

机器学习入门

A short course of machine learning

李德维

ldw@cumtb.edu.cn infhighdim.github.io

中国矿业大学(北京) 理学院

2023.07.05



课程提纲

- ① 机器学习概述
- ② 逻辑斯蒂回归、k近邻和贝叶斯分类器
- ③ 决策树与随机森林
- ④ 支持向量机
- ⑤ 神经网络
- ⑥ 聚类分析

① 机器学习概述

② 逻辑斯蒂回归、k近邻和贝叶斯分类器

③ 决策树与随机森林

④ 支持向量机

⑤ 神经网络

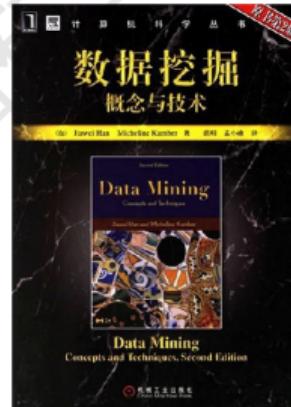
⑥ 聚类分析

课程教材

- a 机器学习, 周志华著. 北京: 清华大学出版社, 2016年1月.
- b 数据挖掘概念与技术, (加)Jiawei Han, Micheline Kamber. 机械工业出版社, 2007年3月.
- c 统计学习方法, 李航, 清华大学出版社, 2012年3月.
- d Foundations of Machine Learning. Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar MIT Press, Second Edition, 2018.(<https://cs.nyu.edu/~mohri/mlbook/>)



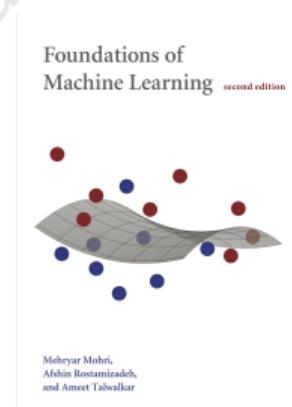
(a)



(b)



(c)



(d)

机器学习概述

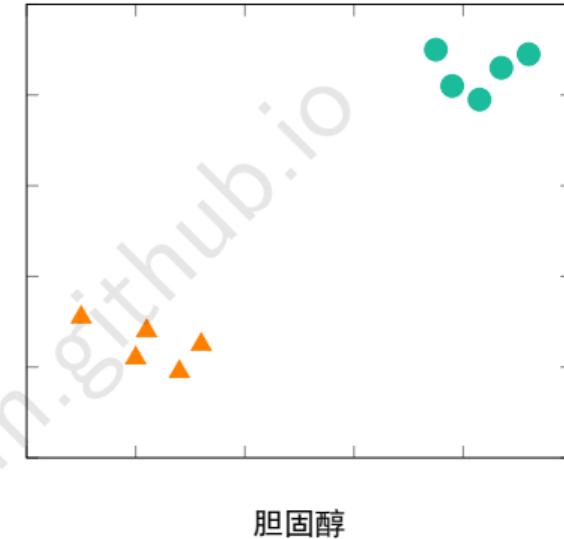
为什么要学机器学习？

色泽	根蒂	敲击声	好瓜
青绿	蜷缩	浊响	是
乌黑	蜷缩	浊响	是
青绿	硬挺	清脆	否
乌黑	稍蜷	沉闷	否

机器学习概述

为什么要学机器学习？

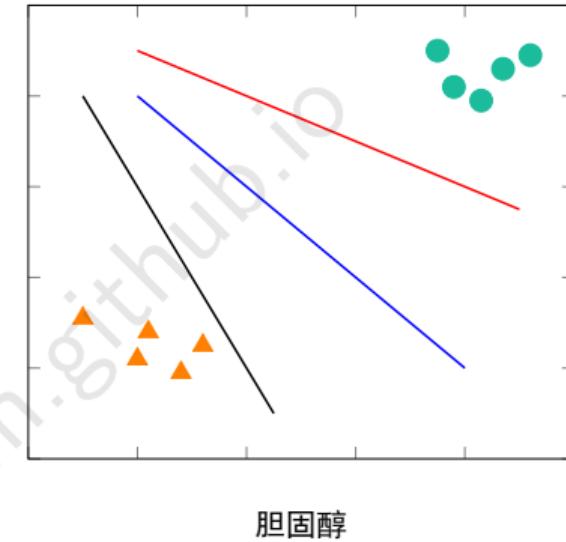
色泽	根蒂	敲击声	好瓜
青绿	蜷缩	浊响	是
乌黑	蜷缩	浊响	是
青绿	硬挺	清脆	否
乌黑	稍蜷	沉闷	否



机器学习概述

为什么要学机器学习？

色泽	根蒂	敲击声	好瓜
青绿	蜷缩	浊响	是
乌黑	蜷缩	浊响	是
青绿	硬挺	清脆	否
乌黑	稍蜷	沉闷	否



机器学习概述



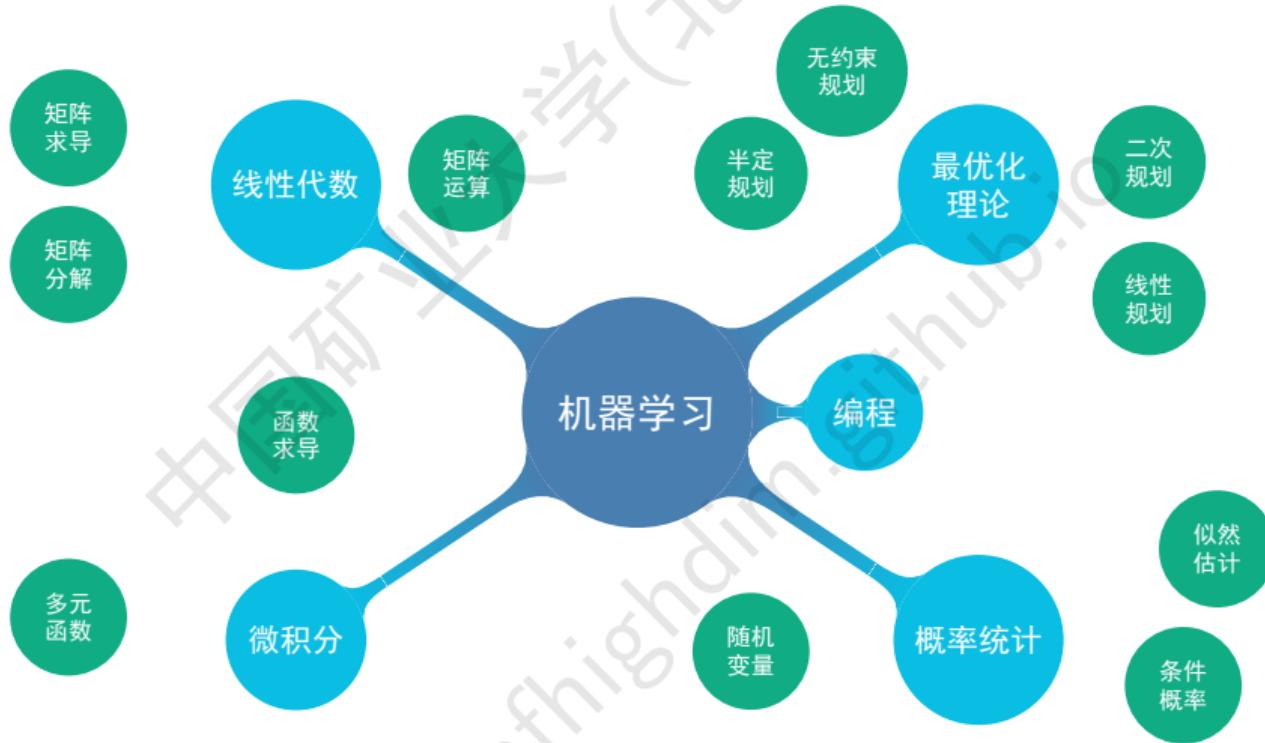
(a) 人脸识别



(b) 游戏玩家匹配

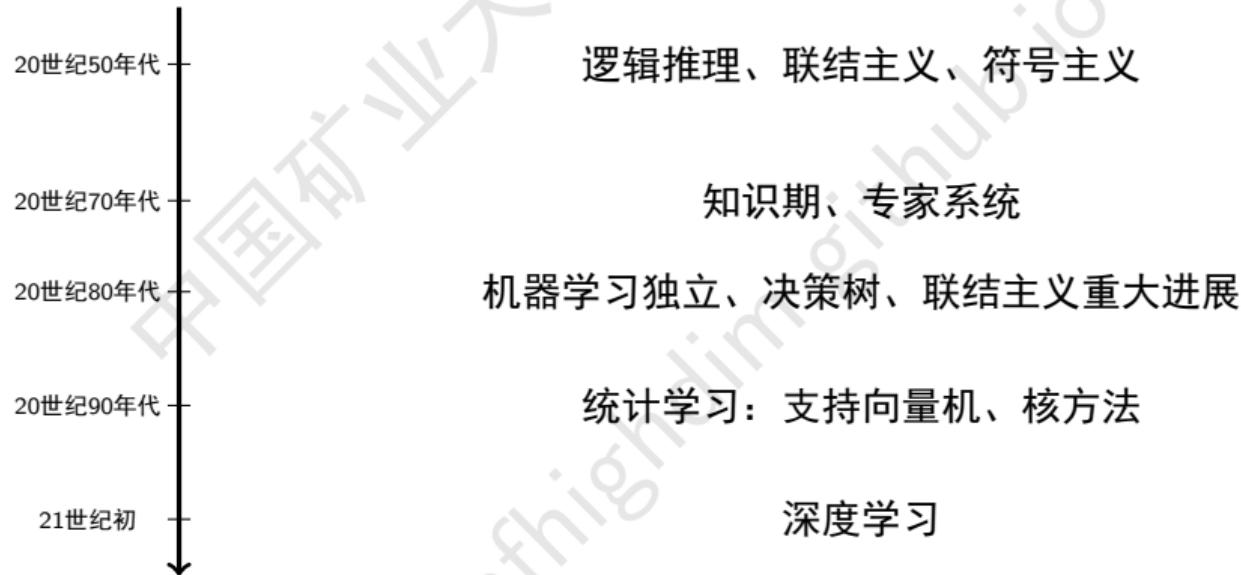
语音识别(微信语音锁)、新闻推荐(今日头条)、无人驾驶(特斯拉)、医学诊断、金融欺诈...

机器学习概述-知识基础

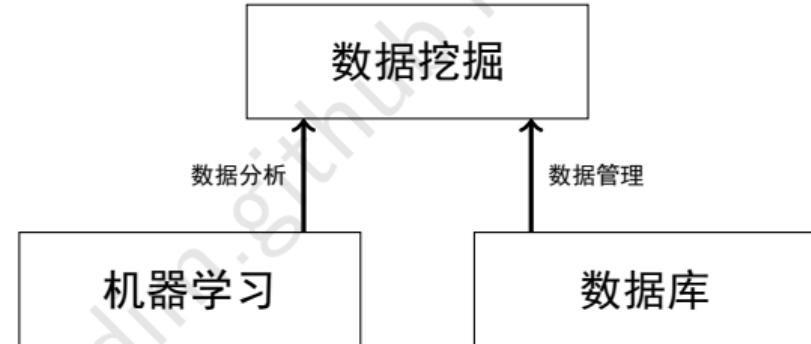
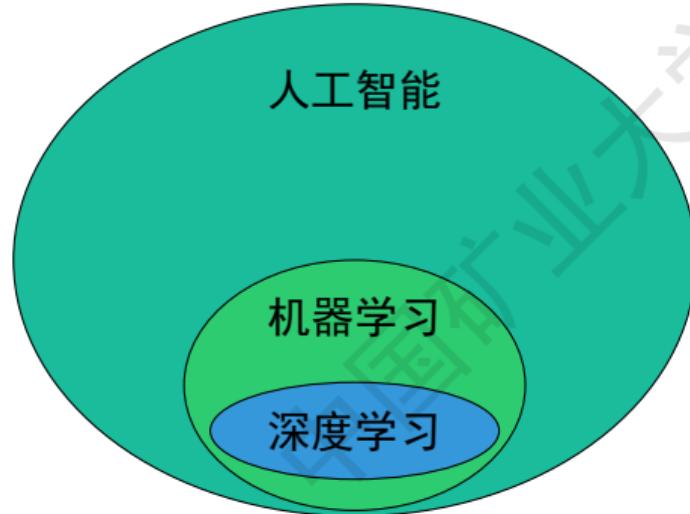


机器学习概述

机器学习 (Machine Learning) 是对研究问题进行模型假设，利用计算机从训练数据中学习得到模型参数，并最终对数据进行预测和分析的一门学科。机器学习涉及多个领域的理论交叉，包括最优化理论、概率论、统计学、逼近论、算法复杂度理论等。

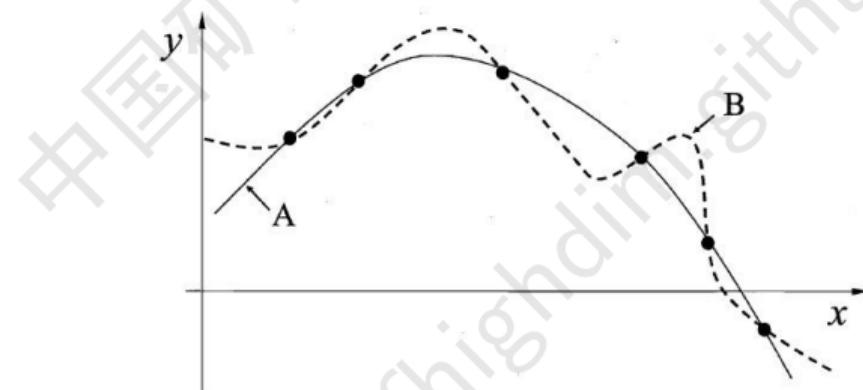


机器学习概述



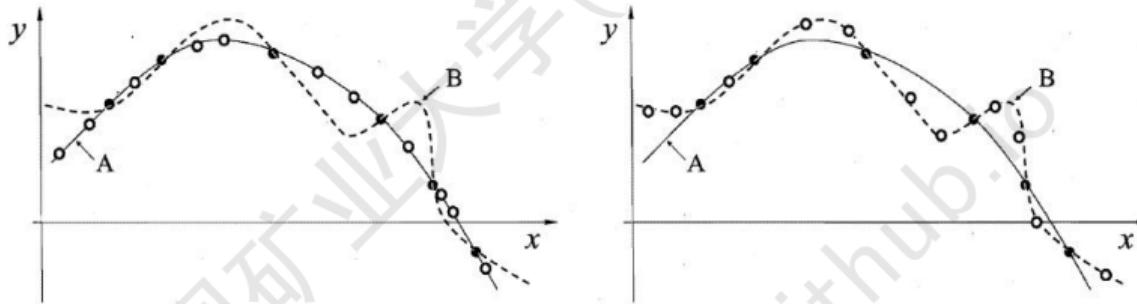
机器学习概述-基本概念

令数据集 $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\} \in R^{m \times n}$, 它包含了 m 个数据 **样本**, 以及对应的**标签**信息 y , 每个样本的**特征**(属性)个数为 n , 也称为样本的维数。标签一般为标量, 可以是连续或离散。数据集一般分为两部分, **训练集**和**测试集**。从训练数据中学得模型的过程称为“**学习**”(learning)或“**训练**”(training), 训练完成后利用测试集对模型进行**评估**。模型评估的目标是验证学到的模型能否具有适用新数据的能力, 这种能力称为**泛化**(generalization) 能力。



