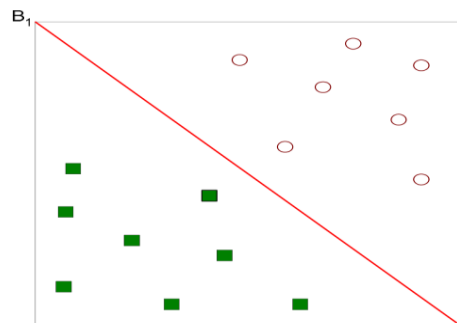


Web信息处理与应用

第十二节 分 类

徐童 2021.11.29

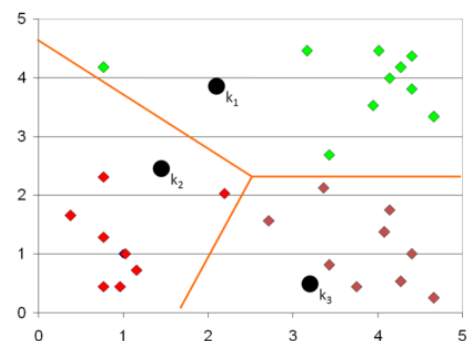
- 数据挖掘的基本方法



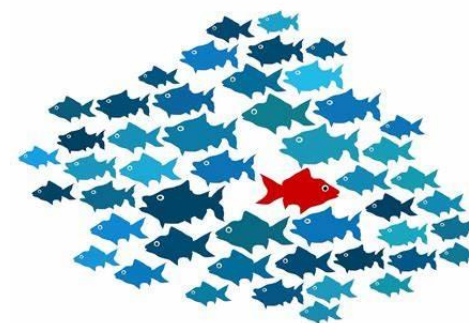
分类

| <i>TID</i> | <i>Items</i> |
|------------|---------------------------|
| 1 | Bread, Milk |
| 2 | Bread, Diaper, Beer, Eggs |
| 3 | Milk, Diaper, Beer, Coke |
| 4 | Bread, Milk, Diaper, Beer |
| 5 | Bread, Milk, Diaper, Coke |

关联规则



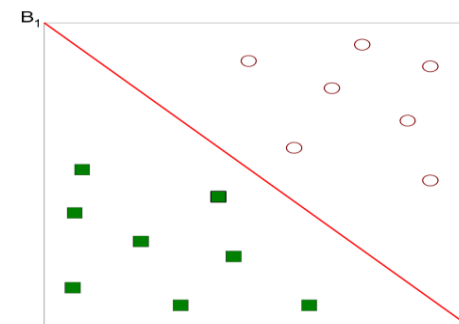
聚类



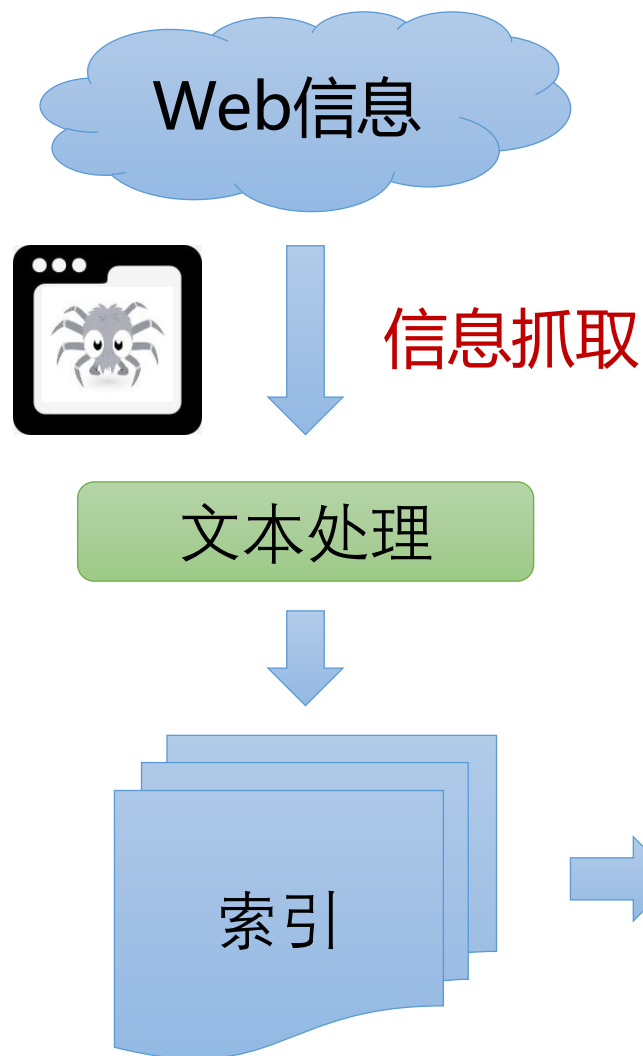
离群检测

- **基本方法 (1) 分类**

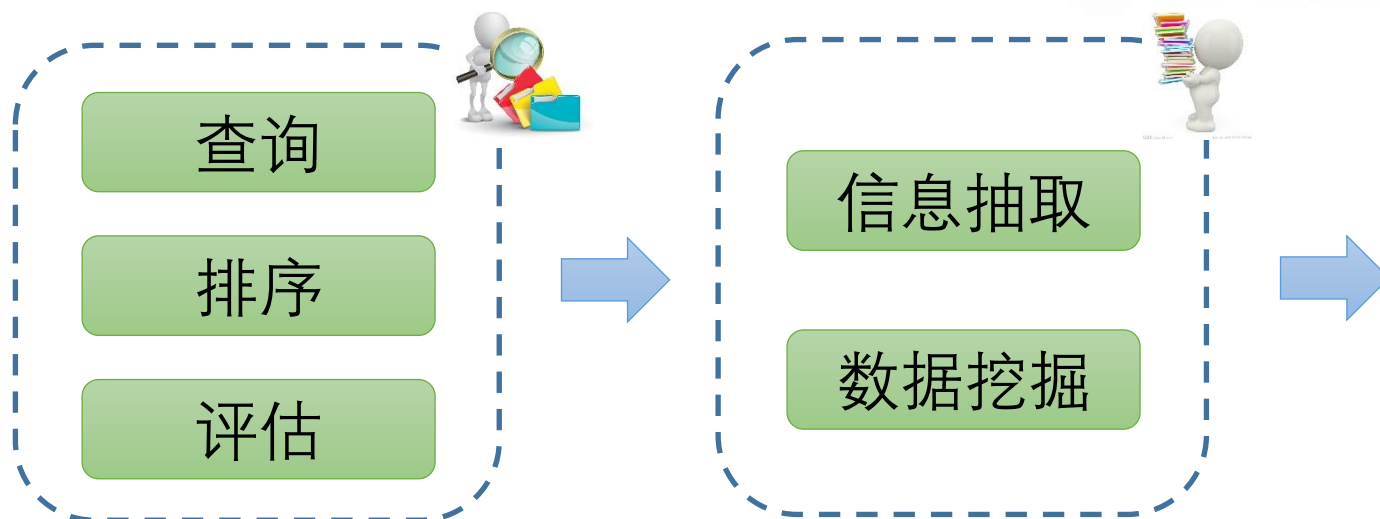
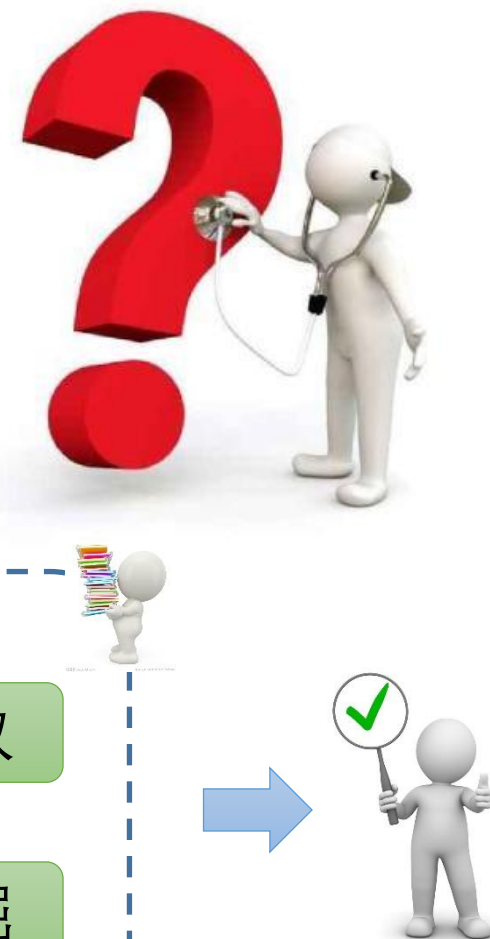
- 给定一组有标签记录（作为训练集），其目标在于训练一个合适的模型，使其能够有效地区分无标签的新数据，将其归为合适的类别
 - 有监督学习，面向预定义的类别
 - 分类问题是我们的老朋友了（传统艺能）
 - 在各种信息检索问题中广泛存在，因其丰富的解决方案，而成为诸多问题转化的对象







- 本课程所要解决的问题



第十一个问题：
常见的分类方法有哪些？



- 时常会面临的困扰
- 这封邮件是垃圾邮件吗？

Important Message     发起会议

发件人: Maggie M... <archive@tfi-urfo.ru>

时 间: 2019年11月22日 00:03:09 (星期五)

收件人: Recipients <archive@tfi-urfo.ru>

Hi,

I have a business proposal to share with you. Contact me back for more details.

Thanks.

Maggie M. Wang

- 时常会面临的困扰

- 如何判别垃圾邮件?

莫名其妙的收信人 →

Important Message     发起会议

发件人: Maggie M... <archive@tfi-urfo.ru> ← 罕见的邮箱后缀

时 间: 2019年11月22日 00:03:09 (星期五)

收件人: Recipients <archive@tfi-urfo.ru>

Hi,

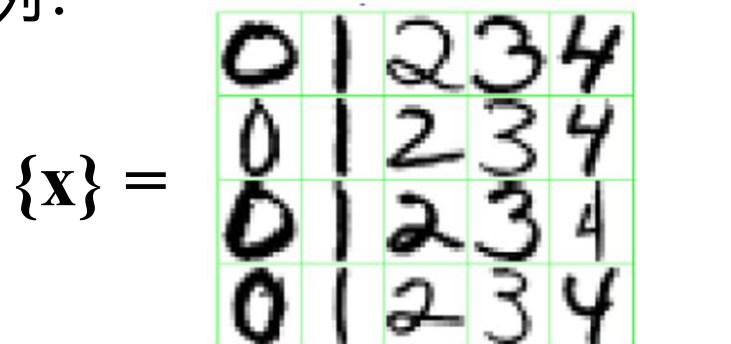
不合常理的
邮件内容 → I have a business proposal to share with you. Contact me back for more details.

