

(1) 参考考察范围

	A	B	C	D
2	机器学习概述	1 机器学习概述	很多可拓展方面, 罗老师PPT!	
3	监督学习			
4		1.1 贝叶斯决策		
5		1.2 线性模型	两个班都有涉及, 重要!	
6		1.3 决策树		
7		2.4 集成学习		
8		2.5 支持向量机		
9	神经网络与深度学习			
10		3.1 前馈神经网络		
11		3.2 卷积神经网络	不在考试范围	
12		3.3 循环神经网络	不在考试范围	
13		3.4 深度学习进阶技术	注意力、LLM不在考试范围	
14	无监督学习			
15		4.1 聚类		
16		4.2 降维	流形学习不在考试范围	
17		4.3 无监督与自监督神经网络	不在考试范围	
18	机器学习的责任与未来			
19		5.1 机器学习伦理	(看罗老师课件这一遍)	
20		5.2 机器学习未来展望	不在考试范围	

写在开头:

机器学习课程考核本身并不是很难, 重点在于理解与记忆, 本笔记结合了课程ppt和一些个人理解进行汇总, 覆盖了全面的考试范围, 但是还是建议有时间一定要把老师将的ppt重点查看

Ps:

AI班: 提前一年学习本课程, 可以问大三软工的学长学姐要一下对应的ppt, 综合汇总下(文章中的以AI班ppt为主+汇合了罗老师班的ppt)

Introduction

一些拓展性问题

1.1 机器学习

补充: 数据挖掘

无国定定义

① 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

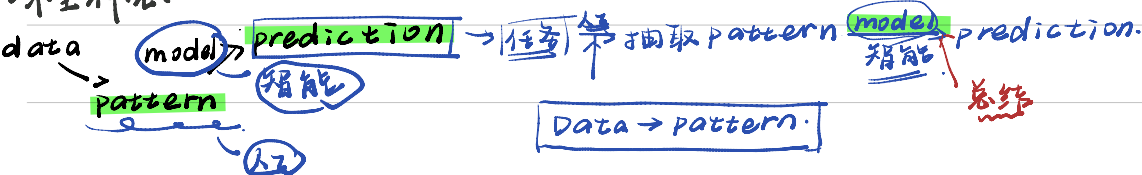
经验在计算机内通常被表达为数据。
从数据中产生模型的算法, 即学习算法

数据挖掘是通过对①(大规模)②观测数据集的分析, 寻找③确信的关系, 并将数据以一种可理解的且利于使用的④新颖方式概括数据的方法。⑤⑥

② 主要任务: 从数据中学习知识并进行预测

③ 三要素DMP: D(Training Data), M(model), P(Prediction)

课堂补充:



1.2 数据

特征: 做出某个判断的证据, 标记是结论

数据 = 特征 + (未必一定存在的) 标记

1.3 监督学习 vs 无监督学习

根据训练数据是否拥有标记信息, 学习任务可大致分为两类: 监督学习 和 无监督学习

训练数据含有标记与学习范式

训练数据不含标记与学习范式

① In the ML literature, a supervised learning problem has the following characteristics:

- We are primarily interested in prediction. 主要研究兴趣在于预测
- We are interested in predicting only one thing 仅对单一预测结果有兴趣
- The possible values of what we want to predict are specified, and we have some training cases for which its value is known.

指定需要预测的可能取值范围, 并已获得若干已知其取值的训练样本

② 训练集 vs 测试集 vs 验证集

Supervised Learning

classification 分类

MIMA



training data
训练数据/训练集
训练样本
有特征, 有标签



test data 测试数据/测试集

测试样本，有特征，无标签
预测标签

训练集 (Training Set) 用于训练模型的数据集，模型通过学习这些数据来调整参数。
测试集 (Test Set) 用于评估模型性能的数据集，模型在训练完成后用测试集进行最终验证，确保泛化能力。
验证集 (Validation Set) 是用于模型调优和评估的数据子集，介于训练集和测试集之间。它的核心作用是帮助开发者调整超参数、选择模型结构或提前终止训练，而不会直接接触测试集（保证最终评估的客观性）
(用于调参和模型选择)

③ 分类 vs 回归

分类 (classification) 问题，我们想要预测一个项目的类别。

- 例如，疾病的类型、文档的主题、图像中包含的语义、客户是否会购买产品。

回归 (regression) 问题，我们想要预测一个数值。

- 例如，客户花费的金额、患者的血压等。

主要方法



无监督学习

不专注于对任何特定事物的预测，而是试图发现数据中的有趣特征

Clustering: 聚类

半监督学习

定义：训练所需标注数据与未标注数据的数据量而言处于中间状态

目的：减少与监督学习相比所需监督量，同时将无监督聚类的结果提升至用户预期水平。

① 约束聚类算法 Constrained Clustering

1. 第一部分：半监督聚类的“辅助信息来源”

这段说的是半监督聚类（介于无监督聚类和有监督分类之间的方法）可以利用的额外信息，包括：

- “数据子集类别标签”：比如只有部分数据标了类（不是全标，全标就是分类了）；
- “关于聚类的领域知识”：比如你做用户聚类时，已知“用户会按消费能力分群”这种行业经验；
- “对象间的相似性信息”：比如提前知道“两个用户浏览记录重合度高则更相似”；
- “用户偏好”：比如你希望聚类结果更偏向某种划分逻辑（比如优先按地域分）。

2. 第二部分：辅助信息的“常见形式”

这些信息常以“成对约束”的形式存在：

- **Must-link (必须链接)**：指定“这两个对象必须分到同一簇”；
- **Cannot-link (不能链接)**：指定“这两个对象必须分到不同簇”；
- 而“带标签的子集”也能转成成对约束（如同标签的对象→Must-link，不同标签→Cannot-link）。

3. 第三部分：半监督聚类的核心逻辑 + 趋势

- 聚类过程：找满足这些约束的划分方式（比如 K-means 的改进版会把约束加入损失函数）；
- 近年趋势：更常用“基于相似性”的方法（比如先根据约束调整对象间的相似性矩阵，再用普通聚类算法）。

② 主动学习 Active Learning

• 传统的监督学习算法是被动接收训练数据的。

• 与之不同，主动学习会从无标签数据中，主动请求标注那些包含关键信

→ 采用不确定性采样

思维：对于分类存在不确定性自

息的样本。

图像.其信息可能具有多个价值。

- 理论结果表明：主动学习能大幅减少所需的训练数据规模！

- What is the notion of uncertainty?
 - Idea: Train a classifier like SVM on the training set.
 - For each unlabeled image, output probabilities indicating class membership.
 - Estimate probabilities can be used to infer uncertainty.
 - A one-vs-one SVM approach can be used to tackle multiple classes.

(4) 强化学习

基本概念

- 智能体(agent)**:智能体是强化学习算法的主体,它能够根据经验做出主观判断并执行动作,是整个智能系统的核心。
- 环境(environment)**:智能体以外的一切统称为环境,环境在与智能体的交互中能被智能体采取的动作影响,同时环境也能向智能体反馈状态和奖励。虽说智能体以外的一切都可视为环境,但在设计算法时常常会排除不相关的因素,建立一个理想的环境模型来对算法功能进行模拟。
- 状态(state)**:状态可以理解为智能体对环境的一种理解和编码,通常包含对智能体所采取决策产生影响的信息。
- 动作(action)**:动作是智能体对环境产生影响的方式,这里说的动作常常指概念上的动作,在设计机器人时还须考虑动作的执行机构。
- 策略(policy)**:是智能体在所处状态下执行某个动作的依据,即给定一个状态,智能体可根据一个策略来选择应该采取的动作。
- 奖励(reward)**:奖励是智能体序贯式采取一系列动作后从环境获得的收益。注意,奖励概念是现实中奖励和惩罚的统合,一般用正值代表实际奖励用负值代表实际惩罚。

强化学习

MIMA



- 强化学习实际上是智能体在与环境交互中学习能帮助其获得最大化奖励这一策略的过程。
 - 学习目标:选择能够获取最大收益的状态到动作的映射
- 在每一次迭代中,智能体根据当前策略选择一个动作,该动作影响环境,导致环境发生改变智能体此时从环境得到状态变化和奖励反馈等信息,并根据这些反馈更新其内部策略。这一过程与人类从与环境交互中学习这一方式具有相似之处。
- 强化学习求解的是探索性和交互性的问题。

(5) 迁移学习

迁移学习

MIMA

- 迁移学习 (Transfer Learning) 其核心思想是将从 **源域 (Source Domain)** “任务中学习到的知识，迁移到与源领域相关但不完全相同的 **目标域 (Target Domain)**” 任务中，以解决目标领域数据稀缺、标注成本高或训练难度大的问题。
- 传统机器学习的核心假设是：源域与目标域的数据分布、任务完全一致。
- 但实际研究与应用中，这一假设常不成立，主要面临三大挑战：
 - **数据稀缺问题**：目标领域标注数据不足（如医疗影像诊断中，罕见病标注样本极少），直接训练模型易过拟合；
 - **分布偏移问题**：源领域与目标领域数据分布差异大（如用“白天交通场景”数据训练的目标检测模型，应用于“雨夜场景”时性能骤降）；
 - **任务适配问题**：源任务与目标任务不完全一致（如用“图像分类”任务学到的特征，需适配“图像分割”任务）。

迁移学习

MIMA

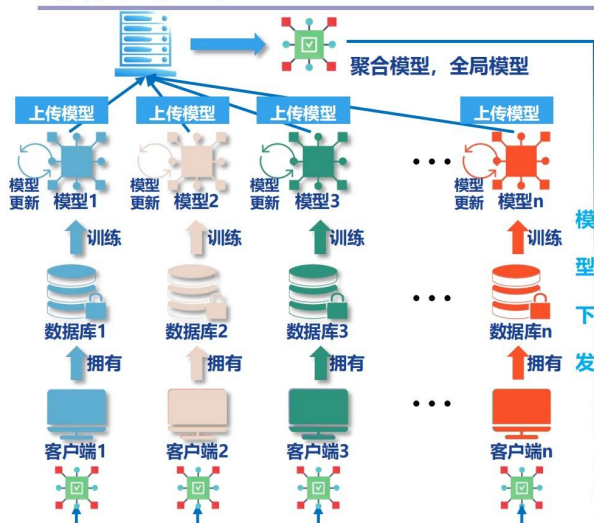
- **特征迁移学习 (Feature-Based Transfer Learning)**
 - 通过学习领域不变特征，将源领域与目标领域的的数据映射到统一的特征空间，使两领域在该空间中的分布差异最小化，从而实现知识迁移
- **参数迁移学习 (Parameter-Based Transfer Learning)**
 - 假设源领域任务与目标领域任务共享部分模型参数（或参数分布），将源领域模型训练得到的参数作为目标领域模型的初始参数，或通过正则化约束目标领域参数与源领域参数的差异，实现知识复用
- **实例迁移学习 (Instance-Based Transfer Learning)**
 - 从源领域中筛选出与目标领域“相似”的样本（称为迁移样本），对这些样本赋予较高的权重，与目标领域样本（若有）联合训练模型，实现知识迁移

预训练与微调
Pre-training & Fine-tuning

16) 联邦学习

联邦学习

MIMA



- 联邦机器学习 (Federated machine learning/Federated Learning) 又名联邦学习
- 联邦学习是一种在保障数据隐私安全、符合法律法规的前提下，让多个持有本地数据的参与方（如机构、设备）无需上传原始数据，通过协同训练共享模型知识，最终构建出泛化能力强的全局模型的分布式机器学习技术。
- 一些术语：全局模型、本地数据、局部模型、联邦聚合算法、全局聚合

联邦学习

MIMA

- **横向联邦学习**
 - 适用场景：参与者的数据**特征重叠较多**，而**样本 ID 重叠较少**。如不同地区银行，业务相似，用户特征相同，但用户群体交集小。
 - 学习过程：参与者本地计算训练梯度，加密更新后发送到服务器；服务器聚合梯度更新模型参数，再将汇总结果模型发回给各参与者，参与者解密梯度更新模型。

- **纵向联邦学习**
 - 适用场景：参与者训练**样本 ID 重叠较多**，而数据**特征重叠较少**。如同一地区银行和电商，用户群体交集大，但用户特征交集小。
 - 学习过程：第三方 C 加密样本对齐；合作者 C 创建加密对，将公钥发送给 A 和 B；A 和 B 分别计算和自己相关的特征中间结果，并加密交互，用来求得各自梯度和损失；A 和 B 分别计算各自加密结果，C 汇总计算总梯度值并解密，回传给 A 和 B，A 和 B 更新模型参数。
- **联邦迁移学习**
 - 适用场景：**数据和特征均不重叠**，通过迁移学习关联知识。如跨语言文本分类。
 - 学习过程：在不同数据集和特征空间下，通过迁移学习技术，挖掘不同领域数据间的潜在联系，进行模型训练。

	集中式学习 (CL)	联邦学习 (FL)
数据处理模式	数据动，模型不动：各参与方将原始数据上传至中心服务器，服务器用全量数据训练模型，再下发模型至参与方。	数据不动，模型动：原始数据始终留在参与方本地，仅将“模型参数更新值”（如梯度）上传至中心，聚合后生成新模型下发。
隐私安全性	隐私安全性极低：原始数据跨机构 / 设备传输，存在泄露、篡改风险（如医疗数据上传时被窃取）。	隐私安全性极高：原始数据不离开本地，参数传输需加密（如差分隐私、同态加密），从源头规避隐私泄露，天然符合数据合规要求。
协作形态	单向被动协作：参与方仅负责提供数据，无模型训练决策权，所有核心操作（模型初始化、训练、聚合）由中心服务器主导。	双向主动协作：参与方不仅提供数据，还需执行本地训练，中心服务器仅负责“规则制定”与“参数聚合”，部分场景（去中心化联邦）可无中心节点。
对数据的约束	要求数据“同构化”：需统一原始数据的格式、特征维度、标签类型（如所有医院需按同一标准整理病历），否则无法集中训练。	支持数据“异构化”：允许参与方数据格式、分布存在差异（如医院 A 以早期癌症样本为主，医院 B 以晚期为主），通过聚合算法（如 FedProx）适配差异。
通信与计算成本	通信成本高（需传输海量原始数据），计算成本集中于中心服务器（需处理全量数据）。	通信成本低（仅传输轻量化参数更新），计算成本分布式分摊（参与方各自承担本地训练），减轻中心压力。

二、模型评估与选择

1. 模型选择.

- ① 从一组候选模型中选择统计模型
- ② 同一个机器学习模型下选择不同超参

2.

一些术语:

1. error(误差): 学习器的实际预测输出与样本的真实输出之间的差异
 - training error (训练误差): 学习器在训练集上的误差
 - generalization error (泛化误差): 在新样本上的误差

2. error rate (错误率): 分类错误的样本数占样本总数的比例

3. accuracy (精度): 1-错误率

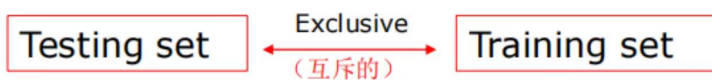
目标: 希望得到泛化误差小的学习器

4. 过拟合 (overfitting): 当学习器把训练样本学得太好了的时候, 很可能已经把训练样本自身的一些特点当作所有潜在样本都会具有的一般性质, 导致泛化性能下降
5. 欠拟合 (underfitting): 对训练样本的一般性质尚未学好

3. 模型评估.

1. In practice, we consider the **generalization ability, time consuming, storage consuming, interpretability** of a model and finally make a decision.

通常实验测试对学习器的泛化误差进行评估并进而做出选择, 为此我们需要使用一个“测试集”来测试学习器对新样本的判别能力, 以测试集的测试误差作为泛化误差的近似



2. 注意事项: 测试集尽可能与训练集互斥, 即测试样本尽量不在训练集中出现、未在训练过程中使用过

4. 划分数据集的方法

(1) Hold-out 留出法

✨ Hold-out (留出法)

基本思想: 直接将数据集D划分成两个互斥的集合, 其中一个集合作为训练集S, 另一个作为测试集T在S上训练出模型, 用T来评估测试误差, 作为泛化误差的估计

注意事项:

- 训练/测试集的划分 尽量保持数据分布的一致性

eg: 例如在分类任务中至少要保持样本的类别比例相似。如果从采样 (sampling) 的角度来看待数据集的划分过程, 则保留类别比例的采样方式通常称为“分层采样”(stratified sampling)。例如通过对D进行分层采样而获得含70%样本的训练集S和含30%样本的测试集T, 若D包含500个正例、500个反例, 则分层采样得到的S应包含350个正例、350个反例, 而T则包含150个正例和150个反例; 若S、T中样本类别比例差别很大, 则误差估计将由于训练/测试数据分布的差异而产生偏差。

- 单次使用留出法得到的估计结果往往不够稳定可靠, 在使用留出法时, 一般要采用若干次随机划分、重复进行实验评估后取平均值作为留出法的评估结果
- Training samples/Testing samples=2/1 ~4/1.

(2) Cross validation 交叉验证.

✨ Cross validation (交叉验证法)

K折交叉检验/k折交叉验证

基本思想: 先将数据集D划分为k个大小相似的互斥子集

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

每个子集都尽可能保持数据分布的一致性, 即从D中通过分层采样得到。然后, 每次用k-1个子集的并集作为训练集, 余下的那个子集作为测试集。这样就可获得k组训练/测试集, 从而可进行k次训练和测试, 最终返回的是这k个测试结果的均值。

(4) 补充(不在PPT中)

02 Model Evaluation and Selection

Bootstrapping (自助法)

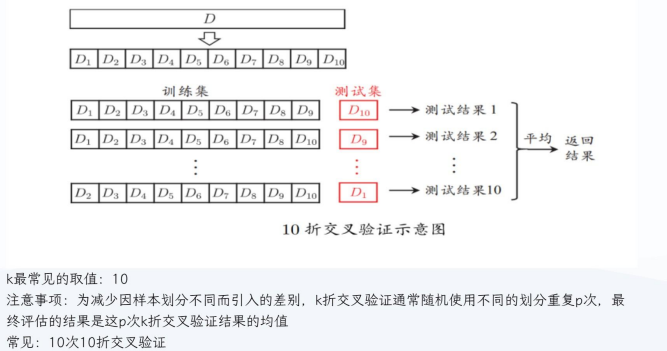
可重复采样/有放回的采样

前提:

希望评估的是用D训练出的模型, 但在留出法和交叉验证法中, 由于保留了一部分样本用于测试, 因此实际评估的模型所使用的训练集比D小——引入一些因训练样本规模不同而导致的估计偏差

留一法虽然受训练样本规模变化的影响较小, 但计算的复杂度又太高了

我们希望评估的是用D训练出的模型。但在留出法和交叉验证法中, 由于



保留了部分样本用于测试，因此实际估计的模型所使用的训练集比 D' 小，这必然会引入一些因训练样本规模不同而导致的估计偏差。留一法受训练样本规模变化的影响较小，但计算复杂度又太高了。有没有什么办法可以减少训练样本规模不同造成的影响，同时还能比较高效地进行实验估计呢？

“自助法” (bootstrapping) 是一个比较好的解决方案，它直接以自助采样法 (bootstrap sampling) 为基础 [Efron and Tibshirani, 1993]。给定包含 m 个样本的数据集 D ，我们对它进行采样产生数据集 D' ：每次随机从 D 中挑选一个样本，将其拷贝放入 D' ，然后再将该样本放回初始数据集 D 中，使得该样本在下次采样时仍有可能被采到；这个过程重复执行 m 次后，我们就得到了包含 m 个样本的数据集 D' ，这就是自助采样的结果。显然， D 中有一部分样本会在 D' 中多次出现，而另一部分样本不出现。可以做一个简单的估计，样本在 m 次采样中始终不被采到的概率是 $(1 - \frac{1}{m})^m$ ，取极限得到

$$\lim_{m \rightarrow \infty} \left(1 - \frac{1}{m}\right)^m \mapsto \frac{1}{e} \approx 0.368, \quad (2.1)$$

即通过自助采样，初始数据集 D 中约有 36.8% 的样本未出现在采样数据集 D' 中。于是我们可将 D' 用作训练集， $D \setminus D'$ 用作测试集；这样，实际评估的模型与期望评估的模型都使用 m 个训练样本，而我们仍有数据总量约 1/3 的、没在训练集中出现的样本用于测试。这样的测试结果，亦称“包外估计” (out-of-bag estimate)。

1.3 Leave-One-Out (留一法)

Leave-One-Out (留一法)

交叉验证的一个特例：包含 m 个样本，令 $k=m$

- 不受随机样本划分方式的影响

m 个样本只有唯一的方式划分 m 个子集——每个子集包含一个样本

- 评估结果往往比较精确

留一法使用的训练集与初始的数据集相比只少了一个样本，使得在绝大多数情况下，留一法中被实际评估的模型与期望评估的用 D 训练出的模型很相似

- 在数据集比较大的情况下，训练 m 个模型的计算开销难以忍受的

5. 性能度量 → 衡量模型泛化能力的评价标准

(1) 回归任务

① 均方误差

$$E(f; D) = \frac{1}{n} \sum_{i=1}^n (f(x_i) - y_i)^2$$

预测值 (上方)
真实值 (下方)

② 分类任务

1° error rate / accuracy

2° precision 查准率

recall 查全率

$$F_1 = \frac{2 \times P \times R}{P + R}$$

定义

Precision (查准率): The positive predictive value. — 查到的东西是否准
Recall (查全率): The true positive rate. — 该要结果是否找到

→ P-R 曲线

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：预测的正例多少是对的

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

查全率：真正的正例找到了多少

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

混淆矩阵的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。

查准率 (precision)：所有预测为阳性的样本中真正为阳性的比例 $p = TP / (TP + FP)$

查全率 (recall)：所有真正为阳性的样本中预测为阳性的比例 $r = TP / (TP + FN)$

ROC 曲线 横坐标：FPR (False Positive Rate) 假阳率 $FPR = FP / (TN + FP)$

纵坐标：TPR 真阳率 $TPR = TP / (TP + FN)$

AUC: AUC (Area Under Curve) 被定义为ROC曲线下的面积, 显然这个面积的数值不会大于1

当横坐标越小, 纵坐标数值越大时, 分类器效果好, 因此AUC越大效果越好

3° 代价敏感错误率

为权衡不同类型的错误所造成的不同损失. 可为错误赋予非均等代价

表 2.2 二分类代价矩阵

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

$cost_{ij}$ 表示将第*i*类样本预测为第*j*类样本的代价

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

Total cost

4° 偏差与方差

bias(偏差): error from erroneous assumptions in the learning algorithm

期望输出与真实标记的差别

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

其中学习算法的期望预测为:

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

偏差度量了学习算法的期望预测与真实结果的偏离程度——刻画了学习算法本身的拟合能力

variance(方差): error from sensitivity to small fluctuations in the training set

方差度量了同样大小的训练集的变动所导致的学习性能的变化——刻画了数据扰动所造成的影响

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

噪声: 表达在当前任务上任何学习算法所能达到的期望泛化误差的下界——刻画学习问题本身的难度

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

泛化误差

三. 贝叶斯分类器

(1) 贝叶斯公式推导

$$P(A|B) = P(A|B) P(B)$$

$$P(A|B) = P(B|A) P(A)$$

$$P(A|B) P(B) = P(B|A) P(A) \rightarrow \text{先验概率}$$

$$\boxed{P(A|B)} = \frac{P(B|A) P(A)}{P(B)} \rightarrow \text{后验概率}$$

$$P(c_i|x) = \frac{P(x|c_i) P(c_i)}{P(x)} \rightarrow \text{类条件概率密度/似然}$$

$$= \frac{P(x|c_i) P(c_i)}{\sum_{i=1}^N P(x|c_i) P(c_i)}$$

(2) 最小错误率贝叶斯决策

$$P(\text{error}|x) = \min [P(w_1|x), P(w_2|x)]$$

若 $P(w_1|x) > P(w_2|x)$ x 被分类为 w_1
 若 $P(w_1|x) < P(w_2|x)$ x 被分类为 w_2

$$\arg \max_{w_i} P(w_i|x)$$

二分类问题

选择后验概率最大的类别恰好能让每个样本的分类错误率达到最小

特例

最小风险贝叶斯决策

$y = \{c_1, c_2, \dots, c_n\}$ 有限的 N 个状态的标签集合

λ_{ij} : 将 c_i 误判为 c_j 带来的损失

$$R(c_i|x) = \sum_{j=1}^N \lambda_{ij} P(c_j|x)$$

$$\arg \min_{c \in y} R(c|x)$$

(3) 朴素贝叶斯

假设属性间相互独立的

$$P(c_i|x) = \frac{P(x|c_i) P(c_i)}{P(x)} = \frac{P(c_i)}{P(x)} \prod_{i=1}^d P(x_i|c_i)$$

Bayes optimal classifier

$$h^*(x) = \arg \max_{c \in y} P(c|x)$$

$$P(c_i|x) = \frac{P(c_i)}{P(x)} \prod_{i=1}^d P(x_i|c_i)$$

NB optimal classifier

$$h_{nb}(x) = \arg \max_{c \in y} P(c_i) \prod_{i=1}^d P(x_i|c_i)$$

$$P(c_i) = \frac{|D_{c_i}|}{|D|}$$

假设 $P(x_i|c_i) \sim N(\mu_{c_i}, \sigma_{c_i}^2)$ 类别 c_i 下第 i 个特征的值分布
 $P(x_i|c_i) = \frac{1}{\sqrt{2\pi} \sigma_{c_i}} e^{-\frac{(x_i - \mu_{c_i})^2}{2\sigma_{c_i}^2}}$

连续

离散 $P(x_i|c_i) = \frac{|D_{c_i, x_i}|}{|D_{c_i}|}$

(4) 拉普拉斯修正

零概率问题:

在计算事件的概率时，如果某个事件在观察样本库（训练集）中没有出现过，会导致该事件的概率结果是0。

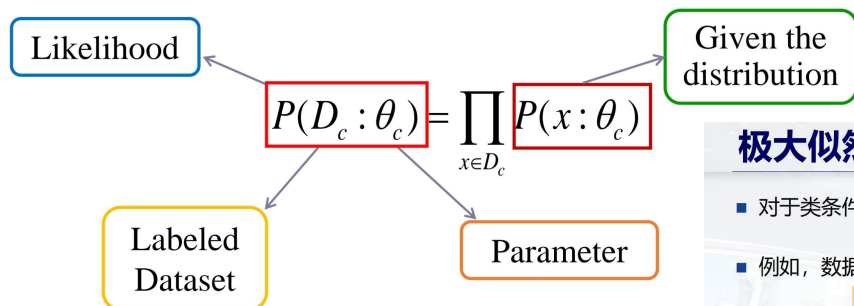
这是不合理的，不能因为一个事件没有观察到，就被认为该事件一定不可能发生（即该事件的概率为0）。

拉普拉斯平滑(Laplacian smoothing)就是为了解决零概率的问题。

理论假设：假定训练样本很大时，每个分量x的计数加1造成的估计概率变化可以忽略不计，但可以方便有效的避免零概率问题

$$P(w_i | y = k) = \frac{\text{在训练集某类样本中，同时又是 } w_i \text{ 某个特征值得样本量} + \alpha}{\text{训练集中某类样本量} + k\alpha}$$

四. MLE 最大似然估计



选择使观察数据概率最大的 θ_c

Log-likelihood (对数似然)

$$P(D_c | \theta_c) = \prod_{x \in D_c} P(x | \theta_c)$$

$$LL(\theta_c) = \log P(D_c | \theta_c) = \sum_{x \in D_c} \log P(x | \theta_c)$$

对数似然函数

$$x_i \sim N(\mu, \sigma^2)$$

$$lik(\mu, \sigma^2) = \prod_{i=1}^n \left(\frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}} \right)$$

$$L(\mu, \sigma^2) = \sum_{i=1}^n \ln \left(\frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}} \right)$$

$$\begin{aligned} &= -\sum_{i=1}^n \ln(\sigma) - \sum_{i=1}^n \ln(\sqrt{2\pi}) - \sum_{i=1}^n \frac{(x_i - \mu)^2}{2\sigma^2} \\ &= -n \ln(\sigma) - n \ln(\sqrt{2\pi}) - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_i - \mu)^2 \end{aligned}$$

$$\frac{\partial L(\mu, \sigma^2)}{\partial \mu} = \frac{1}{\sigma^2} \cdot \sum_{i=1}^n (x_i - \mu)$$

$$\frac{\partial L(\mu, \sigma^2)}{\partial \sigma^2} = -\frac{1}{2} n \cdot \frac{1}{\sigma^2} + \frac{1}{2\sigma^4} \sum_{i=1}^n (x_i - \mu)^2$$

$$\text{令 } \frac{\partial L(\mu, \sigma^2)}{\partial \mu} = 0 \quad \hat{\mu} = \frac{1}{n} \sum_{i=1}^n x_i$$

$$\frac{\partial L(\mu, \sigma^2)}{\partial \sigma^2} = 0 \quad \hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

$$-\frac{1}{2} \cdot n \cdot \frac{1}{\sigma^2} + \frac{1}{2\sigma^4} \sum_{i=1}^n (x_i - \mu)^2 = 0$$

$$\frac{1}{\sigma^2} \sum_{i=1}^n (x_i - \mu)^2 = n$$

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

极大似然估计

■ 对于类条件概率密度，虽然其未知，但我们可能知晓其有一定的参数形式

■ 例如，数据可能满足某种正态分布，但分布的具体参数未知：

$$p(\mathbf{x} | \omega_j) \sim N(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$$

$\theta_j \rightarrow \boldsymbol{\theta}_j = (\theta_1, \theta_2, \dots, \theta_m)^T$

■ 如果可以得到参数的值，即均值和方差（协方差）的值，就可以得到对应的类条件概率密度。

■ 通过观测到的训练样本，推断这些未知参数的最优可能取值的过程叫作参数估计 Parameter Estimation。

■ 概率与似然



概率

描述了参数已知时的随机变量的输出结果



似然

用来描述已知随机变量输出结果时，未知参数的可能取值

五. 参数估计

通过观测到的训练样本，推断这些未知参数的最优可能取值的过程叫作参数估计 Parameter Estimation.

1. 有监督参数估计

• 有监督参数估计:

样本所属的类别与条件以“整体概率密度函数的形式”已知，但刻画该概率密度函数的部分参数是未知的。

• 举例:

仅已知样本的整体分布（例如已知是正态分布），但正态分布的参数是未知的。我们的目标是从已知类别的样本集中，对总体分布的部分参数进行统计推断，这种估计方式就称为有监督下的参数估计。

• 无监督参数估计

• 无监督参数估计:

整体概率密度函数是已知的，但样本所属的类别是未知的，需要确定该概率密度函数的部分参数。

• 补充说明:

“有监督”与“无监督”的核心区别在于“样本所属类别是否已知”。无监督参数估计常用的两种方法是极大似然估计和贝叶斯估计。

• 极大似然估计:

将参数视为确定但未知的量，通过“让实际观测样本出现的概率最大化”这一条件，得到参数的最优估计值。

• 贝叶斯估计:

将未知参数视为具有某种分布的随机变量，利用样本的观测结果将“参数的先验分布”转化为“后验分布”，再基于后验分布修正对参数的原始估计。

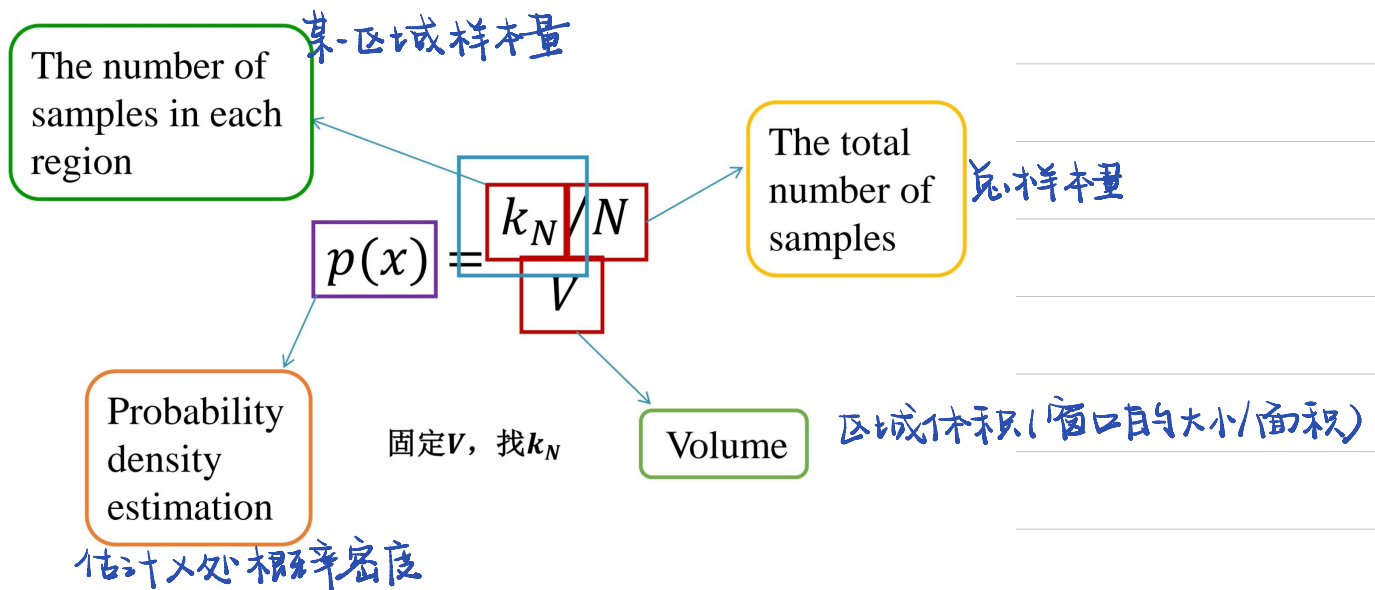
• 非参数估计

• 非参数估计:

样本所属的类别是已知的，但概率密度函数的形式是未知的，因此需要直接推断概率密度函数本身。

方法 1. 近邻 Parzen 窗

2. Parzen 窗



我们可以认为每个样本对于空间上的每个点的概率密度都有一定的贡献，但是随着空间点距离样本点的距离增大，这个贡献率会减小。一个样本点对其所在位置的概率密度贡献最大，随着距离样本点的距离会依次减小，这个贡献率分布函数就是由窗函数定量给出的。将每个样本点对所有空间点的概率密度贡献累加，就是空间上的概率密度分布函数，这便是Parzen窗的算法原理。

04 Decision Tree

定义

一般的，一个决策树包含一个根结点，若干个内部节点和若干个叶结点

- 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
 - 每个测试的结果或是导出最终结论或是导出进一步的判定问题
- 决策树通过从树的根结点开始，沿分支逐层移动直至到达叶结点，对实例或样本进行分类

每个集合包含的样本集合根据属性测试的结果被划分到子节点中，根节点包含样本全集

- 从根结点到每个叶结点的路径对应一个判定测试序列
- 决策过程的最终结论对应于一个目标值

How to Construct a Decision Tree

显然，决策树的生成是一个递归过程

解决的三个问题

- (1) Which **attribute** to start? (root)
- (2) Which **attribute** to proceed?
- (3) When to stop and obtain the target value?

One key problem of decision tree algorithm: **attribute selection**

决策树算法的基本思想

- 选择最佳属性对剩余实例进行分类，并将该属性设为节点
- 对后续节点递归重复此过程
- 停止当：
 - 当前结点包含的样本全属于同一类别，无需划分
 - 当前属性集为空，或是所有样本在现存所有属性上取值相同，无法划分
 - 把当前结点标记为叶结点，将其类别设定为该结点所含样本最多的类别
 - 当前结点包含的样本集合为空，不能划分
 - 把当前结点标记为叶结点，将其类别设定为其父结点所含样本最多的类别

属性选择

决策树学习的关键是如何选择最优划分属性

随着划分的不断进行，我们希望决策树的分支节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

信息增益(Information gain)—ID3

ID3 selects attributes according to their information gain

ID3决策树学习算法以信息增益为准则来选择划分属性

信息熵 (information entropy)

定义：信息熵是度量样本集合纯度最常用的一种指标

假定当前样本集合 D 中第 k 类样本所占的比例为 p_k ($k = 1, 2, \dots, |\mathcal{Y}|$), 则 D 的信息熵定义为

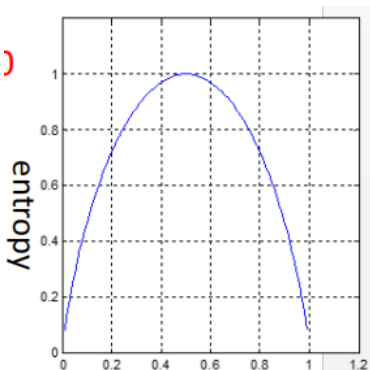
$$\text{Ent}(D) = - \sum_{k=1}^{|\mathcal{Y}|} p_k \log_2 p_k . \quad (4.1)$$

$\text{Ent}(D)$ 的值越小，则 D 的纯度越高

信息熵在二分类问题中的计算公式：

$$1 \quad \text{Entropy} = -p_1 \log(p_1) - (1-p_1) \log(1-p_1)$$

- 如果所有的样本属于同一个类别，信息熵是0 (highest purity)
- 如果样本均匀混合，则信息熵为1 (最混乱) (lowest purity)
- 纯度越高，熵值越小；纯度越低，熵值越大。



信息增益

属性的信息增益是指基于该属性进行划分时预期产生的熵减少量

假定离散属性 a 有 V 个可能的取值 $\{a^1, a^2, \dots, a^V\}$, 若使用 a 来对样本集 D 进行划分, 则会产生 V 个分支结点, 其中第 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 . 我们可根据式(4.1) 计算出 D^v 的信息熵, 再考虑到不同的分支结点所包含的样本数不同, 给分支结点赋予权重 $|D^v|/|D|$, 即样本数越多的分支结点的影响越大, 于是可计算出用属性 a 对样本集 D 进行划分所获得的“信息增益” (information gain)

$$\text{Gain}(D, a) = \text{Ent}(D) - \sum_{v=1}^V \frac{|D^v|}{|D|} \text{Ent}(D^v). \quad (4.2)$$

样本集合
权重
基于 a 对样本集进行划分

一般而言, 信息增益越大, 则意味使用属性 a 来进行划分所获得纯度提升越大
举例理解:

Training set							
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	稍粗	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	稍粗	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	稍粗	平坦	软粘	否
13	青绿	稍蜷	浊响	稍粗	凹陷	硬滑	否
14	浅白	稍蜷	沉闷	稍粗	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	稍粗	稍凹	硬滑	否

$\text{Ent}(D) = -\sum_{k=1}^2 P_k \log_2 P_k$
 $= -(\frac{14}{17} \log_2 \frac{14}{17} + \frac{3}{17} \log_2 \frac{3}{17})$
 $= 0.998$

以色泽为例:

$D^1(\text{色泽}=\text{青绿}) = \{1, 4, 6, 10, 12, 17\}$
 $D^2(\text{色泽}=\text{乌黑}) = \{2, 3, 7, 8, 9, 15\}$
 $D^3(\text{色泽}=\text{浅白}) = \{5, 11, 13, 14, 16\}$

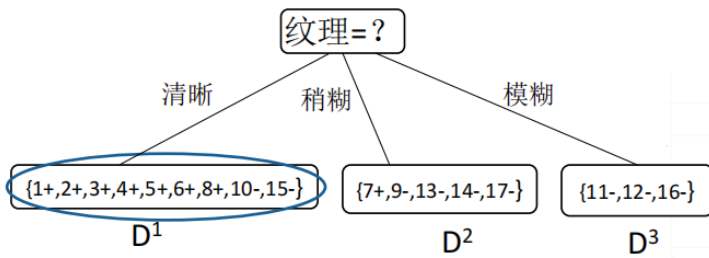
$\text{Ent}(D^1) = -(\frac{6}{14} \log_2 \frac{6}{14} + \frac{8}{14} \log_2 \frac{8}{14}) = 1$
 $\text{Ent}(D^2) = -(\frac{6}{17} \log_2 \frac{6}{17} + \frac{7}{17} \log_2 \frac{7}{17}) = 0.918$
 $\text{Ent}(D^3) = -(\frac{5}{17} \log_2 \frac{5}{17} + \frac{4}{17} \log_2 \frac{4}{17}) = 0.722$

$\frac{3}{17} \text{Ent}(D^3) = \frac{5}{17} \times 1 + \frac{6}{17} \times 0.918 + \frac{5}{17} \times 0.722$
 $= 0.889$
 $\text{Gain}(D, \text{色泽}) = \text{Ent}(D) - \frac{3}{17} \text{Ent}(D^3)$
 $= 0.998 - 0.889 = 0.109$

Similarly:

$\text{Gain}(D, \text{根蒂}) = 0.143$; $\text{Gain}(D, \text{敲声}) = 0.141$;
 $\text{Gain}(D, \text{纹理}) = 0.381$; $\text{Gain}(D, \text{脐部}) = 0.289$;
 $\text{Gain}(D, \text{触感}) = 0.006$

显然属性纹理的信息增益最大, 于是被选定为划分属性, 然后决策树学习算法将对每个分支结点做进一步的划分, 此时划分的样例集合为各分支结点包含的样例集合, 不是training set

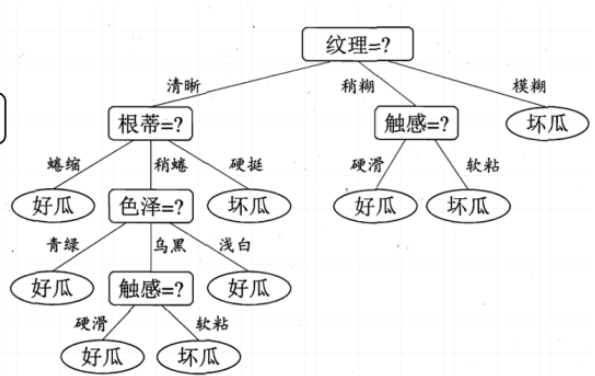


$D^1 = \{1+, 2+, 3+, 4+, 5+, 6+, 8+, 10-, 15-\}$

$\text{Gain}(D^1, \text{色泽}) = 0.043$; $\text{Gain}(D^1, \text{根蒂}) = 0.458$;

$\text{Gain}(D^1, \text{敲声}) = 0.331$; $\text{Gain}(D^1, \text{脐部}) = 0.458$;

$\text{Gain}(D^1, \text{触感}) = 0.458$



AI识图

可见3个属性均取得最大的信息增益，可以任选其中之一作为划分属性即可，类似的，对每个分支结点进行上述操作

局限性

如果我们把编号作为西瓜的一个属性，它将会被选择为最优属性

远大于其他候选划分属性。这很容易理解：“编号”将产生 17 个分支，每个分支结点仅包含一个样本，这些分支结点的纯度已达最大。然而，这样的决策树显然不具有泛化能力，无法对新样本进行有效预测。

局限性：信息增益准则对可取值数目较多的属性有所偏好

增益率—C4.5

定义：

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

The set of samples

The term is used to measure the number of the values of an attribute

An attribute

The number of the values of the attribute

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

IV(a)称为属性a的固有值，属性a的可能取值越多，IV(a)的值通常会越大
 增益率准则对可取值数目较少的属性有所偏好，因此**C4.5算法并不是直接选择增益率最大的候选划分属性**，而是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，再从中选取增益率最高的

剪枝处理

剪枝是决策树学习算法对付过拟合的主要手段
 基本思想：减少树结构中使用的分支数量

原因：

- 决策树学习中，为了尽可能正确分类训练样本，结点划分过程将不断重复，**有时会造成决策树分支过多**，这时就可能因为训练样本学得太好以致于把训练集自身的一些特点当作所有数据都具有的一般性质而导致**过拟合**

判断泛化能力提升：通过验证集上的准确率

预剪枝

规则：**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**决策树泛化能力**的提升，则停止划分并将当前结点标记为叶结点

举例说明：

Training set	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
Training set	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
	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
	10	青绿	硬挺	清脆	清晰	平坦	软粘	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
Validation set	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
	编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
	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
	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
	11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
Validation set	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

Step1：基于信息增益准则，选取属性脐部来对训练集进行划分，产生3个分支

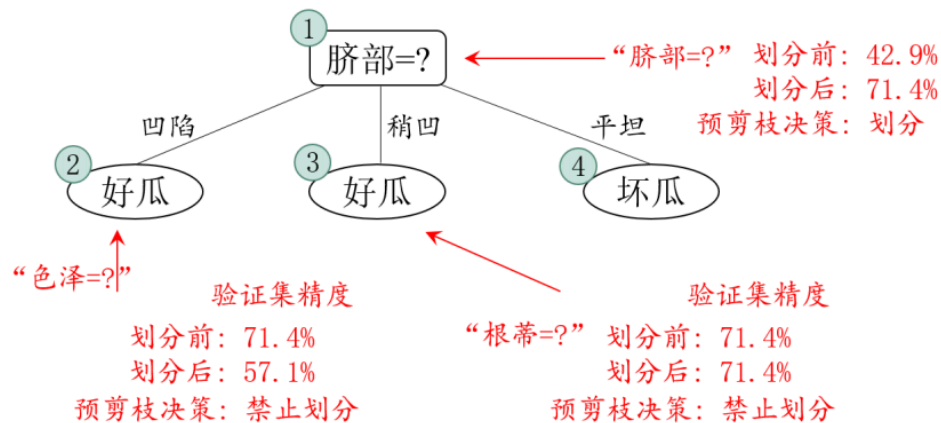
Step2：对泛化前后的泛化性能进行估计以确定是否进行此划分

- 若不进行划分，则该结点被标记为叶结点，其类别为训练样例数最多的类别（当样例最多的类不唯一时可任选其中一类）

唯一 将这个叶结点标记为“好瓜”。用表 4.2 的验证集对这个单结点决策树进行评估, 则编号为 {4, 5, 8} 的样例被分类正确, 另外 4 个样例分类错误, 于是, 验证集精度为 $\frac{3}{7} \times 100\% = 42.9\%$ 。