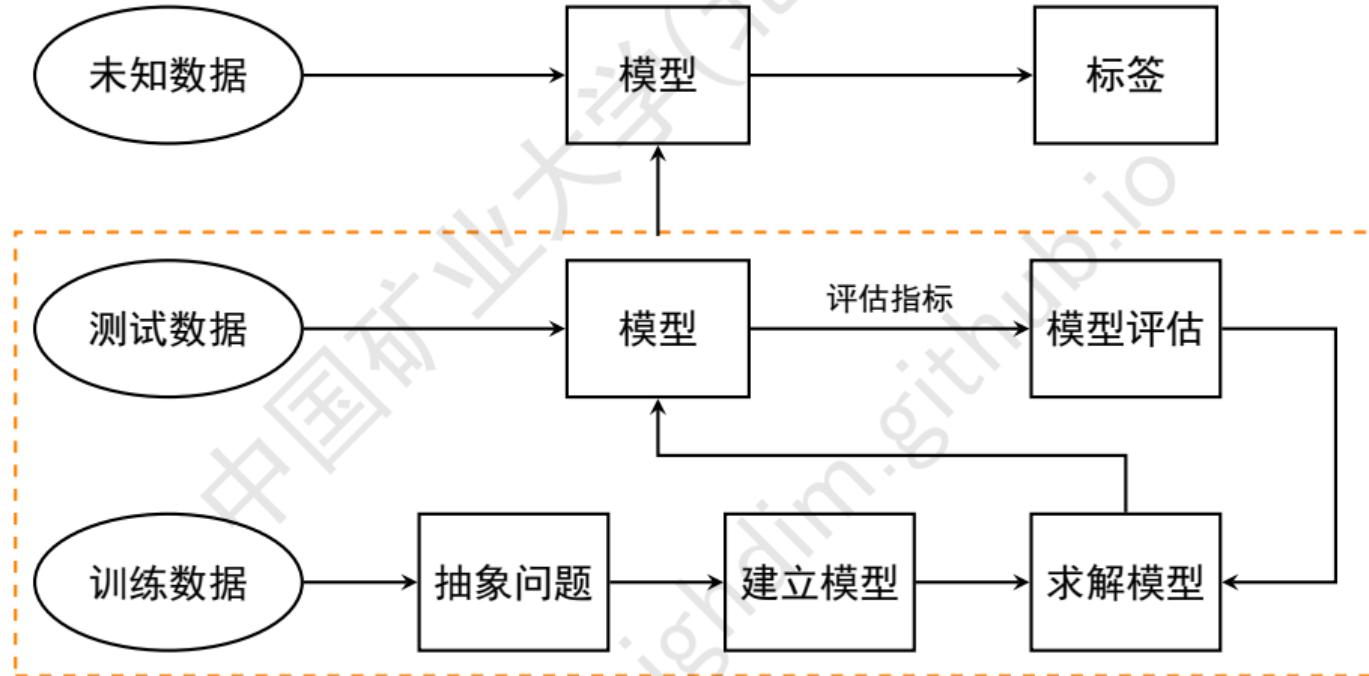
奥卡姆(Occam's razor)剃刀原理: 若有多个假设与观察一致, 则选最简单的那个。

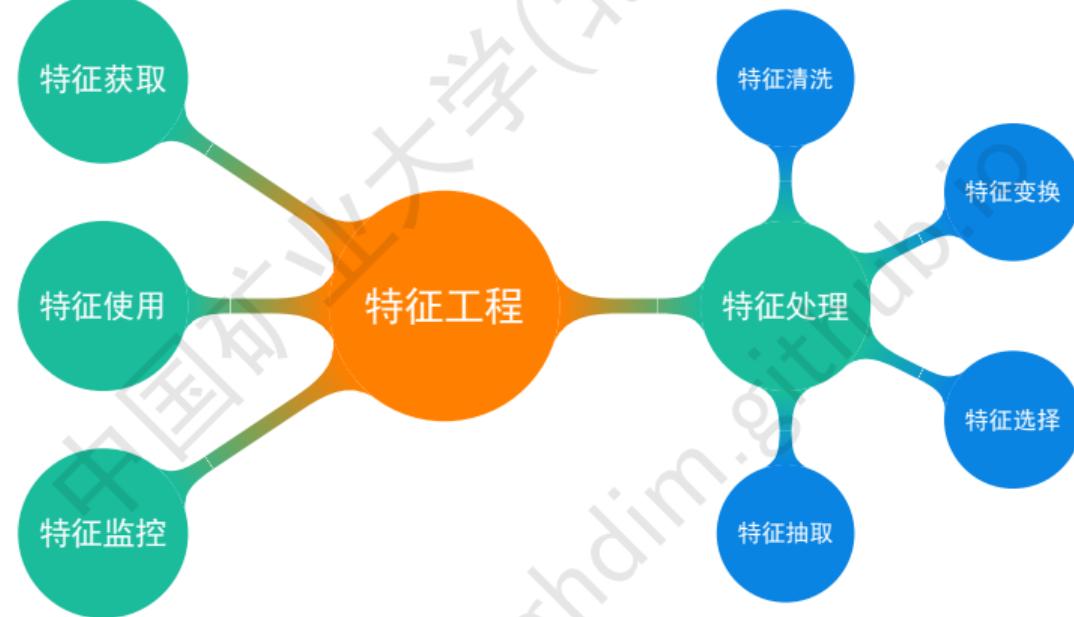
“没有免费的午餐” (No Free Lunch Theorem)



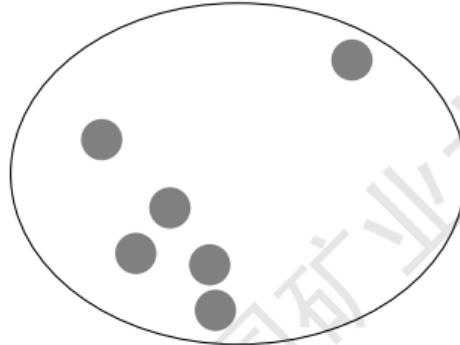
任何模型算法的性能和效果都与具体的问题相关，没有一个算法能在所有任务上都表现最优。

机器学习概述-学习步骤

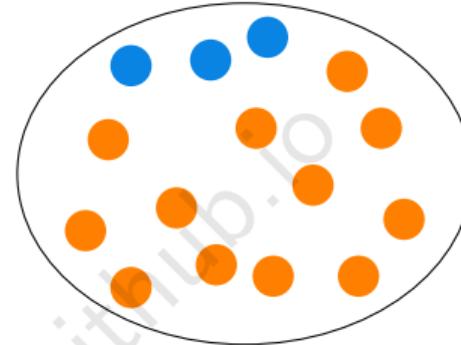




机器学习概述-特征清洗



(a) 离群点: 剔除、平滑



(b) 不平衡: 上下采样、数据增强

编号	特征1	特征2	特征3
1	5	20.1	3.6
2	4.3	9.5	5.9
3	5	20.1	3.6

(c) 重复值: 保留、剔除

编号	特征1	特征2	特征3
1	5	20.1	
2	4.3		5.9
3	5	20.1	3.6

(d) 缺失值: 剔除、人工填写、全局常量、属性平均值、同类属性平均值、算法推测

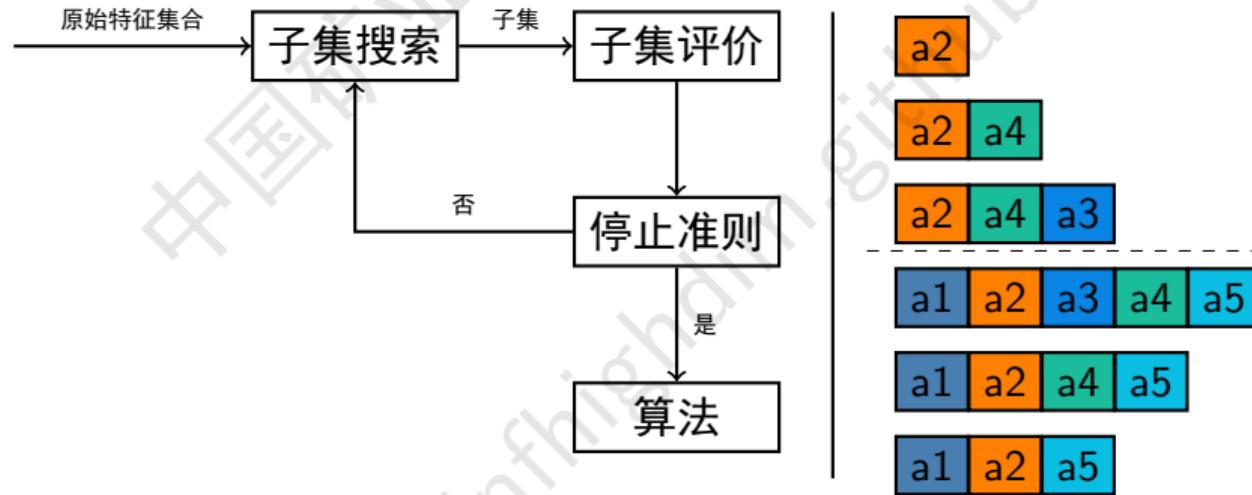
- 聚集：对数据集进行汇总和聚集。例如对于一份销售数据，可以考虑聚集每天的数据，计算月销售额或年销售额；
- 数据泛化：用高层次概念替换原始数据。数值属性age，可以离散化成young, middle, senior等；
- 规范化：标准化、归一化。将数据按比例缩放，映射到一个标准区间内，比如 $[0, 1]$, $[-1, 1]$ ；(思考：为什么要归一化？)

$$\text{max-min} : x' = \frac{x - \min}{\max - \min}; z-score : x' = \frac{x - \hat{X}}{\sigma}; \text{小数定标} : x' = \frac{x}{10^j}$$

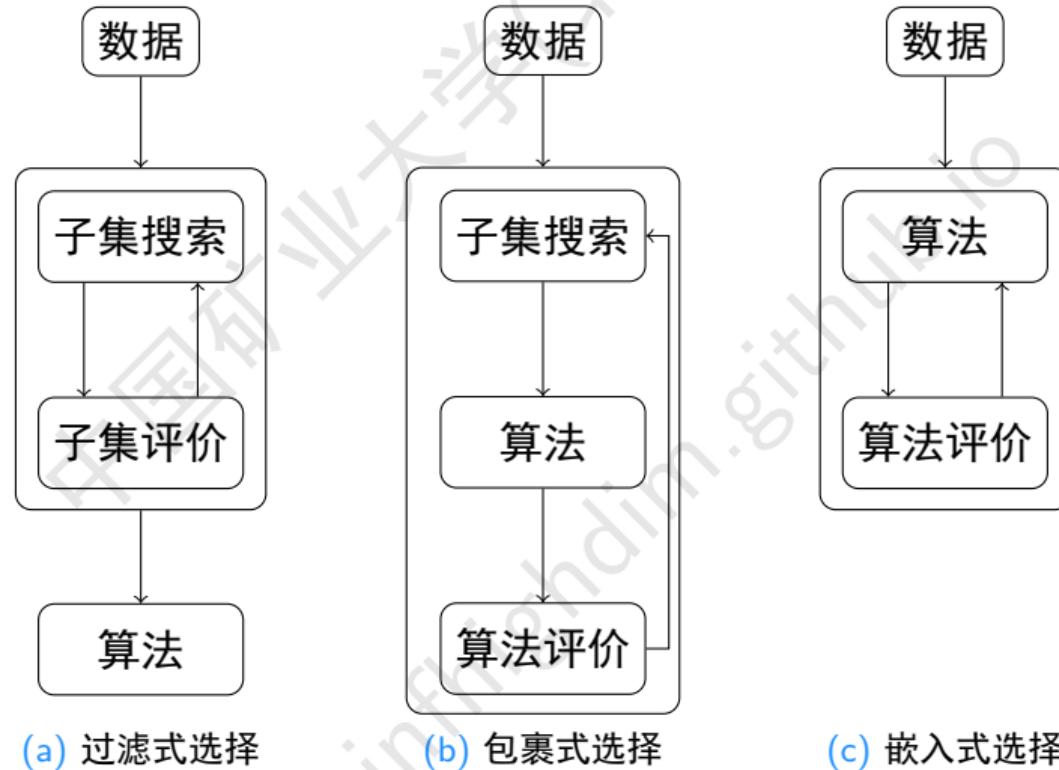
非线性归一化、批量归一化(BatchNormalization)

- 特征构造：例如在电商领域，用户行为数据表中每条记录为某个用户的一次浏览行为或一次点击行为，可以由此构造出用户最近一次浏览的时长、用户最近一次登录的点击次数等特征。

- **动机:** 相关特征、无关特征
- **优点:** 维数灾难–随着维数的增加，计算量呈指数倍增长的一种现象；降低学习任务的难度–更容易学习到好的模型
- **基本思路:** 子集搜索–前向、后向、双向；子集评价



机器学习概述-特征选择



Relief (Relevant Features): 设计了一个"相关统计量"来度量特征的重要性

给定训练集，对于每个样本 x_i ，分别寻找空间中的同类近邻 $x_{i,nh}$ 和异类近邻 $x_{i,nm}$ ，那么使用如下方式计算特征 j 的重要性，

$$\delta^j = \sum_i -\text{diff}(x_i^j, x_{i,nh}^j)^2 + \text{diff}(x_i^j, x_{i,nm}^j)^2 \quad (1)$$

其中

$$\text{diff}(a, b) = \begin{cases} \mathbf{1}_{a=b}, & \text{离散} \\ |a - b|, & \text{连续} \end{cases} \quad (2)$$

对基于不同样本得到的估计结果进行平均，就得到各属性的相关统计量分量，分量值越大，则对应属性的分类能力就越强。（思考：为什么有效？）

机器学习概述-特征选择

LVW (Las Vegas Wrapper): 使用随机策略来进行子集搜索，并以最终分类器的误差为特征子集评价准则。

输入: 数据集 D ;
特征集 A ;
学习算法 \mathcal{L} ;
停止条件控制参数 T .

过程:

```
1:  $E = \infty$ ;
2:  $d = |A|$ ;
3:  $A^* = A$ ;
4:  $t = 0$ ;
5: while  $t < T$  do
6:   随机产生特征子集  $A'$ ;
7:    $d' = |A'|$ ;
8:    $E' = \text{CrossValidation}(\mathcal{L}(D^{A'}))$ ;
9:   if  $(E' < E) \vee ((E' = E) \wedge (d' < d))$  then
10:     $t = 0$ ;
11:     $E = E'$ ;
12:     $d = d'$ ;
13:     $A^* = A'$ 
14:   else
15:     $t = t + 1$ 
16:   end if
17: end while
输出: 特征子集  $A^*$ 
```

机器学习概述-特征选择

给定数据集 D , 考虑线性回归模型
岭回归

$$\min_w \sum_{i=1}^m (y_i - w^\top x_i)^2 + \lambda ||w||^2$$

LASSO

$$\min_w \sum_{i=1}^m (y_i - w^\top x_i)^2 + \lambda ||w||$$

哪一种可以做特征选择?

机器学习概述-特征选择

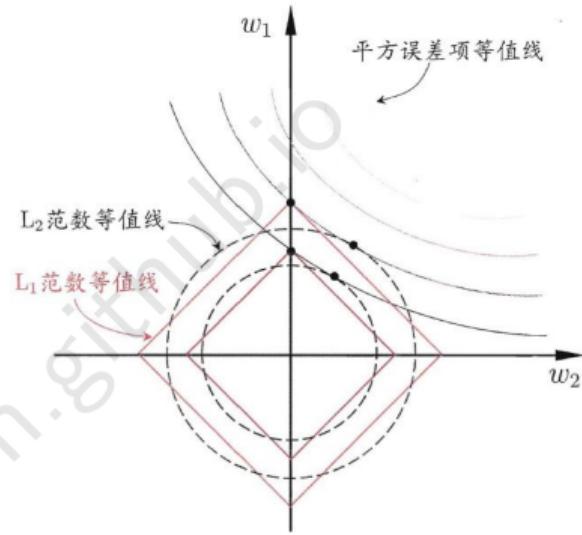
给定数据集 D , 考虑线性回归模型
岭回归

$$\min_w \sum_{i=1}^m (y_i - w^\top x_i)^2 + \lambda \|w\|^2$$

LASSO

$$\min_w \sum_{i=1}^m (y_i - w^\top x_i)^2 + \lambda \|w\|$$

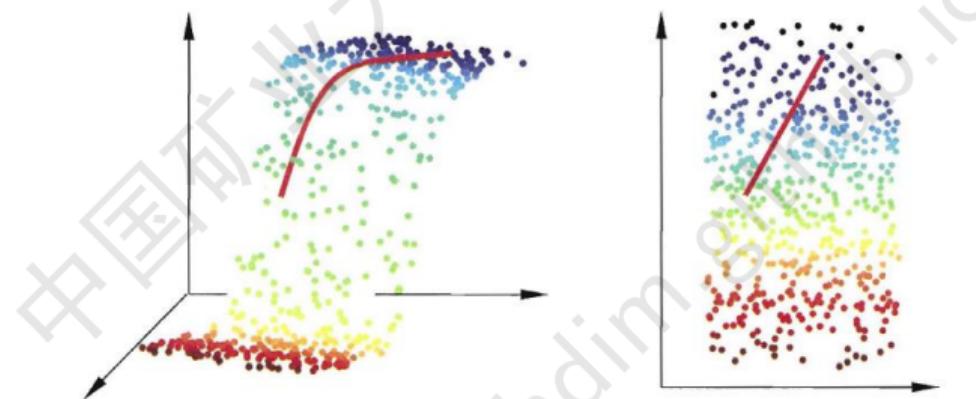
哪一种可以做特征选择?