Thanks.
Maggie M. Wang

基于一些特征与规则，我们可以将垃圾邮件的判别视作一个分类问题

• 什么是分类问题

- 给定一组样本 $\{(\mathbf{x}_i, y_i)\}_{i=1..N}$, 其中 \mathbf{x} 为样本的特征, y 为样本对应的标签
 - 分类问题的目标在于基于规则或特征训练一个有效的分类器 (Classifier)
 - 借助分类器, 当面对新样本 \mathbf{x} 时, 应将其准确映射到对应的标签 $y \in \mathcal{Y}$
 - 如何表征样本是一个问题, 尤其是面向文本、图像、视频等多模态信息
- 一个基础的实例:



digits recognition;
 $\mathcal{Y} = \{0, \dots, 9\}$

- **分类学习是一种有监督学习**
- 不同的分类方法，其学习算法各不相同，但基本流程是一致的



- **最基本的分类方法：人工分类**

- 最为基础和直接的方法，借助领域专家的支持可以保证分类的准确性

- 有多少人工，就有多少智能（手动🐱）

- 广泛应用于早期的各种搜索引擎

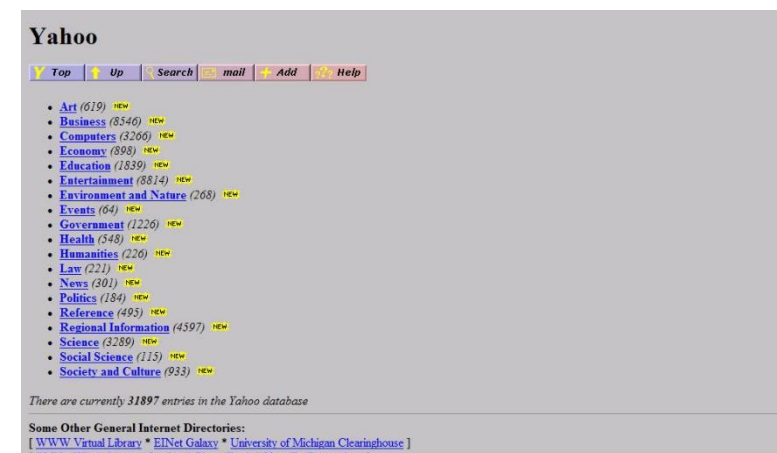
- 例如，时常被拉出来当靶子的Yahoo！

- 在问题规模较小时较为有效

- 严重缺乏可扩展性，代价高昂效率低下

1995年的Yahoo！

我们已经多次见到它了



- 基于规则的分类
- 基于监督学习的分类
 - 决策树
 - 最近邻分类
 - 支持向量机
 - 不平衡分类问题

- **基于规则的分类器**

- 未来的程序猿（媛）们都知道的基本语法：if...then...
- 基于规则的分类器（Rule-based Classifier）就是使用一组if-then来进行分类
 - 基本形式：Condition（规则） \rightarrow y（标签）
 - 其中，Condition是一组属性的组合，也被称作规则的前提
 - 例如：
 - (胎生 = 否) \wedge (飞行动物 = 是) \rightarrow 鸟类
 - (胎生 = 是) \wedge (体温 = 恒温) \rightarrow 哺乳类

- **规则分类器的基本原理**

- 基于规则的分类器所产生的规则集，应具有以下两个重要性质：
- **互斥原理** (Mutually Exclusive Rule)
 - 如果规则集中不存在两条规则被同一条记录出触发，则规则是互斥的
 - 该性质确保每条记录**至多**被一条规则所覆盖
- **穷举原理** (Exhaustive Rule)
 - 对于属性值的任一组合，规则集中都存在某条规则加以覆盖
 - 该性质确保每条记录**至少**被一条规则所覆盖

- 规则分类器的基本原理

R1: (Give Birth = no) (Can Fly = yes) → Birds

R2: (Give Birth = no) (Live in Water = yes) → Fishes

R3: (Give Birth = yes) (Blood Type = warm) → Mammals

R4: (Give Birth = no) (Can Fly = no) → Reptiles

R5: (Live in Water = sometimes) → Amphibians

| Name | Blood Type | Give Birth | Can Fly | Live in Water | Class |
|---------------|------------|------------|---------|---------------|-------|
| lemur | warm | yes | no | no | ? |
| turtle | cold | no | no | sometimes | ? |
| dogfish shark | cold | yes | no | yes | ? |

- Turtle同时触发R4与R5，因此出现了矛盾。
 - 解决方法：重新制定规则、采用有序规则（只满足第一个触发的规则）或采用投票机制
- Dogfish Shark无法对应任何一条规则，因此无法分类
 - 解决方法：制定一条“兜底”规则，解决所有不被其他规则包含的样本

- 规则分类器的有序性

- 在互斥原理未被遵守的情况下，一条记录可能被多条规则覆盖
 - 此时，这些规则的结果之间可能存在冲突
- 一种解决方法是：对规则集按照优先级降序排列
 - 对于一条记录，按照所触发的排序最高的规则进行执行

则执行

☒ 规则

☐ 移动到 [新建文件夹](#)

☐ 自动转发

☐ 自动回复

☐ 拒收

☒ 执行本规则后，不继续下一条规则

- 规则分类器的有序性

- 常见的两种规则排序方法

- 基于规则的排序方案：按照规则的质量（如准确性）进行排序
- 基于类的排序方案：同类的规则排在一起，相对顺序被忽略

Rule-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Class-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income<80K) ==> No

(Refund=No, Marital Status={Married}) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income>80K) ==> Yes

- 如何制定规则分类器

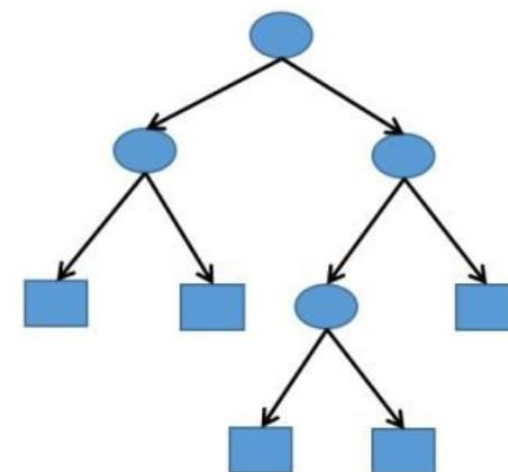
- 最基础的方法：人工制定规则进行分类
 - 相较于纯手工打造，效率更高，运用更广泛
 - 例如，各大门户网站持续不断维护更新的敏感词列表.....
 - 一般而言，各种商业IR系统都会制定复杂的查询规则
 - 如果人工规则得到定期更新，效果往往较好
 - 然而，维护规则成本较高，需要持续更新



发表内容含有敏感词, 请检查
后再重新发布...1000

返回

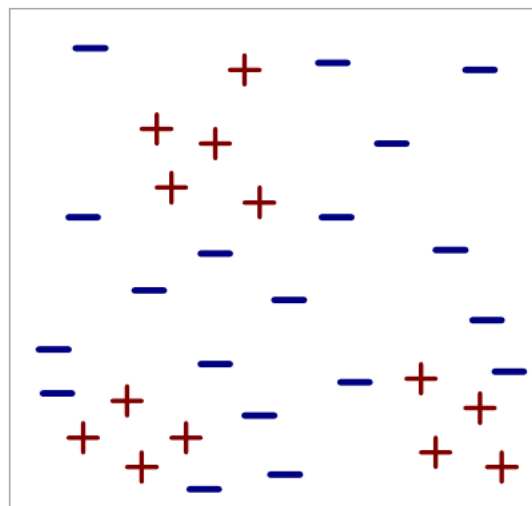
- **如何制定规则分类器**
- 如何利用算法自动生成规则？
 - 直接方法：从数据中自动学习规则
 - 例如，顺序覆盖（Sequential Covering）、RIPPER、CN2等算法
 - 间接算法：借助其他分类模型学习规则
 - 典型例子：从决策树中提炼规则



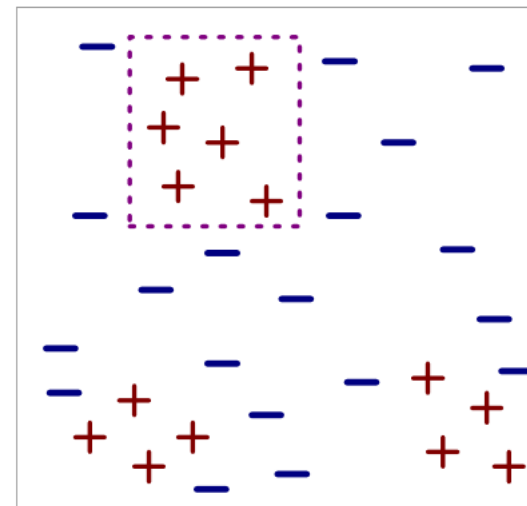
- **直接方法：顺序覆盖**

- 一种基础的规则生成算法，采用某种评估以贪心方式进行学习
- 算法的基本流程如下：
 - 算法开始时，决策表为空，即不包含任何规则
 - 每一步针对某个类 y ，提取覆盖当前训练集的最佳规则
 - 什么是好的规则？覆盖的样本尽可能的多，同时样本类别尽可能一致
 - 如果规则能覆盖大多数正例（即 y ），而没有或覆盖极少负例（非 y ），则保留该规则
 - 将该规则加入决策表的尾端，同时删除该规则覆盖的所有训练样本
 - 重复上述过程，直至满足终止条件（例如：某个增益的阈值，如熵或准确率）

- **直接方法：顺序覆盖**
- 顺序覆盖 (Sequential Covering) 的图示

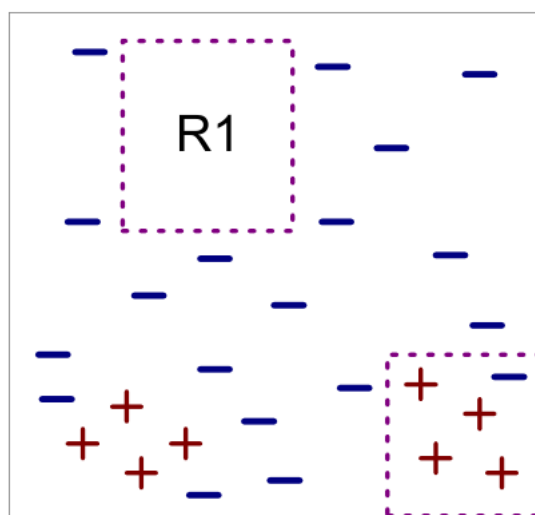


(i) Original Data

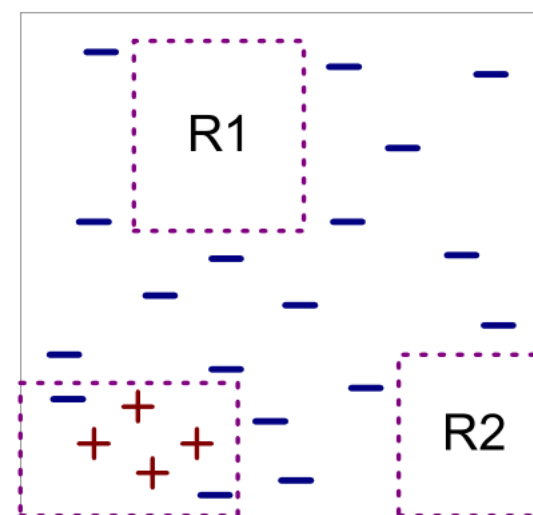


(ii) Step 1

- **直接方法：顺序覆盖**
- 顺序覆盖 (Sequential Covering) 的图示



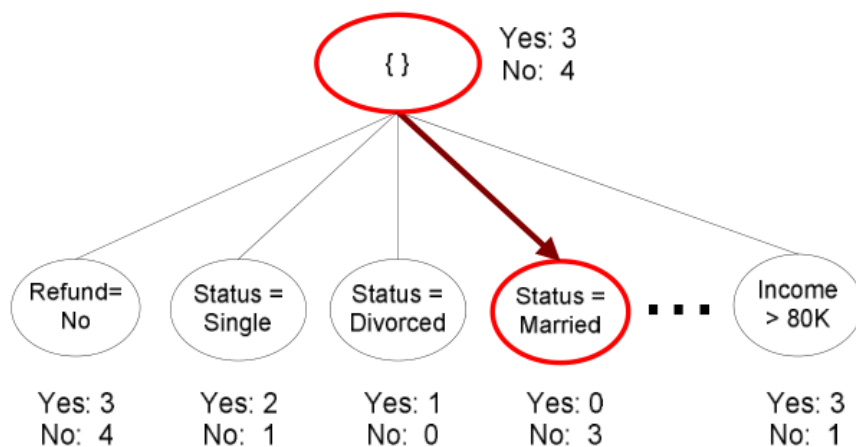
(iii) Step 2



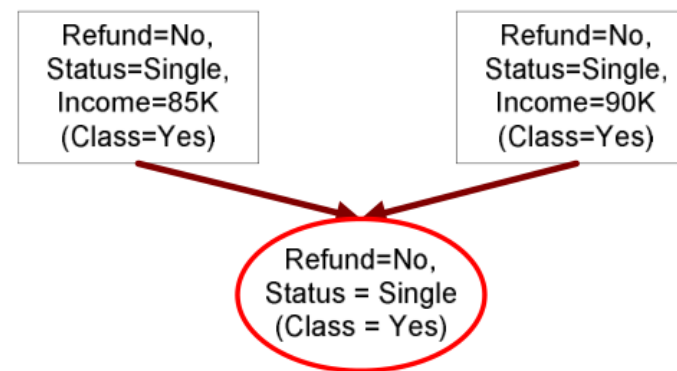
(iv) Step 3

• 规则增长策略

- 由于搜索空间呈指数增长，找到一个最佳规则的计算开支很大
- 一般而言，采用一种贪心的增长规则来解决指数搜索问题
- 常见的分类规则增长策略包括 “一般到特殊” 和 “特殊到一般” 两种



