合概率，难以从有限的训练样本估计获得
6. **朴素贝叶斯分类器**：采用了“属性条件独立性假设：每个属性独立地对分类结果发生影响。
7. **拉普拉斯修正**：为了避免其他属性携带的信息被训练集中未出现的属性值“抹去”

令 N 表示训练集 D 中可能的类别数，N_i 表示第 i 个属性可能的取值数则式 (7.16) 和 (7.17) 分别修正为

$$\hat{P}(c) = \frac{|D_c| + 1}{|D| + N} \quad (7.19) \qquad \hat{P}(x_i \mid c) = \frac{|D_{c,x_i}| + 1}{|Dc| + N_i} \quad (7.20)$$

8. **半朴素贝叶斯分类器**：对属性条件独立假设进行一定程度的放松
9. **独依赖估计**(ODE)：假设每个属性在类别之外最多仅依赖一个其他属性



(a) NB (b) SPODE (c) TAN

10. TAN：以属性为结点构建完全图，任意两个结点之间边的权重为I（），构建最大带权生成树，权重刻画了属性间的相关性，通过最大生成树算法，保留了强相关属性之间的依赖性
11. AODE：尝试将每个属性作为超父构建SPODE
12. 贝叶斯网：借助有向无环图刻画属性间的依赖关系

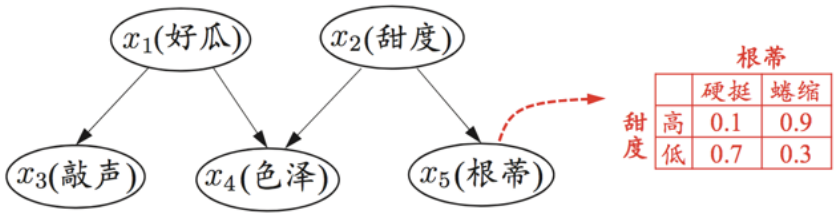
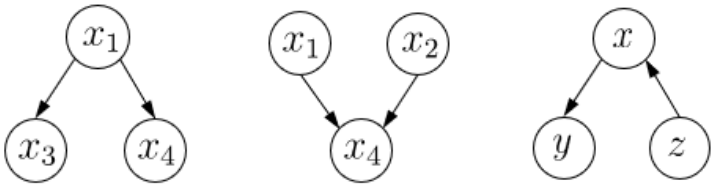


图7.2 西瓜问题的一种贝叶斯网结构以及属性“根蒂”的条件概率表

- 从网络图结构可以看出 -> “色泽”直接依赖于“好瓜”和“甜度”
- 从条件概率表可以得到 -> “根蒂”对“甜度”的量化依赖关系
  P(根蒂＝硬挺|甜度＝高)＝0.1

- 贝叶斯网中三个变量之间的典型依赖关系：



13. 道德图：V型结构父节点相连，有向边变成无向边，无法连接的变量间条件独立
14. EM算法：如果使用基于最大似然估计的模型，模型中存在隐变量，就要用EM算法做参数估计。
    1. E步：对当前参数seta推断隐变量分布，并计算对数似然关于Z的期望
    2. M步：寻找参数最大化期望似然
    3. EM算法两个步骤：第一步计算期望，利用当前估计的参数值计算对数似然的参数值；第二部最大化，寻找能使E步产生的似然期望最大化的参数值；直到收敛到全局最优解。
15. 高斯朴素贝叶斯、多项式贝叶斯、伯努利朴素贝叶斯

第八章 集成学习

1. 两个问题：如何得到若干个**个体学习器**，如何选择一种**结合策略**
2. 一般都是**同质个体学习器**：CART决策树、神经网络
3. 分类：
    1. **boosting**：个体学习器之间存在强依赖关系，需要串行生成
        1. 代表算法：Adaboosting、GBDT、XGBOOST

    2. Adaboosting：分类正确得票多，分类错误得票少，误差为权重之和，每个学习器根据上一个学习器的权重计算误差再分类、
2. **bagging**：不存在依赖关系，并行生成
    1. **随机森林**：选择一部分样本，再从这一部分样本中选一部分特征,使用了CART决策树作为弱学习器
    2. 自助采样法
3. **stacking**：使用两层学习模型，将训练好的所有基模型对训练集进行预测，第j个基模型对第i个训练样本的预测值将作为新的训练集中第i个样本的第j个特征值，最后基于新的训练集进行训练，**缺点**：耗时间

4. 结合策略：
    1. **平均法**：用于回归
    2. **投票法**：用于分类（硬：选多的，软：概率大的，准确率比硬高）
    3. **学习法**：Stacking
5. **误差-分歧分解**：个体学习器精确性越高、多样性越大，则集成效果越好

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

6. 多样性度量：不合度量、相关系数、K统计量、Q统计量

第九章 聚类

1. **聚类**：根据个体属性对一系列未分类的个体进行类别的识别，把一组个体按照相似性归成若干个类，属于无监督学习方法
2. 聚类既可以作为一个单独的过程，也可以作为分类等其他学习任务的前驱过程
3. 聚类性能度量，也被称为聚类"有效性指标"
4. 聚类结果：簇内相似度高，簇间相似度低
5. 外部指标：将聚类结果与某个"参考模型"(reference model)进行比较
6. 内部指标：直接考察聚类结果而不用任何参考模型
7. 原型聚类：此类算法假设**聚类结构能通过一组原型刻画**
8. 算法过程：先对原型进行初始化，再对原型进行迭代更新求解。
9. k-means算法

输入： 样本集$D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
聚类簇数k.

过程：

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$C_i=\emptyset\ (1\leq i\leq k)$
  for $j=1,\ldots,m$ **do**
    计算样本$\boldsymbol{x}_j$与各均值向量$\boldsymbol{\mu}_i\ (1\leq i\leq k)$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：$C_{\lambda_j}=C_{\lambda_j}\bigcup\{\boldsymbol{x}_j\}$;
  end for
  for $i=1,\ldots,k$ **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
return 簇划分结果

输出： 簇划分$\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

1. 主要优点：解决聚类问题的一种经典算法，简单、快速；当结果簇是密集的，而簇与簇之间区别明显时，它的效果较好；对处理大数据集，该算法是相对可伸缩和高效率的。
2. 主要缺点：必须事先确定k值，很多情况下并不清楚类别个数。但是聚类结果对初始值敏感，对于不同的初始值，可能导致不同结果；初始聚类中心的选择对聚类结果有较大的影响。初值选择不当，可能无法得到好的聚类结果。可以多设一些不同初值，对比最后的运算结果，一直到结果趋于稳定来解决这一问题，但比较耗时耗资源。在簇的平均值被定义的情况下才能使用，这对于处理符号属性的数据不适用；l对于噪声和孤立点数据是敏感的，少量的该类数据能够对平均值产生较大的影响

10. 学习向量量化（LVQ）：LVQ假设数据样本带有类别标记，学习过程中利用样本的这些监督信息来辅助聚类

输入： 样本集$D=\{(\boldsymbol{x}_1,y_1),\ldots,(\boldsymbol{x}_m,y_m)\}$;
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$\eta\in(0,1)$.

过程：

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; $i^*=\arg\min_{i\in\{1,2,\ldots,q\}}d_{ji}$;
  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
return 当前原型向量

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

11. 高斯混合聚类（GMM聚类）：多个高斯分布函数的线性组合
    1. 混合系数：代表了点属于某个分布的概率。
    2. 含有隐含变量：用EM算法来求解