机器学习概述-特征抽取

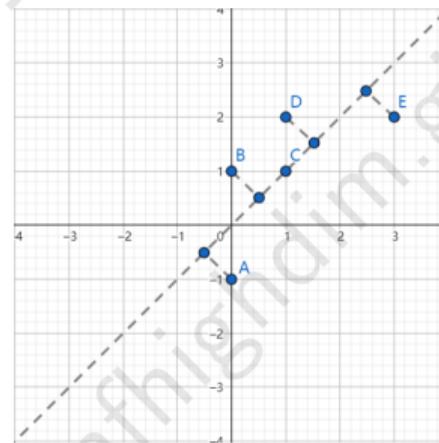
降维：高维空间，低维映射，学习难度降低，学习效果不变甚至更好
学习映射矩阵 W ，获得

$$z = Wx$$



主成分分析(Principal Component Analysis , 简称PCA): 将 n 维原始特征映射到 k 维($k < n$)上, 称这 k 维特征为主成分。其主要目标是将特征维度变小, 同时尽量减少信息损失。
PCA将原始样本点投影到理想的超平面上:

- **最大可分性:** 样本点在这个超平面上的投影能尽可能分开;
- **最近重构性:** 样本点到这个超平面的距离都足够近.



机器学习概述-特征抽取

下面从最大可分性角度推导。给定 m 个样本点 x'_1, x'_2, \dots, x'_m , 首先进行中心化, 令 $x_i = x'_i - \frac{1}{m} \sum_{i=1}^m x'_i$, 那么有

$$\mu_x = \sum_{i=1}^m x_i = 0 \quad (3)$$

为了将原始样本投影到低维空间, 计算样本 x_i 与单位方向向量 w 的内积 $y_i = x_i^\top w$, 投影后的方差为

$$Var(y) = \frac{1}{m} \sum_{i=1}^m (y_i - \mu_y)^2 = \frac{1}{m} \sum_{i=1}^m (y_i - \frac{1}{m} \sum_{i=1}^m x_i w)^2 = \frac{1}{m} \sum_{i=1}^m y_i^2 \quad (4)$$

$$Var(y) = \frac{1}{m} \sum_{i=1}^m (x_i^\top w)^2 = w^\top \left(\frac{1}{m} \sum_{i=1}^m x_i x_i^\top \right) w = w^\top \Sigma w \quad (5)$$

基于最大可分性，我们构建如下最优化问题

$$\max_w \quad w^\top \Sigma w \quad (6)$$

$$s.t. \quad w^\top w = 1 \quad (7)$$

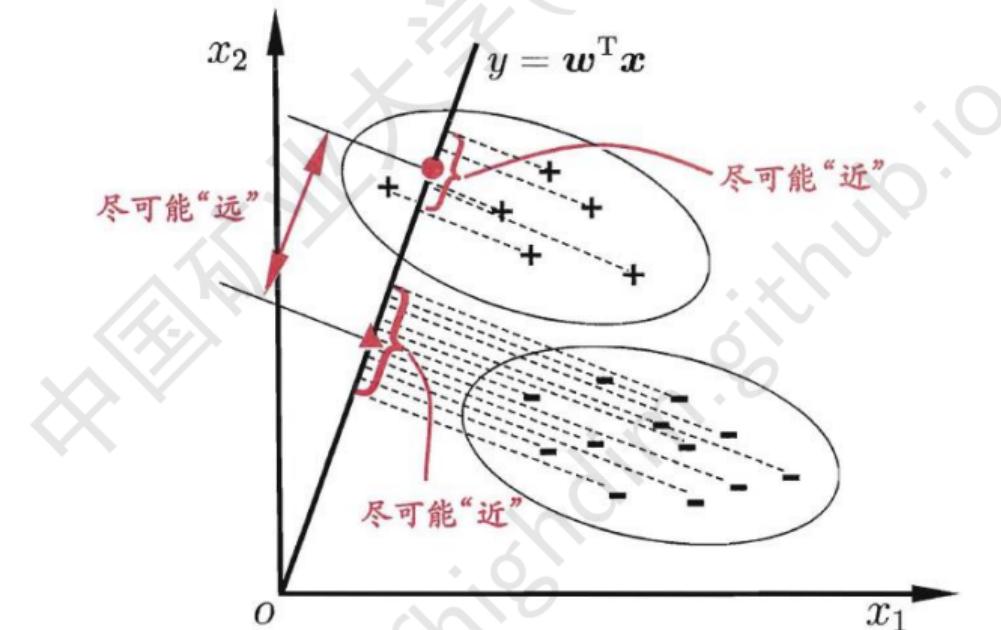
对上式构建拉格朗日函数 $L(w, \lambda) = w^\top \Sigma w + \lambda(1 - w^\top w)$, 令 $\nabla_w L = 0$ 可得

$$\Sigma w = \lambda w \quad (8)$$

将上式带入到方差可得 $Var(y) = \lambda$.

机器学习概述-特征抽取

线性判别分析(Linear Discriminant Analysis, 简称LDA)



机器学习概述-特征抽取

定义类内散度矩阵

$$S_w = \Sigma_0 + \Sigma_1 = \sum_{x \in X_0} (x - \mu_0)(x - \mu_0)^\top + \sum_{x \in X_1} (x - \mu_1)(x - \mu_1)^\top$$

和类间散度矩阵

$$S_b = (\mu_0 - \mu_1)(\mu_0 - \mu_1)^\top$$

可以构建如下最优化问题

$$\max \frac{w^\top S_b w}{w^\top S_w w}$$

可以等价转化为如下最优化问题

$$\min_w -w^\top S_b w \tag{9}$$

$$s.t. \quad w^\top S_w w = 1 \tag{10}$$

机器学习概述-特征抽取

利用拉格朗日乘子法，有

$$S_b w = \lambda S_w w$$

由于 $S_b w$ 方向恒为 $\mu_0 - \mu_1$ ，不妨令 $S_b w = \lambda(\mu_0 - \mu_1)$ ，那么有

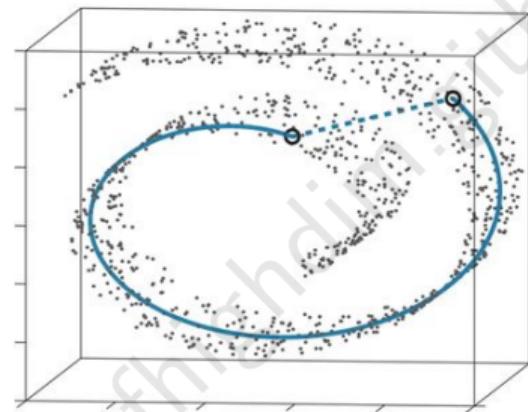
$$w = S_w^{-1}(\mu_0 - \mu_1)$$

考虑到数值解的稳定性，一般会先对 S_w 进行奇异值分解，然后再求逆矩阵。
LDA 也可以推广至多分类情形。

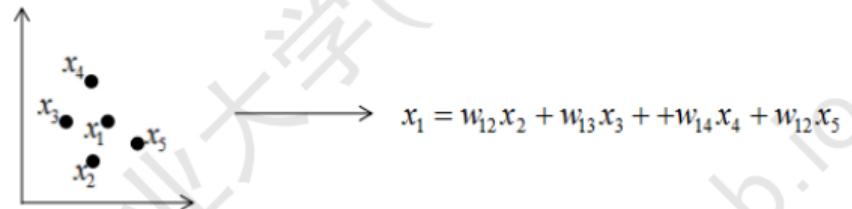
- 多维标度分析(MDS): 降维前后能够保持距离关系不变

$$\min_w \sum_{ij} (||Wx_i - Wx_j|| - d(x_i, x_j))^2$$

- 等距特征映射ISOMAP: 引入测地距离



- 局部线性嵌入LLE: 保持局部线性关系



- 拉普拉斯特征映射LE: 基于图构建邻接矩阵，降维后仍能保持原有的数据结构信息

$$\min \sum_{ij} ||z_i - z_j||^2 w_{ij}$$

其中 $w_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma^2}\right)$

- t-分布随机近邻嵌入tsne: 利用概率分布定义距离关系

$$\begin{aligned} p_{ij} &= \frac{\exp(-\|x_i - x_j\|^2)/2\sigma^2}{\sum_{k \neq l} \exp(-\|x_k - x_l\|^2)/2\sigma^2} \\ q_{ij} &= \frac{\exp(-\|z_i - z_j\|^2)}{\sum_{k \neq l} \exp(-\|z_k - z_l\|^2)} \end{aligned}$$

目标函数

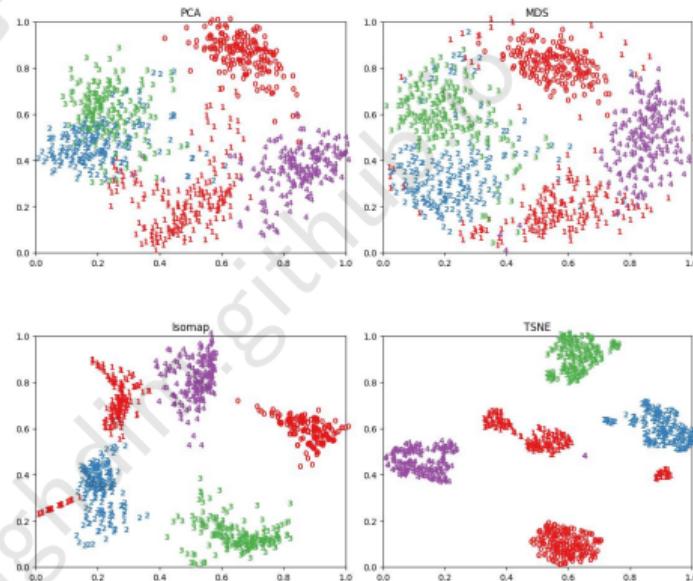
$$C = KL(P||Q) = \sum_{ij} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

机器学习概述-特征抽取

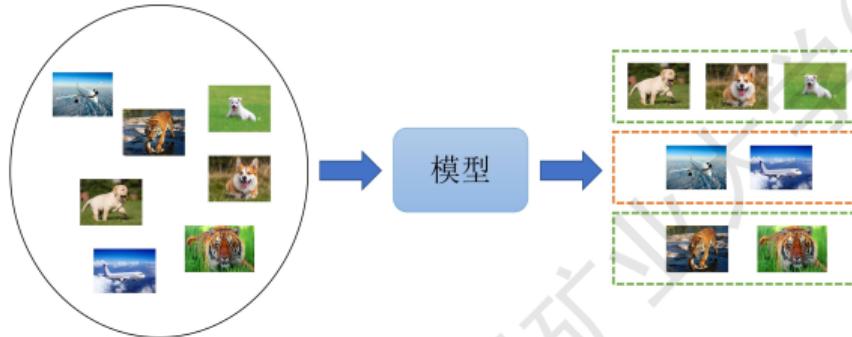
手写字体集合

0 1 2 3 4 0 1 1 3 4 0 1 2 3 4 0 0 4 1 3
1 0 0 2 2 2 0 1 2 3 3 3 3 4 4 1 0 2 2 0
0 1 3 2 1 4 3 1 3 1 4 3 4 4 0 7 1 4 9 2
2 2 4 4 0 0 1 2 3 4 0 1 2 3 4 0 1 2 3 4
0 0 4 1 3 4 0 0 2 2 2 0 4 2 3 3 3 3 4 4
1 0 2 1 0 0 1 3 2 1 3 1 3 4 4 3 4 4 0 7
1 4 4 1 2 2 4 4 0 0 1 2 3 4 0 4 2 3 4 0
4 2 3 4 0 0 4 1 3 4 0 0 2 2 2 0 4 2 3 3
3 3 4 4 1 0 2 2 0 0 1 3 2 4 4 3 4 3 1 4
3 1 9 6 3 1 4 4 2 2 2 4 4 0 3 0 1 1 3 4
0 1 2 3 4 0 1 2 3 4 0 0 4 1 3 1 0 0 1 2
2 0 1 2 3 3 3 3 4 4 1 0 1 2 0 0 1 3 2 1
4 3 1 3 1 4 3 1 4 0 3 1 4 4 2 2 2 4 4 0
0 1 2 3 4 0 1 2 3 4 0 1 2 3 4 0 0 4 1 3
4 0 0 1 1 2 0 1 1 3 3 3 3 4 4 1 0 1 2 0
0 1 3 1 4 3 1 3 1 4 3 1 4 0 3 1 4 9 2
2 1 4 4 0 0 1 2 3 4 0 1 2 3 4 0 1 2 3 4
0 0 4 1 3 4 0 0 2 2 2 0 1 2 3 3 3 3 4 4
1 0 2 1 0 0 1 3 2 1 4 3 1 3 4 3 4 3 4 0
3 1 0 4 2 2 2 4 4 0 0 1 2 3 4 0 1 2 3 4