在用属性“脐部”划分之后, 图 4.6 中的结点 ②、③、④ 分别包含编号为 {1, 2, 3, 14}、{6, 7, 15, 17}、{10, 16} 的训练样例, 因此这 3 个结点分别被标记为叶结点“好瓜”、“好瓜”、“坏瓜”。此时, 验证集中编号为 {4, 5, 8, 11, 12} 的样例被分类正确, 验证集精度为 $\frac{5}{7} \times 100\% = 71.4\% > 42.9\%$ 。于是, 用“脐部”进行划分得以确定。

Step3: 重复上述步骤对分支结点进行划分和预剪枝



优点:

- 降低过拟合的风险, 显著减少决策树的训练时间开销和测试时间开销

缺点:

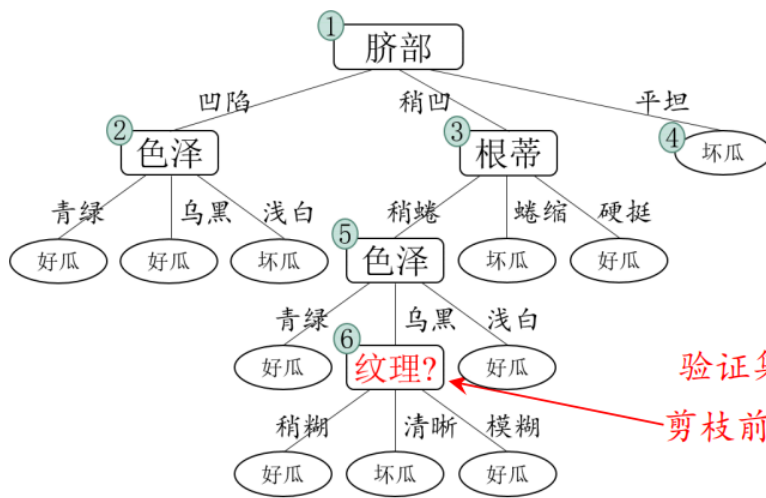
- 有些分支的当前划分虽不能提升泛化性能, 甚至可能导致泛化性能的暂时下降, 但在其基础上进行的后续划分可能导致性能显著提升, 可能带来欠拟合的风险

后剪枝

规则: 先从训练集生成一颗完整的决策树, 然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能力提升, 则将子树替换为叶结点

举例说明:

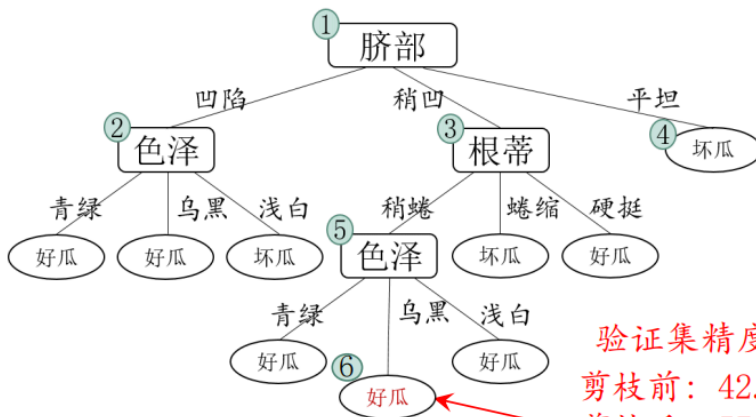
Step1: 先从训练集中生成一颗完整的决策树



对于节点6，剪枝前
验证集精度：3/7=42.9%

验证集精度
剪枝前：42.9%

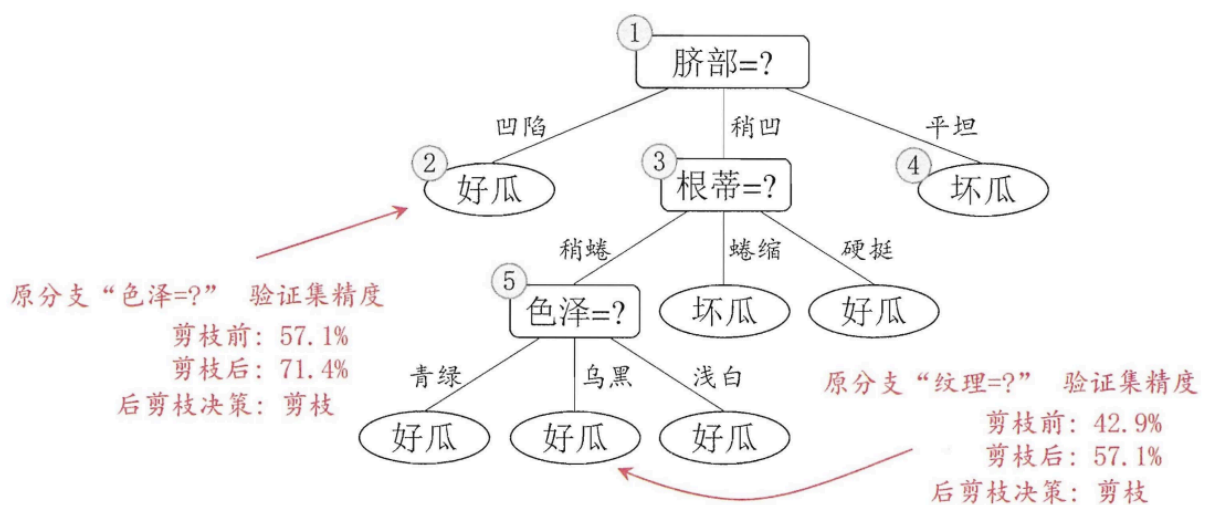
Step2: 自底向上对非叶节点进行考察，首先考察结点6，若将其领衔的分支剪除，相当于将结点6替换为叶结点（包含的样本实例集中哪个类别占比大贴谁的label）



此时验证集精度：57.1% > 42.9%
因此剪枝。

验证集精度
剪枝前：42.9%
剪枝后：57.1%
后剪枝决策：剪枝

Step3: 重复上述过程，直至根节点



优点：

- 后剪枝决策树的欠拟合风险很小

- 泛化能力往往优于预剪枝决策树

缺点：训练开销比未剪枝决策树和预剪枝决策树都大得多（生成完整的决策树+自底向上对所有非叶节点进行逐一考察）——但是预测速度变快

特殊情况处理

连续值处理

在决策树学习中使用连续属性

C4.5中使用二分法对连续属性进行处理

思想：

确定一个阈值 T_a ，将连续属性 A_c 转化为具有两个值的离散属性 A_d

Find a threshold T_a to change continuous attribute A_c to discrete attribute A_d which has two values

$$A_d = \begin{cases} \text{true}, & \text{if } A_c < T_a \\ \text{false}, & \text{otherwise} \end{cases}$$

选定阈值步骤：

T_a 中的为候选划分点，选取最优的划分点进行样本集合的划分

在区间 $[a^i, a^{i+1})$ 中取任意值所产生的划分结果相同。因此，对连续属性 a ，我们可考察包含 $n-1$ 个元素的候选划分点集合

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} \mid 1 \leq i \leq n-1 \right\}, \quad (4.7)$$

即把区间 $[a^i, a^{i+1})$ 的中位点 $\frac{a^i + a^{i+1}}{2}$ 作为候选划分点。然后，我们就可像离散属性值一样来考察这些划分点，选取最优的划分点进行样本集合的划分。例如，可对式(4.2)稍加改造：

$$\begin{aligned} \text{Gain}(D, a) &= \max_{t \in T_a} \text{Gain}(D, a, t) \\ &= \max_{t \in T_a} \text{Ent}(D) - \sum_{\lambda \in \{-, +\}} \frac{|D_t^\lambda|}{|D|} \text{Ent}(D_t^\lambda), \end{aligned} \quad (4.8)$$

上述 $\text{Gain}(D, a, t)$ 是样本集 D 基于划分点 t 二分后的信息增益，**最终选择使 $\text{Gain}(D, a, t)$ 最大的划分点**

注意：与离散属性不同的是，若当前结点划分属性为连续属性，该属性还可作为其后代结点的划分属性

离散属性：一条路径上属性不可重复使用的

缺失值处理

- 如何在属性值缺失的情况下进行划分属性选择
 - 仅可根据训练集 D 在属性 a 上没有缺失值的子集来判断属性 a 的优劣

给定训练集 D 和属性 a , 令 $\tilde{D}$ 表示 D 中在属性 a 上没有缺失值的样本子集。对问题(1), 显然我们仅可根据 $\tilde{D}$ 来判断属性 a 的优劣。假定属性 a 有 V 个可取值 $\{a^1, a^2, \dots, a^V\}$, 令 $\tilde{D}^v$ 表示 $\tilde{D}$ 中在属性 a 上取值为 a^v 的样本子集, $\tilde{D}_k$ 表示 $\tilde{D}$ 中属于第 k 类 ($k = 1, 2, \dots, |\mathcal{Y}|$) 的样本子集, 则显然有 $\tilde{D} = \bigcup_{k=1}^{|\mathcal{Y}|} \tilde{D}_k$, $\tilde{D} = \bigcup_{v=1}^V \tilde{D}^v$ 。假定我们为每个样本 x 赋予一个权重 w_x , 并定义

$$\rho = \frac{\sum_{x \in \tilde{D}} w_x}{\sum_{x \in D} w_x}, \quad \text{在 } a \text{ 属性上有值的样本所占比例} \quad (4.9)$$

$$\tilde{p}_k = \frac{\sum_{x \in \tilde{D}_k} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq k \leq |\mathcal{Y}|), \quad \text{无缺失值样本中第 } k \text{ 类所占比例} \quad (4.10)$$

$$\tilde{r}_v = \frac{\sum_{x \in \tilde{D}^v} w_x}{\sum_{x \in \tilde{D}} w_x} \quad (1 \leq v \leq V), \quad \text{无缺失值样本中在属性 } a \text{ 上取值为 } a^v \text{ 的样本所占比例} \quad (4.11)$$

基于上述定义, 我们可将信息增益的计算式(4.2)推广为

$$\begin{aligned} \text{Gain}(D, a) &= \rho \times \text{Gain}(\tilde{D}, a) \\ &= \rho \times \left(\text{Ent}(\tilde{D}) - \sum_{v=1}^V \tilde{r}_v \text{Ent}(\tilde{D}^v) \right), \end{aligned} \quad (4.12)$$

其中由式(4.1), 有

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

举例

Training set

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

色泽: $\tilde{D} = \{2+, 3+, 4+, 6+, 7+, 8+, 9-, 10-, 11-, 12-, 14-, 15-, 16-, 17-\}$

色泽: $\tilde{D} = \{2+, 3+, 4+, 6+, 7+, 8+, 9-, 10-, 11-, 12-, 14-, 15-, 16-, 17-\}$

Information gain:

$$\begin{aligned} \text{Gain}(\tilde{D}, \text{色泽}) &= \text{Ent}(\tilde{D}) - \sum_{v=1}^3 \tilde{r}_v \text{Ent}(\tilde{D}^v) \\ &= 0.985 - \left(\frac{4}{14} \times 1.000 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0.000 \right) \\ &= 0.306 \end{aligned}$$

我们这里把含有色泽属性样本集的权重所占的比例考虑进去 (每个样本的初始权重为1): $\tilde{D}$ 含有14个样本, 每个样本的权重为1, 所以 $\tilde{D}$ 总权重为14; 训练集 D 共包含17个样本, 每个样本的权重为1, 所以训练集 D 的总权重为17; $\tilde{D}$ 所占权重比例为 $\frac{14}{17}$;

$$\sum_{v=1}^3 \tilde{r}_v \text{Ent}(\tilde{D}^v) = \frac{4}{14} \times 1.000 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0.000 = 0.679$$

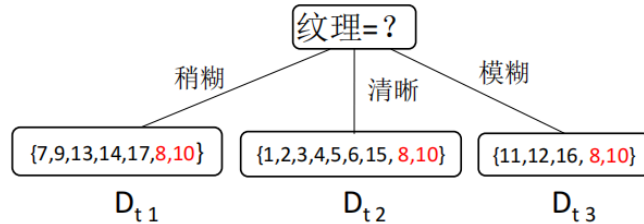
$$\text{Gain}(D, \text{色泽}) = \rho \times \text{Gain}(\tilde{D}, \text{色泽}) = \frac{14}{17} \times 0.306 = 0.252$$

- 给定划分属性, 如果样本在该属性上的值有缺失, 如何对样本进行划分
 - 若样本 x 在划分属性 a 上的取值已知, 则将 x 划入与其取值对应的子结点, 且样本权重在子结点中保持为 w_x
 - 若样本 x 在划分属性 a 上的取值未知

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 a^v 对应的子结点中调整为 $\tilde{r}_v \cdot w_x$; 直观地看, 这就是让同一个样本以不同的概率划入到不同的子结点中去.

直观来看: 让同一个样本以不同的概率划入到不同的子结点中去

- ✓ 纹理(15个样本): {1, 2, 3, 4, 5, 6, 7, 9, 11, 12, 13, 14, 15, 16, 17}
- 其中: 稍糊(5个样本): {7, 9, 13, 14, 17}
- 清晰(7个样本): {1, 2, 3, 4, 5, 6, 15}
- 模糊(3个样本): {11, 12, 16}
- ✓ 缺失纹理属性取值的样本: {8, 10}



选择纹理属性后, 我们把在纹理属性上有取值的样本划分到三个分支, 权重不变; 同时把在纹理属性上没有取值的样本8, 10同时放进三个分支, 在三个子节点的权重调整为 $\tilde{r}_v \cdot w_x$, 即 $\frac{5}{15}, \frac{7}{15}, \frac{3}{15}$ 。则:

1. D_{t1} 各个样本权重: 样本7, 9, 13, 14, 17的权重为1, 样本8, 10的权重为 $\frac{5}{15}$
2. D_{t2} 各个样本权重: 样本1, 2, 3, 4, 5, 6, 15的权重为1, 样本8, 10的权重为 $\frac{7}{15}$
3. D_{t3} 各个样本权重: 样本11, 12, 16的权重为1, 样本8, 10的权重为 $\frac{3}{15}$

优缺点

优点:

- 生成容易理解的规则
- 进行分类无需大量计算
- 明确指出哪些属性对预测或分类最为重要
- 擅长处理“矩形区域”的分类任务
 - 决策树所形成的分类边界的明显特点:
 - **轴平行**, 它的分类边界由若干个与坐标轴平行的分段组成, 这样的分类边界使得学习结果有**较好的可解释性**
 - 当学习任务的真实分类边界比较复杂, 必须使用很多段划分才能获得较好的近似

缺点:

- 决策树可能出现“误差传播”问题
 - 决策树是“串行判断”的: 从根节点开始, 每一步的判断结果会决定下一个节点的选择。如果某一层的属性划分出现了小误差, 这个误差会沿着后续的节点传递下去, 最终可能导致分类结果的错误
- 不擅长处理“非矩形区域”的分类任务

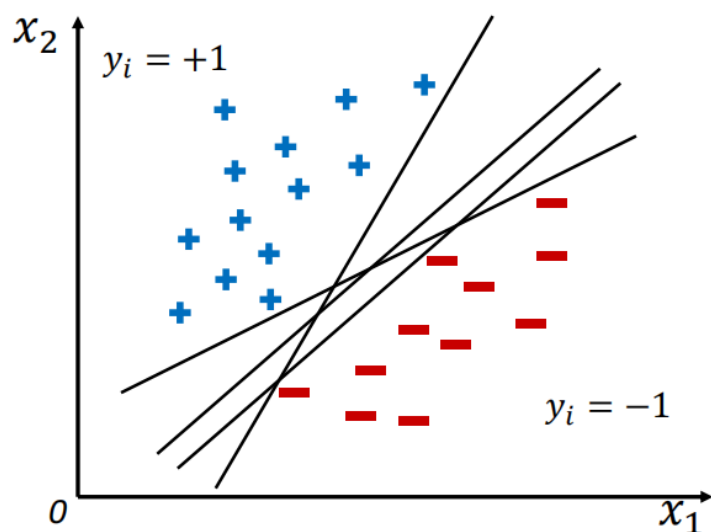
05 Support vector machine

间隔与支持向量

思想引入

给定训练样本集 D ，分类学习的最基本的想法就是基于训练集 D 在样本空间中划分超平面，将不同类别的样本分开

Train data set $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$, $y_i \in \{-1, +1\}$



Any of these
would be fine...

...but which one is
the best?

如何寻找划分效果最好【这个划分超平面产生的分类结果是最鲁棒的，对未见示例的**泛化能力最强**】的超平面？

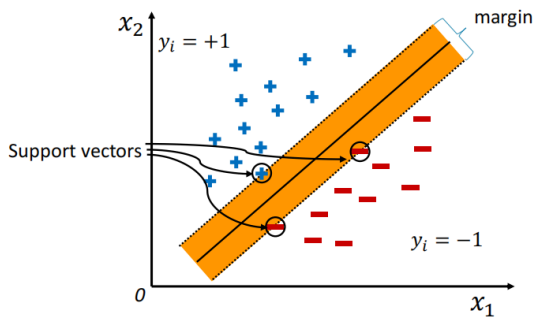
直观：应该去找位于两类训练样本“正中间”的划分超平面，因为该划分超平面对训练样本局部扰动的“容忍性”最好，直线与训练样本之间的最小距离应尽可能大——间隔最大

定义

支持向量 (support vector)：距离超平面最近的几个训练样本点（间隔上下边界上的点）

间隔 (margin)：将线性分类器的边界宽度定义为边界在触及数据点前可增加的最大宽度

间隔大小与支持向量相关，支持向量决定间隔宽度
每一个分类器对应一个间隔，选择最大间隔！



推导求解过程

(具体过程见pad)

一定要会从原始问题到对偶问题的推导！！+后续KKT条件分析

summary

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

$$\begin{aligned} \max_{\alpha} \quad & L(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \\ & \alpha_i \geq 0, \quad i = 1, 2, \dots, m \end{aligned}$$

$$\begin{aligned} & L(w, b, \alpha) \\ & \min_{w,b} \max_{\alpha: \alpha_i \geq 0} L(w, b, \alpha) \\ & \max_{\alpha: \alpha_i \geq 0} \min_{w,b} L(w, b, \alpha) \end{aligned}$$

- Solve the dual problem, get the value of α

- Calculate 'w' and 'b': $w = \sum_{i=1}^m \alpha_i y_i x_i$ $b = y_s - \sum_{i \in t} \alpha_i y_i x_i^T x_s$

- The final model:

$$f(x) = w^T x + b = \sum_{i=1}^m \alpha_i y_i x_i^T x + b = \sum_{\alpha_i \neq 0} \alpha_i y_i x_i^T x + b$$

28

- 大多数 $\alpha_i = 0$, 不会对 $f(x)$ 产生影响
- 若 $\alpha_i > 0$, 则必有 $y_i f(x_i) = 1$, 对应的样本点位于最大间隔边界上, 是一个支持向量

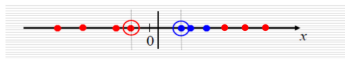
支持向量机的一个重要性质: 训练完成后, 大部分的训练样本都不需要保留, 最终模型只有支持向量相关

核函数

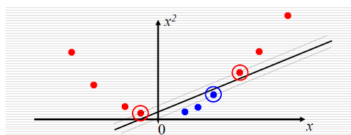
解决的是在原始样本空间线性不可分的情况

在上述讨论中，我们假设样本是线性可分的，存在一个超平面能将训练样本正确分类，然而在实际任务中，原始样本空间也许并不能存在一个能正确划分两类样本的超平面

- Datasets are linearly inseparable.