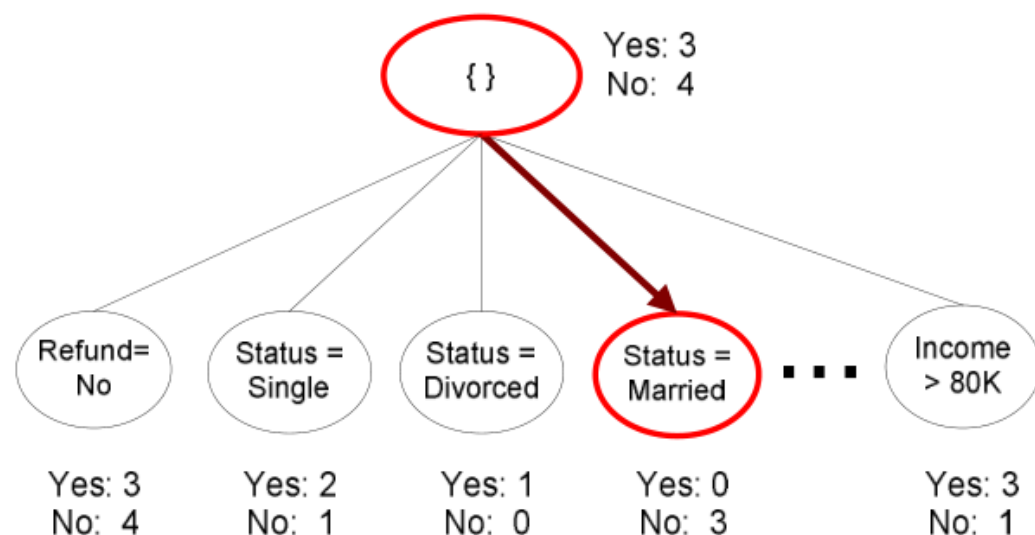
(a) General-to-specific



(b) Specific-to-general

- 规则增长策略

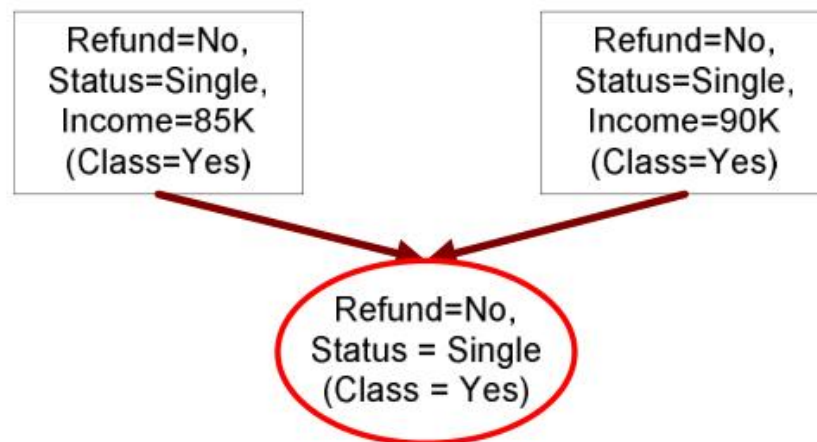
- “从一般到特殊”：初始规则条件为空，给定目标标签 y
- 逐步加入合取项（通过AND运算相连）来提高规则质量
- 例如图中，通过添加规则 “Status = Married”，提升了规则准确率



(a) General-to-specific

- 规则增长策略

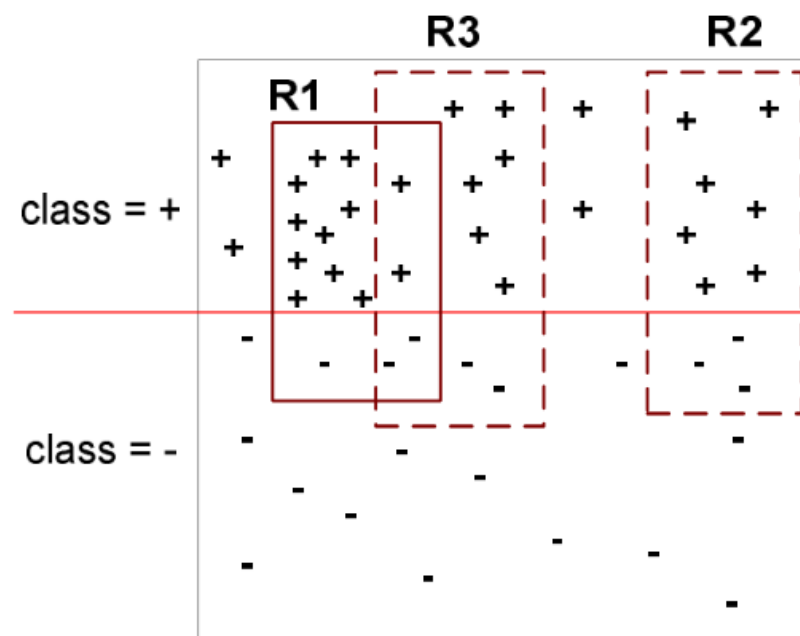
- “从特殊到一般”：随机选择一个正样本作为初始种子
- 逐步删除规则中的合取项，来覆盖更多的同类别正例
- 例如图中，为了覆盖更多正例，删除了原来对Income 的限定



(b) Specific-to-general

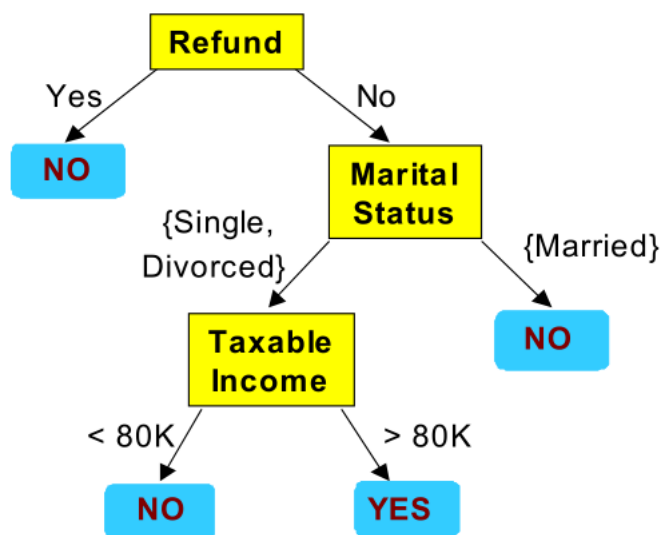
- 为何要删除样本

- 顺序覆盖的流程中有一步：“删除该规则覆盖的所有样本”
 - 为何要删除这些样本？
- 顺序覆盖的本质是一种贪心法思想，新规则对于整体分类的增益具有边际效益
 - 例如右图，如果在选定R1后不删除所覆盖的样本，那么会影响R3准确率的评估



- 间接方法：基于决策树生成规则

- 事实上，决策树从根节点到叶节点的每一条路径都对应一条分类规则
 - 路径中的条件构成规则的前提，而叶节点的类标号对应规则的分类结果
 - 注意：这种情况下生成的规则是满足互斥和穷举两条原理的



Classification Rules

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single, Divorced}, Taxable Income<80K) ==> No

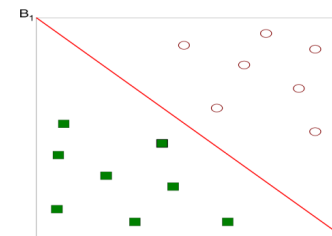
(Refund=No, Marital Status={Single, Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

- **基于规则分类器的优点**
- 基于规则的分类器具有较好的可解释性和直观性
 - 这一点与决策树类似，区别主要在于呈现方式
- 基于规则的分类器生成较为简便，分类也较为迅速
 - 当然，如果面临违反互斥原则的情况，可能涉及多条规则投票，相对较为复杂

- 基于规则的分类
- **基于监督学习的分类**
 - 决策树
 - 最近邻分类
 - 支持向量机
 - 不平衡分类问题

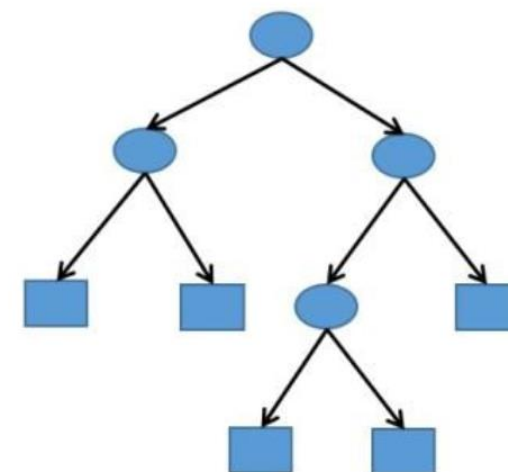
- **更为普遍的分类方法：基于有监督学习**
- 通过训练样本，学习从样本到类别标签的映射方式，实现标签分类
 - 常见的有监督分类方法包括
 - 决策树：类似于基于规则的分类方法，简单易行
 - K-最近邻分类法：简单，有效，但取决于数据分布
 - 支持向量机：效果较好，但核函数的选取是个问题
 - 朴素贝叶斯、逻辑回归、人工神经网络.....
 - 通常情况下，实用中往往采用多种技术的混合模型进行分类



- 基于规则的分类
- 基于监督学习的分类
 - 决策树
 - 最近邻分类
 - 支持向量机
 - 不平衡分类问题

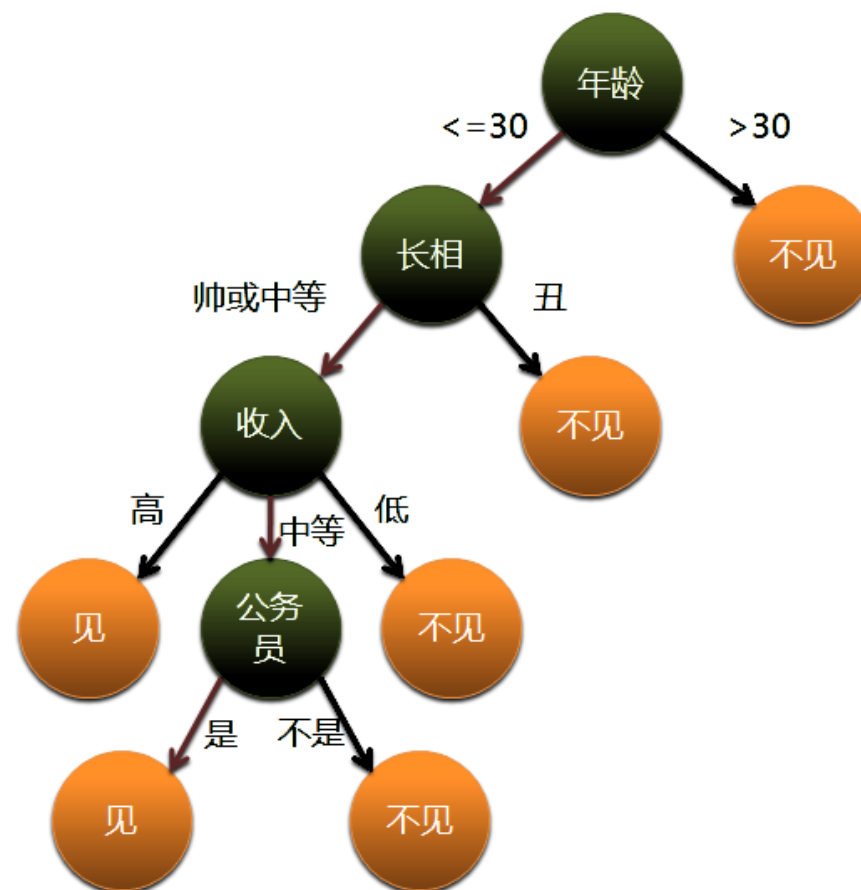
- 什么是决策树

- 在前面，我们介绍了基于规则进行分类的技术
 - 在满足特定要求（两大原理）的条件下，将规则以树状形式呈现出来
- 决策树（Decision Tree）的定义
 - 一种典型的分类方法，首先对数据进行处理，利用归纳算法生成可读的规则和决策树，然后使用决策对新数据进行分析。
 - 本质上，决策树仍是通过一系列规则对数据进行分类的过程。