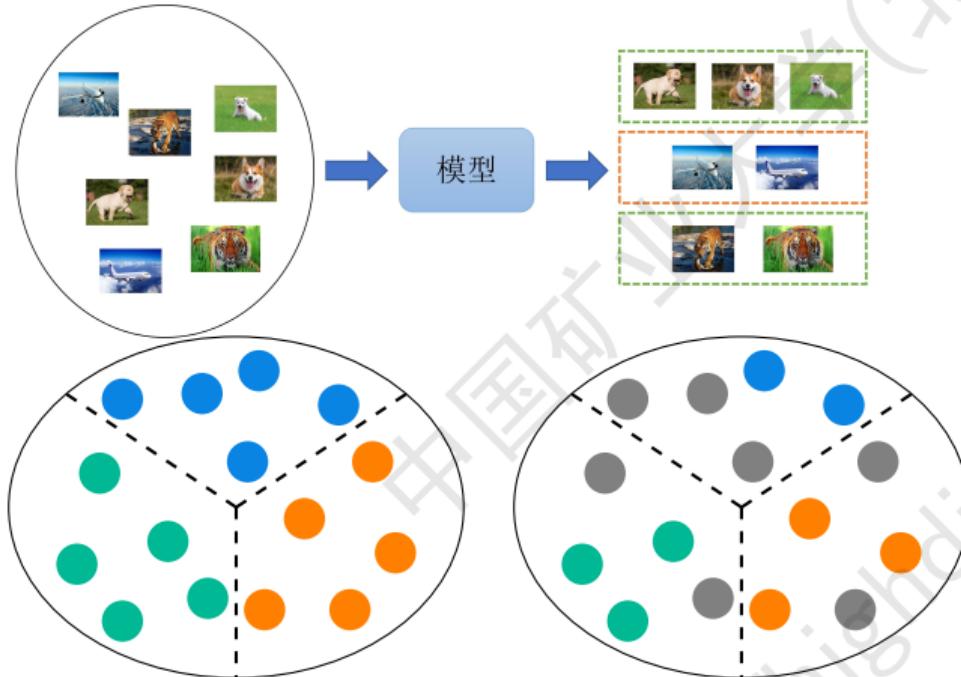
降维方法对比



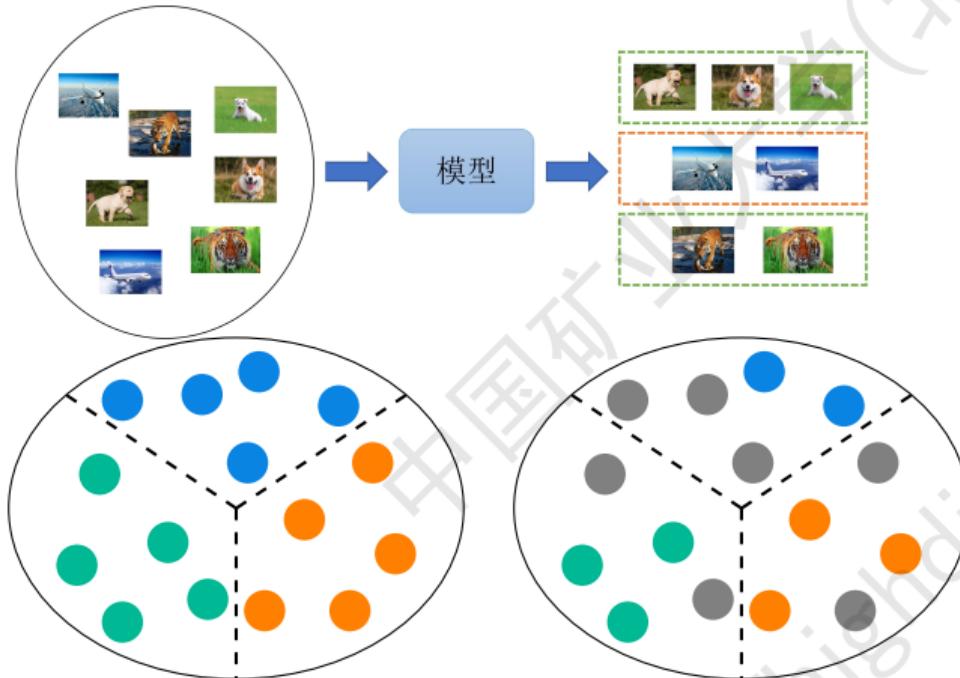
机器学习概述-分类问题



机器学习概述-分类问题

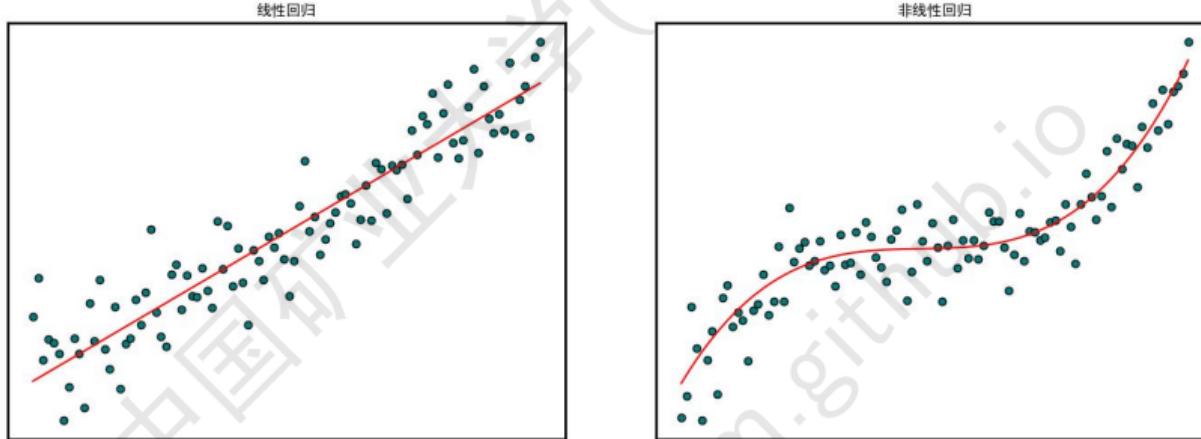


机器学习概述-分类问题



- 医疗诊断：患者生理指标，是否患病
- 金融欺诈：客户信息、还款历史，是否违约
- 人脸识别：面部信息，谁
- 市场营销：消费者行为，是否目标客户

机器学习概述-回归

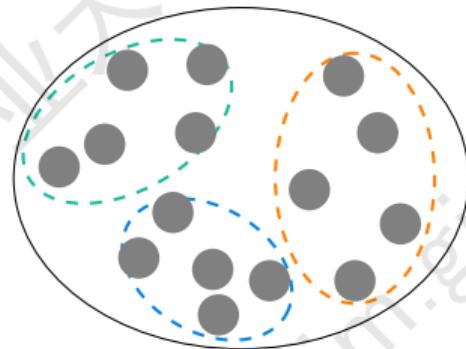


- 金融领域：历史股票价格，未来股票价格
- 医疗领域：患者信息、用药情况，血药浓度
- 人脸识别：面部信息，用户年龄

机器学习概述-回归问题评估指标

- MAE
- MSE
- RMSE
- MAPE
- R2 score
- Adjusted R2 score

聚类学习是按照某种特定标准(如距离等)把一个数据集划分为不同的类或簇（子集），使得同一个簇内的数据对象的相似性尽可能大，不在同一个簇中的数据对象的差异性也尽可能地大（即聚类后同一类的数据尽可能聚集到一起，不同类数据尽量分离）。



- 社交网络：用户特征，社交圈子
- 电子商务：客户信息，分群
- 新闻分类：新闻文本，新闻类型

机器学习概述-聚类问题评估指标

评估标准：类内相似度高，类间相似度低

外部评价指标

- 纯度
- 归一化互信息(Normalized Mutual Information, NMI)
- 兰德指数(Rand index, RI)
- 调整兰德系数(Adjusted Rand index, ARI)
- R2 score
- Adjusted R2 score

内部评价指标

- 轮廓系数(Silhouette Coefficient)
- Calinski-Harabaz指数

机器学习概述-综合案例

2004年3月，在美国的自动驾驶车比赛，斯坦福大学机器学习专家S. Thrun的小组研制的参赛车用6小时53分钟成功走完了132英里赛程获得冠军。感知：语音识别、目标识别、物体追踪；预测：车辆行人的行为预测。



机器学习概述-综合案例

2004年3月，在美国的自动驾驶车比赛，斯坦福大学机器学习专家S. Thrun的小组研制的参赛车用6小时53分钟成功走完了132英里赛程获得冠军。感知：语音识别、目标识别、物体追踪；预测：车辆行人的行为预测。



2012美国大选，奥巴马麾下有一支机器学习团队，他们对各类选情数据进行分析，为奥巴马成功竞选提供了有力支持。涉及选民分类、选民的偏好画像。

① 机器学习概述

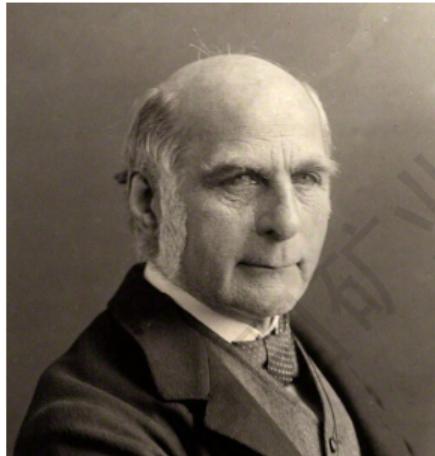
② 逻辑斯蒂回归、k近邻和贝叶斯分类器

③ 决策树与随机森林

④ 支持向量机

⑤ 神经网络

⑥ 聚类分析



线性回归：

对于一个回归数据集 $D = \{(x_i, y_i)\}_{i=1}^m$ ，
线性回归旨在学习一个线性模型，通过对
特征信息线性组合来预测 y ，即

$$\begin{aligned}f(x) &= w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b \\&= w^\top x + b\end{aligned}$$

最小二乘法：

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

- 1078对父、子身高的散点图
- 身材高大的父辈的孩子要矮些，而身材矮小的父辈的孩子要高些
- 遗传现象—身高趋于一般、“退化到平庸”

经典算法–Logistic回归

为了使表达形式更加简洁，将原始问题转换为向量形式。令 $\hat{w} = (w_1, \dots, w_n, b)$,
 $\hat{x}_i = (x_{i1}, \dots, x_{in}, 1)$, 那么有

$$f(\mathbf{x}_i) = \mathbf{w}^\top \mathbf{x}_i + b = \hat{\mathbf{w}}^\top \hat{\mathbf{x}}_i^\top$$

令

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^\top & 1 \\ \mathbf{x}_2^\top & 1 \\ \vdots & \\ \mathbf{x}_n^\top & 1 \end{pmatrix}$$

目标函数(思考：如何求解？)

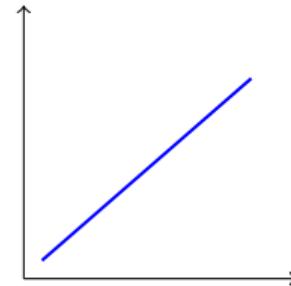
$$E = \sum_{i=1}^m (y_i - \hat{\mathbf{w}}^\top \hat{\mathbf{x}}_i^\top)^2 = (\mathbf{y} - \mathbf{X}\hat{\mathbf{w}})^\top (\mathbf{y} - \mathbf{X}\hat{\mathbf{w}})$$

经典算法-Logistic回归

分类问题：对于一个数据集 $D = \{(x_i, y_i)\}_{i=1}^m$, 标签 y 是离散、标量, 机器学习旨在寻找一个决策函数 $f(x)$, 使得对任意 x , 都能预测出其对应的标签 y 。如果 y 有两种取值, 则为二分类问题, 多于两种取值则为多分类问题。

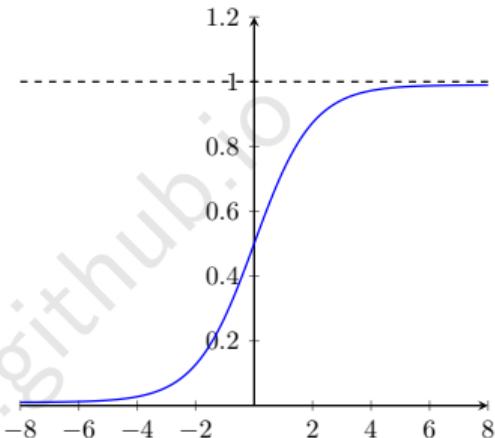
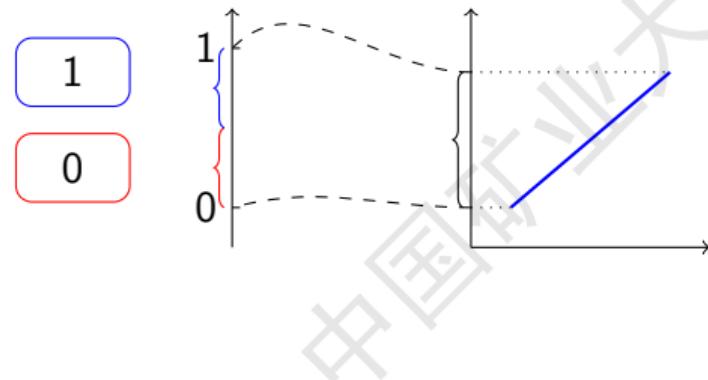
对于一个二分类数据集, 即标签 $y \in \{0, 1\}$, 是否可以借鉴回归的思想?

$$\begin{cases} 0 \\ 1 \end{cases} = y = ? = f(x) = w^\top x + b =$$



经典算法-Logistic回归

期望找到一个连续可微函数，将 $f(x)$ 映射到 $[0, 1]$ 。



Sigmoid函数

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

经典算法–Logistic回归

使用如下函数来预测标签

$$f(x) = \frac{1}{1 + e^{-(w^\top x + b)}}$$

一个事件的几率 (odds) 是指该事件发生的概率与该事件不发生的概率的比值. 如果事件发生的概率是 p , 那么该事件的几率是 $\frac{p}{1-p}$. 所以对数几率

$$\ln \frac{f(x)}{1 - f(x)} = w^\top x + b$$

对数几率回归–logistic regression

- 无需事先假设数据分布
- 近似概率预测
- Sigmoid函数任意阶可导

经典算法–Logistic回归

如果将 $f(x)$ 看作是 $y = 1$ 的概率估计，那么有

$$\ln \frac{P(y = 1|x)}{P(y = 0|x)} = w^\top x + b$$

那么有

$$P(y = 1|x) = \frac{e^{w^\top x + b}}{1 + e^{w^\top x + b}} = \pi(x)$$

$$P(y = 0|x) = \frac{1}{1 + e^{w^\top x + b}} = 1 - \pi(x)$$

利用极大似然法估计参数模型，似然函数为

$$\prod_{i=1}^m [\pi(x_i)]^{y_i} [1 - \pi(x_i)]^{1-y_i}$$

经典算法-Logistic回归

目标函数

$$\begin{aligned} L &= - \sum_{i=1}^m [y_i \ln \pi(x_i) + (1 - y_i) \ln(1 - \pi(x_i))] \\ &= - \sum_{i=1}^m \left[y_i \ln \frac{\pi(x_i)}{1 - \pi(x_i)} + \ln(1 - \pi(x_i)) \right] \\ &= - \sum_{i=1}^m [y_i(w^\top x_i + b) - \ln(1 + \exp(w^\top x_i + b))] \\ &= \sum_{i=1}^m [-y_i(\hat{w}^\top \hat{x}_i) + \ln(1 + \exp(\hat{w}^\top \hat{x}_i))] \end{aligned}$$

可以利用梯度下降法、牛顿法求解。

经典算法-Logistic回归

牛顿法

$$h'(x^{(k+1)}) = h'(x^{(k)}) + h''(x^{(k)})(x^{(k+1)} - x^{(k)}) = 0$$

本例中的迭代

$$\hat{w}^{(k+1)} = \hat{w}^{(k)} - \left(\frac{\partial^2 L}{\partial \hat{w} \partial \hat{w}^\top} \right)^{-1} \frac{\partial L}{\partial \hat{w}}$$

其中

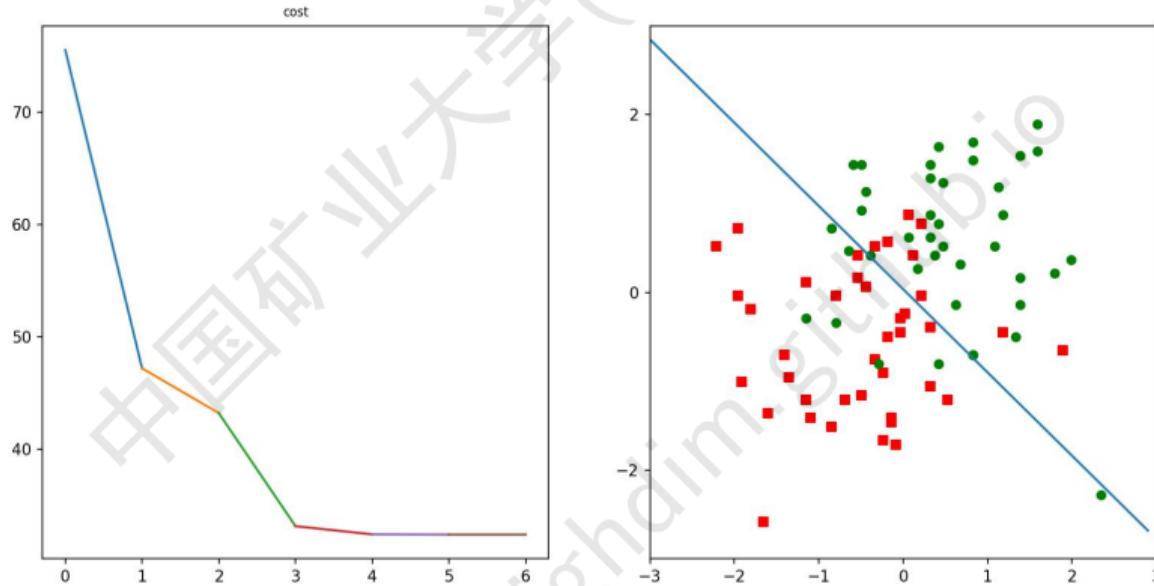
$$\frac{\partial L}{\partial \hat{w}} = - \sum_{i=1}^m \hat{x}_i (y_i - \pi(x_i))$$

$$\frac{\partial L}{\partial \hat{w} \partial \hat{w}^\top} = \sum_{i=1}^m \hat{x}_i \hat{x}_i^\top \pi(x_i) (1 - \pi(x_i))$$

(思考：如何推广到多分类？)

经典算法-Logistic回归

实例



对于一个二分类问题，定义如下指标

- 真正(True Positive, TP): 被模型预测为正的正样本;
- 假正(False Positive, FP): 被模型预测为正的负样本;
- 假负(False Negative, FN): 被模型预测为负的正样本;
- 真负(True Negative, TN): 被模型预测为负的负样本;

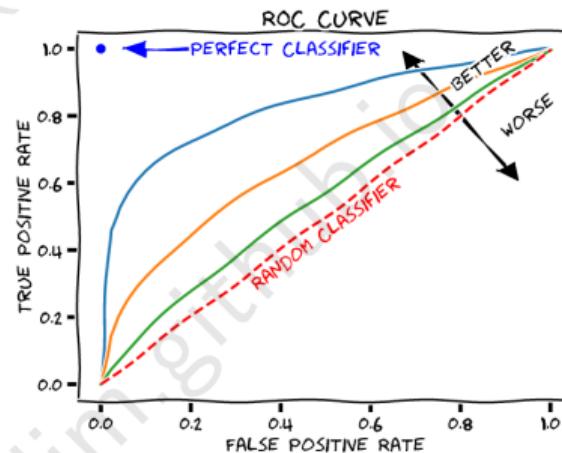
计算比例

- 真正率= $\frac{TP}{TP+FN}$
- 假正率= $\frac{FP}{TN+FP}$
- 假负率= $\frac{FN}{TP+FN}$
- 真负率= $\frac{TN}{TN+FP}$

常见的指标

- 准确率(Accuracy)= $\frac{TP+TN}{TP+FP+FN+TN}$
- 精度(Precision)= $\frac{TP}{TP+FP}$
- 召回率(Recall)= $\frac{TP}{TP+FN}$
- $F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$
- $F_\beta = \frac{(1+\beta^2) \times Precision \times Recall}{\beta^2 \times Precision + Recall}$
- P-R曲线