- How about mapping data to a higher dimensional space :



解决措施：

将样本从原始空间映射到一个更高维的特征空间，使样本在高维特征空间内线性可分

结论：如果原始空间是有限维的，即属性数有限，那么一定存在一个高维特征空间使样本可分

核函数的引入

核函数

令 $\phi(x)$ 表示 x 映射后的特征向量。

在特征空间划分超平面所对应模型

$$f(x) = w^T \phi(x) + b$$

$$\min_{w, b} \frac{1}{2} \|w\|^2$$

$$\text{s.t. } y_i (w^T \phi(x_i) + b) \geq 1 \quad i=1, 2, \dots, m$$

其对应偶问题

$$\max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j)$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \quad i=1, 2, \dots, m$$

→ 由于特征空间维数可能很高，甚至可能是无穷维，直接计算困难。

核函数可避开上述困难

$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle = \phi(x_i)^T \phi(x_j)$$

特征空间中特征向量内积计算 → 在原始输入空间中计算 K 。

$$\Rightarrow \max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \quad i=1, 2, \dots, m$$

$$f(x) = w^T \phi(x) + b$$

$$= \sum_{i=1}^m \alpha_i y_i \phi(x_i)^T \phi(x) + b$$

$$= \sum_{i=1}^m \alpha_i y_i K(x_i, x) + b$$

核函数的选取

定理 6.1 (核函数) 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 $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$, “核矩阵” (kernel matrix) $\mathbf{K}$ 总是半正定的:

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

只要一个对称函数所对应的核矩阵半正定, 它就能作为核函数使用
常用的核函数:

表 6.1 常用核函数

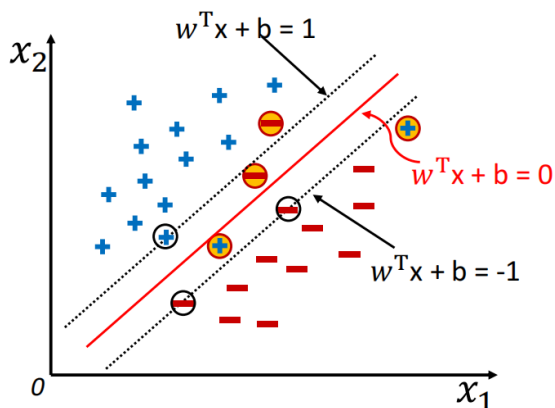
名称	表达式	参数
线性核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$	
多项式核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j)^d$	$d \geq 1$ 为多项式的次数
高斯核	$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\ \mathbf{x}_i - \mathbf{x}_j\ ^2}{2\sigma^2}\right)$	$\sigma > 0$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

软间隔

在现实任务中往往很难确定合适的核函数使得训练样本在特征空间中线性可分, 即便恰好找到了某个函数使训练集在特征空间中线性可分, 也很难断定这个貌似可分的结果不是由于过拟合所造成的

解决方案: 允许支持向量机在一些样本上出错

软间隔: 允许某些样本不满足约束条件



原则: 在最大化间隔的同时, 使得不满足约束的样本数量尽可能少

Maximize margin

Minimize # training samples that don't satisfy the constraint

$$\min_{w,b} \left[\frac{1}{2} \|w\|^2 + C \sum_{i=1}^m l_{0/1}(y_i(w^T x_i + b) - 1) \right]$$

↑ 惩罚系数
↑ 损失函数

- $l_{0/1}$ --- loss function

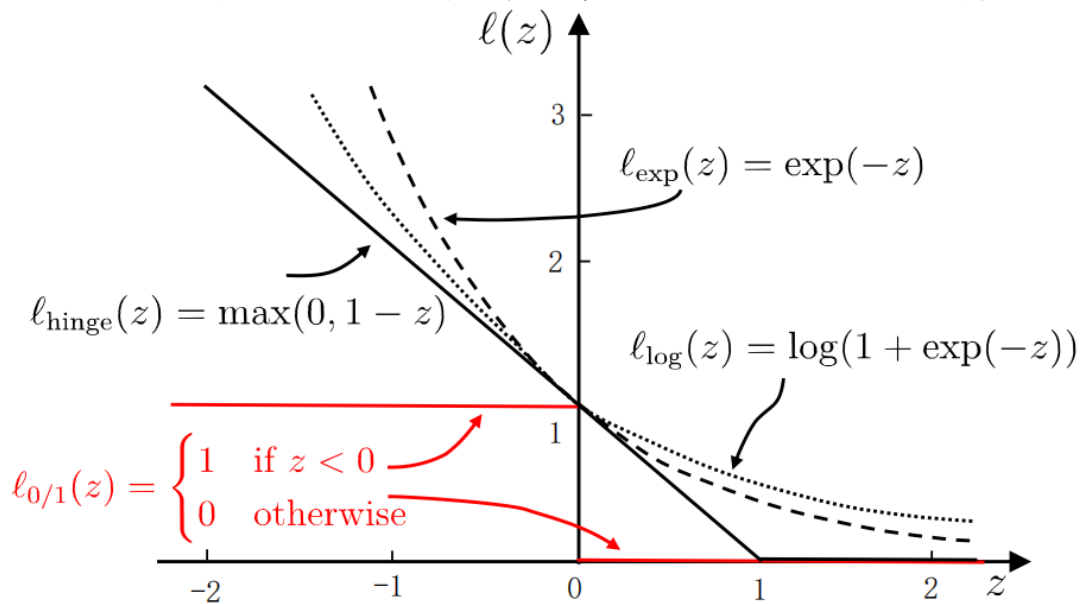
$$l_{0/1}(z) = \begin{cases} 1, & \text{if } z < 0; \\ 0, & \text{otherwise.} \end{cases}$$

$C > 0$, tradeoff parameter; $C = \infty \longrightarrow$ hard margin SVM

But $l_{0/1}$ is not convex, discontinuous (非凸、非连续)

通常用其他一些函数替代 l ---替代损失

(替代损失函数一般具有较好的数学性质，通常是凸的连续函数且是 l 的上界)



hinge loss: $l_{hinge}(z) = \max(0, 1 - z)$

exponential loss(指数损失): $l_{exp}(z) = \exp(-z)$

logistic loss(对率损失): $l_{log}(z) = \log(1 + \exp(-z))$

以使用hinge loss举例:

若采用 hinge 损失

则上述问题变成

$$\min_{w, b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \max(0, 1 - y_i(w^T x_i + b))$$

引入松弛变量 $\xi_i \geq 0$

$$\min_{w, b, \xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \xi_i$$

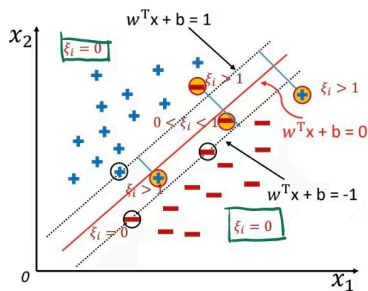
$$\text{s.t. } y_i(w^T x_i + b) \geq 1 - \xi_i$$

最大化
间隔

最小化
越界程度

每个样本都有一个对应的松弛变量
用以表征该样本不满足约束的程度。

→ 软间隔
支持向量机



$\xi_i = 0$: 在间隔边界外侧

$0 < \xi_i < 1$: 样本在间隔边界与超平面间

$\xi_i > 1$: 样本跨越超平面, 分类错误。

具体求解过程:

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

$$\alpha_i \geq 0, \mu_i \geq 0$$

$$\frac{\partial L}{\partial w} = \|w\| - \sum_{i=1}^m \alpha_i y_i x_i = 0 \Rightarrow w = \sum_{i=1}^m \alpha_i y_i x_i$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^m \alpha_i y_i = 0$$

$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - \mu_i = 0$$

$$C = \alpha_i + \mu_i$$

$$\max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\text{s.t. } \sum_{i=1}^m \alpha_i y_i = 0$$

$$0 < \alpha_i \leq C \quad i = 1, 2, \dots, m$$

→ 与硬间隔下的对偶问题
唯一差别在于对对偶变量的约束不同。

KKT 条件

$$\left. \begin{aligned} w &= \sum_{i=1}^m \alpha_i y_i x_i \\ \sum_{i=1}^m \alpha_i y_i &= 0 \\ C &= \alpha_i + \mu_i \end{aligned} \right\} \text{驻点条件}$$

$$\left. \begin{aligned} \alpha_i (y_i f(x_i) - 1 + \xi_i) &= 0 \\ \mu_i \xi_i &= 0 \end{aligned} \right\} \text{互补松弛}$$

$$\left. \begin{aligned} y_i f(x_i) - 1 + \xi_i &\geq 0 \\ \xi_i &\geq 0 \end{aligned} \right\} \text{原问题可行}$$

$$\alpha_i \geq 0, \mu_i \geq 0 \rightarrow \text{对偶问题可行.}$$

$v(x_i, y_i)$ 是 $\alpha_i = 0$ 或 $y_i f(x_i) = 1 - \xi_i$

① 若 $\alpha_i = 0$ 则对 $f(x)$ 无影响

② 若 $\alpha_i > 0$, $y_i f(x_i) = 1 - \xi_i$ → 该样本为支持向量。

若 $\alpha_i < C$, $\mu_i > 0$, $\xi_i = 0$ 样本恰在最大间隔边界上

$\alpha_i = C$, $\mu_i = 0$ $\xi_i \leq 1$ 样本落在最大间隔内部

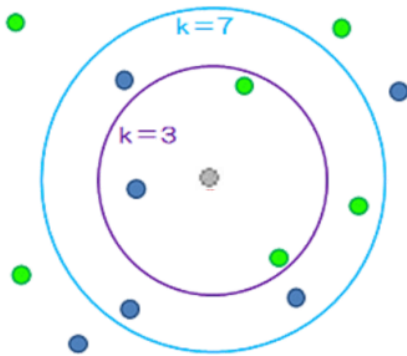
$\xi_i > 1$ 样本被错误分类。

06 K Nearest Neighbors algorithm

定义

K近邻算法被认为是一种惰性学习算法【无学习过程】，其通过数据集与邻近样本的相似度进行分类

- “K” 代表在分类过程中被纳入考量的数据集样本数量
- K会影响结果



注意：

K 近邻算法被称为**非参数方法**

与其他监督学习算法不同，**K 近邻算法**不会从训练数据中学习一个明确的映射关系（函数 f ），它只是在**测试阶段**直接利用训练数据来进行预测

K-NN技术角度

问题定义

解决问题：

已知训练数据为 $(x_1, y_1), \dots, (x_N, y_N)$ ，以及一个测试样本 x ，共包含 N 组数据； x_i 是一个由 D 个特征组成的**向量**， y_i 是对应的**标签**

目标：为未见过的测试样本 x 预测输出结果 y

预测规则：找出与测试样本最相似的 K 个训练样本

输出的形式：

- 分类问题
 - 输出是**离散变量**，即 $y_i \in \{1, \dots, C\}$
 - 通过**多数投票**(majority voting)，为测试样本分配占比最多的类别标签
- 回归问题
 - 输出是**连续（实数值）变量**，即 $y_i \in \mathbb{R}$
 - 通过计算**平均值**，为测试样本分配邻近样本的平均响应值

算法所需要素

- **参数 K**: 需要选取的近邻样本数量
- **距离函数**: 用于计算样本之间的相似度

距离度量

1. 欧氏距离被广泛使用 Euclidean distance

在欧几里得平面中, 若 $\mathbf{p} = (p_1, p_2)$ 且 $\mathbf{q} = (q_1, q_2)$, 则两者的距离公式为:

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$$

在三维欧几里得空间中, 距离公式为:

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2}$$

一般地, 在n维空间, 距离公式为:

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}$$

2. 曼哈顿距离 (城市街区距离) Manhattan Distance

二维空间下的曼哈顿距离公式:

$$d(\mathbf{p}, \mathbf{q}) = |p_1 - q_1| + |p_2 - q_2|$$

一般地, 在n维空间中, 曼哈顿距离为:

$$d(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\| = \sum_{i=1}^n |p_i - q_i|$$

3. 闵可夫斯基距离 (泛概念) Minkowski Distance

两个点 $\mathbf{p} = (p_1, p_2, \dots, p_n)$ 和 $\mathbf{q} = (q_1, q_2, \dots, q_n) \in \mathbb{R}^n$ 之间的 $\mathbf{p}$ 阶闵可夫斯基距离定义为:

$$\left(\sum_{i=1}^n |p_i - q_i|^m \right)^{1/m}$$

闵可夫斯基距离通常采用 $m=1$ 或 2 的情况, 其中分别对应曼哈顿距离和欧式距离

4. 余弦相似度 (Cosine similarity)

余弦相似度是内积空间中两个非零向量之间相似性的度量, 通过计算它们之间夹角的余弦值来衡量

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i \times B_i}{\sqrt{\sum_{i=1}^n (A_i)^2} \times \sqrt{\sum_{i=1}^n (B_i)^2}}$$

高维向量用得更多

5. 汉明距离 (Hamming distance)

统计两个样本在特征上不一致的程度

换句话说, 它衡量将一个字符串转换为另一个字符串所需的最小替换次数

eg: 1011101 和 1001001 间的汉明距离2

特征归一化

特征应该处于同一尺度

Features should be on the same scale!

eg: mm&cm---统一尺度

一种方法: 正态化

One way is:

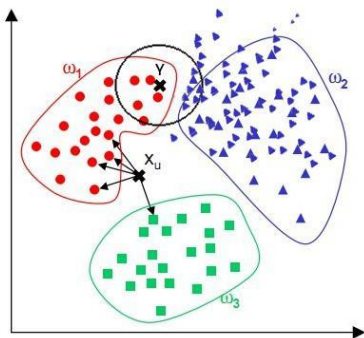
Replace p_i by $z_{im} = \frac{(x_{im} - \bar{x}_m)}{\sigma_m}$ (make them zero mean, unit variance)

$\bar{x}_m = \frac{1}{N} \sum_{i=1}^N x_{im}$: mean of m^{th} feature

$\sigma_m^2 = \frac{1}{N} \sum_{i=1}^N (x_{im} - \bar{x}_m)^2$: variance of m feature

特征加权

每一维度的特征不是同等重要的, 根据特征对分类的重要性进行分级



- 等权
- 距离更近权重更高
- 先验知识
- 自适应权重---在模型中学 (交叉验证学习权重)

$$d(p, q) = \sqrt{\sum_k W_k (p_k - q_k)^2}$$

怎样选择k

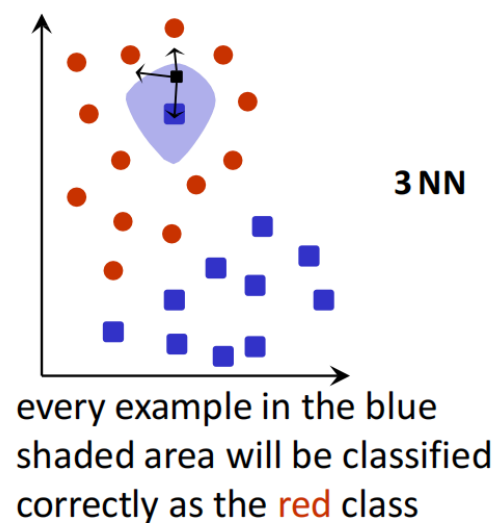
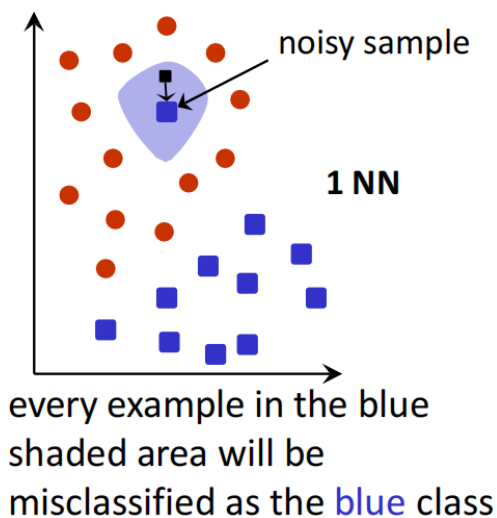
理论分析

理论上，若样本数量无限，k值越大，分类效果越好

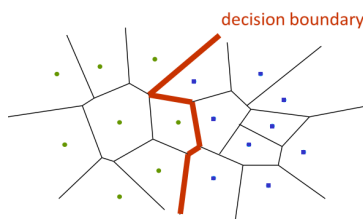
实际上样本是有限的

k=1 最近的一个

- 效率高但对噪声敏感



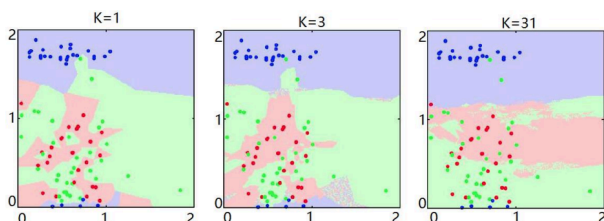
- 维诺图Voronoi Diagram



- Voronoi图将空间划分为若干单元格
- 每个单元格包含一个样本，且单元格内的任意位置与该样本的距离，都小于其与其他样本的距离。

k=N 大多数

K 值（近邻数量）的选择对分类结果的影响



Small K

为每个类别划分出大量小区域,对噪声敏感

可能导致决策边界不平滑，引发过拟合

Large K

划分出更少但更大的区域；通常决策边界更平滑；能降低类别标签噪声的影响
过于平滑的决策边界可能引发欠拟合

选择方法

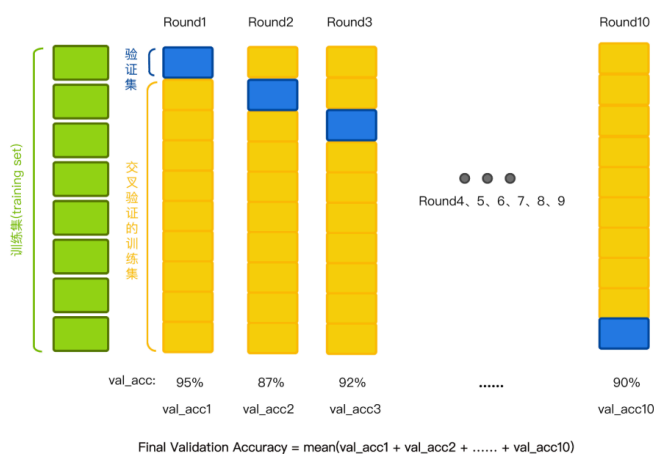
Holdout 验证法

机器学习中常用的模型评估与参数选择方法

- 数据划分
 - 将带有标签的数据分为两组
 - 训练集（占比 80%）和验证集（占比 20%）
- 模型验证
 - 用训练集中的样本对验证集的类别标签进行预测，以此估计模型的泛化误差
- 参数选择
 - 选择能使分类准确率最大化的近邻数k
- 最终评估
 - 确定模型后，在独立的测试集上报告其最终性能

K-fold cross-validation

1. 数据划分：将训练集划分为K个大小相近的子集
2. 多轮验证：
 - 每一轮（Round）选取 1 个子集作为验证集，剩余(K-1)个子集作为训练集。
 - 用训练集训练模型，在验证集上计算准确率
3. 最终评估：将K轮验证的准确率取平均值，作为模型的最终验证准确率



Leave-One-Out Method

每次留一个样本做验证，用剩余所有样本做训练

1. 遍历候选 K 值：对每个待选的近邻数k，单独计算其误差
2. 逐个验证样本：
 - 对数据集中的每个样本 x_i ，用除 x_i 外的所有样本训练模型，预测 x_i 的标签。
 - 若预测错误，就给当前k对应的误差 $\text{err}(k)$ 加 1。

3. 选择最优 K 值：最终选取误差(err(k))最小的k作为最优近邻数 k^*

总结

对于K的选择

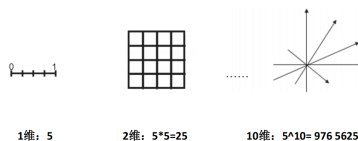
- 奇数---避免平局
- 1-NN在实际应用中通常表现良好
- 可以选择交叉验证等方式来选择k
- 对于距离度量的选择
- 通常使用欧式距离
- 局部自适应度量（会根据样本的局部分布调整距离计算方式）
- 当处理非数值型数据，或不同维度的特征尺度【归一化】存在差异时，距离度量会更复杂

维数灾难

K-NN算法在高维空间中失效，“邻居”变得极为庞大且易产生误导

维度灾难是指在高维空间中出现的、日常经验里低维场景不会发生的各类现象

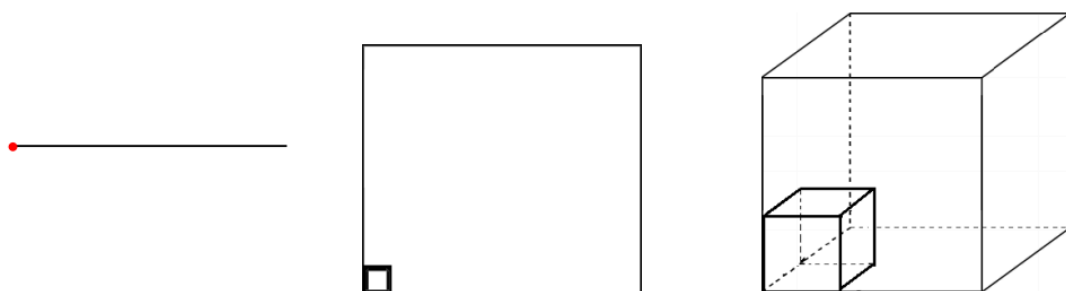
- 存储复杂度问题
- 计算复杂度问题
- 采样问题



- 组合问题
- 近邻搜索问题

● Assume 5000 points uniformly distributed in the unit hypercube and we want to apply 5-nn. (Suppose our query point is at the origin)

- In 1-dimension, we must go a distance of $5/5000 = 0.001$ on the average to capture 5 nearest neighbors
- In 2 dimensions, we must go $\sqrt{0.001}$ to get a square that contains 0.001 of the volume.
- In d dimensions, we must go $(0.001)^{1/d}$



随着维度 d 增加，要包含 5 个近邻点所需的距离会**急剧增大**：

比如当 $d = 10$ 时， $(0.001)^{1/10} \approx 0.56$ ，几乎要覆盖超立方体的一半范围；当 $d = 20$ 时，距离接近 0.9，意味着“近邻”已经和“全局点”没有区别——这就是高维空间中“近邻概念失效”的直观体现。

维数很大，找5-NN几乎搜索整个空间

- 距离函数问题
- 非参数估计问题

优缺点

优点：

- 简单直观、训练速度极快、易于实现；
- 特别适用于**多分类问题**；
- 若训练数据无限多且 K 值足够大， K -NN 的效果会很好

缺点：

- 对含噪声的特征敏感；
- 测试阶段仍需将所有训练数据存储在内存在中；
- 查询（预测）速度慢：每个测试样本需执行($O(nd)$)的计算（ n 为训练样本数， d 为特征维度）；
- 高维空间中，距离的概念会变得违背直觉；
- 高维场景下性能可能很差（受维度灾难影响）

K-d Trees

k-d 树的深度保证为 $\log_2 n$, 传统上k-d树用于存储d维空间上的数据点

局限性：K-D树算法在极高维空间中难以高效实现最近邻搜索——需要回溯的分支太多！

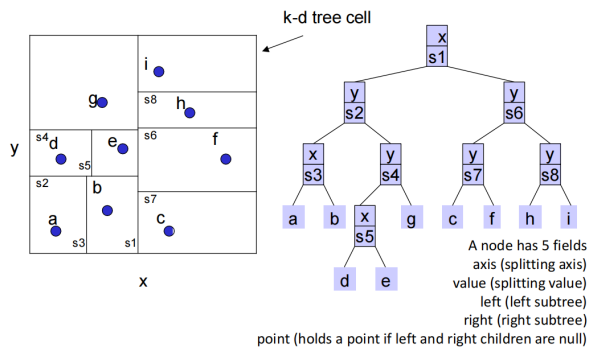
核心思想与示例

核心思想：树的每一层仅针对**1个维度**进行划分

k-d 树的核心价值是**解决K近邻算法“查询速度慢”**的，k-d 树能将查询复杂度降低到 $O(d \log n)$ ，大幅提升高样本量场景下的查询效率。

树的每个节点包含 5 个字段：

- **axis**：当前层的分割维度（如 x 或 y ）；
- **value**：分割阈值（如 s_1 是 x 轴的分割值）；
- **left/right**：左 / 右子树（对应分割后的左右 / 上下区域）；
- **point**：若节点无左右子树（叶子节点），则存储对应的数据点



K-d树构建流程

根节点：通过垂直于 x_1 轴的超平面，将数据点集分割为大小相等的两个子集

根节点的子节点：基于第二个坐标进行划分 x_2 轴

深度循环：当树的深度为 d 时，重新从第一个坐标开始交替划分【按维度顺序交替划分】

递归终止：直到每个子集只剩1个点，将其作为叶节点存储

构建的两个核心问题：

- 选择哪个维度进行划分
 - 选择分布范围最宽的轴、方差最大的轴，或按维度顺序交替选择
- 该维度的分割点如何确定
 - 通常选择该维度的**中位数**，或区间的中点

k-d树的邻近搜索步骤

k-d Tree Nearest Neighbor Search

- 递归定位查询点所在的单元格
通过 k-d 树的分层维度划分，递归遍历树结构，找到与查询点处于同一个空间单元格的那个数据点
- 回溯搜索可能的更近邻点
在返回（回溯）遍历树的过程中，检查每个子树 —— 只要该子树对应的空间区域，有可能存在比当前已知近邻**更近**的点，就对这个子树进行搜索

07 Linear Regression

回归概念

回归分析用于估计输入值与输出值之间的关系，并建立数学模型，以准确预测新样本的输出值

- 回归分析是一个监督学习问题

分类：

根据输入值的维度

- 单元回归
- 多元回归

根据输入值和输出值的关系

- 线性回归
- 非线性回归

线性回归模型

定义

general model:

$$f(x) = w_1x_1 + w_2x_2 + \cdots + w_dx_d + b$$

$x_1, x_2, \cdots, x_d$: 样本的特征

$w_1, w_2, \cdots, w_d$: 权重，体现相应特征的重要性

vector formal:

$$f(x) = w^T x + b$$

$w = (w_1; w_2; \cdots; w_d)$

$x = (x_1; x_2; \cdots; x_d)$

举个例子：

模型训练:

training set:

S1: 楼层=6, 地段=5, 面积=120。房价=100
S2: 楼层=3, 地段=1, 面积=100。房价=200
S3: 楼层=16, 地段=8, 面积=200。房价=120

$$\begin{cases} 6w_1 + 5w_2 + 120w_3 = 100 \\ 3w_1 + 1w_2 + 100w_3 = 200 \\ 16w_1 + 8w_2 + 200w_3 = 120 \end{cases} \Rightarrow \begin{cases} w_1 = a \\ w_2 = b \\ w_3 = c \end{cases} \Rightarrow \text{通过训练集再导出权重向量 } w$$

$$f(x) = a \cdot x_{\text{楼层}} + b \cdot x_{\text{地段}} + c \cdot x_{\text{面积}}$$

预测:

将样本特征值代入得输出值

模型评估 (which model is better?)

$$f_{\text{房价}}(x) = 0.2 \cdot x_{\text{楼层}} + 0.5 \cdot x_{\text{地段}} + 0.3 \cdot x_{\text{面积}} + 1$$

$$g_{\text{房价}}(x) = 0.1 \cdot x_{\text{楼层}} + 0.4 \cdot x_{\text{地段}} + 0.5 \cdot x_{\text{面积}} + 3$$

a known data: 楼层=6, 地段=5, 面积=120, 房价=100 预测值与真实值对比

model f predictive value: $0.2 \cdot 6 + 0.5 \cdot 5 + 0.3 \cdot 120 + 1 = 40.7$

model g predictive value: $0.1 \cdot 6 + 0.4 \cdot 5 + 0.5 \cdot 120 + 3 = 65.6$

model f error: $100 - 40.7 = 59.3$

model g error: $100 - 65.6 = 34.4$

$34.4 < 59.3$

model g is better

优化目标:

能够最小化模型预测值与真实值之间误差---回归模型训练的核心逻辑

最小二乘法

least square method

在回归任务中, 常采用最小二乘法来测量模型预测值与真实值之间的误差

$$error = \sum_{i=1}^m (f(x_i) - y_i)^2$$

$f(x_i)$: 预测值

y_i : 真实值

最小二乘法:

$$\min \sum_{i=1}^m (f(x_i) - y_i)^2$$

单特征线性回归问题

single linear regression

求解 w 和 b

$$model: f(x_i) = wx_i + b$$

$$\begin{aligned} loss \quad function: E_{(w,b)} &= \sum_{i=1}^m (f(x_i) - y_i)^2 \\ &= \sum_{i=1}^m (wx_i + b - y_i)^2 \end{aligned}$$

最终任务

$$\min E_{(w,b)}$$

basic linear regression

• solve w and b

single linear regression:

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

$$\frac{\partial E_{(w,b)}}{\partial w} = 2 \left(w \sum_{i=1}^m x_i^2 - \sum_{i=1}^m (y_i - b) x_i \right) = 0$$

$$\frac{\partial E_{(w,b)}}{\partial b} = 2 \left(mb - \sum_{i=1}^m (y_i - wx_i) \right) = 0$$

new sample

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

$$b = \frac{1}{m} \sum_{i=1}^m (y_i - wx_i) \quad \bar{x} = \frac{1}{m} \sum_{i=1}^m x_i$$

model: $f(x_i) = wx_i + b$

dataset: $D = \{(x_i, y_i)\}_{i=1}^m$

17

多特征线性回归

$$\text{model: } f(x_i) = w_1 x_{i1} + w_2 x_{i2} + \dots + w_d x_{id} + b$$

$$\text{vector model: } f(x_i) = w^T x_i + b$$

$$\text{loss function: } E_{\hat{w}} = (y - X\hat{w})^T (y - X\hat{w})$$

vector model $f(x_i) = w^T x_i + b$

$$f(x) = \begin{pmatrix} w_1 & w_2 & \dots & w_d & b \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \\ 1 \end{pmatrix}$$

$\hat{w} = (w; b) \rightarrow (d+1) \times 1$ 向量

$$X = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1d} & 1 \\ x_{21} & x_{22} & \dots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \dots & x_{md} & 1 \end{pmatrix} = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_m^T \end{pmatrix} \quad m \times (d+1) \text{ 矩阵}$$

$$\hat{w} \in \mathbb{R}^{d+1}$$

$$x \in \mathbb{R}^{m \times (d+1)}$$

$$y \in \mathbb{R}^{m \times 1}$$

$y = (y_1, y_2, \dots, y_m)$ $m \times 1$ 向量 $\Rightarrow$ 样本真实值

$$\text{loss function: } E_{\hat{w}} = (y - X\hat{w})^T (y - X\hat{w}) \Leftrightarrow \|y - X\hat{w}\|^2$$

$$= (y^T - \hat{w}^T X^T) (y - X\hat{w})$$

$$= y^T y - y^T X \hat{w} - \hat{w}^T X^T y + \hat{w}^T X^T X \hat{w}$$

$$\frac{\partial E_{\hat{w}}}{\partial \hat{w}} = -(y^T X)^T - X^T y + (X^T X)^T + X^T X \hat{w}$$

$$= -X^T y - X^T y + 2 X^T X \hat{w}$$

$$= -2 X^T y + 2 X^T X \hat{w}$$

$$\text{令 } \frac{\partial E_{\hat{w}}}{\partial \hat{w}} = 0$$

$$X^T X \hat{w} = X^T y$$

$$\hat{w} = (X^T X)^{-1} X^T y$$

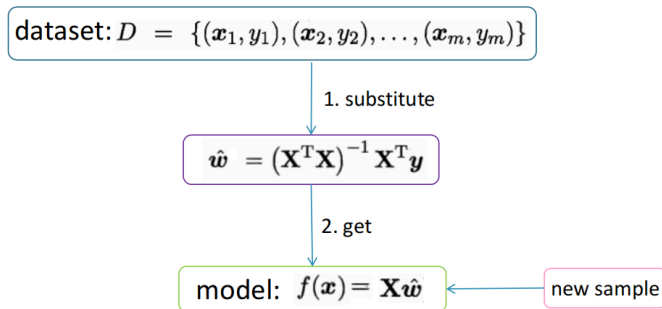
向量求导公式:

$$\frac{\partial (X\theta)}{\partial \theta} = X^T$$

$$\frac{\partial (\theta^T X)}{\partial \theta} = X$$

$$\frac{\partial (\theta^T X \theta)}{\partial \theta} = (X^T + X) \theta$$

•multiple linear regression:



补充视野:

SVM靠边界上的点解 w , b

线性回归用所有的样本点解 w , b

缺点与应用场景

缺点

1. 最小二乘法的损失函数有可能造成模型过拟合
2. 解析解中 $X^T X$ 有可能不是满秩矩阵, 那样 $(X^T X)^{-1}$ 不存在
 - 样本的特征数远远超过样本数, 这样 $\hat{w}$ 多个解

应用场景

1. 已有样本的值比较准确
2. 样本的特征数小于样本数

ridge regression & lasso regression

$$\text{model: } f(x_i) = w_1 x_{i1} + w_2 x_{i2} + \dots + w_d x_{id} + b$$

$$\text{vector model: } f(x_i) = w^T x_i + b$$

$$f(x) = X\hat{w}$$

loss function

基本线性回归:

$$\text{basic linear regression: } E_{\hat{w}} = (y - X\hat{w})^T (y - X\hat{w})$$

ridge regression(岭回归):

$$\text{ridge regression: } E_{\hat{w}} = (y - X\hat{w})^T (y - X\hat{w}) + \lambda \|\hat{w}\|_2^2 \quad \lambda > 0$$

lasso regression:

$$\text{ridge regression: } E_{\hat{w}} = (y - X\hat{w})^T (y - X\hat{w}) + \lambda \|\hat{w}\|_1 \quad \lambda > 0$$

ridge回归和lasso回归可以防止模型过拟合, 也可以解决样本的特征数远远超过样本数的问题

Norm范数

范数是用来衡量“向量或矩阵的大小（规模）”的指标，最终会输出一个数值

- Vector norm 向量范数

$$w \in \mathbb{R}^d$$

L_0 -norm: the number of non-zero elements in the vector

$$L_1\text{-norm: } \|w\|_1 = |w_1| + |w_2| + \dots + |w_d| = \sum_{i=1}^d |w_i|$$

$$L_2\text{-norm: } \|w\|_2 = \sqrt{w_1^2 + w_2^2 + \dots + w_d^2} = \left(\sum_{i=1}^d w_i^2 \right)^{\frac{1}{2}}$$

$$L_\infty\text{-norm: } \|w\|_\infty = \max\{|w_1|, |w_2|, \dots, |w_d|\} = \max_{1 \leq i \leq d} \{|w_i|\}$$



$$w = [1 \quad 2 \quad -2]$$

$$\|w\|_0 = 3 \quad \|w\|_1 = 5$$

$$\|w\|_2 = 3 \quad \|w\|_\infty = 2$$

- Matrix norm 矩阵范数

ridge regression

$$\text{model: } f(x_i) = w_1 x_{i1} + w_2 x_{i2} + \dots + w_d x_{id} + b$$

$$f(x_i) = w^T x_i + b$$

loss function:

$$E\hat{w} = (y - X\hat{w})^T (y - X\hat{w}) + \lambda \|\hat{w}\|^2 \quad \lambda > 0$$

$$= (y - X\hat{w})^T (y - X\hat{w}) + \lambda \hat{w}^T w$$

$$= (y - X\hat{w})^T (y - X\hat{w}) + \lambda \hat{w}^T I w$$

$$\frac{\partial E\hat{w}}{\partial \hat{w}} = 2X^T(X\hat{w} - y) + \lambda(I^T + I)\hat{w}$$

$$= 2X^T(X\hat{w} - y) + 2\lambda I \hat{w}$$

$$\text{令 } \frac{\partial E\hat{w}}{\partial \hat{w}} = 0$$

$$X^T(X\hat{w} - y) + \lambda I \hat{w} = 0$$

$$(X^T X + \lambda I) \hat{w} = X^T y$$

$$\hat{w} = (X^T X + \lambda I)^{-1} X^T y$$

一定满秩

向量求导公式:

$$\frac{\partial (X\theta)}{\partial \theta} = X^T$$

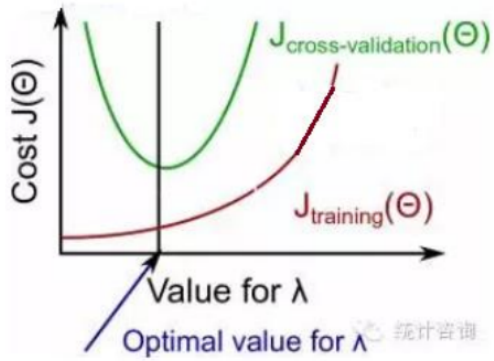
$$\frac{\partial (\theta^T X)}{\partial \theta} = X$$

$$\frac{\partial (\theta^T X \theta)}{\partial \theta} = (X^T + X)\theta$$

ans

有关 λ 选取

要选能让验证集损失最小的那个值



lasso regression

Least Absolute Shrinkage and Selection Operator (LASSO)

model: $f(\mathbf{x}_i) = w_1x_{i1} + w_2x_{i2} + \dots + w_dx_{id} + b$

vector model: $f(\mathbf{x}_i) = \mathbf{w}^T \mathbf{x}_i + b$

$f(\mathbf{x}) = \mathbf{X}\hat{\mathbf{w}}$

loss function: $E_{\hat{\mathbf{w}}} = (\mathbf{y} - \mathbf{X}\hat{\mathbf{w}})^T (\mathbf{y} - \mathbf{X}\hat{\mathbf{w}}) + \lambda \|\hat{\mathbf{w}}\|_1 \quad \lambda > 0$

$$\|\mathbf{w}\|_1 = |w_1| + |w_2| + \dots + |w_d| = \sum_{i=1}^d |w_i|$$

Lasso回归更容易得到稀疏解

稀疏解：“稀疏”指的是权重向量 $\mathbf{w}$ 中大部分元素是0，只有少数非零元素

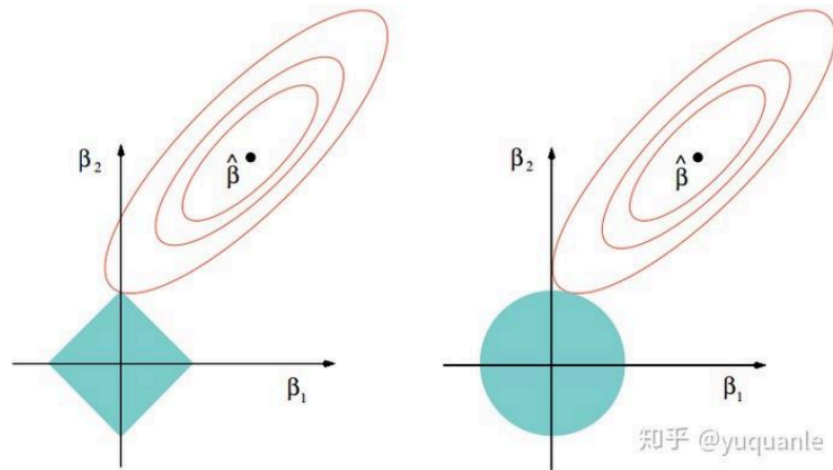
可以用来进行特征选择：

当某个特征对应的权重 $w_d = 0$ 时，意味着这个特征 x_{id} （第 i 个样本的第 d 个特征）对当前任务不重要

两者对比

不同点：lasso回归更容易得到稀疏解

lasso vs ridge regression



Lasso和岭回归的区别很好理解，在优化过程中，最优解为约束空间的交集，正则项可以看作是约束空间。可以看出二范的约束空间是一个球形，一范的约束空间是一个方形，这也就是二范会得到很多参数接近0的值，而一范会尽可能非零参数最少。

共同点：

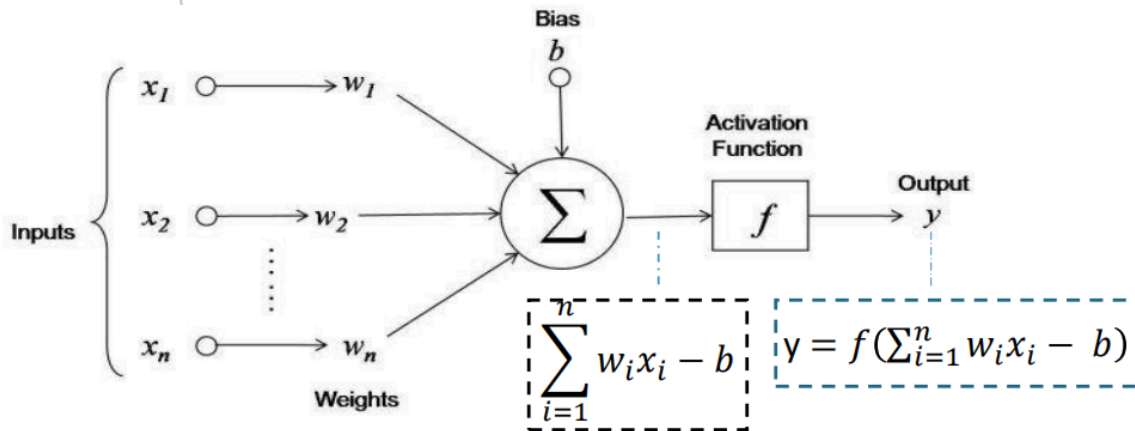
- ridge回归和lasso回归可以防止模型过拟合
- 解决样本的特征数远超过样本数的问题

08 Neural Networks and Deep Learning

人工神经网络

Artificial Neural Networks

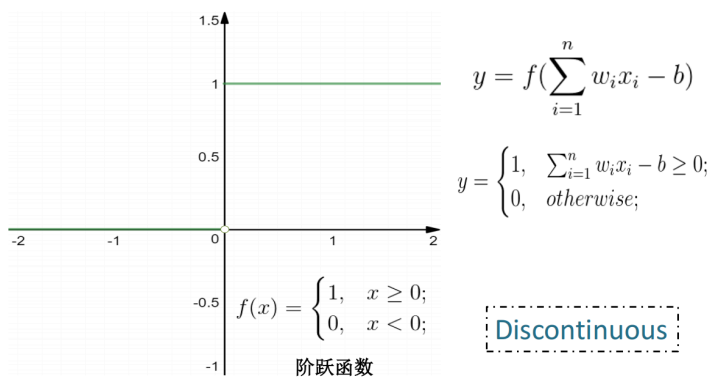
M-P Neuron Model



Activation Function

激活函数

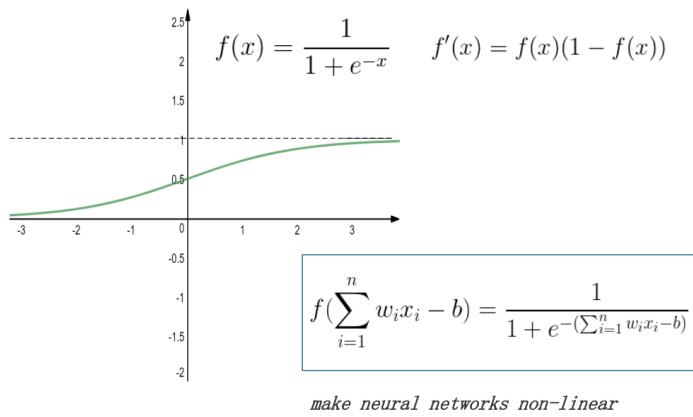
阶跃函数



M-P 神经元的工作流程是：

1. 输入 x_1 、 x_2 分别乘以对应的权重 w_1 、 w_2 ，得到加权和
2. 对加权和加上偏置项，再通过**激活函数**（感知机常用阶跃函数）处理；
3. 最终输出结果 y

Sigmoid function

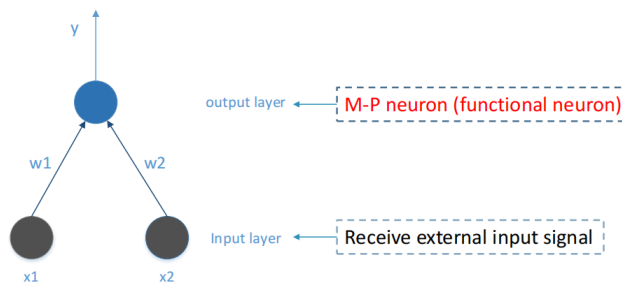


Perception

单层感知机

Single layer perceptrons

基本结构



输入层：负责接收外部输入的特征信号

输出层：由一个M-P神经元（功能神经元组成）

关键属性：

- 它是线性分类器，只能处理**线性可分**的数据集
- 收敛性保证：如果训练数据是线性可分的，感知机的学习算法（通过调整权重来最小化分类误差）**一定能收敛**，找到正确的分类边界

感知机的局限性：

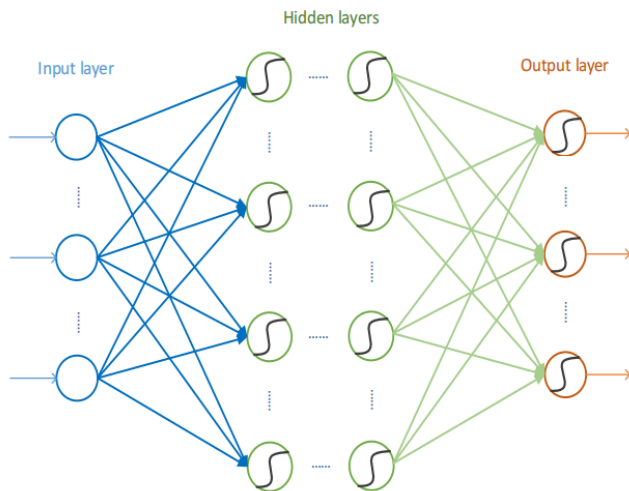
单层感知机仅能学习线性可分模式，学习无法处理非线性可分的数据

多层感知机

Multilayer Perceptron MLP

局限性：内存开销大、计算成本高

基本结构



输入层 (Input layer): 接收外部输入的特征 (图中蓝色节点), 每个节点对应一个特征维度

隐藏层 (Hidden layers): 位于输入层和输出层之间的中间层 (图中绿色节点), 是 MLP 的核心——感知机没有隐藏层, 而 **MLP 通过隐藏层实现非线性建模** (可以有 1 层或多层隐藏层)

输出层 (Output layer): 输出最终结果 (图中橙色节点), 比如分类任务的类别概率、回归任务的预测值

核心特性:

全连接 (Full connected): 每一层的每个神经元, 都会和上一层的所有神经元相连, 且每条连接都有独立的权重

前馈 (Feedforward): 信息在网络中**单向向前流动**——从输入层→隐藏层→输出层, 没有循环或反馈连接

类比复合函数思想

学习过程/训练过程

基本流程

核心: 通过训练数据迭代调整参数, 让模型预测结果接近真实标签

- 参数初始化
 - 随机初始化网络的参数
 - 包括所有连接的权重 ($w_1, w_2, \dots, w_n$)
 - 以及每个神经元的偏置 ($b_1, b_2, \dots, b_k$)

- 核心循环: 前向预测-计算误差-反向调参

1. **输入训练数据**: 把训练样本喂给网络的输入层

2. **前向传播 (Predict label)**: 数据从输入层→隐藏层→输出层, 得到模型的“预测标签”

3. **计算损失 (loss)**: 对比“预测标签”和“真实标签”的差异, 用损失函数量化这个差异 (差异越大, loss 值越高)
4. **调整参数 (Adjust parameters)**: 通过反向传播算法, 根据 loss 的大小, 迭代更新权重和偏置 (目标是让 loss 越来越小)

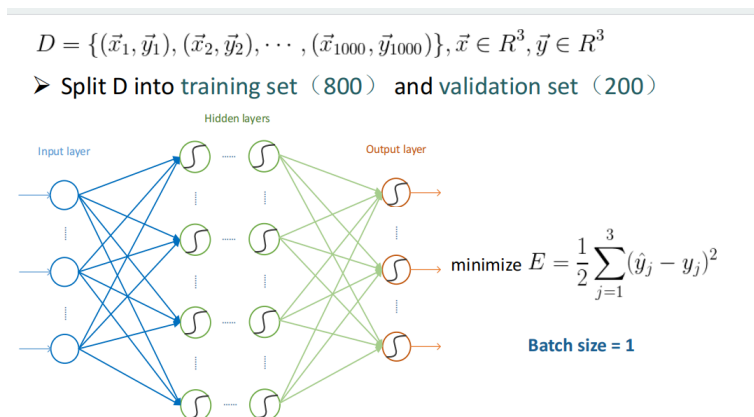
神经网络的学习过程就是基于训练数据, 不断调整神经元的连接权重、以及每个功能神经元的阈值, 最终让模型的结果尽可能贴合真实情况