• 喜闻乐见的决策树实例

- 假如你的家人给你安排了一次相亲，你去不去呢？



- 决策树如何生成？

- 基本的决策树学习过程，可以归纳为以下三个步骤：
 - 特征选择：选取对于训练数据有着**较强区分能力**的特征
 - 例如，相亲时“有房”是个具有竞争力的要素
 - 生成决策树：基于选定的特征，逐步生成完整的决策树
 - 例如，将所有相亲条件依次整合，最终形成上一页择偶标准
 - 决策树剪枝：简化部分枝干，避免过拟合因素影响
 - 例如，把“学历”等条件剪去，最后保留“真爱”这一项

- 决策树的特征选择问题

- 选取对训练数据具有区分能力的特征，从而提高决策树学习的效率
 - 如果某特征分类的结果与随机结果没有很大的差别，则称这个特征是没有分类能力的
 - 经验上，扔掉这样的特征对决策树学习的精度影响不大

表 5.1 贷款申请样本数据表

| ID | 年龄 | 有工作 | 有自己的房子 | 信贷情况 | 类别 |
|----|----|-----|--------|------|----|
| 1 | 青年 | 否 | 否 | 一般 | 否 |
| 2 | 青年 | 否 | 否 | 好 | 否 |
| 3 | 青年 | 是 | 否 | 好 | 是 |
| 4 | 青年 | 是 | 是 | 一般 | 是 |
| 5 | 青年 | 否 | 否 | 一般 | 否 |
| 6 | 中年 | 否 | 否 | 一般 | 否 |
| 7 | 中年 | 否 | 否 | 好 | 否 |
| 8 | 中年 | 是 | 是 | 好 | 是 |
| 9 | 中年 | 否 | 是 | 非常好 | 是 |
| 10 | 中年 | 否 | 是 | 非常好 | 是 |
| 11 | 老年 | 否 | 是 | 非常好 | 是 |
| 12 | 老年 | 否 | 是 | 好 | 是 |
| 13 | 老年 | 是 | 否 | 好 | 是 |
| 14 | 老年 | 是 | 否 | 非常好 | 是 |
| 15 | 老年 | 否 | 否 | 一般 | 否 |

- **基本概念：信息熵**

- 了解特征选择的准则之前，先了解基本的信息熵（Entropy）的概念
 - 熵是表示随机变量不确定性的度量，不确定性越高，熵越高
 - 设D是一个取自有限个值的离散随机变量，其概率分布如下：

$$P(D = d_k) = p_k, \quad k = 1, 2, \dots, |y|$$

- 那么，随机变量D的熵定义为：

$$Ent(D) = - \sum_{K=1}^{|y|} p_k \log_2 p_k$$

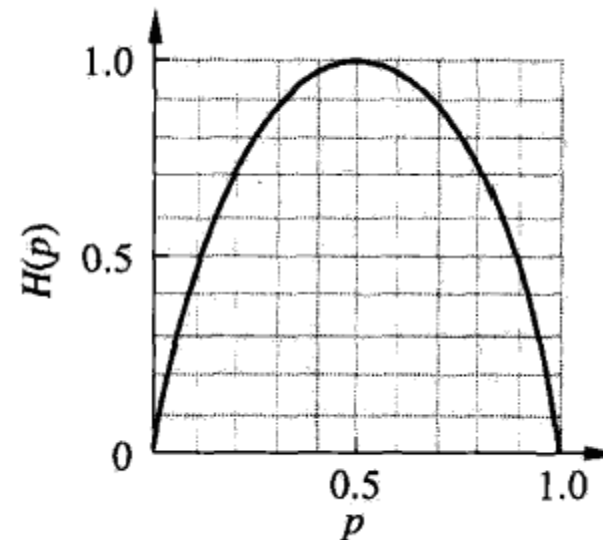
- **基本概念：信息熵**

- 假设随机变量的取值为非0即1时，可以直观看出熵的变化趋势

- 此时，熵的计算公式为： $\text{Ent}(X) = -p \log_2 p - (1-p) \log_2 (1-p)$

- 熵随概率 p 变化的曲线如图所示。

- 可以看出， $p=0$ 或 1 时， $\text{Ent}(X)=0$ ，即没有不确定性。
 - 当 $p=0.5$ 时，熵取值最大，同时随机变量不确定性最大。



- **特征选择准则 (1) 信息增益**

- 特征A对训练数据集D的信息增益 $Gain(D, a)$ ，定义为集合D的经验熵 $Ent(D)$ 与特征A在给定条件下对D的经验条件熵 $Ent(D|a)$ 之差，即：

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

- 其中，公式的后半部分，也就是经验条件熵 $Ent(D|a)$ 的部分，即

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

- 特征选择准则 (1) 信息增益

- 一个计算实例，来自周志华老师的西瓜书（《机器学习》）

表 4.1 西瓜数据集 2.0

| 编号 | 色泽 | 根蒂 | 敲声 | 纹理 | 脐部 | 触感 | 好瓜 |
|----|----|----|----|----|----|----|----|
| 1 | 青绿 | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 是 |
| 2 | 乌黑 | 蜷缩 | 沉闷 | 清晰 | 凹陷 | 硬滑 | 是 |
| 3 | 乌黑 | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 是 |
| 4 | 青绿 | 蜷缩 | 沉闷 | 清晰 | 凹陷 | 硬滑 | 是 |
| 5 | 浅白 | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 是 |
| 6 | 青绿 | 稍蜷 | 浊响 | 清晰 | 稍凹 | 软粘 | 是 |
| 7 | 乌黑 | 稍蜷 | 浊响 | 稍糊 | 稍凹 | 软粘 | 是 |
| 8 | 乌黑 | 稍蜷 | 浊响 | 清晰 | 稍凹 | 硬滑 | 是 |
| 9 | 乌黑 | 稍蜷 | 沉闷 | 稍糊 | 稍凹 | 硬滑 | 否 |
| 10 | 青绿 | 硬挺 | 清脆 | 清晰 | 平坦 | 软粘 | 否 |
| 11 | 浅白 | 硬挺 | 清脆 | 模糊 | 平坦 | 硬滑 | 否 |
| 12 | 浅白 | 蜷缩 | 浊响 | 模糊 | 平坦 | 软粘 | 否 |
| 13 | 青绿 | 稍蜷 | 浊响 | 稍糊 | 凹陷 | 硬滑 | 否 |
| 14 | 浅白 | 稍蜷 | 沉闷 | 稍糊 | 凹陷 | 硬滑 | 否 |
| 15 | 乌黑 | 稍蜷 | 浊响 | 清晰 | 稍凹 | 软粘 | 否 |
| 16 | 浅白 | 蜷缩 | 浊响 | 模糊 | 平坦 | 硬滑 | 否 |
| 17 | 青绿 | 蜷缩 | 沉闷 | 稍糊 | 稍凹 | 硬滑 | 否 |

- 特征选择准则 (1) 信息增益

- 一个计算实例，来自周志华老师的西瓜书（《机器学习》）
 - 首先，整体的熵可以根据“是否为好瓜”进行计算如下：

$$\text{Ent}(D) = - \sum_{k=1}^2 p_k \log_2 p_k = - \left(\frac{8}{17} \log_2 \frac{8}{17} + \frac{9}{17} \log_2 \frac{9}{17} \right) = 0.998 .$$

- 其次，考虑特定特征，以“色泽”为例

D^1 (色泽=青绿), D^2 (色泽= 乌黑), D^3 (色泽=浅白)

$$\text{Ent}(D^1) = - \left(\frac{3}{6} \log_2 \frac{3}{6} + \frac{3}{6} \log_2 \frac{3}{6} \right) = 1.000$$

$$\text{Ent}(D^2) = - \left(\frac{4}{6} \log_2 \frac{4}{6} + \frac{2}{6} \log_2 \frac{2}{6} \right) = 0.918$$

$$\text{Ent}(D^3) = - \left(\frac{1}{5} \log_2 \frac{1}{5} + \frac{4}{5} \log_2 \frac{4}{5} \right) = 0.722$$

- **特征选择准则 (1) 信息增益**

- 一个计算实例，来自周志华老师的西瓜书（《机器学习》）
 - 由此，可以计算特征“色泽”的经验条件熵如下：

$$\begin{aligned}\text{Gain}(D, \text{色泽}) &= \text{Ent}(D) - \sum_{v=1}^3 \frac{|D^v|}{|D|} \text{Ent}(D^v) \\ &= 0.998 - \left(\frac{6}{17} \times 1.000 + \frac{6}{17} \times 0.918 + \frac{5}{17} \times 0.722 \right) \\ &= 0.109 .\end{aligned}$$

- 类似的，计算其他特征的信息增益，可以最终选择最合适的特征

• 特征选择准则 (2) 信息增益率

- 信息增益虽然能够较好地体现某个特征在降低信息不确定性方面的贡献
 - 信息增益越大, 说明信息纯度提升越快, 最后结果的不确定性越低
- 但是, 信息增益也具有一定的局限性, 尤其体现在更偏好可取值较多的属性
 - 取值较多, 不确定性相对更低, 因此得到的熵偏低

• 如何改进? 引入信息增益率:

- 分母相当于取值 (标签) 的不确定性
 - 本质上是引入一个惩罚项
 - 取值越少, 惩罚项越小

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

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

- **特征选择准则 (2) 信息增益率**

- 信息增益率有矫枉过正的危险
 - 采用信息增益率的情况下，往往倾向于选择取值较少的特征
 - 当取值较少时， $IV(a)$ 较小，因此惩罚项相对较小
 - 目前通常采用折中的方法
 - 先从候选特征中，找到信息增益高于平均水平的集合
 - 再从这一集合中，找到信息增益率最大的特征

- **特征选择准则 (3) 基尼指数**

- 信息的纯度的衡量标准

- 基尼指数的目的，在于表示样本集合中一个随机样本被分错的概率
- 基尼指数越低，表明被分错的概率越低，相应的信息纯度也就越高
- 假设有K个类，样本点属于第k类的概率为 p_k ，则基尼指数定义为

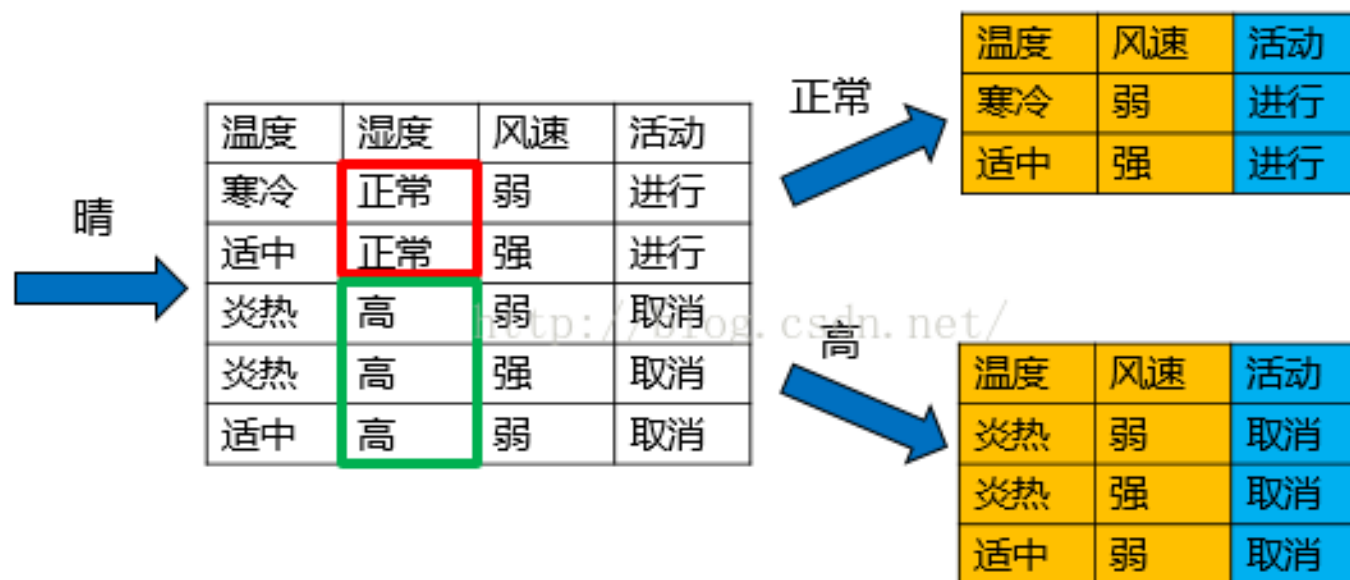
$$\text{Gini}(p) = \sum_{k=1}^K p_k(1 - p_k) = 1 - \sum_{k=1}^K p_k^2$$