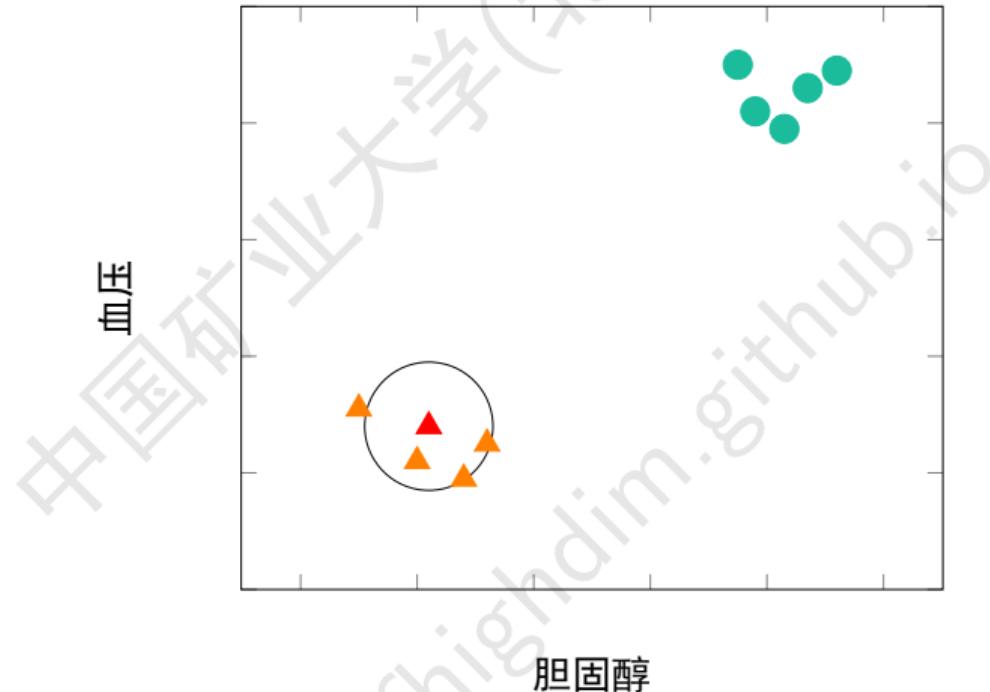
ROC(接受者操作特征曲线), AUC



排序关系

思考：为什么不用评估指标做损失函数？

经典算法-k近邻



经典算法-k近邻

k近邻: 给定测试样本, 基于某种距离度量找出训练集中与其最靠近的 k 个训练样本, 然后基于这 k 个“邻居”的信息来进行预测。这 k 个样本的多数属于某个类, 就把该样本分为这个类。

模型三要素: 距离度量、 k 值和分类决策规则。

优点:

- 算法简单、直观
- 可用于分类和回归
- 更适用于类域的交叉或重叠较多的分类样本集

缺点:

- 时间复杂度和空间复杂度高
- 训练样本不平衡时, 对稀有类别的预测准确率低

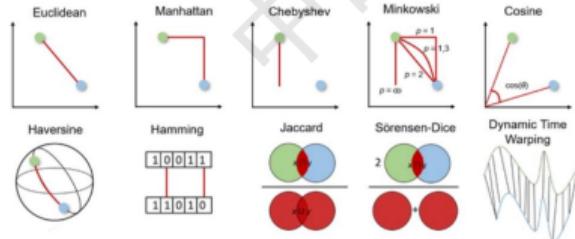
经典算法-k近邻

距离度量: 欧氏距离 $d = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$ 、

曼哈顿距离 $d = \sum_{i=1}^n |x_i - y_i|$ 、切比雪夫距

离 $d = \max_i |x_i - y_i|$ 、闵可夫斯基距

离 $d = \sqrt[p]{\sum_{i=1}^n (x_i - y_i)^p}$ 、余弦距离



例子: 已知二维空间的3个点, $x_1 = (1, 1)$, $x_2 = (5, 1)$, $x_3 = (4, 4)$, 试求在 p 取不同值时, L_p 距离下 x_1 的最近邻点.

解:

$$L_p(x_1, x_2) = 4$$

$$L_1(x_1, x_3) = 6$$

$$L_2(x_1, x_3) = 4.24$$

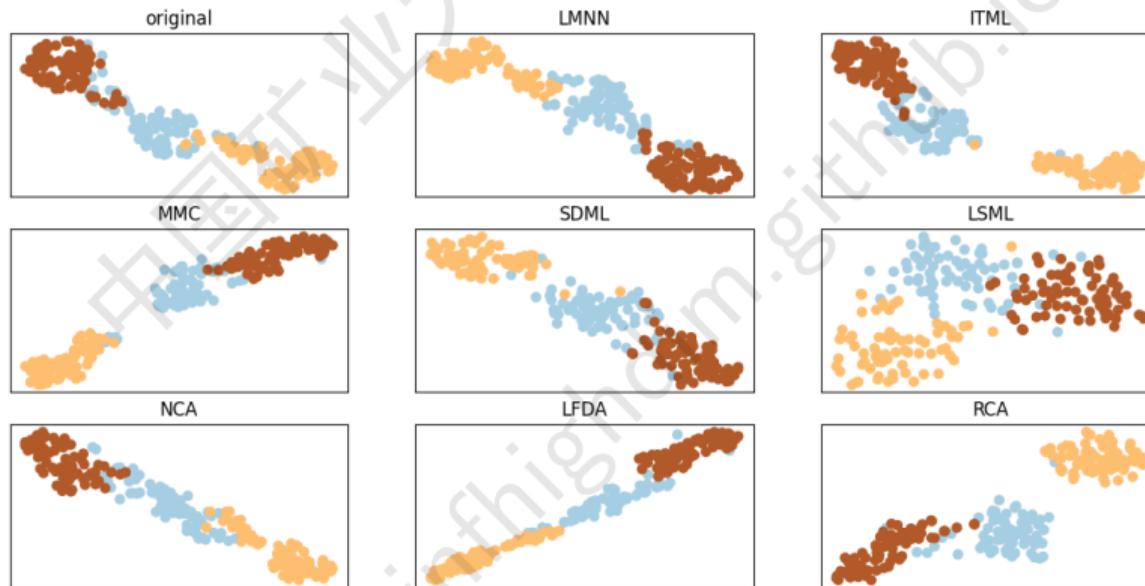
$$L_3(x_1, x_3) = 3.78$$

$$L_4(x_1, x_3) = 3.57$$

经典算法- k 近邻

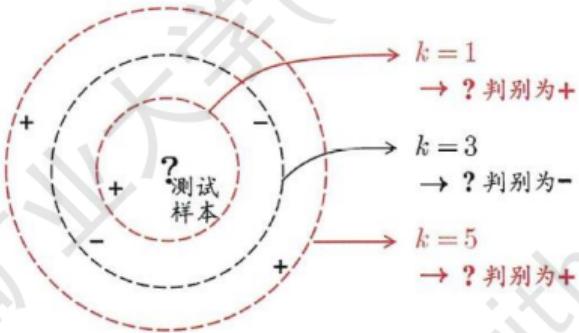
学习距离度量 M , 类内距离和类间距离分别为

$$d_S = \sum_{y_i=y_j} (x_i - x_j)^\top M(x_i - x_j), d_D = \sum_{y_i \neq y_j} (x_i - x_j)^\top M(x_i - x_j)$$



经典算法- k 近邻

k 值



近似误差、估计误差

k 值大小与模型复杂度的关系

经典算法-k近邻

分类决策规则-多数表决：如果分类的损失函数为0-1损失函数，分类函数为

$$f : R^n \rightarrow \{c_1, c_2, \dots, c_K\}$$

那么误分类的概率为

$$P(y \neq f(x)) = 1 - P(y = f(x))$$

对于给定的样本 x ，其最近邻的 k 个样本构成集合 $N_k(x)$ ，如果涵盖 $N_k(x)$ 的区域类别是 c_j ，那么误分类率是

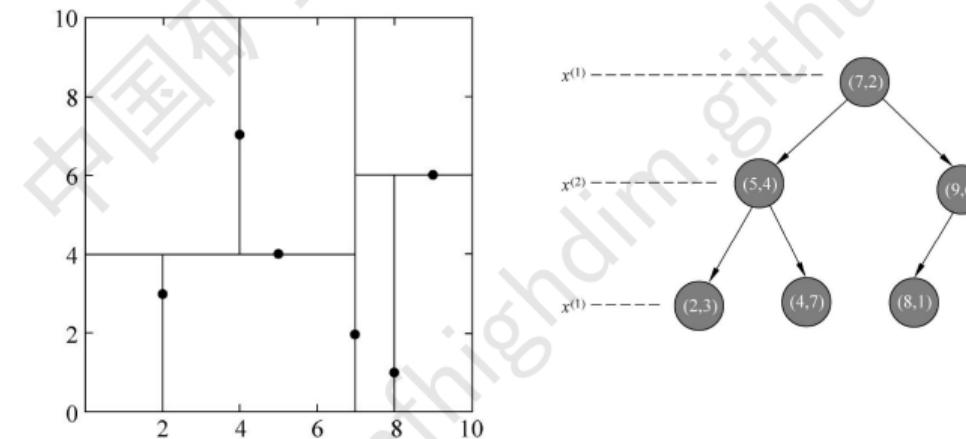
$$\frac{1}{k} \sum_{x_i \in N_k(x)} I(y_i \neq c_j) = 1 - \frac{1}{k} \sum_{x_i \in N_k(x)} I(y_i = c_j)$$

可以看到只有当 $\frac{1}{k} \sum_{x_i \in N_k(x)} I(y_i = c_j)$ 最大时，才能使误分类率最小即经验风险最小。

经典算法- k 近邻

kd 树构建

- 对训练数据进行快速 k 近邻搜索
- 对 k 维空间中的实例点进行存储以便对其进行快速检索的树形数据结构.
- 不断地用垂直于坐标轴的超平面将 k 维空间切分，构成一系列的 k 维超矩形区域.



kd树搜索：

- 二叉树搜索比较待查询节点和分裂节点的分裂维的值，（小于等于就进入左子树分支，大于就进入右子树分支直到叶子结点）
- 顺着“搜索路径”找到最近邻的近似点
- 回溯搜索路径，并判断搜索路径上的结点的其他子结点空间中是否可能有距离查询点更近的数据点，如果有可能，则需要跳到其他子结点空间中去搜索
- 重复这个过程直到搜索路径为空

例子：查找点(2, 4.5)的最近邻点

- 搜索路径中的结点为(7,2),(5,4),(4,7), 最近邻为(4,7), 距离为3.202;
- 回溯至(5,4), 距离为3.04, 更新为最近邻。以(2,4.5)为圆心, 以3.202为半径画一个圆, 它与超平面 $y=4$ 相交, 所以需要跳到(5,4)的左子空间去搜索。搜索路径中的结点为(7,2),(2, 3);
- 回溯至(2,3), (2,3)是叶子节点, 距离为1.5, 最近邻更新为(2,3), 距离更新为1.5;
- 回溯至(7,2), 距离大于1.5, 以(2,4.5)为圆心, 以1.5为半径画圆, 不和超平面 $x=7$ 相交, 搜索结束。

经典算法-朴素贝叶斯

机器学习所要实现的是基于有限的训练样本集尽可能准确地估计出后验概率 $P(y|x)$. 大体来说，主要有两种策略：

- 给定 x , 可通过直接建模 $P(y|x)$ 来预测 y , 这样得到的是"判别式模型" (discriminative models);
- 先对联合概率分布 $P(x,y)$ 建模, 然后再由此获得 $P(y|x)$, 这样得到的是"生成式模型" (generative models)。

对于生成式模型, 利用贝叶斯定理有

$$P(c|x) = \frac{P(x,c)}{P(x)} = \frac{P(c)P(x|c)}{P(x)}$$

$$P(Y = c_k | X = x) = \frac{P(X = x | Y = c_k)P(Y = c_k)}{\sum_k P(X = x | Y = c_k)P(Y = c_k)}$$

经典算法-朴素贝叶斯

条件概率分布 $P(X = x|Y = c_k)$ 有指数级数量的参数，其估计实际是不可行的。
朴素贝叶斯-条件独立性的假设：

$$\begin{aligned} P(Y = c_k|X = x) &= P(X^{(1)} = x^{(1)}, \dots, X^{(n)} = x^{(n)}|Y = c_k) \\ &= \prod_{j=1}^n P(X^{(j)} = x^{(j)}|Y = c_k) \end{aligned}$$

那么后验概率计算公式为

$$P(Y = c_k|X = x) = \frac{P(Y = c_k) \prod_j P(X^{(j)} = x^{(j)}|Y = c_k)}{\sum_k P(Y = c_k) \prod_j P(X^{(j)} = x^{(j)}|Y = c_k)}$$

经典算法-朴素贝叶斯

后验概率最大化准则：假设选择0-1损失函数

$$L(Y, f(X)) = \begin{cases} 1, & Y \neq f(X) \\ 0, & Y = f(X) \end{cases}$$

条件期望风险

$$R_{exp}(f) = E \left[\sum_{k=1}^K P(c_k|X) L(c_k, f(X)) \right]$$

为了使期望风险最小化

$$\begin{aligned} f(x) &= \arg \min_{y \in \mathcal{Y}} \sum_{k=1}^K P(c_k|X=x) L(c_k, y) \\ &= \arg \min_{y \in \mathcal{Y}} \sum_{k=1}^K P(y \neq c_k|X=x) = \arg \max_{y \in \mathcal{Y}} \sum_{k=1}^K P(y = c_k|X=x) \end{aligned}$$

经典算法-朴素贝叶斯

朴素贝叶斯利用极大似然估计来学习模型参数
首先估计先验概率

$$P(Y = c_k) = \frac{\sum_{i=1}^m I(y_i = c_k)}{m}$$

设第 j 个特征的可能取值集合为 $\{a_{j1}, a_{j2}, \dots, a_{jS_j}\}$, 那么条件概率的估计为

$$P(X^{(j)} = a_{jl}|Y = c_k) = \frac{\sum_{i=1}^m I(x_i^{(j)} = a_{jl}, y_i = c_k)}{\sum_{i=1}^m I(y_i = c_k)}$$

经典算法-朴素贝叶斯

利用如下数据学习一个朴素贝叶斯分类器并确定 $x = (2, S)$ 的标签 y 。

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$X^{(1)}$	1	1	1	1	1	2	2	2	2	2	3	3	3	3	3
$X^{(2)}$	S	M	M	S	S	S	M	M	L	L	L	M	M	L	L
Y	-1	-1	1	1	-1	-1	-1	1	1	1	1	1	1	1	-1

解：

$$P(Y = 1), P(Y = -1), P(X^{(1)} = 1|Y = 1), P(X^{(1)} = 2|Y = 1), P(X^{(1)} = 3|Y = 1)$$

$$P(X^{(2)} = S|Y = 1), P(X^{(2)} = M|Y = 1), P(X^{(2)} = L|Y = 1)$$

$$P(X^{(1)} = 1|Y = -1), P(X^{(1)} = 2|Y = -1), P(X^{(1)} = 3|Y = -1)$$

$$P(X^{(2)} = S|Y = -1), P(X^{(2)} = M|Y = -1), P(X^{(2)} = L|Y = -1)$$

经典算法-朴素贝叶斯

特殊情况下可能会出现某些要估计的概率为0，考虑使用拉普拉斯平滑。

$$P(X^{(j)} = a_{jl} | Y = c_k) = \frac{\sum_{i=1}^m I(x_i^{(j)} = a_{jl}, y_i = c_k) + \lambda}{\sum_{i=1}^m I(y_i = c_k) + S_j \lambda}$$

$$P(Y = c_k) = \frac{\sum_{i=1}^m I(y_i = c_k) + \lambda}{m + k \lambda}$$

经典算法-朴素贝叶斯

实例：基于上个例子的数据，使用拉普拉斯平滑学习一个朴素贝叶斯分类器

$$P(Y=1) = \frac{10}{17}, \quad P(Y=-1) = \frac{7}{17}$$

$$P(X^{(1)}=1|Y=1) = \frac{3}{12}, \quad P(X^{(1)}=2|Y=1) = \frac{4}{12}, \quad P(X^{(1)}=3|Y=1) = \frac{5}{12}$$

$$P(X^{(2)}=S|Y=1) = \frac{2}{12}, \quad P(X^{(2)}=M|Y=1) = \frac{5}{12}, \quad P(X^{(2)}=L|Y=1) = \frac{5}{12}$$

$$P(X^{(1)}=1|Y=-1) = \frac{4}{9}, \quad P(X^{(1)}=2|Y=-1) = \frac{3}{9}, \quad P(X^{(1)}=3|Y=-1) = \frac{2}{9}$$

$$P(X^{(2)}=S|Y=-1) = \frac{4}{9}, \quad P(X^{(2)}=M|Y=-1) = \frac{3}{9}, \quad P(X^{(2)}=L|Y=-1) = \frac{2}{9}$$

对于给定的 $x = (2, S)^T$ 计算：

$$P(Y=1)P(X^{(1)}=2|Y=1)P(X^{(2)}=S|Y=1) = \frac{10}{17} \cdot \frac{4}{12} \cdot \frac{2}{12} = \frac{5}{153} = 0.0327$$

$$P(Y=-1)P(X^{(1)}=2|Y=-1)P(X^{(2)}=S|Y=-1) = \frac{7}{17} \cdot \frac{3}{9} \cdot \frac{4}{9} = \frac{28}{459} = 0.0610$$