核心概念

1. Epoch (训练轮次)
1 个 Epoch = 把**整个训练集完整过一遍**, 通常一个Epoch包含多个迭代
2. Batch (批次)
把训练集拆成若干“小份”, 每一份就是 1 个 Batch
3. Iteration (迭代)
1 个 Iteration = 用**1 个 Batch** 的数据训练一次

每完成 1 个 Epoch (遍历完整个训练集), 用**验证集评估当前训练效果**
重复多个Epoch, 直到模型在验证集上表现稳定 (“收敛”)

SGD (随机梯度下降)

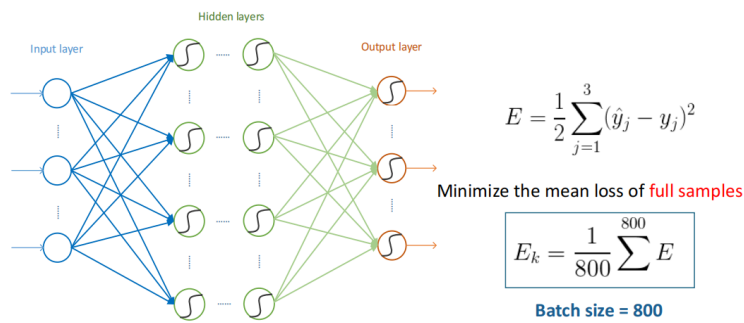


Batch size = 1: SGD 的特点是**每轮迭代只用 1 个样本来计算损失、更新参数**
迭代逻辑: 每次拿 1 个训练样本, 计算损失后立刻调整网络的权重和偏置
上例: 由于训练集有800个样本, 1个Epoch需要做800次参数更新

BGD (批量梯度下降)

$$D = \{(\vec{x}_1, \vec{y}_1), (\vec{x}_2, \vec{y}_2), \dots, (\vec{x}_{1000}, \vec{y}_{1000})\}, \vec{x} \in R^3, \vec{y} \in R^3$$

➤ Split D into training set (800) and validation set (200)



Batch size = 800: BGD 的特点是每轮迭代用“整个训练集”的所有样本来计算损失

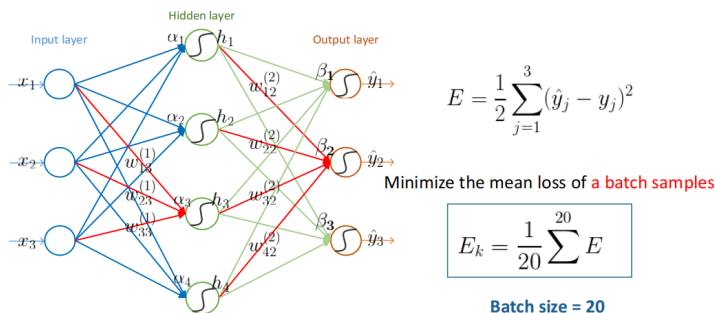
损失计算：先算每个样本的损失 E ，再求训练集所有样本的损失均值 $E_k = \frac{1}{800} \sum E$ ，目标是最小化这个“全样本平均损失”

上例：1 个 Epoch（遍历完整个训练集）只做**1 次参数更新**（用全训练集的损失均值调整权重和偏置）

Mini-batch Gradient Descent (小批量梯度下降)

$$D = \{(\vec{x}_1, \vec{y}_1), (\vec{x}_2, \vec{y}_2), \dots, (\vec{x}_{1000}, \vec{y}_{1000})\}, \vec{x} \in R^3, \vec{y} \in R^3$$

➤ Split D into training set (800) and validation set (200)



Batch size = 20: 每轮迭代用 **20 个样本组成的“小批量”** 计算损失

损失计算：先算该批量内每个样本的损失 E ，再求批量内的损失均值 $E_k = \frac{1}{20} \sum E$ ，目标是最小化这个批量的平均损失

上例：1 个 Epoch（遍历完整个训练集）需要做 $800 \div 20 = 40$ 次参数更新

更新参数

Gradient Descent

梯度下降

梯度下降的本质是：沿着函数下降最快的方向（负梯度），逐步调整参数，直到函数值最小化

Gradient Descent

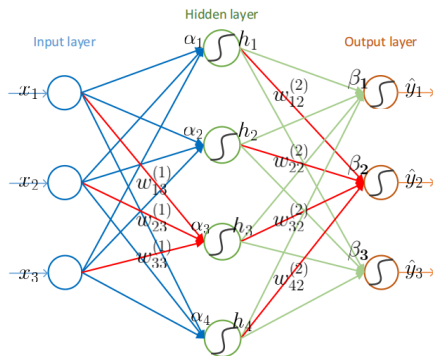
- What is gradient ?

For example: $f(\theta_0, \theta_1) \xrightarrow{\nabla} \left(\frac{\partial f}{\partial \theta_0}, \frac{\partial f}{\partial \theta_1} \right)^T$

- The gradient ∇f points in the direction of the greatest rate of increase of the function
- $-\nabla f$ points in the direction of the greatest rate of descent of the function

$$\text{minimize } f(\theta_0, \theta_1) \Rightarrow \boxed{\vec{\theta} \leftarrow \vec{\theta} - \eta \nabla f(\vec{\theta})} \Rightarrow \begin{aligned} \theta_0 &\leftarrow \theta_0 - \eta \frac{\partial f}{\partial \theta_0} \\ \theta_1 &\leftarrow \theta_1 - \eta \frac{\partial f}{\partial \theta_1} \end{aligned}$$

Update weights and bias



minimize loss function

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

Update Parameters:

$$\vec{w} \leftarrow \vec{w} - \eta \nabla E$$

$$\vec{b} \leftarrow \vec{b} - \eta \nabla E$$

反向传播

推导过程

假设使用均方差损失函数+sigmoid函数

如何避免陷入局部最小值

- 多组不同参数初始化
- 模拟退火
- 随机梯度下降

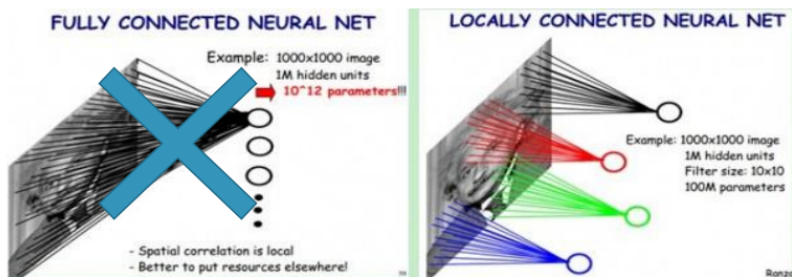
CNN

核心要素

1. Receptive Field

局部感受野

7 / 10



2. Parameter Sharing 权值共享



The same patterns appear in different regions



Sharing the same set of parameters

3. Pooling 池化

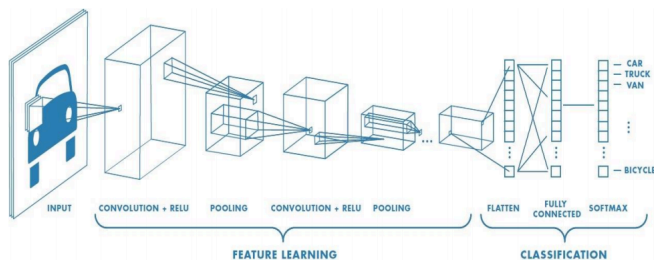
对像素进行抽样以缩小图像——图像的规模减少，但主要的特征依然保留

CNN结构 ☀

要求知道每一个模块是干什么的

采用三种主要类型的层来构建卷积神经网络架构

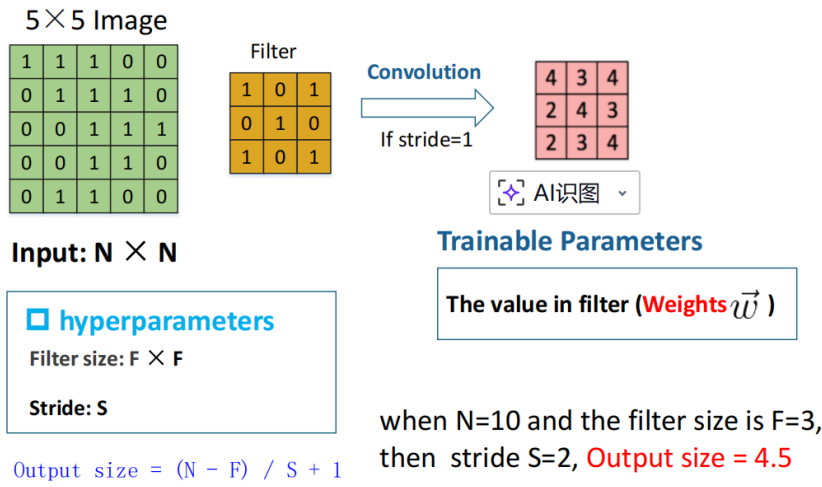
- 卷积层
- 池化层
- 全连接层



输入：与神经网络中输入为向量不同，此处输入为多通道图像

卷积层

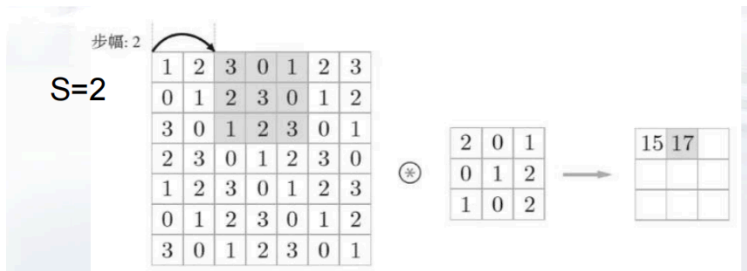
1. 卷积运算：
具体操作



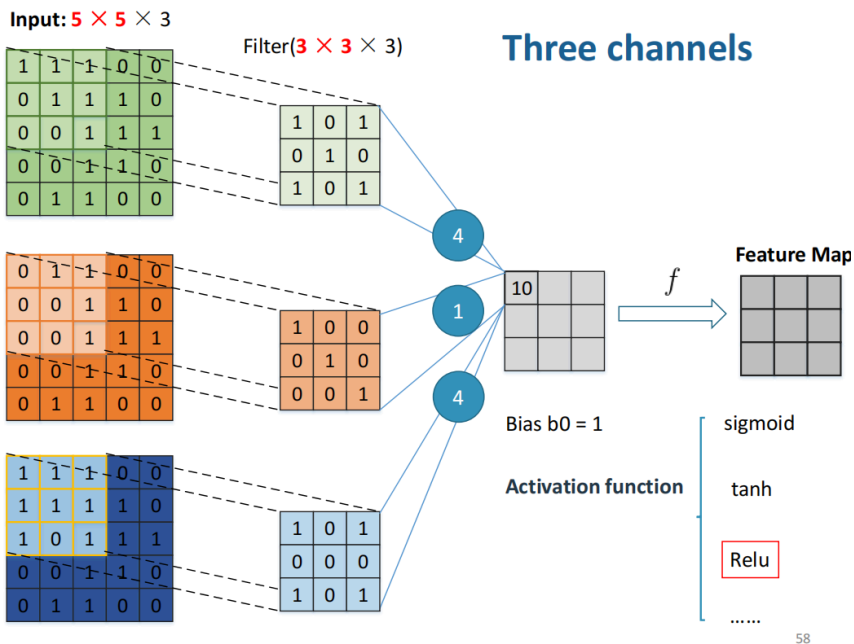
2. 填充padding

- 零补充 zero-padding
 - 在输入图像的边缘添加“0 值像素”
 - Output size = $(N+2P - F) / S + 1$ (此处P指表示在输入图像的每个边缘(上下左右)添加的 0 值层数)

3. 步幅stride



4. 多通道卷积操作



5. 作用: [卷积神经网络 \(CNN\) 中, 卷积层的作用是什么 卷积层的主要作用-CSDN博客](#)

池化层

池化层是缩小高、长方向的运算

mean pooling / max pooling

3	1	1	0
0	1	1	1
2	0	4	1
0	0	1	1

Stride = 2

Filter = 2×2

 hyperparameters

Filter size: $F \times F$

Stride: S

Output size = $(N - F) / S + 1$

Max Pooling

3	1
2	4

Mean Pooling

$5/4$	$3/4$
$1/2$	$7/4$

No Trainable Parameters !

特点:

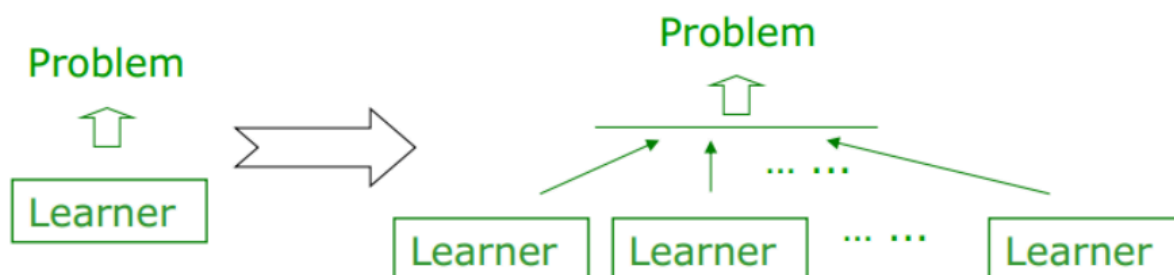
- 特征不变形：池化操作是模型更加关注是否存在某些特征而不是特征具体的位置
- 特征降维：池化相当于在空间范围内做了维度约减，从而使模型可以抽取更加广范围的特征，同时减小了下一层的输入大小，进而减少计算量和参数个数
- 在一定程度上防止过拟合，更方便优化
- 增大感受野

09 Ensemble Learning

定义

定义：集成学习通过构建并结合多个学习器来完成学习任务

一般结构：先产生一组“个体学习器”，再由某种策略将它们结合起来，个体学习器通常由一个现有的学习算法从训练数据产生



- 同质集成
 - 集成中只包含同种类型的个体学习器
 - 同质集成中的个体学习器也称为基学习器，相应的学习算法称为基学习算法
- 异质集成
 - 集成中包含不同类型的个体学习器
 - 弱学习器：学习器的泛化性能优于随机猜测

要获得好的集成，个体学习器应好而不同，即个体学习器要有一定的准确性（学习器不能太坏）且要有多样性（学习器间具有差异）

集成学习方法

如何生成基学习器

分类

- 个体学习器间存在强依赖关系，必须串行生成的序列化方法
 - Boosting
- 个体学习器间不存在强依赖关系，可同时生成的并行化方法
 - Bagging和随机森林

并行方法

Bagging

Bootstrap aggregating

基于自助采样法获取与等量的不同训练集

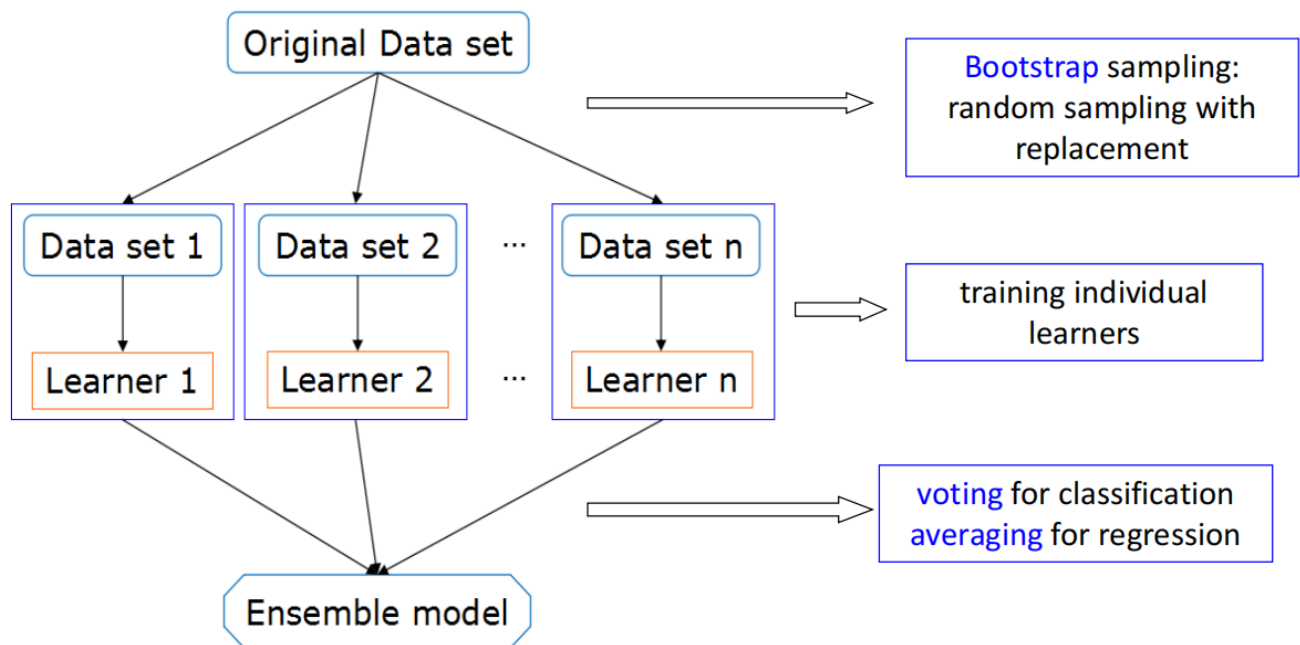
图

可看出, 它直接基于我们在 2.2.3 节介绍过的自助采样法 (bootstrap sampling). 给定包含 m 个样本的数据集, 我们先随机取出一个样本放入采样集中, 再把该样本放回初始数据集, 使得下次采样时该样本仍有可能被选中, 这样, 经过 m 次随机采样操作, 我们得到含 m 个样本的采样集, 初始训练集中有的样本在采样集里多次出现, 有的则从未出现. 由式(2.1)可知, 初始训练集中约有 63.2% 的样本出现在采样集中.

每个训练子集中仅使用原始训练集的63.2%样本, 因此可将剩余的36.8%样本用作验证集。

依据自助采样法可采样出 T 个含 m 个训练样本的采样集, 基于每个采样集训练出一个基学习器, 再将这些基学习器进行结合

在对预测输出进行结合时, Bagging通常对分类任务使用简单投票法, 对回归任务使用简单平均法



Bagging特点

- 能不加修改地应用于多分类、回归等任务
- 主要关注降低方差, 因此在不剪枝决策树、神经网络等易受样本干扰的学习器上效用更为明显
- 多样性仅通过样本扰动

随机森林

Bagging的一个扩展变形

特点:

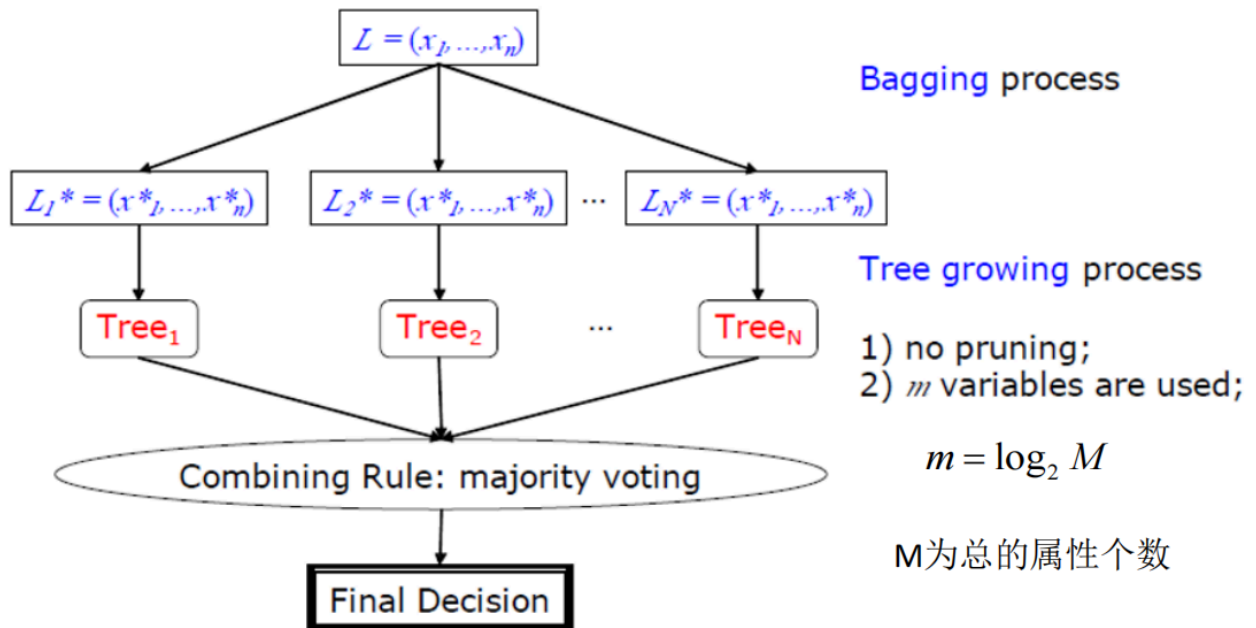
- 基学习器: 决策树

- 又放回构造训练集
- 随机选取 m 个特征构建决策树
- 多样性不仅来自于样本扰动还来自属性扰动

随机森林在以决策树为基学习器构建Bagging集成的基础上，进一步在决策树的训练过程中引入了随机属性选择

与传统的决策树不同：

- 引入随机性质扰动
- 每棵树都未经剪枝

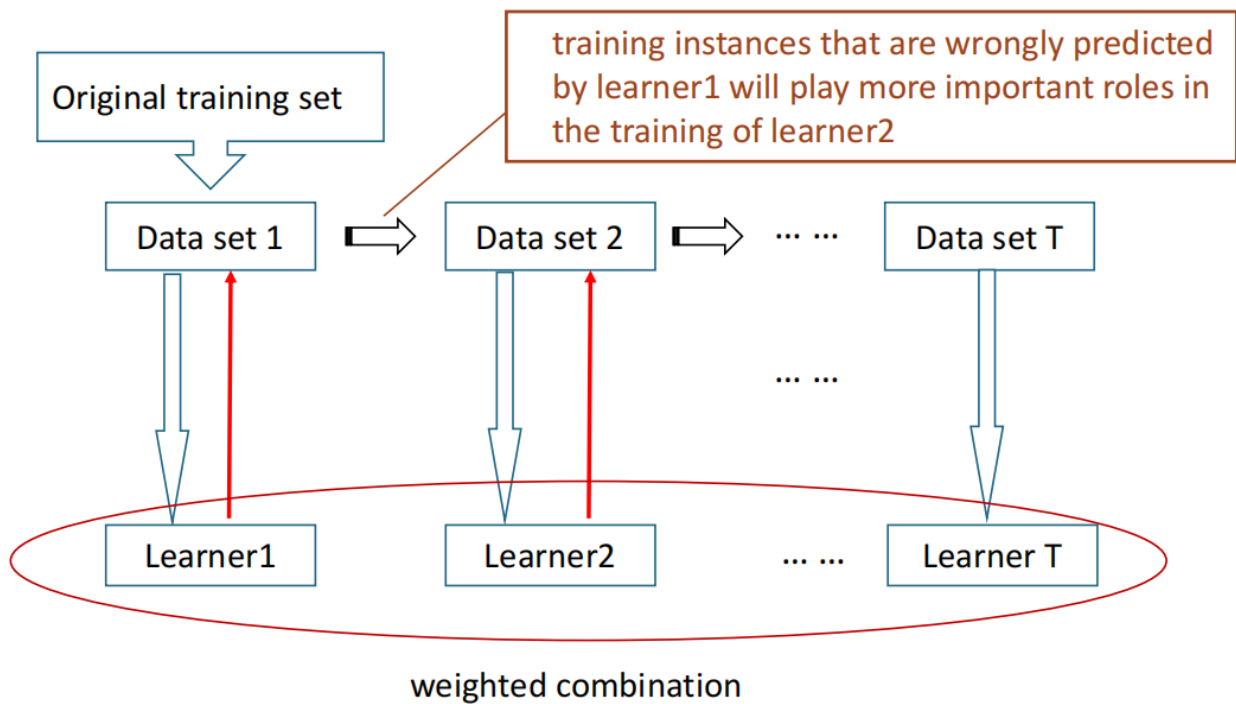


串行方法

AdaBoost

Boosting是一族可将弱学习器提升为强学习器的算法

工作机制：先从初始训练集训练出一个基学习器，再根据基学习器的表现对训练样本分布进行调整，使得先前基学习器做错的训练样本在后续受到更多的关注，然后基于调整后的样本分布来训练下一个基学习器，如此重复进行，直到基学习器的数目达到事先指定的 T ，最终将这 T 个基学习器加权结合