- **决策树的生成过程**

- 决策树的最终目标，在于使每个节点所对应的样本类别均为“纯”的
- 以十大经典算法之一的C4.5为例，当某个节点对应的样本集合“不纯”时
 - 计算当前节点的类别信息熵
 - 计算当前节点各个属性的信息熵，并进而计算得到该属性对应的信息增益率
 - 基于最大信息增益率的属性，对节点对应的样本集合进行分类
 - 重复上述过程，直至节点对应的样本集合为“纯”的集合（即样本类别统一）
- 其他决策树生成算法过程类似，区别在于准则不同
 - ID3（原型算法）采用信息增益，而CART采用基尼指数

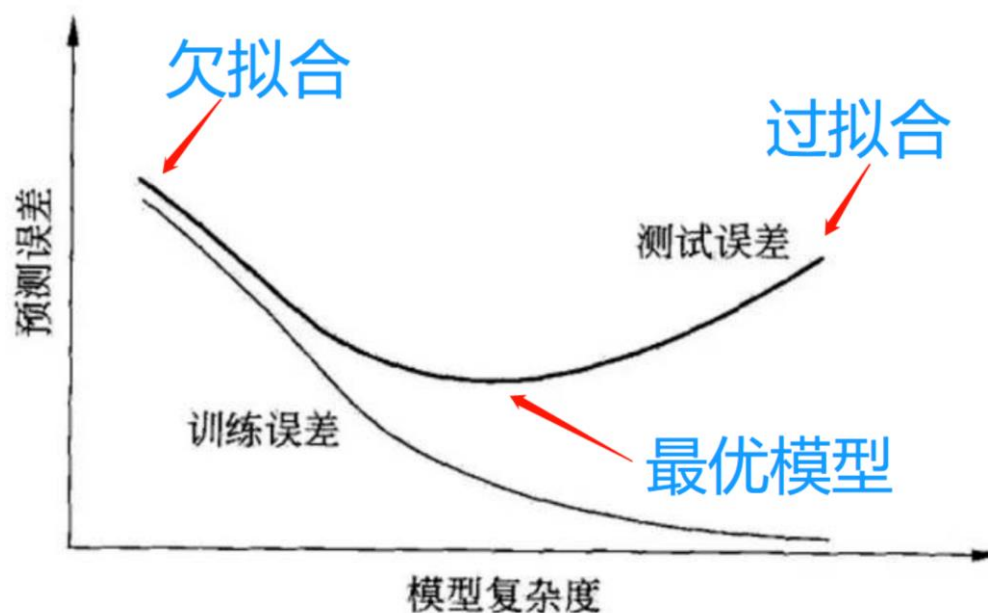
- 决策树的生成实例

- 迭代式地对节点进行分类，直到某个节点上的样本类别统一
 - 此时，该节点为决策树的叶子节点



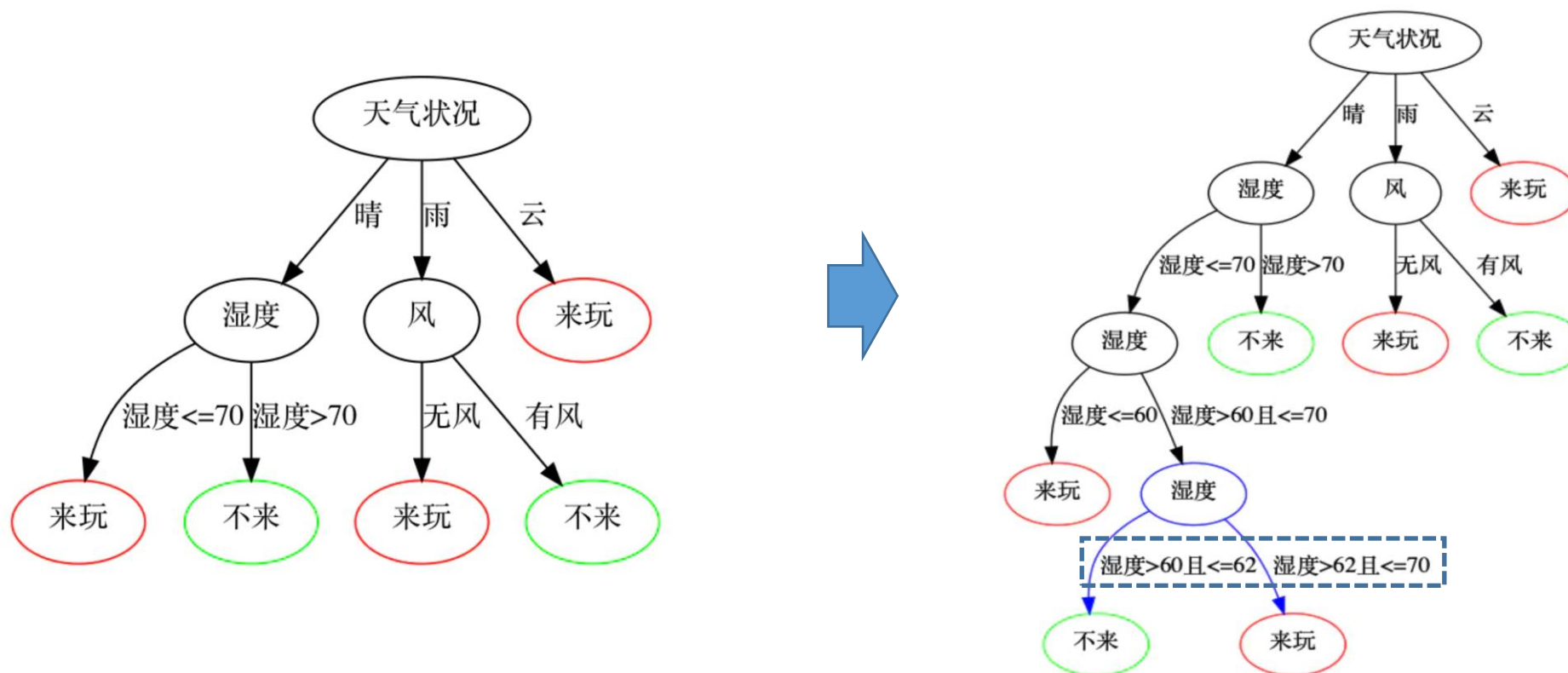
- 决策树生成是否一劳永逸？

- 事实上，训练任何分类模型都要“有度”，否则可能“过犹不及”
 - 如果训练集与测试集效果都不好，说明出现“欠拟合”现象
 - 如果训练集效果好，而测试集效果不好，说明出现“过拟合”现象