由于 $P(Y=-1)P(X^{(1)}=2|Y=-1)P(X^{(2)}=S|Y=-1)$ 最大，所以 $y = -1$ 。

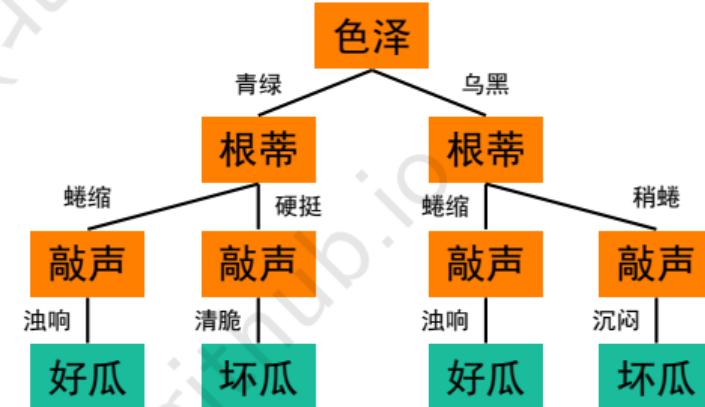
- ① 机器学习概述
- ② 逻辑斯蒂回归、k近邻和贝叶斯分类器
- ③ 决策树与随机森林
- ④ 支持向量机
- ⑤ 神经网络
- ⑥ 聚类分析

决策树

色泽	根蒂	敲击声	好瓜
青绿	蜷缩	浊响	是
乌黑	蜷缩	浊响	是
青绿	硬挺	清脆	否
乌黑	稍蜷	沉闷	否

决策树

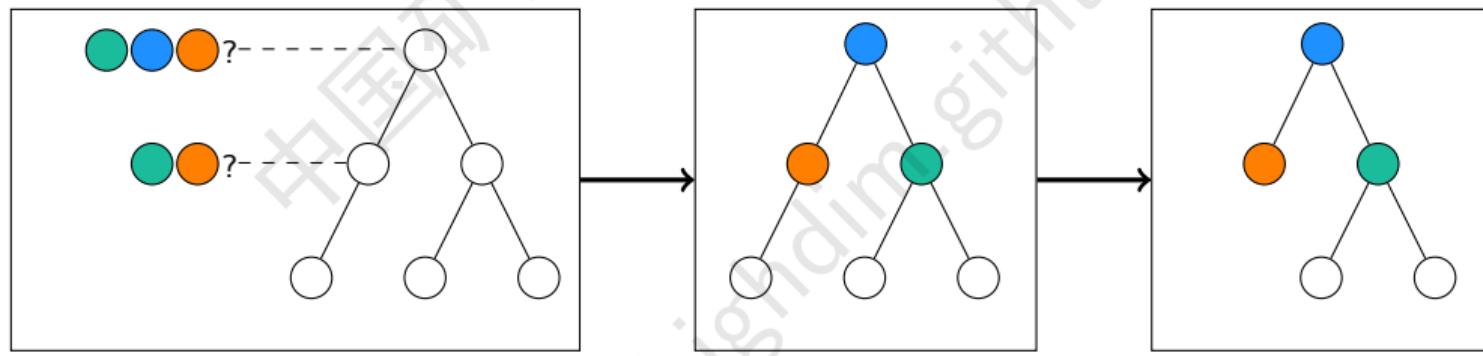
色泽	根蒂	敲击声	好瓜
青绿	蜷缩	浊响	是
乌黑	蜷缩	浊响	是
青绿	硬挺	清脆	否
乌黑	稍蜷	沉闷	否



决策树

定义：分类决策树模型是一种描述对实例进行分类的树形结构。决策树由结点（node）和有向边（directed edge）组成。结点有两种类型：内部结点（internal node）和叶结点（leaf node）。内部结点表示一个特征或属性，叶结点表示一个类。

算法步骤：特征选择、决策树的生成和决策树的修剪。



特征选择

决策树生成

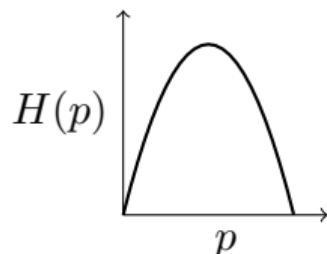
决策树剪枝

决策树

熵: 随机变量不确定性。设 X 是一个取有限个值的离散随机变量，其概率分布为 $P(X = x_i) = p_i$ ，则随机变量 X 的熵定义为

$$H(X) = H(p) = - \sum_{i=1}^n p_i \log p_i$$

熵越大，随机变量的不确定性就越大。
熵 $H(p)$ 随概率 p 变化的曲线



条件熵: $H(Y|X)$ 在已知随机变量 X 的条件下随机变量 Y 的不确定性。 X 给定条件下 Y 的条件概率分布的熵对 X 的数学期望

$$H(Y|X) = \sum_{i=1}^n p_i H(Y|X = x_i)$$

当熵和条件熵中的概率由数据估计(特别是极大似然估计)得到时，所对应的熵与条件熵分别称为**经验熵**(empirical entropy)和**经验条件熵**(empirical conditional entropy)。

决策树

信息增益(information gain): 得知特征 X 的信息而使得类 Y 的信息的不确定性减少的程度。特征 A 对训练数据集 D 的信息增益 $g(D, A)$, 定义为集合 D 的经验熵 $H(D)$ 与特征 A 给定条件下 D 的经验条件熵 $H(D|A)$ 之差

$$g(D, A) = H(D|A) - H(D)$$

- 经验熵 $H(D)$ 表示对数据集 D 进行分类的不确定性;
- 经验条件熵 $H(D|A)$ 表示在特征 A 给定的条件下对数据集 D 进行分类的不确定性;
- 特征 A 使得对数据集 D 的分类的不确定性减少的程度。

信息增益比: 在训练数据集的经验熵大的时候, 信息增益值会偏大。

$$g_R(D, A) = \frac{g(D, A)}{H(D)}$$

决策树

特征选择：决策树的分支结点所包含的样本尽可能属于同一类别，即结点的“纯度”(purity)越来越高。

利用信息增益来做特征选择：首先计算数据集的经验熵

$$H(D) = - \sum_{k=1}^K \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}$$

然后计算特征 A 对数据集 D 的经验条件熵

$$H(D|A) = \sum_{i=1}^m \frac{|D_i|}{|D|} H(D_i) = - \sum_{i=1}^m \frac{|D_i|}{|D|} \sum_{k=1}^K \frac{|D_{ik}|}{|D_i|} \log_2 \frac{|D_{ik}|}{|D_i|}$$

决策树

特征选择-信贷数据集

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	香
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

$$H(D) = -\frac{9}{15} \log_2 \frac{9}{15} - \frac{6}{15} \log_2 \frac{6}{15} = 0.971$$

分别以 A_1, A_2, A_3, A_4 表示年龄、有工作、有自己的房子和信贷情况4个特征

$$\begin{aligned} g(D, A_1) &= H(D) - \left[\frac{5}{15} H(D_1) + \frac{5}{15} H(D_2) + \frac{5}{15} H(D_3) \right] \\ &= 0.083 \end{aligned}$$

$$\begin{aligned} g(D, A_2) &= H(D) - \left[\frac{5}{15} H(D_1) + \frac{10}{15} H(D_2) \right] \\ &= 0.324 \end{aligned}$$

$$g(D, A_3) = 0.420$$

$$g(D, A_4) = 0.363$$

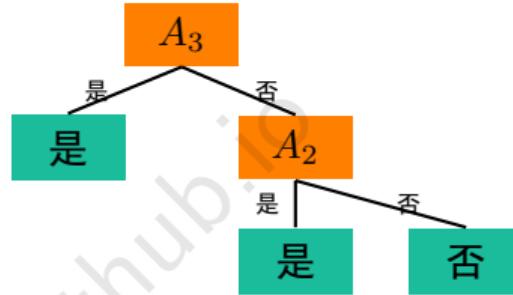
决策树

ID3：在决策树各个结点上应用信息增益准则选择特征，递归地构建决策树。

- 特征 A_3 (有自己的房子)的信息增益值最大，将训练数据集 D 划分为两个子集 $D_1(A_3$ 取值为“是”) $)$ 和 $D_2(A_3$ 取值为“否”);
- 对 D_2 则需从特征 A_1 (年龄)， A_2 （有工作）和 A_4 （信贷情况）中选择新的特征。分别计算信息增益

$$g(D_2, A_1) = 0.251, g(D_2, A_2) = 0.918, g(D_2, A_4) = 0.474$$

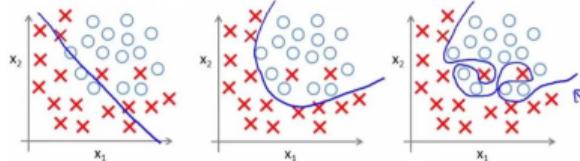
- A_2 将训练数据集 D 划分为两个子集



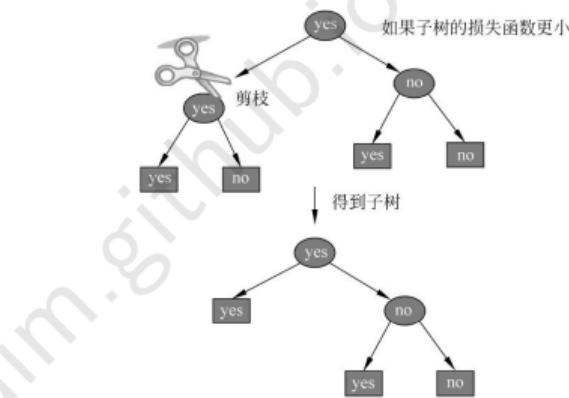
C4.5：使用信息增益比来选择特征。

决策树

过拟合(overfitting): 对训练数据的分类很准确，但对未知的测试数据的分类却没有那么准确。



剪枝(pruning): 极小化决策树整体的损失函数(loss function)或代价函数(cost function)



- 预剪枝(prunning): 生成过程中剪枝
- 后剪枝(postpruning): 生成完成后剪枝

决策树–CART分类树

基尼指数(Breiman, 1984): 分类问题中, 假设有 K 个类, 样本点属于第 k 类的概率为 p_k , 则概率分布的基尼指数定义为

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

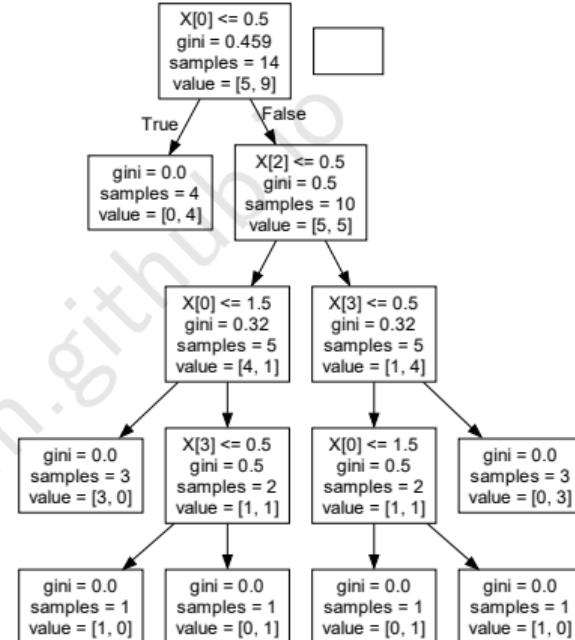
基尼指数 $Gini(D)$ 表示集合 D 的不确定性, 基尼指数 $Gini(D, A)$ 表示经 $A = a$ 分割后集合 D 的不确定性。基尼指数值越大, 样本集合的不确定性也就越大。

算法步骤:

- 计算现有特征对该数据集的基尼指数;
- 选择基尼指数最小的特征及其对应的切分点作为最优特征与最优切分点;
- 依次递归, 生成二叉决策树。

决策树-CART分类树

年龄	收入	学生	信用	买电脑
<= 30	高	否	一般	否
<= 30	高	否	好	否
31 – 40	高	否	一般	是
> 40	中	否	一般	是
> 40	低	是	一般	是
> 40	低	是	好	否
31 – 40	低	是	好	是
<= 30	中	否	一般	否
<= 30	低	是	一般	是
> 40	中	是	一般	是
<= 30	中	中	好	是
31 – 40	中	中	好	是
31 – 40	高	否	一般	否
> 40	中	否	好	是



决策树-CART回归树

对于回归问题，构建回归树，基于平方误差最小化的原则寻找最优切分变量和切分点。

(1) 对于第 j 个变量 $x^{(j)}$ 和取值 s ，定义两个区域

$$R_1(j, s) = \{x | x^{(j)} \leq s\}, R_2(j, s) = \{x | x^{(j)} \geq s\}$$

(2) 然后通过求解如下最优化问题来获得 (j, s)

$$\min_{j, s} \left[\min_{c_1} \sum_{x_i \in R_1(j, s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j, s)} (y_i - c_2)^2 \right]$$

(3) 计算输出值

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m(j, s)} y_i, m = 1, 2$$

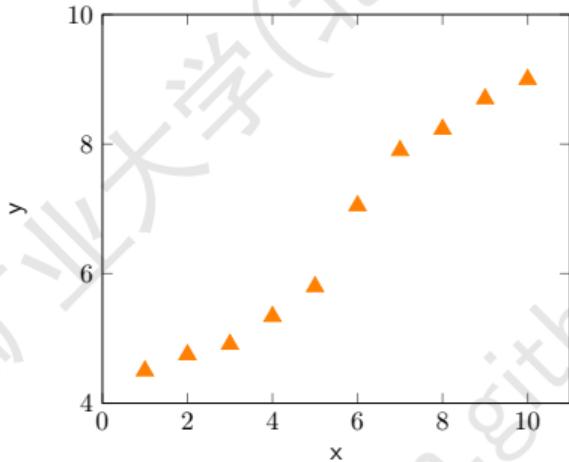
决策树-CART回归树

- (4) 重复如上步骤，直至满足停止条件(树的深度、叶子结点数量)；
(5) 将输入空间划分为 M 个区域 R_1, R_2, \dots, R_M ，最终决策树为

$$f(x) = \sum_{m=1}^M \hat{c}_m I(x \in R_m)$$

最小二乘回归树(least squares regression tree)

决策树-CART回归树



步骤：对特征 x ，取 $s = 1$ ，先求两个区域，再分别求得区域均值，然后计算方差，依次计算 $s = 2, \dots, 10$

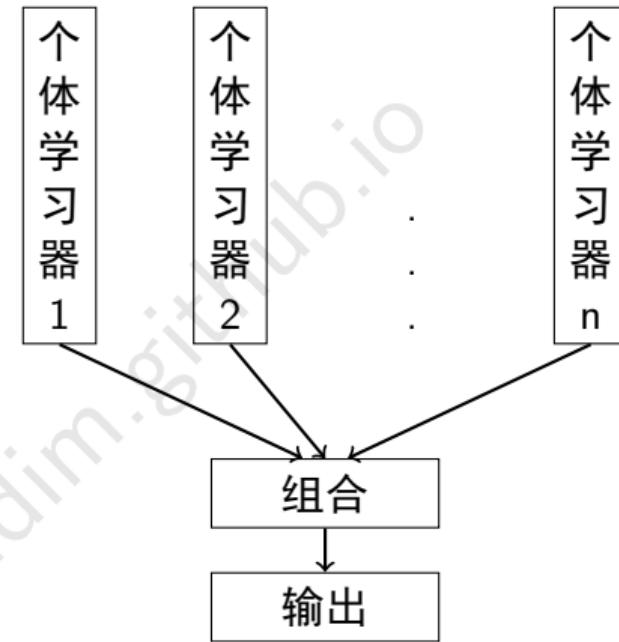
s	1	2	3	4	5	6	7	8	9	10
c_1	4.50	4.63	4.72	4.88	5.06	5.39	5.75	5.18	6.35	6.62
c_2	6.85	7.12	7.43	7.78	8.18	8.46	8.64	8.85	9.00	0
$v(s)$	22.65	17.7	12.19	7.38	3.36	5.07	10.05	15.18	21.33	27.63