Bagging vs Boosting

Bagging

- 核心特点:

- 基学习器之间相互独立;
- 基学习器是并行生成的;
- 采用 Bootstrap (自助采样) 方式获取训练数据;
- 作用: 降低模型的方差。

AdaBoost

- 核心特点:

- 基学习器之间存在依赖关系;
- 基学习器是按序列生成的;
- 通过调整样本的权重分布 (错分样本权重升高) 优化学习
- 作用: 降低模型的偏差 (缓解过拟合)

结合策略

优点

- 从统计的方面来看, 由于学习任务的假设空间往往很大, 可能有多个假设在训练集上达到同等性能, 此时若使用单学习器可能因误选而导致泛化性能不佳, 结合多个学习器则会减小这一风险
- 从计算的方面来看, 学习算法往往会陷入局部极小, 有的局部极小点所对应的泛化性能可能很糟糕, 而通过多次运行之后进行结合, 可降低陷入糟糕局部极小点的风险
- 从表示的方面来看, 某些学习任务的真实假设可能不在当前学习算法所考虑的假设空间中, 此时若使用单学习器则肯定无效, 而通过结合多个学习器, 由于相应的假设空间有所扩大, 有可能学得更好的近似

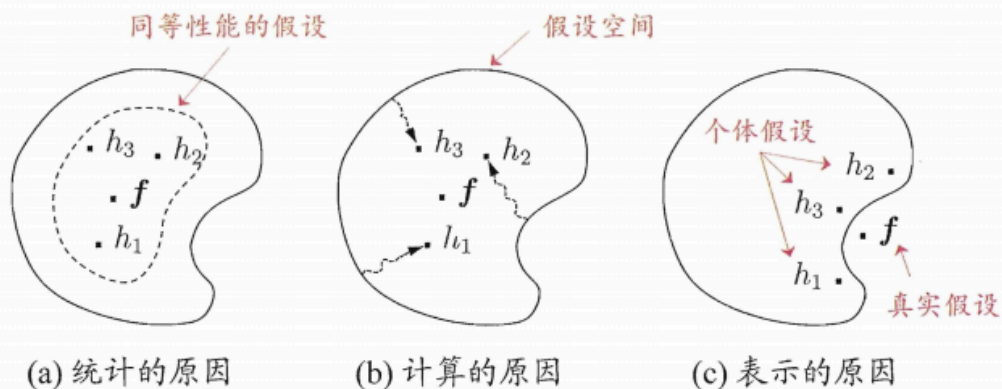


图 8.8 学习器结合可能从三个方面带来好处 [Dietterich, 2000]

平均法

对数值型输出，最常见的结合策略是平均法

- **simple averaging**

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

- **weighted averaging**

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

$$w_i \geq 0, \sum_{i=1}^T w_i = 1$$

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，个体学习器性能相近时宜使用简单平均法

投票法

学习器 h_i 将从类别标记集合 $\{c_1, c_2, \dots, c_N\}$ 中预测出一个标记, 我们将 h_i 在样本 $\mathbf{x}$ 上的预测输出表示为一个 N 维向量 $(h_i^1(\mathbf{x}); h_i^2(\mathbf{x}); \dots; h_i^N(\mathbf{x}))$, 其中 $h_i^j(\mathbf{x})$ 是 h_i 在类别标记 c_j 上的输出.

- **majority voting** (绝对多数投票法)

$$H(\mathbf{x}) = \begin{cases} c_j & \text{if } \sum_{i=1}^T h_i^j(\mathbf{x}) > \frac{1}{2} \sum_{k=1}^l \sum_{i=1}^T h_i^k(\mathbf{x}) \\ \text{rejection} & \text{otherwise.} \end{cases}$$

- **plurality voting** (相对多数投票法)

$$H(\mathbf{x}) = c_{\arg \max_j \sum_{i=1}^T h_i^j(\mathbf{x})}$$

- **weighted voting** (加权投票法)

$$H(\mathbf{x}) = c_{\arg \max_j \sum_{i=1}^T w_i h_i^j(\mathbf{x})}$$

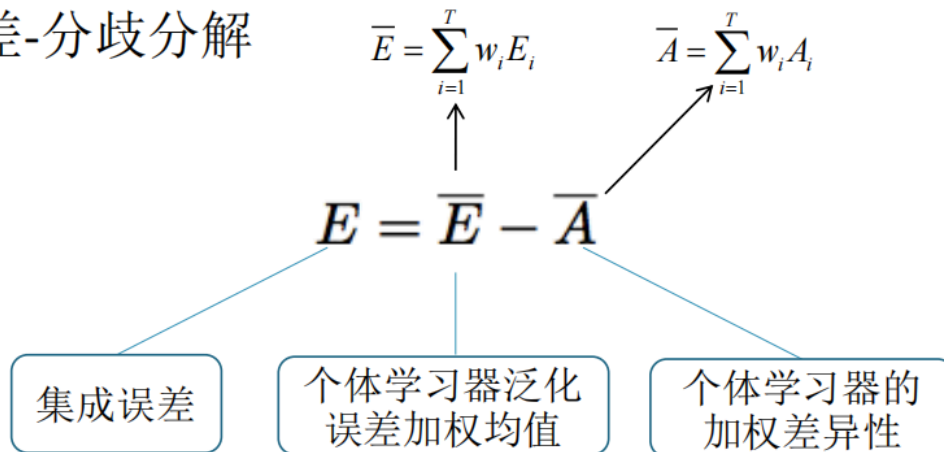
式(8.24)~(8.26)没有限制个体学习器输出值的类型. 在现实任务中, 不同类型个体学习器可能产生不同类型的 $h_i^j(\mathbf{x})$ 值, 常见的有:

- 类标记: $h_i^j(\mathbf{x}) \in \{0, 1\}$, 若 h_i 将样本 $\mathbf{x}$ 预测为类别 c_j 则取值为 1, 否则为 0. 使用类标记的投票亦称“硬投票”(hard voting).
- 类概率: $h_i^j(\mathbf{x}) \in [0, 1]$, 相当于对后验概率 $P(c_j | \mathbf{x})$ 的一个估计. 使用类概率的投票亦称“软投票”(soft voting).

多样性

Diversity

• 误差-分歧分解



个体学习器的准确率越高，多样性越大，则集成越好。

多样性度量

度量集成中个体分类器的多样性，估算个体学习器的多样化程度

• Disagreement Measure（不合度量）

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

dis_{ij} 的值域为 $[0, 1]$. 值越大则多样性越大.

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

• Correlation Coefficient（相关系数）

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

ρ_{ij} 的值域为 $[-1, 1]$. 若 h_i 与 h_j 无关, 则值为 0; 若 h_i 与 h_j 正相关则值为正, 否则为负.

ρ 的绝对值越大，两个分类器的相关性越大，反之，相关性越小。

Diversity Measure

- Q-Statistic (Q-统计量)

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

$$|Q_{ij}| \leq |\rho_{ij}|$$

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

- Kappa-Statistic (K-统计量)

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

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

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

p_1 : 两个分类器取得一致的概率

p_2 : 两个分类器偶然达成一致的概率

κ 越大, 相关性越大, 反之, 相关性越小。

多样性增强

在集成学习中需有效的生成多样性大的个体学习器, 一般思路是在学习过程中引入随机性

- 数据样本扰动
给定初始数据集, 可从中产生出不同的数据子集, 再利用不同的数据子集训练出不同的个体学习器。数据样本扰动通常是基于采样法: Boosting、Bagging
对于不稳定基学习器 (决策树、神经网络...) 很有效。
- 输入属性扰动
对于稳定基学习器 (LDA、SVM、KNN...), 从初始属性集中抽取出若干个属性子集, 基于每个属性子集训练一个基学习器。对包含大量冗余属性的数据, 在子空间中训练个体学习器不仅能产生多样性大的个体, 还会因属性数的减少而节省时间开销
若数据仅包含少量属性或冗余属性很少, 则不宜使用输入属性扰动法
- 输出表示扰动
将原任务拆解为多个可同时求解的子任务或者是将分类输出转化为回归输出后构建个体学习器
- 算法参数扰动
基学习算法一般都有参数需要设置, 例如神经网络的神经元数, 初始连接权值等, 通过随机设置不同的参数。往往可产生差别较大的个体学习器

10 Clustering

定义

- 无监督学习任务
- 在无监督学习中，训练样本的标记信息是未知的，目标是通过对无标记训练样本的学习来揭示数据的内在性质及规律
- 聚类试图将数据集中的样本划分为若干个通常不相交的子集，每个子集称为一个簇

定义：聚类是指将一组对象进行分组的任务，使得同一组内的对象彼此之间的相似度高于与其他组内对象的相似度（组间相似度）

组内相似，组间相似

距离计算

相似性度量：相似性度量用于量化两个对象之间的相似程度。常用方法是计算样本间的距离
距离计算

- 距离越大，相似性越小
- 距离在某种程度上是相似度的倒数

□ Minkowski distance

$$dist_{mk}(x_i, x_j) = \left(\sum_{u=1}^n |x_{iu} - x_{ju}|^p \right)^{\frac{1}{p}}$$

where, $x_i = (x_{i1}; x_{i2}; \dots x_{in})$, $x_j = (x_{j1} \ x_{j2}; \dots x_{jn})$

□ Euclidean distance (p=2)

□ Manhattan distance (p=1)

$x_1 = [2, 1]$	$\xrightarrow{\text{Manhattan distance}}$	$d = (2-3 ^1 + 1-3 ^1)^1 = 3$
$x_2 = [3, 3]$	$\xrightarrow{\text{Euclidean distance}}$	$d = (2-3 ^2 + 1-3 ^2)^{1/2} = \sqrt{5}$

算法

K-means

核心步骤:

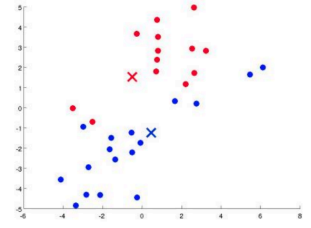
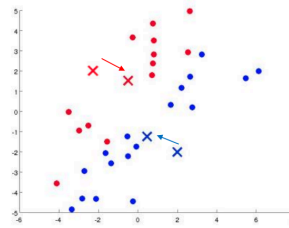
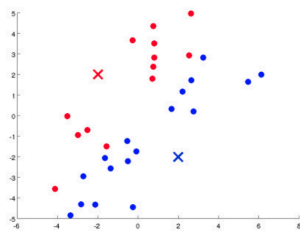
- 随机选择K个样本作为聚类的初始质心
- 计算每个样本到这K个质心的距离
- 选择距离最小的聚类质心 $c_i (i=1, 2 \dots K)$, 将样本划分至对应簇
- 重新确定各簇的质心 (通常为簇内样本的均值)
- 迭代上述步骤, 直至算法收敛 (质心变化小于阈值)

- For all green samples, compute distances from the blue point and red point.
- Choose the smallest distance and assign it to one cluster.

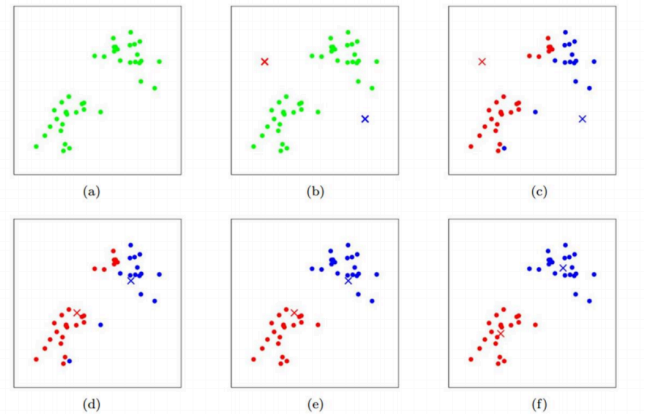
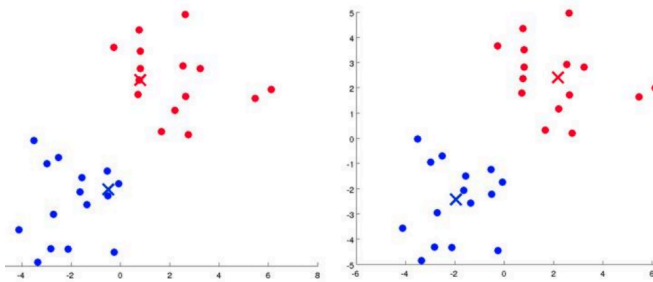
- Compute the average for two clusters.
- Re-determine cluster centroids.

移动聚类中心

簇划分



Iterate until convergence (the cluster centroids will not change).



输入: 样本集 $D = \{x_1, x_2, \dots, x_m\}$;
聚类簇数 k .

过程:

```

1: 从  $D$  中随机选择  $k$  个样本作为初始均值向量  $\{\mu_1, \mu_2, \dots, \mu_k\}$ 
2: repeat
3:   令  $C_i = \emptyset$  ( $1 \leq i \leq k$ )
4:   for  $j = 1, 2, \dots, m$  do
5:     计算样本  $x_j$  与各均值向量  $\mu_i$  ( $1 \leq i \leq k$ ) 的距离:  $d_{ji} = \|x_j - \mu_i\|_2$ ;
6:     根据距离最近的均值向量确定  $x_j$  的簇标记:  $\lambda_j = \arg \min_{i \in \{1, 2, \dots, k\}} d_{ji}$ ;
7:     将样本  $x_j$  划入相应的簇:  $C_{\lambda_j} = C_{\lambda_j} \cup \{x_j\}$ ;
8:   end for
9:   for  $i = 1, 2, \dots, k$  do
10:    计算新均值向量:  $\mu'_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$ ;
11:    if  $\mu'_i \neq \mu_i$  then
12:      将当前均值向量  $\mu_i$  更新为  $\mu'_i$ 
13:    else
14:      保持当前均值向量不变
15:    end if
16:  end for
17: until 当前均值向量均未更新
输出: 簇划分  $C = \{C_1, C_2, \dots, C_k\}$ 

```

图 9.2 k 均值算法

注意:

有关初始化: 由于初始化方式的不同, 每个结果均存在差异

- ❑ We need all possible initializations and get the best result.
- ❑ The measure to find the best result is **minimizing square error E(SSE, sum of the Squared Error)**.

$$E = \sum_{i=1}^k \sum_{x \in C_i} \|x - u_i\|_2^2 \quad \text{or} \quad E = \sum_{i=1}^m \|x_i - u_{\lambda_i}\|_2^2$$

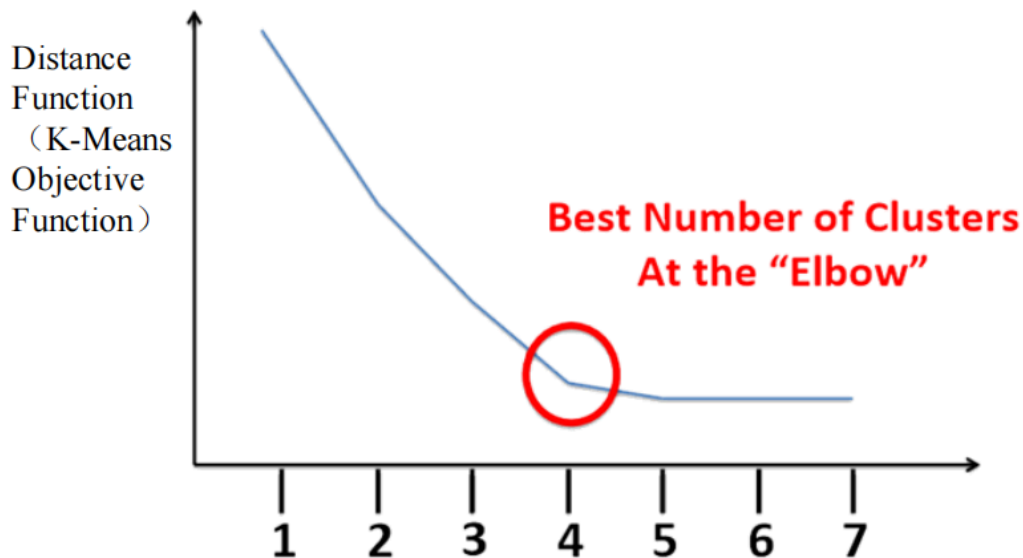
$$u_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

怎样选择 k

肘部法则:

绘制“簇数 K - 距离函数 (K-Means 目标函数, 通常为簇内样本到质心的距离平方和)”的曲线, 曲线中“肘部”对应的 K 即为最优簇数

Comparing multiple clusters “The Elbow Method”



高斯混合聚类

1. 高斯混合分布的定义

高斯混合分布是**多个高斯分布的加权组合**，用于描述数据的概率分布：

- 公式：

$$p_m(x) = \sum_{i=1}^k \alpha_i p(x|\mu_i, \Sigma_i), \quad \sum_{i=1}^k \alpha_i = 1$$

其中：

- $p(x|\mu_i, \Sigma_i)$ ：第*i*个高斯分量的概率密度函数；
- μ_i 、 Σ_i ：第*i*个高斯分量的均值向量、协方差矩阵；
- α_i ：第*i*个高斯分量的混合系数（权重），满足所有 α_i 之和为 1。

2. 目标函数（后验概率）

GMM 通过**后验概率**判断样本属于某一高斯分量的概率：

$$p_m(z_j = i|x_j) = \frac{p_m(x_j|z_j=i) \cdot p(z_j=i)}{p(x_j)} = \frac{\alpha_i \cdot p_m(x_j|\mu_i, \Sigma_i)}{\sum_{l=1}^k \alpha_l \cdot p_m(x_j|\mu_l, \Sigma_l)}$$

- 含义：样本 x_j 属于第*i*个高斯分量（簇）的概率，是 GMM 实现“软聚类”的核心（样本不是被硬性划分到某一簇，而是得到属于各簇的概率）。

DBSCAN

基于密度的聚类方法，此类算法假设聚类结构能通过样本分布的紧密程度确定

- ϵ -邻域: 对 $\mathbf{x}_j \in D$, 其 ϵ -邻域包含样本集 D 中与 $\mathbf{x}_j$ 的距离不大于 ϵ 的样本, 即 $N_\epsilon(\mathbf{x}_j) = \{\mathbf{x}_i \in D \mid \text{dist}(\mathbf{x}_i, \mathbf{x}_j) \leq \epsilon\}$;

- 核心对象(core object): 若 $\mathbf{x}_j$ 的 ϵ -邻域至少包含 $MinPts$ 个样本, 即 $|N_\epsilon(\mathbf{x}_j)| \geq MinPts$, 则 $\mathbf{x}_j$ 是一个核心对象;
- 密度直达(directly density-reachable): 若 $\mathbf{x}_j$ 位于 $\mathbf{x}_i$ 的 ϵ -邻域中, 且 $\mathbf{x}_i$ 是核心对象, 则称 $\mathbf{x}_j$ 由 $\mathbf{x}_i$ 密度直达;
- 密度可达(density-reachable): 对 $\mathbf{x}_i$ 与 $\mathbf{x}_j$, 若存在样本序列 $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n$, 其中 $\mathbf{p}_1 = \mathbf{x}_i$, $\mathbf{p}_n = \mathbf{x}_j$ 且 $\mathbf{p}_{i+1}$ 由 $\mathbf{p}_i$ 密度直达, 则称 $\mathbf{x}_j$ 由 $\mathbf{x}_i$ 密度可达;
- 密度相连(density-connected): 对 $\mathbf{x}_i$ 与 $\mathbf{x}_j$, 若存在 $\mathbf{x}_k$ 使得 $\mathbf{x}_i$ 与 $\mathbf{x}_j$ 均由 $\mathbf{x}_k$ 密度可达, 则称 $\mathbf{x}_i$ 与 $\mathbf{x}_j$ 密度相连.