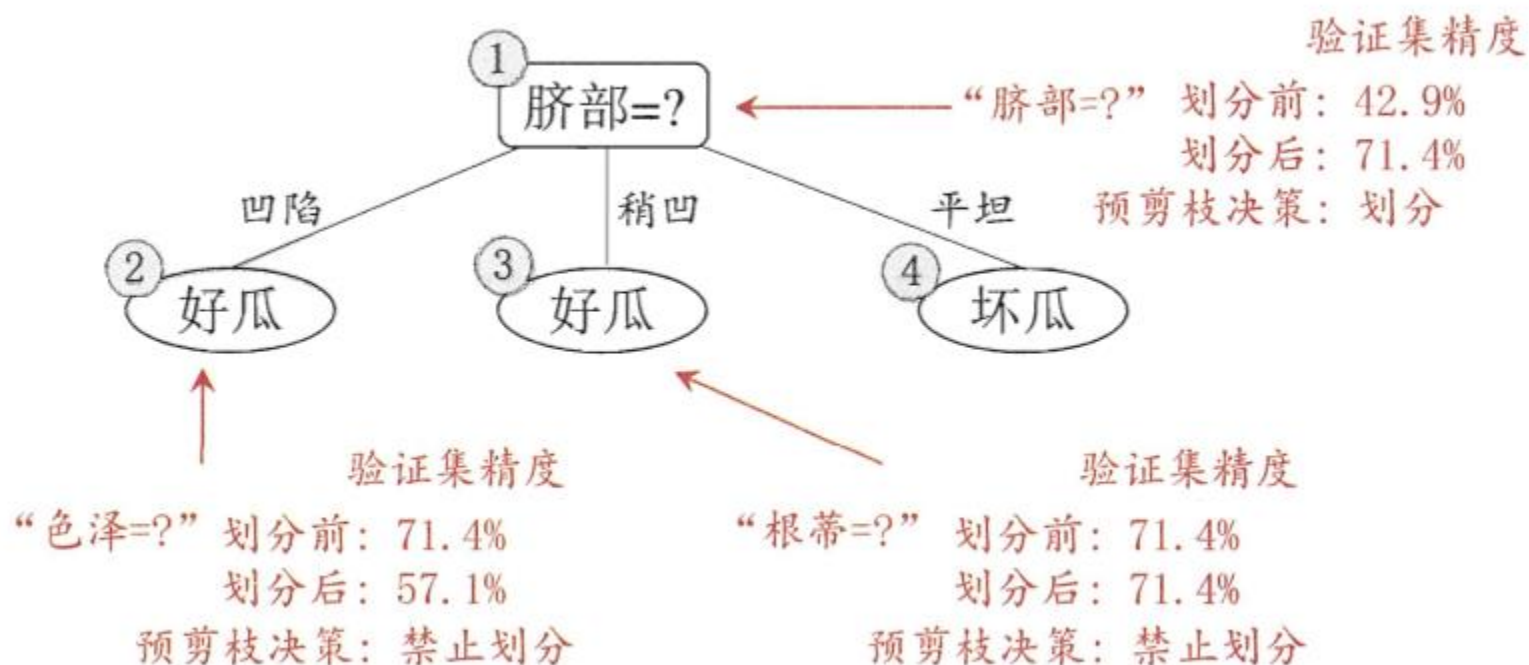
- 决策树的剪枝

- 在决策树问题中，过拟合来自于过度迁就训练数据特性，而导致构造出过于细枝末节的决策树，泛化能力较差，因此需要剪去细枝末节的部分：剪枝



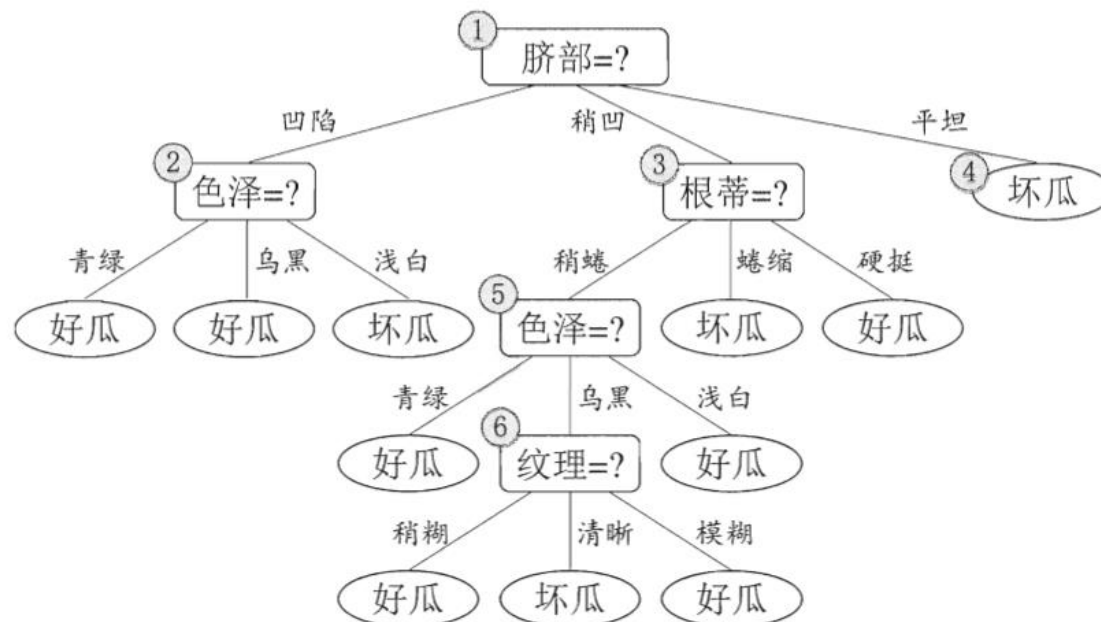
- 决策树的剪枝策略（1）预剪枝

- 在生成决策树的过程中即进行剪枝，称作“预剪枝”
 - 每个节点划分前，衡量当前节点的划分能否提高决策树的泛化能力



• 决策树的剪枝策略（2）后剪枝

- 在生成决策树之后再行剪枝，称作“后剪枝”
 - 自底向上考察每个非叶子节点，考虑将该节点替换成叶子节点后能否提高泛化性能

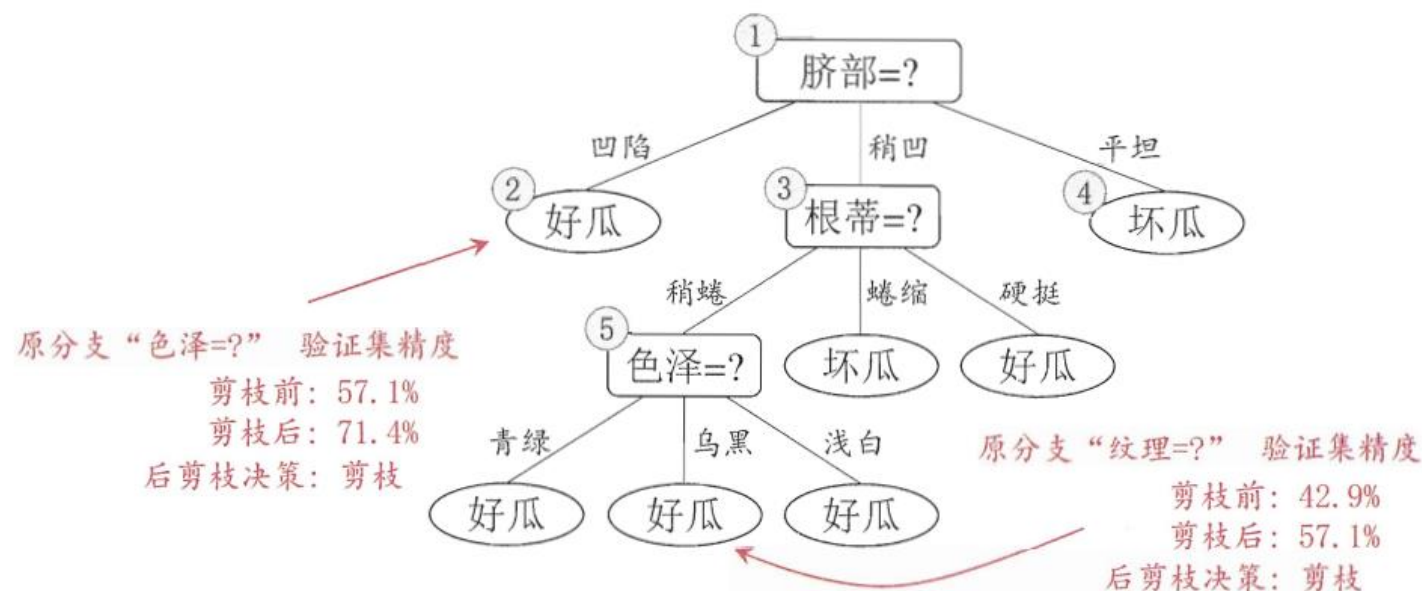


剪枝前

图 4.5 基于表 4.2 生成的未剪枝决策树

• 决策树的剪枝策略（2）后剪枝

- 在生成决策树之后再进行剪枝，称作“后剪枝”
 - 自底向上考察每个非叶子节点，考虑将该节点替换成叶子节点后能否提高泛化性能

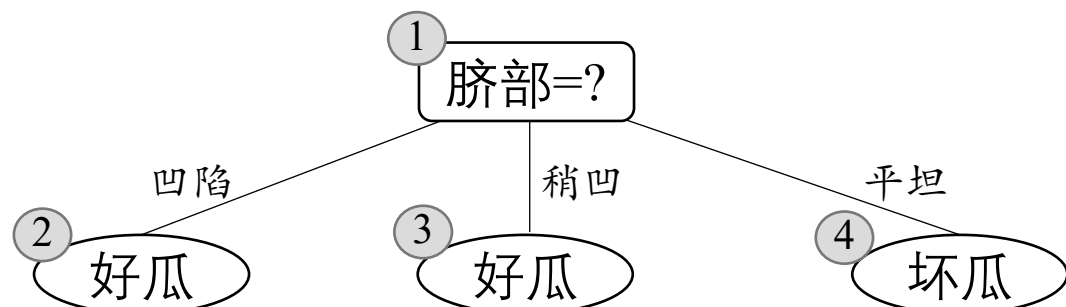


剪枝后

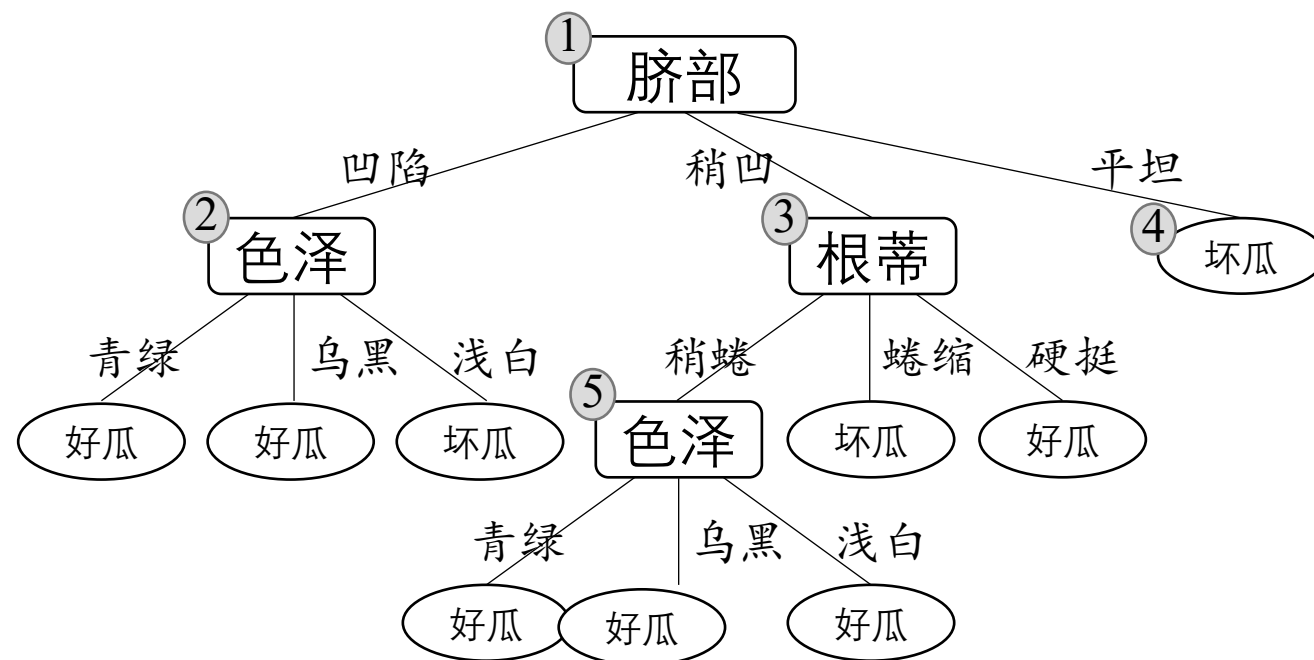
图 4.7 基于表 4.2 生成的后剪枝决策树

- 两种剪枝策略的比较

- 我们来对比一下预剪枝与后剪枝各自得到的最终决策树



预剪枝



后剪枝

- 基于规则的分类
- 基于监督学习的分类
 - 决策树
 - 最近邻分类
 - 支持向量机
 - 不平衡分类问题

• **VSM与文档的向量化表示**

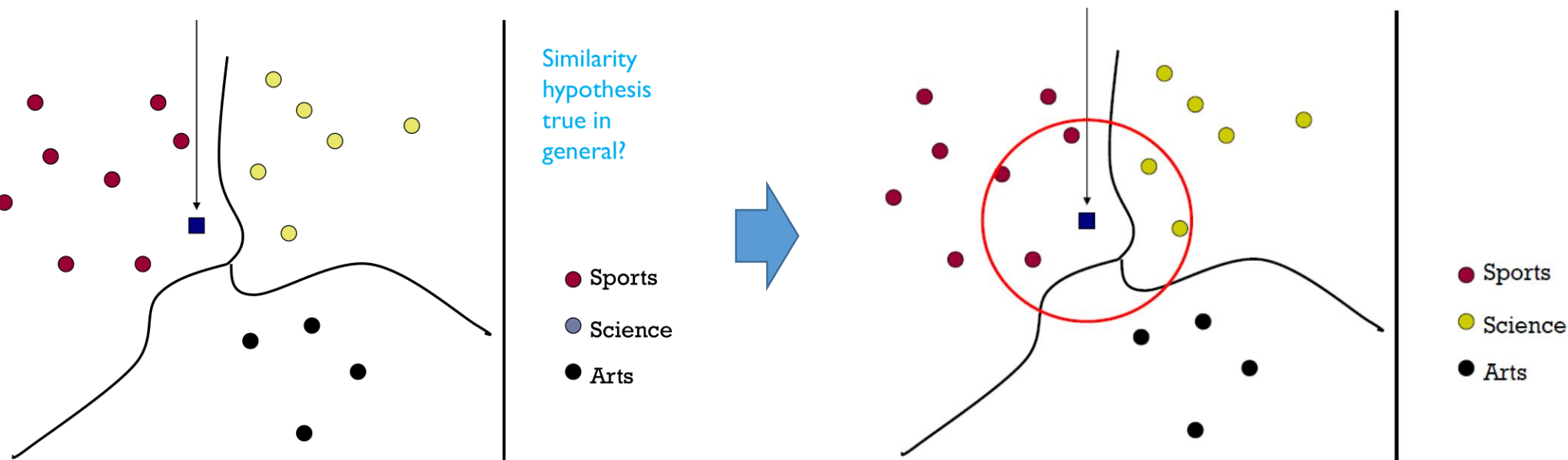
• 向量空间模型 (Vector Space Model) 与文档表示示例

| | Antony and Cleopatra | Julius Caesar | The Tempest | Hamlet | Othello | Macbeth |
|-----------|----------------------|---------------|-------------|--------|---------|---------|
| Antony | 5.25 | 3.18 | 0 | 0 | 0 | 0.35 |
| Brutus | 1.21 | 6.1 | 0 | 1 | 0 | 0 |
| Caesar | 8.59 | 2.54 | 0 | 1.51 | 0.25 | 0 |
| Calpurnia | 0 | 1.54 | 0 | 0 | 0 | 0 |
| Cleopatra | 2.85 | 0 | 0 | 0 | 0 | 0 |
| mercy | 1.51 | 0 | 1.9 | 0.12 | 5.25 | 0.88 |
| worser | 1.37 | 0 | 0.11 | 4.15 | 0.25 | 1.95 |

- 所以, D1对应的向量为(5.25, 1.21, 8.59, 0, 2.85, 1.51, 1.37)
- 基于文档的向量化表示, 可以通过向量之间的相似度来判断相似文档