随机森林

集成学习：构建并结合多个学习器
来完成学习任务

基本步骤

- 学习若干个不同的个体学习器
- 利用某种策略将多个学习器进行组合



随机森林

集成学习的效果

	测试例1	测试例2	测试例3
h_1	✓	✓	✗
h_2	✗	✓	✓
h_3	✓	✗	✓
集成	✓	✓	✓

(a) 集成提升性能

	测试例1	测试例2	测试例3
h_1	✓	✓	✗
h_2	✓	✓	✗
h_3	✓	✓	✗
集成	✓	✓	✗

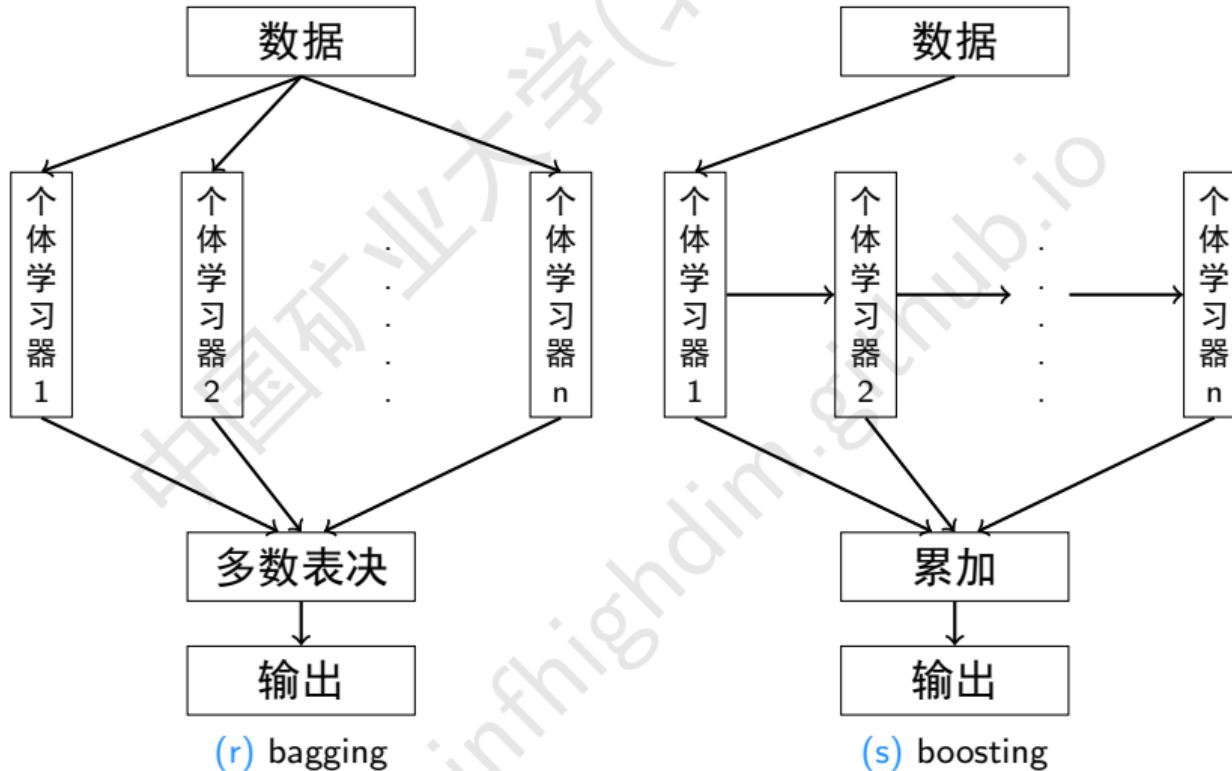
(b) 集成不起作用

	测试例1	测试例2	测试例3
h_1	✓	✗	✗
h_2	✗	✓	✗
h_3	✗	✗	✓
集成	✗	✗	✗

(c) 集成起负作用

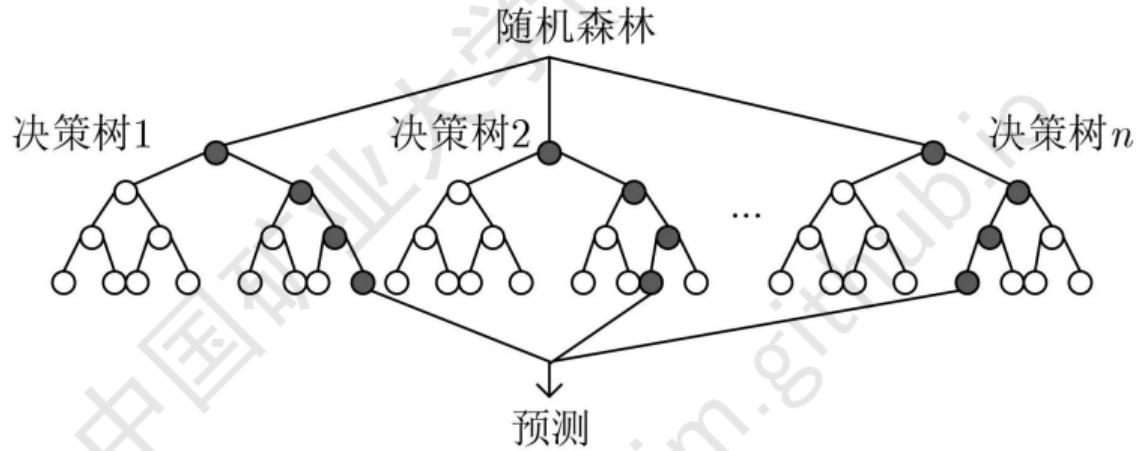
个体学习器应该**好而不同**: 既要有准确性又要有多样性

随机森林



随机森林

随机森林：一种由多个决策树构成的集成算法，不同决策树之间没有关联



随机性：样本随机、特征随机

- 样本：自助采样法(bootstrap sampling);
- 特征：从特征集合中随机选取 k 个特征.

随机森林

优点

- 采用了集成算法，精度优于大多数单模型算法；
- 样本和特征随机性的引入降低了过拟合风险；
- 训练过程中能检测特征重要性，是常见的特征筛选方法；
- 每棵树可以同时生成，并行效率高，训练速度快。

缺点

- 在某些噪音较大的分类或回归问题上会过拟合；
- 对于有不同取值的属性的数据，取值划分较多的属性会对随机森林产生更大的影响。

梯度提升树

对于一般的boosting算法，以树作为基分类器，约定 $f_t(x)$ 表示第 t 轮的模型， $h_t(x)$ 表示第 t 轮学习得到的决策树，迭代步骤

$$f_t(x) = f_{t-1}(x) + h_t(x)$$

损失函数定义为

$$L(f_t(x), y) = L(f_{t-1}(x) + h_t(x), y)$$

最终的模型可以表示为

$$f_t(x) = \sum_{t=1}^T h_t(x)$$

思考：如何确定 $h_t(x)$ ？

梯度提升树

Freidman提出了用损失函数的负梯度来拟合本轮损失的近似值，进而拟合一个CART回归树。

$$r_{t,i} = -\frac{\partial L(f(x_i), y)}{\partial f(x_i)}$$

在第 t 轮迭代中，要拟合的数据为 $(x_i, r_{t,i})$ ，获得第 t 棵回归树 $h_t(x)$

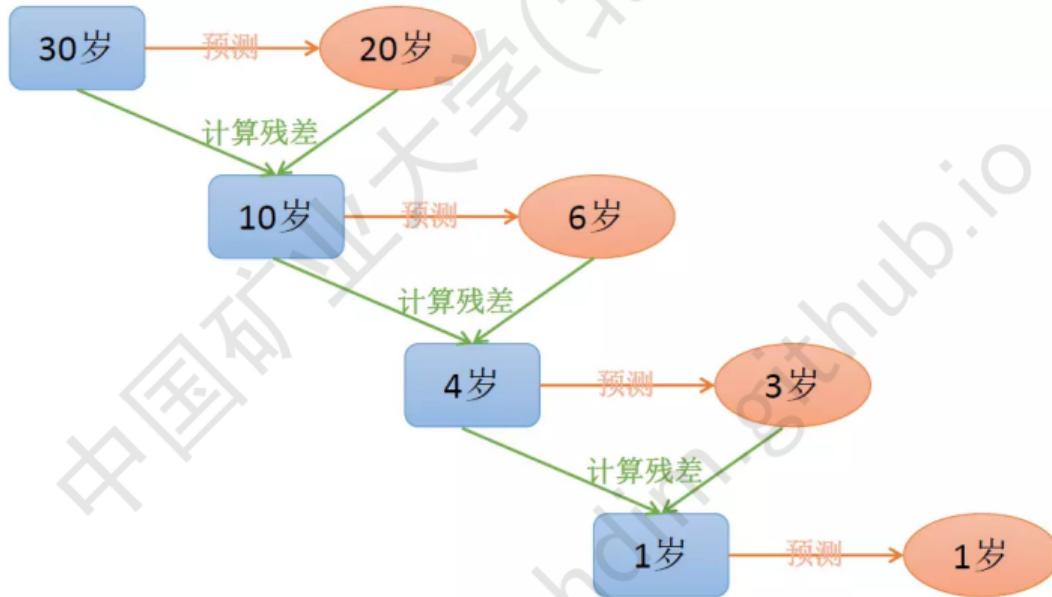
$$h_t(x) = \sum_{j=1}^J c_{t,j} I(x \in R_{t,j})$$

本轮的模型为

$$f_t(x) = f_{t-1}(x) + \sum_{j=1}^J c_{t,j} I(x \in R_{t,j})$$

定义：以CART回归树作为基学习器的boosting算法。

梯度提升树



梯度提升树

优点

- 采用基于负梯度的Boosting集成手段
- 适用于回归和分类任务
- 预测精度比随机森林高
- 对异常值的鲁棒性强(采用了Huber损失和分位数损失)

缺点

- 串行方式的模型训练，难并行，造成计算开销
- 不适合高维稀疏离散特征

梯度提升树

随机森林与梯度提升树对比

区别点	随机森林	GBDT
集成方式	bagging	boosting
决策树类型	分类树、回归树	CART回归树
结合方式	并行生成	顺序生成
优化指标	方差优化	残差优化
训练样本	有放回抽样	全样本
表决方式	多数表决	累加之和

Xgboost

Xgboost(eXtreme Gradient Boost)是分布式梯度提升决策树 (GBDT) 机器学习库

与gbdt区别

- 二阶导数信息
- 特征采样
- 分裂节点寻找的近似算法
- Shrinkage思想

相关资料：

<https://xgboost.readthedocs.io/en/stable/>

```
import xgboost as xgb  
xgb.XGBClassifier()  
xgb.XGBRegressor()
```

实例

① 机器学习概述

② 逻辑斯蒂回归、k近邻和贝叶斯分类器

③ 决策树与随机森林

④ 支持向量机

⑤ 神经网络

⑥ 聚类分析

支持向量机–SVM

支持向量机(Support Vector Machine, SVM)

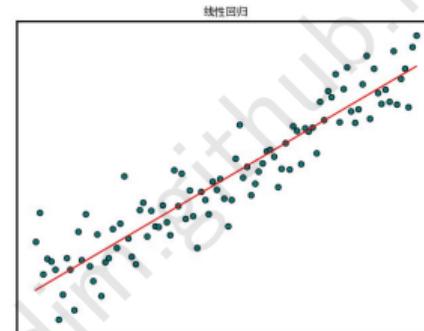
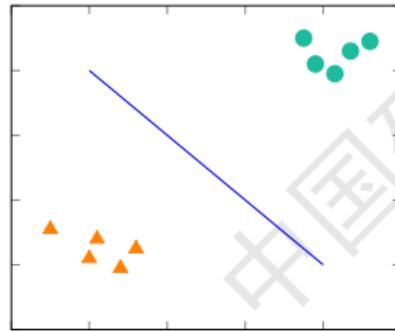


- 1964年由Vapnik提出, 线性SVM
- 二十世纪70-80年代, SVM 成为统计学习一部分
- 引入核函数, 提出非线性SVM
- 二十世纪90年代后得到快速发展

支持向量机-SVM

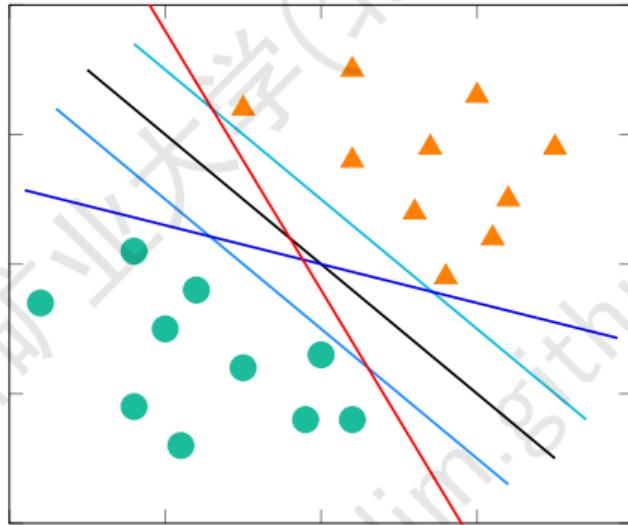
按照问题划分

- 分类问题：支持向量分类机(Support Vector Classifier, SVC)
- 回归问题：支持向量回归机(Support Vector Regressor, SVR)



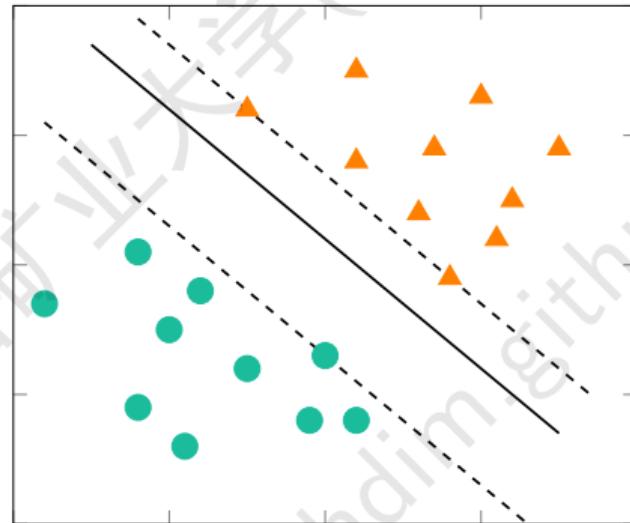
按照形式划分

- 线性支持向量机
- 非线性支持向量机



直观理解：最中间的直线、容错性最好、最鲁棒、泛化能力最强

如何寻找最优直线？



支持向量机-SVC

对于任意一条可以将两类点划分开的直线，假设其方程为 $w^\top x + b = 0$ ，那么直线两边的点满足如下等式

$$\begin{cases} w^\top x_i + b > 0, & y_i = +1 \\ w^\top x_i + b < 0, & y_i = -1 \end{cases}$$

将直线向正负类点平移，遇到第一个正类和负类点时停下，获得两条直线(支持直线)

$$w^\top x_i + b = a_+, w^\top x_i + b = a_-$$

当两条直线的间隔

$$\frac{|a_+ - a_-|}{\|w\|}$$

最大时，中间的那条直线

$$w^\top x_i + b = \frac{a_+ + a_-}{2}$$

就是我们要寻找的最佳分类直线。

支持向量机-SVC

为了获得更简单的表达形式，我们将中间线和两条支持直线的形式改写为

$$w^\top x_i + b - \frac{a_+ + a_-}{2} = 0$$

$$w^\top x_i + b - \frac{a_+ + a_-}{2} = \frac{a_+ - a_-}{2}$$

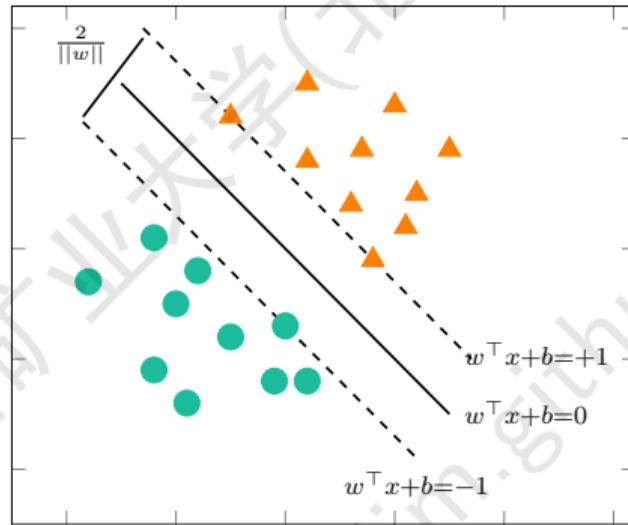
$$w^\top x_i + b - \frac{a_+ + a_-}{2} = \frac{a_- - a_+}{2}$$

将所有等式的两边都除以 $\frac{a_+ - a_-}{2}$ ，并且令

$$w = w / (\frac{a_+ - a_-}{2}), b = (b - \frac{a_+ + a_-}{2}) / (\frac{a_+ - a_-}{2})$$

那么中间线和两条支持直线的形式变为

$$w^\top x_i + b = 0, w^\top x_i + b = \pm 1$$



样本进一步满足

$$\begin{cases} w^\top x_i + b \geq +1, & y_i = +1 \\ w^\top x_i + b \leq -1, & y_i = -1 \end{cases}$$

支持向量机-SVC

构建如下最优化问题

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

等价于

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

知识回顾：凸集、凸函数、凸规划、拉格朗日乘子法

支持向量机-SVC

为求解问题，使用Lagrange乘子法将其转化为对偶问题。首先引入Lagrange函数

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

其中 $\alpha = (\alpha_1, \dots, \alpha_m)$ 是Lagrange乘子。

首先求Lagrange函数关于 w, b 的极小值。由极值条件有：

$$\nabla_w L(w, b, \alpha) = 0, \nabla_b L(w, b, \alpha) = 0$$

得到

$$w = \sum_{i=1}^m \alpha_i y_i x_i$$

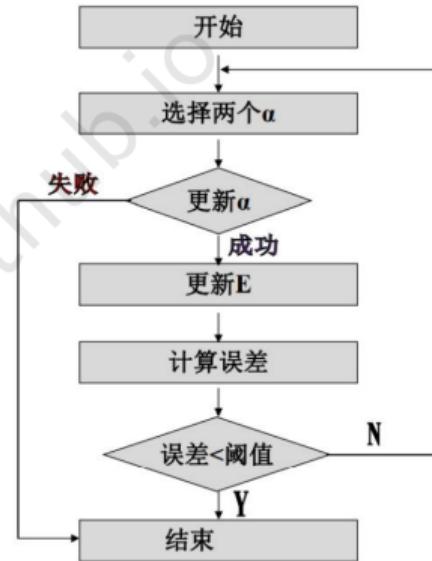
$$0 = \sum_{i=1}^m \alpha_i y_i$$

支持向量机-SVC

代入对偶问题得到

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

利用SMO (Sequential Minimal Optimization)求解



由于原始问题中存在不等式约束，因此上述推导满足KKT(Karush-Kuhn-Tucker)条件，即

$$\begin{cases} \alpha_i \geq 0, \\ y_i f(x_i) - 1 \geq 0, \\ \alpha_i(y_i f(x_i) - 1) = 0. \end{cases}$$

第三个称为互补松弛条件，即对于任意样本 (x_i, y_i) ，总有 $\alpha_i = 0$ 或 $y_i f(x_i) = 1$.