基于以上概念, DBSCAN将簇定义为由密度可达关系导出的最大密度相连样本

密度相连样本集合. 形式化地说, 给定邻域参数 $(\epsilon, MinPts)$, 簇 $C \subseteq D$ 是满足以下性质的非空样本子集:

$$\text{连接性(connectivity): } \mathbf{x}_i \in C, \mathbf{x}_j \in C \Rightarrow \mathbf{x}_i \text{ 与 } \mathbf{x}_j \text{ 密度相连} \quad (9.39)$$

$$\text{最大性(maximality): } \mathbf{x}_i \in C, \mathbf{x}_j \text{ 由 } \mathbf{x}_i \text{ 密度可达} \Rightarrow \mathbf{x}_j \in C \quad (9.40)$$

找出所有核心对象

随机选一个核心对象生长出一个簇，并在核心对象集合里删去该核心对象

```

输入: 样本集  $D = \{x_1, x_2, \dots, x_m\}$ ;
      邻域参数  $(\epsilon, MinPts)$ .
过程:
1: 初始化核心对象集合:  $\Omega = \emptyset$ 
2: for  $j = 1, \dots, m$  do
3:   确定样本  $x_j$  的  $\epsilon$ -邻域  $N_\epsilon(x_j)$ ;
4:   if  $|N_\epsilon(x_j)| \geq MinPts$  then
5:     将样本  $x_j$  加入核心对象集合:  $\Omega = \Omega \cup \{x_j\}$ 
6:   end if
7: end for
8: 初始化聚类簇数:  $k = 0$ 
9: 初始化未访问样本集合:  $\Gamma = D$ 
10: while  $\Omega \neq \emptyset$  do
11:   记录当前未访问样本集合:  $\Gamma_{old} = \Gamma$ ;
12:   随机选取一个核心对象  $o \in \Omega$ , 初始化队列  $Q = \langle o \rangle$ ;
13:    $\Gamma = \Gamma \setminus \{o\}$ ;
14:   while  $Q \neq \emptyset$  do
15:     取出队列  $Q$  中的首个样本  $q$ ;
16:     if  $|N_\epsilon(q)| \geq MinPts$  then
17:       令  $\Delta = N_\epsilon(q) \cap \Gamma$ ;
18:       将  $\Delta$  中的样本加入队列  $Q$ ;
19:        $\Gamma = \Gamma \setminus \Delta$ ;
20:     end if
21:   end while
22:    $k = k + 1$ , 生成聚类簇  $C_k = \Gamma_{old} \setminus \Gamma$ ;
23:    $\Omega = \Omega \setminus C_k$ 
24: end while
25: return 簇划分结果
输出: 簇划分  $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$ 

```

优点:

- **-无需指定簇数K:**
DBSCAN 通过“密度”自动识别簇的数量，无需像 K-Means 那样人工预设K。
- **可发现任意形状的空间聚类:**
相比 K-Means（仅能识别球形簇），DBSCAN 能拟合不规则形状的簇（如环形、带状等）。
- **可剔除离群点（噪声）:**
远离高密度区域的样本会被标记为噪声，不会被强行划分到某一簇中

缺点:

- **不适用于簇密度差异大的场景:**
DBSCAN 依赖全局的“密度参数（邻域半径 ϵ 、最小点数 $MinPts$ ）”，若不同簇的密度差异大，无法同时适配所有簇。
- **参数调整较复杂:**
 ϵ 和 $MinPts$ 的选择对聚类结果影响极大，且无通用的最优参数，需结合数据分布反复调试

AGNES

层次聚类

AGNES采用自底向上聚合策略的层次聚类方法，先将数据集中每个样本看作一个初始的聚类簇，然后在算法运行的每一步找出距离最近的两个聚类簇进行合并，该过程不断重复，直到达到预设的聚类簇个数

$$\text{最小距离: } d_{\min}(C_i, C_j) = \min_{\mathbf{x} \in C_i, \mathbf{z} \in C_j} \text{dist}(\mathbf{x}, \mathbf{z}) , \quad (9.41)$$

$$\text{最大距离: } d_{\max}(C_i, C_j) = \max_{\mathbf{x} \in C_i, \mathbf{z} \in C_j} \text{dist}(\mathbf{x}, \mathbf{z}) , \quad (9.42)$$

$$\text{平均距离: } d_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{\mathbf{x} \in C_i} \sum_{\mathbf{z} \in C_j} \text{dist}(\mathbf{x}, \mathbf{z}) . \quad (9.43)$$

11 数据预处理

概念

噪声：原始值的修改，测量变量中的随机误差或波动

离群点：数据集中具有显著不同于大多数其他数据对象的特征的数据对象

缺失值：

- 信息未收集
- 属性可能不适用于所有情况
- 重复数据：数据集可能包含彼此重复或近乎重复的数据对象
- 从异构数据源合并数据时出现的主要问题

数据质量

为什么要对数据进行预处理

- **准确性**：数据是否正确、精准。
- **完整性**：数据是否存在未记录、不可用等缺失情况。
- **一致性**：数据是否存在部分修改、部分未修改（或存在无效关联）等不一致问题。
- **时效性**：数据是否得到及时更新？
- **可信度**：数据的正确性有多高的可信赖度？
- **可解释性**：数据的易懂程度如何？

主要任务

- 数据清洗
 - 填补缺失值、平滑噪声数据、识别 / 移除离群点、解决数据不一致问题
- 数据集成
 - 整合多个数据库、数据立方体或文件中的数据
- 数据约简
 - 维度约简（降低特征维度）
 - 数量约简（减少数据量）
 - 数据压缩（压缩数据体积）
- 数据变换与数据离散化
 - 归一化（将数据缩放到统一范围）
 - 概念分层生成（将低层次数据抽象为高层次概念，如将“年龄”分为“青年 / 中年 / 老年”）

数据清洗

原因

- **数据不完整**：缺失属性值、缺少目标关注的某些属性，或仅包含聚合数据
 - **忽略包含缺失值的记录**：通常用于分类任务中标签缺失的情况，但当各属性的缺失率差异较大时，这种方法效果不佳
 - **手动填补缺失值**：繁琐且难以实现（大规模数据下不可行）。
 - **自动填补缺失值**：
 - 全局常量：例如用“unknown”或新增一个类别填充
 - 属性均值：用该属性所有样本的均值填充
 - 同类别属性均值：用该样本所属类别的属性均值填充（更智能）
 - 最可能值：基于推理（如贝叶斯公式、决策树）得到的预测值
 - 利用已有数据的分布特征来推断缺失值
 - 聚合函数（均值、众数）
 - 概率估计（结合关联特征）
- **数据含噪声**：包含噪声、错误或异常值
 - 可能的原因
 - 数据采集设备故障
 - 数据录入问题
 - 数据传输问题
 - 技术局限性
 - 命名规范不一致
 - 处理
 - 分箱法
 - 先对数据排序，再划分为等频分箱
 - 之后可通过“箱均值平滑”“箱中位数平滑”“箱边界平滑”等方式处理数据
 - 回归法
 - 通过将数据拟合到回归函数中实现平滑
 - 聚类法
 - 识别并移除离群点
 - 人机结合检查法
 - 识别可疑值并由人工核查

补充平滑：

平滑（Smoothing）是指消除数据中的噪声、异常波动或随机误差，让数据更接近其真实的趋势或分布

- **数据不一致**：编码或名称存在矛盾
- **人为刻意问题**：伪装的缺失数据

数据离散化

简单离散化方法：分箱法

1. 等宽划分：

- 将数据范围划分为N个宽度相等的区间（均匀网格）
- 若A和B是属性的最小值与最大值，区间宽度为： $W=(B-A)/N$ ；
- 最直观，但异常值可能主导区间分布，且不适合偏态数据

2. 等频划分：

- 将数据范围划分为N个区间，每个区间包含的样本数量大致相等
- 数据缩放效果好，但处理分类属性时较复杂

eg:

- Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34
- * Partition into equal-frequency (equi-depth) bins:
 - Bin 1: 4, 8, 9, 15
 - Bin 2: 21, 21, 24, 25
 - Bin 3: 26, 28, 29, 34
- * Smoothing by bin means:
 - Bin 1: 9, 9, 9, 9
 - Bin 2: 23, 23, 23, 23
 - Bin 3: 29, 29, 29, 29
- * Smoothing by bin boundaries:
 - Bin 1: 4, 4, 4, 15
 - Bin 2: 21, 21, 25, 25
 - Bin 3: 26, 26, 26, 34

数据集成

定义：将多源数据合并为一个统一的存储库，将分散在不同来源的数据拼接成可用的整体
核心任务和挑战：

- 模式集成
 - 整合不同来源的元数据
- 实体识别
 - 从多源数据中识别现实世界的同一实体
- 检测 and 解决数据值冲突
 - 同一现实实体在不同数据源中的属性值存在差异
 - 可能原因：表示方式不同、计量尺度不同

数据集成中的冗余处理

- **对象识别**：同一属性或对象在不同数据库中可能有不同名称
- **可推导数据**：某一属性可能是另一表中的“推导属性”
- 冗余属性可通过**相关性分析与协方差分析**检测
 - 核心方法
 - 卡法检验

- **分类数据的相关性分析**

- **卡方 (χ^2) 检验**

$$\text{公式: } \chi^2 = \sum \frac{(\text{实际值} - \text{期望值})^2}{\text{期望值}}$$

- χ^2 值越大，变量间越可能存在关联；
- 对 χ^2 值贡献最大的是“实际计数与期望计数差异显著”的单元格；
- 相关性不代表因果性：
 - 示例：城市的医院数量与汽车盗窃数量存在相关性；
 - 两者实际都与“人口”这一第三方变量存在因果关联。

- 皮尔逊相关系数

- **相关系数（又称皮尔逊积矩系数）**

公式：

$$r_{A,B} = \frac{\sum_{i=1}^n (a_i - \bar{A})(b_i - \bar{B})}{(n-1)\sigma_A\sigma_B} = \frac{\sum_{i=1}^n (a_i b_i) - n\bar{A}\bar{B}}{(n-1)\sigma_A\sigma_B}$$

其中： n 是样本数量， $\bar{A}$ 和 $\bar{B}$ 分别是属性 A 和 B 的均值， σ_A 和 σ_B 分别是属性 A 和 B 的标准差， $\sum(a_i b_i)$ 是 A 与 B 的交叉乘积和。

- 若 $r_{A,B} > 0$ ： A 与 B 正相关（ A 的值随 B 的增大而增大），数值越大，相关性越强；
- 若 $r_{A,B} = 0$ ： A 与 B 相互独立；
- 若 $r_{A,B} < 0$ ： A 与 B 负相关。

数据约简

定义：得到数据集的简化表示 —— 其数据量大幅减少，但能产生与原数据集相同（或几乎相同）的分析结果

- **为何需要数据约简？**：数据库 / 数据仓库可能存储数 TB 级数据，在完整数据集上运行复杂数据分析会耗费大量时间
- **数据约简策略**：
 1. **维度约简**：例如移除不重要的属性，包含：

- 小波变换
- 主成分分析 (PCA)
 - 思想：用少数的主成分替代原始的高维特征，同时保留数据的主要信息
 - 找到一个投影方向，以捕捉数据中最大的变异信息
 - 将原始数据投影到一个维度远更小的空间中，实现维度约简
 - 步骤
 - 给定N个n维数据向量，找到 $k \leq n$ 个正交向量（主成分），用于最优表示数据
 - 标准化输入数据：使每个属性处于相同的数值范围。
 - 计算k个标准正交（单位）向量，即主成分
 - 每个输入数据（向量）是这k个主成分向量的线性组合
 - 主成分按“重要性”或强度从高到低排序
 - 由于主成分已排序，可通过剔除弱成分（即方差低的成分）来缩减数据规模——使用最强的主成分，可较好地重构原始数据
 - 仅适用于数值型数据
- 特征子集选择、特征创建
 - 冗余属性
 - 重复了一个或多个其他属性所包含的大部分或全部信息
- 无关属性
 - 不包含任何对当前数据挖掘任务有用的信息
- 步骤
 - 属性独立假设下的最优单个属性：通过显著性检验选择最优属性
 - 最优逐步特征选择
 - initial attribute set: {A1, A2, A3, A4, A5, A6}
 - initial reduced set: { }
 - step 1: {A1}
 - step 2: {A1, A4}
 - step 3: {A1, A4, A6}
 - 先选择最优的单个属性；
 - 再选择与第一个属性匹配度最高的下一个最优属性，依此类推
 - 逐步属性剔除：重复剔除最差的属性
 - initial attribute set: {A1, A2, A3, A4, A5, A6}
 - initial reduced set: {A1, A2, A3, A4, A5, A6}
 - step 1: {A1, A3, A4, A5, A6}
 - step 2: {A1, A4, A5, A6}
 - step 3: {A1, A4, A6}
 - 最优组合属性选择与剔除

2. 数量约简（部分场景下也直接称为“数据约简”），包含：

- 回归与对数线性模型
- 直方图、聚类、采样
- 数据立方体聚合

3. 数据压缩

维度约简

- **维度灾难：**
 - 当维度增加时，数据会变得越来越稀疏
 - 对聚类、异常值分析至关重要的“点间密度与距离”会变得无意义；
 - 子空间的可能组合会呈指数级增长
- **维度约简：**
 - 避免维度灾难
 - 帮助消除无关特征、减少噪声
 - 降低数据挖掘所需的时间与空间成本
 - 便于数据可视化

数据约简

定义：通过选择更小的替代数据表示形式，减少数据量

- **参数化方法（如回归）：**
 - 假设数据符合某种模型，估计模型参数，仅存储参数（异常值可能保留），舍弃原始数据
- **非参数化方法：**
 - 不假设数据模型；
 - 主要类型：直方图、聚类、采样等

采样

定义：选取一个小样本 s 来代表整个数据集 N

- 使挖掘算法的复杂度可能低于数据规模的线性级别
- 核心原则：选择数据的**代表性子集**
- 简单随机采样在数据偏态时效果很差
- 需采用自适应采样方法，例如分层采样

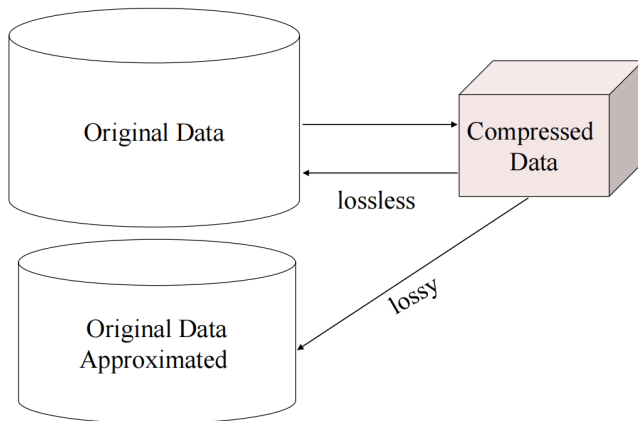
分类

- 简单随机采样：每个样本被选中的概率相等
- 无放回采样：样本被选中后，从总体中移除
- 有放回采样：样本被选中后，仍保留在总体中
- 分层采样：先划分数据集，再从每个分区中按比例抽样（即各分区抽取的样本占比与原数据一致）；适用于偏态数据

数据压缩

类型

- 字符串压缩
- 音视频压缩
- 时间序列



- 原始数据经**无损压缩**后，可完全还原为原始数据
- 原始数据经**有损压缩**后，只能还原为近似的原始数据

数据转换

定义：一种函数，将某一属性的所有值映射为一组新值，每个旧值对应一个新值
方法

- **平滑**：去除数据中的噪声；
- **属性 / 特征构造**：基于已有属性构建新属性；
- **聚合**：汇总数据（如数据立方体构建）；
- **归一化**：将数据缩放到更小的指定范围，包含：
 - min-max归一化
 - z-score归一化
 - 小数定标归一化

- **Min-max normalization**: to $[new_min_A, new_max_A]$

$$v' = \frac{v - min_A}{max_A - min_A} (new_max_A - new_min_A) + new_min_A$$

- Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,000 is mapped to $\frac{73,000 - 12,000}{98,000 - 12,000} (1.0 - 0) + 0 = 0.716$

- **Z-score normalization** (μ : mean, σ : standard deviation):

$$v' = \frac{v - \mu_A}{\sigma_A}$$

- Ex. Let $\mu = 54,000$, $\sigma = 16,000$. Then $\frac{73,600 - 54,000}{16,000} = 1.225$

- **Normalization by decimal scaling**

$$v' = \frac{v}{10^j} \quad \text{Where } j \text{ is the smallest integer such that } \text{Max}(|v'|) < 1$$

- **离散化**：概念分层提升 Concept Hierarchy Generation
 - 概念分层：将概念（属性值）按层级组织，通常与数据仓库的维度关联
 - 概念分层支持数据仓库的“下钻”与“上卷”操作，实现多粒度数据查看

- 概念分层构建：递归将低层级概念（如年龄数值）替换为高层级概念（如青年 / 中年 / 老年），缩减数据
- 可由领域专家 / 数据仓库设计者显式指定
- 可自动为数值型与标称型数据生成（数值型数据需用离散化方法）

12 特征选择与提取

特征定义

定义：用于分类的测量指标

- 在更一般的情况下，会使用 l 个特征 $x_i, i = 1, 2, \dots, l$ ，它们构成**特征向量**：

$$\mathbf{x} = [x_1, x_2, \dots, x_l]^T$$

- 每个特征向量唯一对应一个单独的对象。

- 分类任务中的几个基础问题：**

1. 特征如何生成？

- 取决于具体问题，对应**特征生成阶段**。

2. 最优的特征数量 l 是多少？

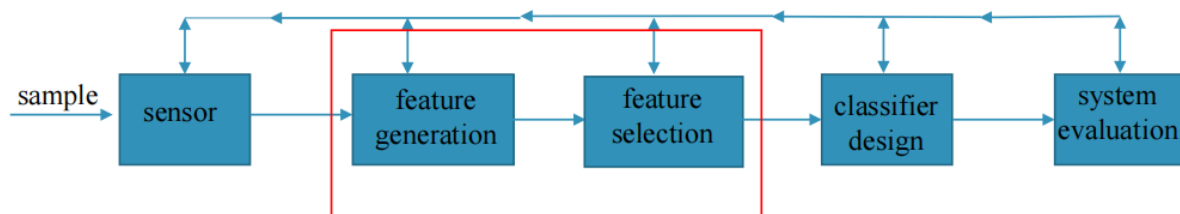
- 对应**特征选择阶段**。实际中通常先生成较多的候选特征，再从中选取“最优”的特征。

3. 选定合适的特征后，如何设计分类器？

- 前文示例中是线性分类器；一般情况下，划分不同类别区域的边界是非线性的。针对特定任务选择分类器类型、确定优化准则，对应**分类器设计阶段**。

4. 如何评估所设计分类器的性能？

- 对应**系统评估阶段**。



特征选择

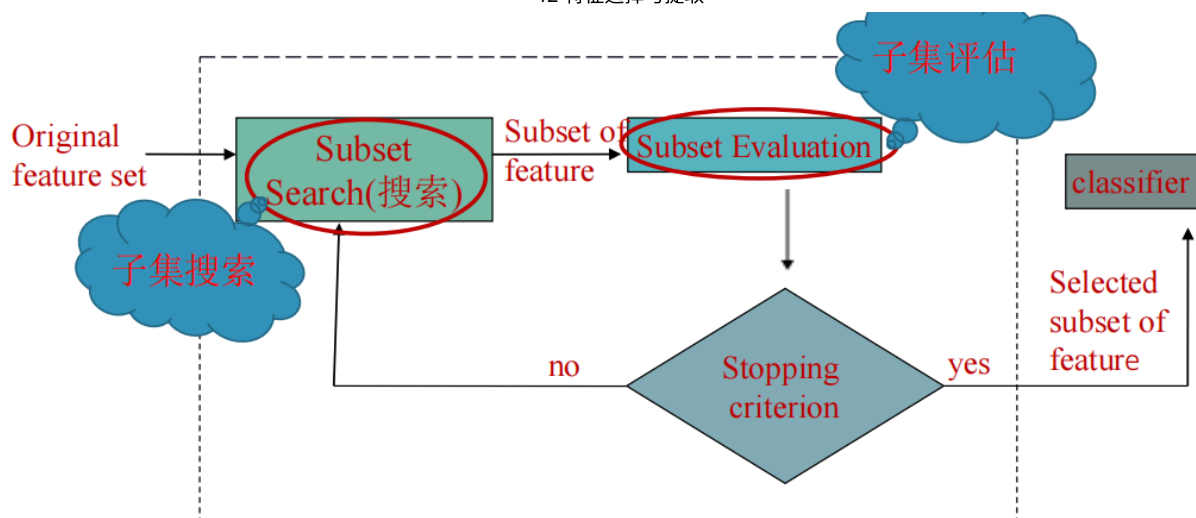
相关特征：与任务相关，且彼此间不相关的属性

核心特点：好而不同，对任务有用且无冗余

无关特征：与任务不相关的属性

特征选择（Feature Selection）：从所有特征中筛选出“与任务相关、且彼此不相关”的特征

步骤：



流程解析

1. **输入**：原始特征集（Original feature set）；
2. **子集搜索（Subset Search）**：从原始特征集中生成候选特征子集；
3. **子集评估（Subset Evaluation）**：通过分类器等工具评估候选子集的有效性（比如特征子集对应的模型性能）；
4. **停止准则（Stopping criterion）**：判断当前候选子集是否满足要求：
 - 若不满足（no），返回“子集搜索”环节生成新的候选子集；
 - 若满足（yes），输出最终的选定特征子集（Selected subset of feature）。

特征提取

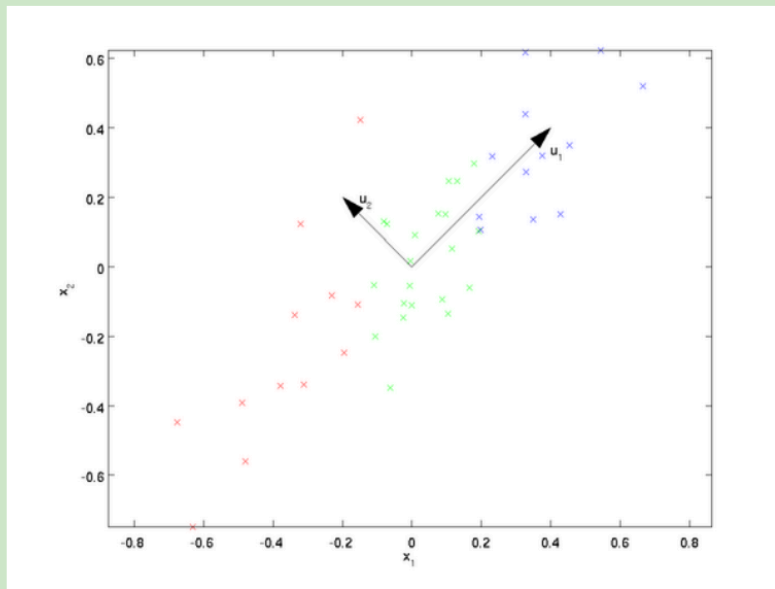
特征提取：是指通过构建特征组合，将原始特征转换为一组新特征的过程

PCA优点：保证新空间中特征之间不相关的情况下，使变换后的特征维数更少
实现降维和特征提取

缺点：被忽略掉的成分可能也包含一些相对独立的信息

PCA顾名思义，就是找出数据里最主要的方面，用数据里最主要的方面来代替原始数据。具体的，假如我们的数据集是 n 维的，共有 m 个数据 $(x^{(1)}, x^{(2)}, \dots, x^{(m)})$ 。我们希望将这 m 个数据的维度从 n 维降到 n' 维，希望这 m 个 n' 维的数据集尽可能的代表原始数据集。我们知道数据从 n 维降到 n' 维肯定会有损失，但是我们希望损失尽可能的小。那么如何让这 n' 维的数据尽可能表示原来的数据呢？

我们先看看最简单的情况，也就是 $n=2$ ， $n'=1$ ，也就是将数据从二维降维到一维。数据如下图。我们希望找到某一个维度方向，它可以代表这两个维度的数据。图中列了两个向量方向， u_1 和 u_2 ，那么哪个向量可以更好的代表原始数据集呢？从直观上也可以看出， u_1 比 u_2 好。



为什么 u_1 比 u_2 好呢？可以有两种解释，第一种解释是样本点到这个直线的距离足够近，第二种解释是样本点在这个直线上的投影能尽可能的分开。

假如我们把 n' 从1维推广到任意维，则我们的希望降维的标准为：样本点到这个超平面的距离足够近，或者说样本点在这个超平面上的投影能尽可能的分开。

Kernel-PCA优点：解决低维空间中非线性映射问题 应用核技巧，不增加计算复杂度

缺点：选择合适的核函数没有固定的方法