- 相似文档与最近邻假设

- 先前的向量空间模型遵循一个思路：表征空间上相近的文档是相似的
- 同样的，可以提出合理假设：表征空间上相近的文档应该属于同一个类别



- 最近邻分类的基本要素

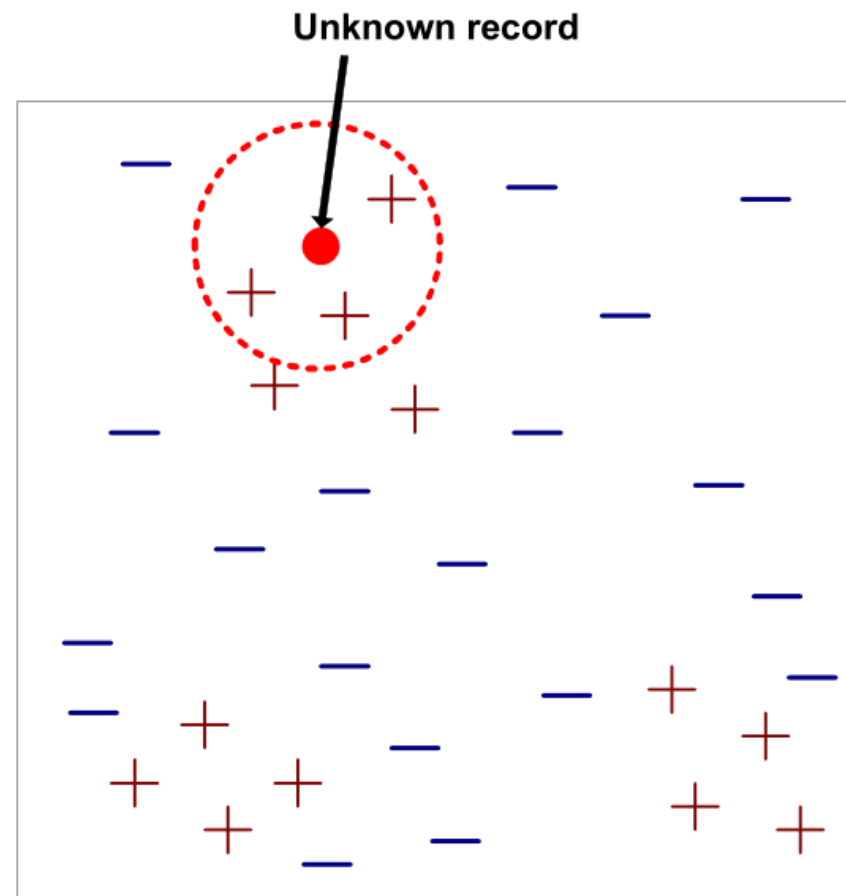
- K-最近邻分类 (K Nearest Neighbor)

- 模型输入:

- 训练样本集合与距离度量
- K值, 用于限定最近邻的数量

- 算法流程:

- 给定未知样本, 首先计算与其他样本的距离, 找到K-最近邻
- 基于K-最近邻的类别确定分类结果



- **最近邻分类的关键问题（1）距离度量**

- 最近邻分类的效果严重依赖于距离（或相似性）的度量

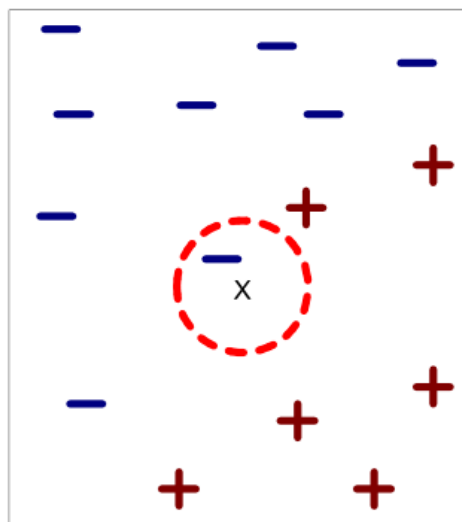
- 对于高维空间而言，最基本的度量方式为欧式距离

$$D(\mathbf{x}, \mathbf{x}') = \sqrt{\sum_i \sigma_i^2 (\mathbf{x}_i - \mathbf{x}'_i)^2}$$

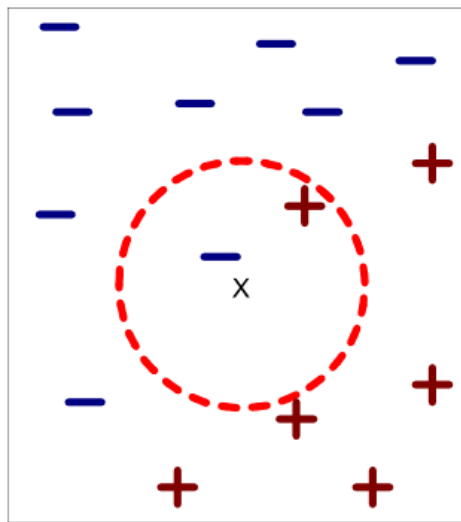
- 如果采用0/1向量，则可使用汉明距离（Hamming）代替
 - 相当于统计有多少维的数字不相同
 - 除此之外，对于文本而言（如采用TF-IDF），可使用余弦相似度
 - 其他可采用的度量如马氏距离、无穷范数（向量最大值）等

- **最近邻分类的关键问题 (2) K的取值**

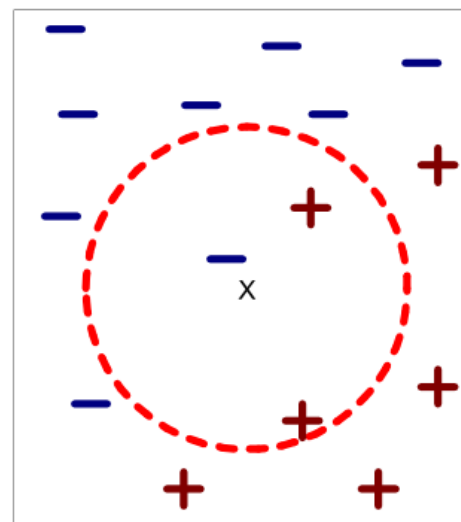
- 最近邻分类的效果同样严重依赖于K的取值 (即邻居的数量)
 - K太小, 则容易受噪声干扰, K太大, 又可能导致错误涵盖其他类别样本



(a) 1-nearest neighbor



(b) 2-nearest neighbor



(c) 3-nearest neighbor

- **最近邻分类的特点**

- 最近邻分类是一种典型的基于实例的学习
 - 使用具体实例进行预测，而不需要对数据进行抽象（如提取特征）
- 最近邻分类是一种消极学习，不需要模型，但分类过程开销很大
 - 相比之下，积极学习方法训练模型较为费时费力，但基于模型分类很快
- 最近邻分类基于局部信息进行判别，受噪声影响很大
- 最近邻分类需要慎重选择度量并预处理数据，否则可能被误导
 - 例如，借助身高体重进行分类，身高波动范围不大，而体重差距巨大

- 基于规则的分类
- 基于监督学习的分类
 - 决策树
 - 最近邻分类
 - 支持向量机
 - 不平衡分类问题

- 写在前面：一本参考书

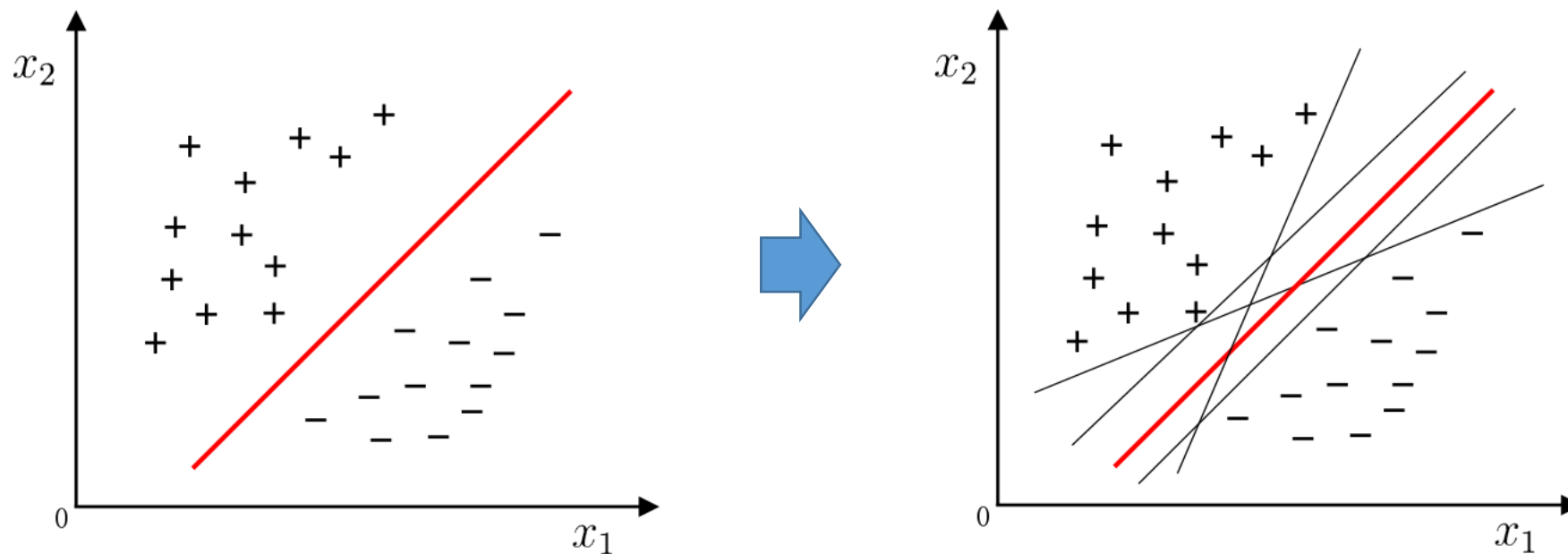
参考书目：支持向量机：理论、算法与拓展，

邓乃扬 / 田英杰，科学出版社，2009



- 高维空间中的节点分割问题

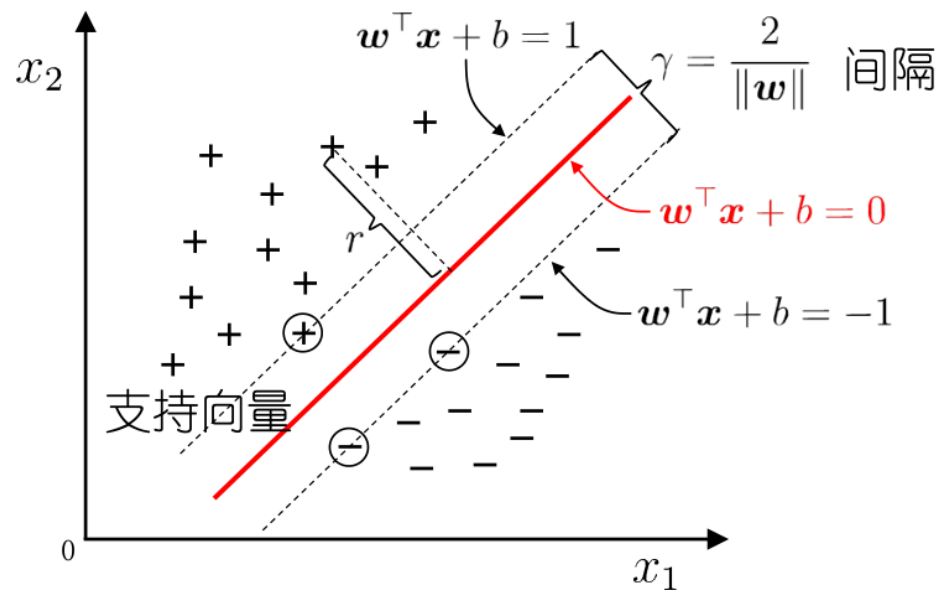
- 如果仅考虑二分类问题，那么分类问题可以转化为寻找一个超平面，实现对于高维空间中的节点进行有效分割



将训练样本分开的超平面可能有很多, 哪一个好呢?

- 高维空间中的节点分割问题

- 显然，应该选择“正中”的最大间隔超平面，容忍性好，泛化能力强
 - 符合这样条件的超平面，在线性可分的条件下“存在且唯一”（为什么？）



其中， w, b 为参数， w 向量方向垂直于超平面。离超平面最近的节点（带圈的部分）被称作“支持向量”

有超平面方程如右： $w^T x + b = 0$

- 支持向量机及其基本形式

- 支持向量机（Support Vector Machine）的目的在于找到这样一个最大间隔超平面，使得超平面对应的间隔 γ 最大化

$$\begin{array}{ll} \arg \max_{\mathbf{w}, b} & \frac{2}{\|\mathbf{w}\|} \\ \text{s.t.} & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, m. \end{array}$$



$$\begin{array}{ll} \arg \min_{\mathbf{w}, b} & \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \dots, m. \end{array}$$

- 支持向量机的求解方式

- 问题：如何求解支持向量机的基本形式？
- 我们注意到，该形式符合凸二次规划问题特征，可以借助拉格朗日对偶性，通过求解对偶问题加以求解

$$\begin{aligned} \min \quad & f_0(x), \quad x \in R^n, \\ \text{s.t.} \quad & f_i(x) \leq 0, \quad i = 1, \dots, m, \\ & h_i(x) = a_i^T x - b_i = 0, \quad i = 1, \dots, p, \end{aligned}$$

满足以上形式的最优化问题为凸规划问题，其中 f_0 与 f_i 为连续可微凸函数， h_i 为一般线性函数
事实上， h_i 是什么对我们来说并不重要，因为支持向量机里没有这一部分.....

- 支持向量机的求解方式

- 凸规划问题对应的拉格朗日函数如下：

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) + \sum_{i=1}^p \nu_i h_i(x),$$

- 其中， λ 与 ν 为拉格朗日算子
- 理论证明可知，该函数的下确界为原凸规划问题最优解的一个下界
 - 证明其实不复杂，只要知道 $L(x, \lambda, \nu) \leq f_0(x)$ 就不难理解
- 因此，求解该问题可以获得原凸规划问题的最优解

- 支持向量机的求解方式

- 接下来，考虑如何将支持向量机的基本形式套入凸规划问题：

$$\begin{aligned} \min \quad & f_0(x) \quad x \in R^n, \\ \text{s.t} \quad & f_i(x) \leq 0, \quad i = 1, \dots, m, \\ & \cancel{h_i(x) = a_i^\top x - b_i = 0, \quad i = 1, \dots, p,} \end{aligned}$$

$$\begin{aligned} \arg \min_{w,b} \quad & \frac{1}{2} \|w\|^2 \\ \text{s.t.} \quad & y_i(w^\top x_i + b) \geq 1, \quad i = 1, 2, \dots, m. \end{aligned}$$

- 支持向量机的求解方式

- 由此，我们采用拉格朗日算子法求解支持向量机

- 第一步：引入拉格朗日乘子 $\alpha_i \geq 0$ 得到拉格朗日函数

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

- 第二步：令 $L(\mathbf{w}, b, \boldsymbol{\alpha})$ 对 \mathbf{w} 和 b 的偏导为零可得

$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i, \quad \sum_{i=1}^m \alpha_i y_i = 0.$$

- 第三步：回代

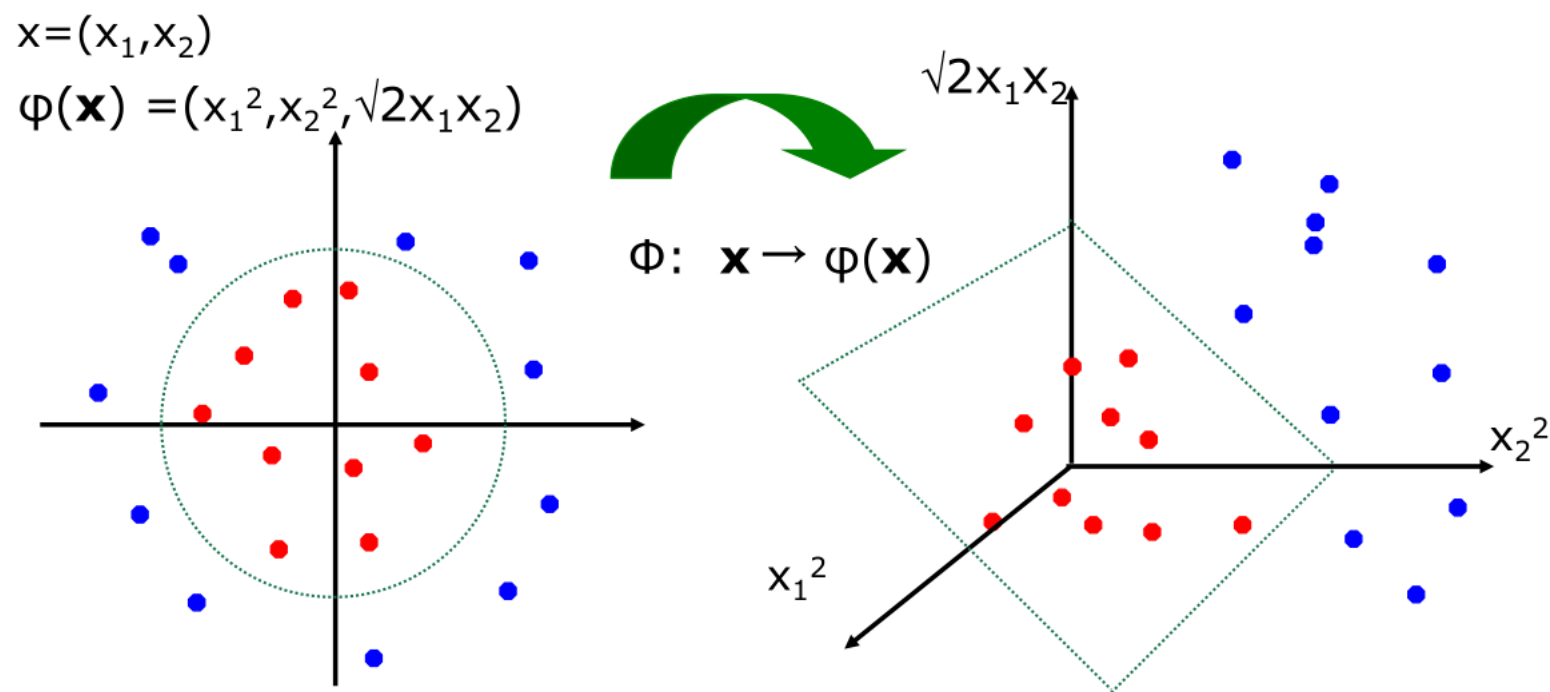
$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j - \sum_{i=1}^m \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad i = 1, 2, \dots, m. \end{aligned}$$

- 支持向量机的求解方式

- 具体而言，求解方式可采用序列最小优化算法（SMO）进行求解
 - 基本思路：不断执行如下两个步骤直至收敛
 - 第一步：选定以对需要更新的向量 α_i 和 α_j
 - 第二步：固定 α_i 和 α_j 以外的参数，求解对偶问题以更新 α_i 和 α_j
 - 挑选 α_i 和 α_j 时，注意挑选差别较大的一对以尽快提升目标函数
 - 在得到所有的 α_i 之后，即可借助 $\alpha_i(y_i f(\mathbf{x}_i) - 1) = 0$ 获得其他参数 w 和 b 的解，其中 $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$

- 线性不可分问题与核函数

- 如果数据集线性不可分，不存在这样一个超平面，怎么办？
 - 将样本映射到一个更高维的特征空间，使得在这个特征空间线性可分



- 为什么需要核函数

- 基于核函数的方法计算两个样本之间的内积

- 从前面的回代式可以发现，优化过程需要计算内积 $\min_{\alpha} \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \mathbf{x}_i^{\top} \mathbf{x}_j - \sum_{i=1}^m \alpha_i$

- 核函数的概念：

- 某些样本在低维空间时线性不可分，通过非线性映射将其映射到高维空间的时候则线性可分，但非线性映射的形式、参数等难以确定。
 - 核函数的目的，在于将高维空间下的内积运算转化为低维空间下的核函数计算，从而避免高维空间可能遇到的“维度灾难”问题。

- 如何选择核函数

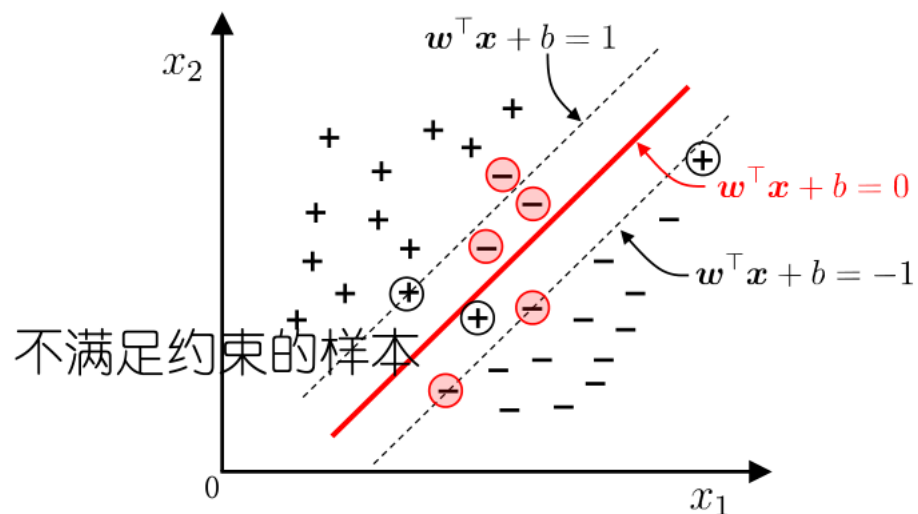
- 常见的核函数如下表所示：

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

- 其中，线性核函数与高斯核函数（径向基）是最为常用的
- 常见挑选方法一般为以下两种：
 - 穷举法：一个个试过来，选择效果最好的一种
 - 混合法：将多个不同的核函数混合起来使用

- 软间隔问题

- 现实中，有时可能既没有办法找到一个线性可分的超平面，也无法确定合适的核函数使得训练样本线性可分。另外，也有过拟合的可能
 - 一种解决方法是：引入“软间隔”，允许少数样本不满足超平面约束



- 软间隔支持向量机

- 引入惩罚项C的同时，对于约束条件部分的目标函数进行修改

原始问题

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \max(0, 1 - y_i(\mathbf{w}^\top \phi(\mathbf{x}_i) + b))$$

对偶问题

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) - \sum_{i=1}^m \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, m. \end{aligned}$$

多了一个额外的约束

- 基于规则的分类
- 基于监督学习的分类
 - 决策树
 - 最近邻分类
 - 支持向量机
 - 不平衡分类问题

- 不平衡分类问题

- 具有不平衡分布的数据集在日常应用中极为常见
 - 例如，考试中及格的学生一般是绝大多数（神捕的课除外）
 - 如果依赖于效果指标优化分类结果，在不平衡分类条件下可能被误导

如何以最低的代价做一个Accuracy接近 100% 的搜索引擎？

百毒

0 个返回结果

- 解决方案 (1) 代价敏感学习

- 引入代价矩阵，衡量将一个类错分到另一个类的代价
 - 例如，宁可多判断一些疑似患者，不能漏掉一个病人的情况下，可将“病人”错分至“健康”的代价设为100，而“健康”错判的代价为1
 - 最终优化目标由原先的准确/召回变更为加权后的代价

| | | True Class j | | | |
|------------------------|----------|-------------------|----------|-----|----------|
| | | 1 | 2 | ... | k |
| Predicted Class i | 1 | $C(1,1)$ | $C(1,2)$ | ... | $C(1,k)$ |
| | 2 | $C(2,1)$ | ... | ... | \vdots |
| | \vdots | \vdots | ... | ... | \vdots |
| | \vdots | \vdots | ... | ... | \vdots |
| | k | $C(k,1)$ | ... | ... | $C(k,k)$ |

- **解决方案（2） 抽样方法**

- 另一种方法是基于采样的方法，通过改变样本分布来缓解不平衡问题
- 常见的策略有以下三种：
 - 过采样：提升少数类的样本比例，例如从少数类中进行重复随机采样
 - 欠采样：降低多数类的样本比例，例如用多数类的采样样本而非全样本
 - 混合采样：同时采用前两种策略
 - 也可以利用已有的少数类样本，通过K-最近邻生成新样本
 - 采样中要注意噪声问题，噪声也可能被复制多次

本章小结

分类

- 基于规则的分类方法
- 基于监督学习的分类方法
 - 决策树
 - 最近邻分类
 - 支持向量机
 - 不平衡分类问题