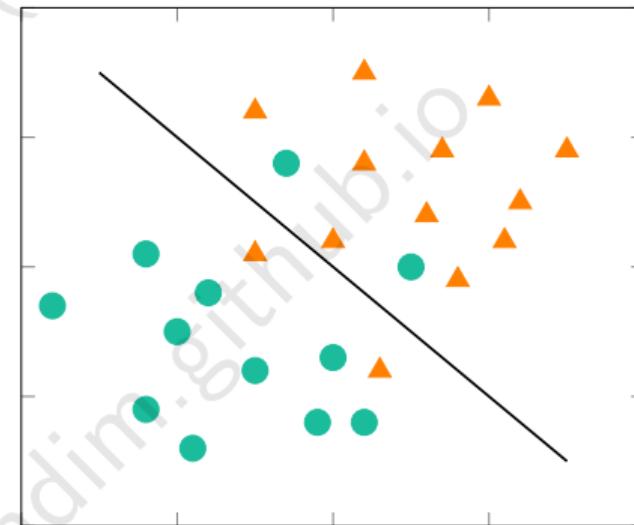
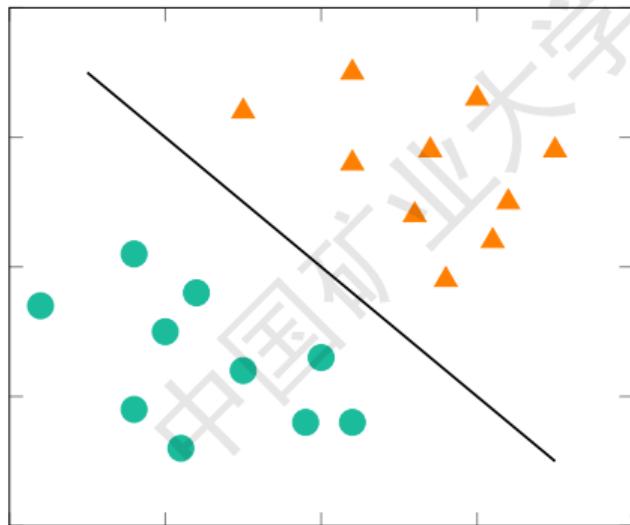
- $\alpha_i = 0$, 样本对分割直线无影响
- $\alpha_i > 0$, 则 $y_i f(x_i) = 1$, 样本为支持向量

分割直线只受到支持向量的影响 (解的稀疏性)

求解b:

$$y_i(w^\top x_i + b) = 1 \Rightarrow b = y_i - w^\top x_i$$

硬间隔与软间隔



支持向量机-SVC

软间隔SVC

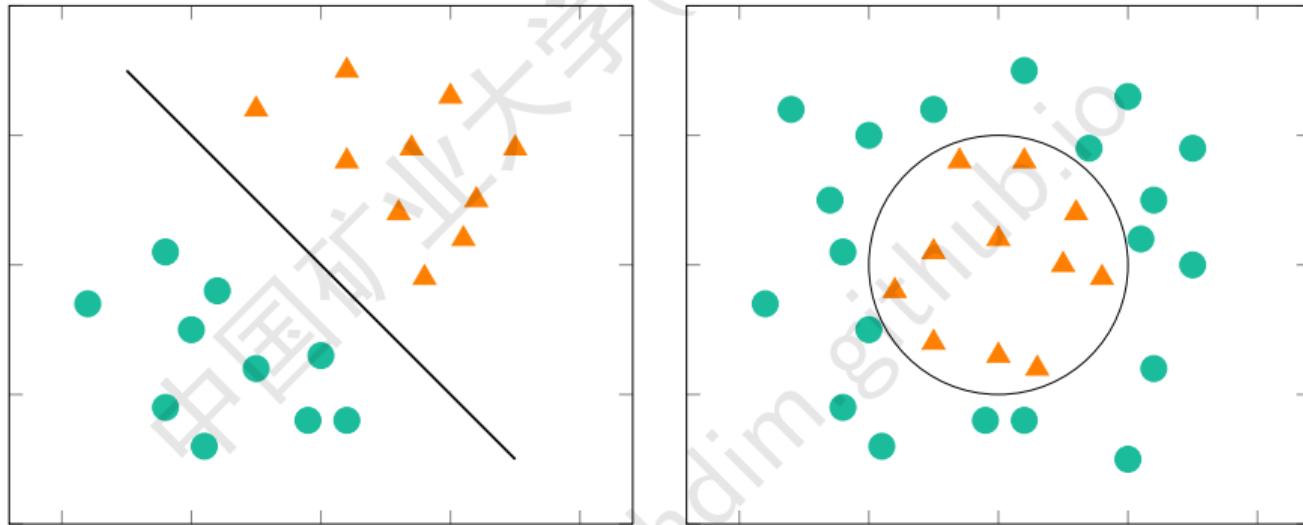
$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \xi_i \\ s.t. \quad & y_i(w^\top x_i + b) \geq 1 - \xi_i, i = 1, 2, \dots, m \end{aligned}$$

对偶问题

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j x_i^\top x_j \\ s.t. \quad & \sum_{i=1}^m \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C, i = 1, \dots, m \end{aligned}$$

支持向量机–kernel

线性不可分



令 $\phi(x)$ 表示将 x 映射后的向量，我们要学习的超平面为

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

那么最优化问题为

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

对偶问题

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

需要计算 $\phi(x_i)^\top \phi(x_j)$, 引入核函数(kernel function)

$$k(x_i, x_j) = \phi(x_i)^\top \phi(x_j)$$

支持向量机–kernel

核函数定义：令 X 为输入空间， $k(\cdot, \cdot)$ 是定义在 $\mathcal{X} \times \mathcal{X}$ 上的对称函数， k 是核函数当且仅当对于任意数据 $D = \{x_1, \dots, x_m\}$ ，核矩阵 K 总是半正定的。

$$K = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \cdots & k(x_m, x_m) \end{bmatrix}$$

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。
核函数的选择影响SVM的效果。

常见的核函数

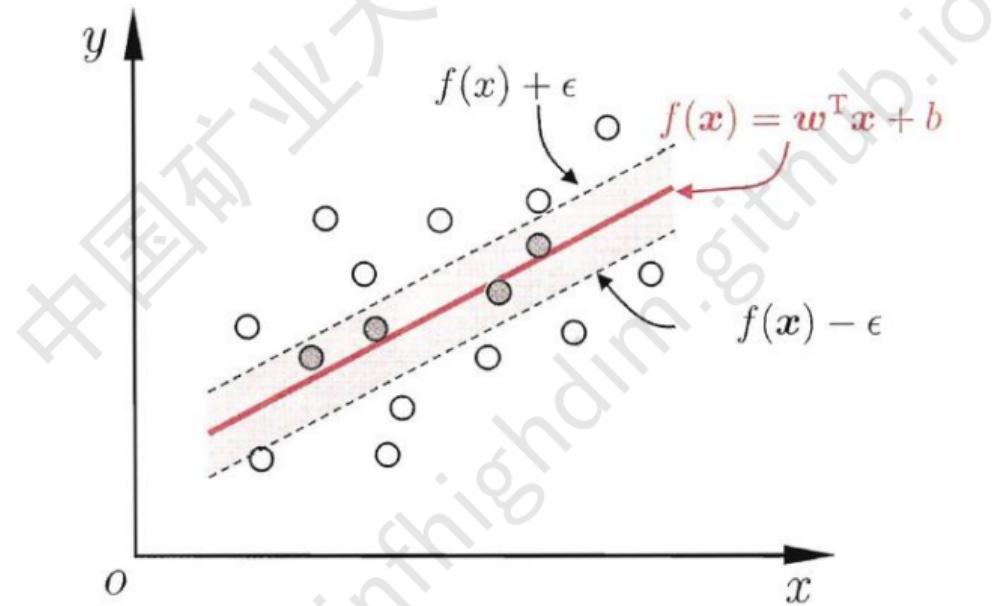
名称	表达式	参数
线性核	$k(x_i, x_j) = x_i^\top x_j$	
多项式核	$k(x_i, x_j) = (x_i^\top x_j)^d$	$d \geq 1$ 为多项式次数
高斯核	$k(x_i, x_j) = \exp(-\frac{\ x_i - x_j\ ^2}{2\sigma^2})$	$\sigma > 0$ 为高斯核的带宽
拉普拉斯核	$k(x_i, x_j) = \exp(-\frac{\ x_i - x_j\ }{\sigma})$	$\sigma > 0$
Sigmoid核	$k(x_i, x_j) = \tanh(\beta x_i^\top x_j + \theta)$	\tanh 为双曲正切函数

组合核函数：加权和、直积

支持向量机-SVR

传统回归与SVR对比

- 传统回归，尽量处于数据分布的中心位置
- 容忍一定的偏差，构建 2ϵ 间隔带



支持向量机-SVR

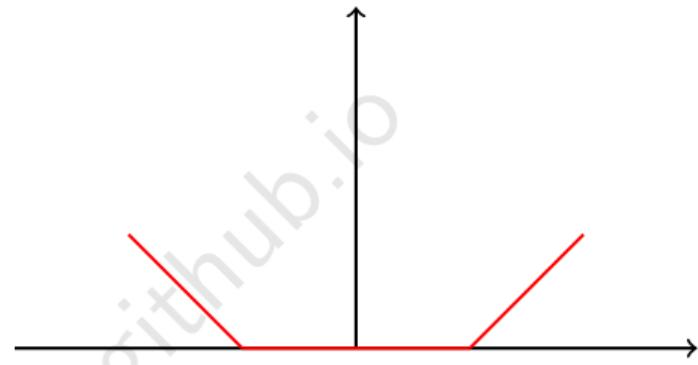
SVR构建如下最优化问题

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

其中 l_ϵ 是 ϵ 不敏感损失(ϵ -insensitive loss)函数

$$l_\epsilon = \begin{cases} 0, & \text{if } |z| \leq \epsilon \\ |z| - \epsilon, & \text{otherwise} \end{cases}$$

l_ϵ 函数图像



支持向量机-SVR

引入松弛变量 $\xi_i, \hat{\xi}_i$, 原始最优化问题改写为

$$\begin{aligned} & \min_{w, b, \xi_i, \hat{\xi}_i} \quad \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m (\xi_i + \hat{\xi}_i) \\ & s.t. \quad f(x_i) - y_i \leq \epsilon + \xi_i, \\ & \quad y_i - f(x_i) \leq \epsilon + \hat{\xi}_i, \\ & \quad \xi_i \geq 0, \hat{\xi}_i \geq 0, i = 1, 2, \dots, m \end{aligned}$$

引入拉格朗日乘子 $\alpha_i \geq 0, \hat{\alpha}_i \geq 0, \mu_i \geq 0, \hat{\mu}_i \geq 0$, 构建拉格朗日函数

$$\begin{aligned} L(w, b, \alpha, \hat{\alpha}, \xi, \hat{\xi}, \mu, \hat{\mu}) &= \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m (\xi_i + \hat{\xi}_i) - \sum_{i=1}^m \mu_i \xi_i - \sum_{i=1}^m \hat{\mu}_i \hat{\xi}_i \\ &+ \sum_{i=1}^m \alpha_i (f(x_i) - y_i - \epsilon - \xi_i) + \sum_{i=1}^m \hat{\alpha}_i (y_i - f(x_i) - \epsilon - \hat{\xi}_i) \end{aligned}$$

支持向量机-SVR

令 L 对 $w, b, \xi_i, \hat{\xi}_i$ 偏导为零，可以得到

$$w = \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) x_i, 0 = \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i)$$

$$C = \alpha_i + \mu_i, C = \hat{\alpha}_i + \hat{\mu}_i$$

带入原始问题，获得如下对偶问题

$$\begin{aligned} \max_{\alpha, \hat{\alpha}} \quad & \sum_{i=1}^m y_i (\hat{\alpha}_i - \alpha_i) - \epsilon_i (\hat{\alpha}_i + \alpha_i) - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m (\hat{\alpha}_i - \alpha_i)(\hat{\alpha}_j - \alpha_j) x_i^\top x_j \\ s.t. \quad & \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) = 0, \\ & 0 \leq \alpha_i, \hat{\alpha}_i \leq C \end{aligned}$$

支持向量机-SVR

根据KKT条件

$$\begin{cases} \alpha_i(f(x_i) - y_i - \epsilon - \xi_i) = 0, \\ \hat{\alpha}_i(y_i - f(x_i) - \epsilon - \hat{\xi}_i) = 0, \\ (C - \alpha_i)\xi_i = 0, (C - \hat{\alpha}_i)\hat{\xi}_i = 0 \end{cases}$$

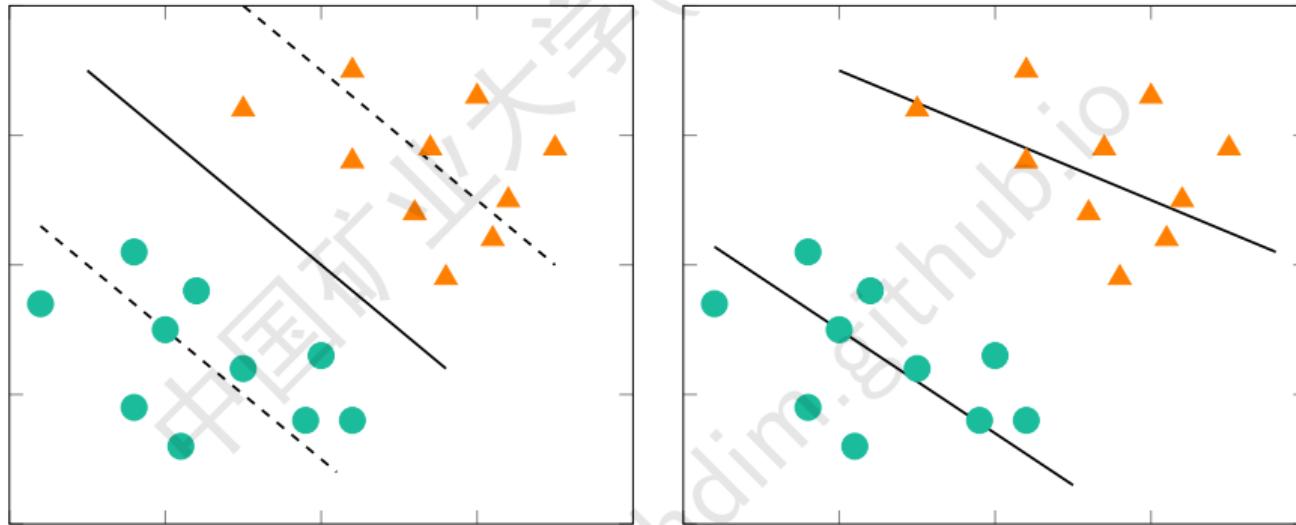
只有当样本在 ϵ 间隔带之外时， $\alpha_i, \hat{\alpha}_i$ 才能取非零值。

根据 w 的表达式，可知 $\alpha_i - \hat{\alpha}_i \neq 0$ 才对模型有贡献，所以SVR的解仍然具有稀疏性。

求解得到 α_i 后，若 $0 < \alpha_i < C$ ，则 $\xi_i = 0$ ，那么

$$b = y_i + \epsilon - \sum_{i=1}^m (\hat{\alpha}_i - \alpha_i) x_i^\top x$$

最小二乘SVM与双胞胎支持向量机



支持向量机

软件包—libsvm, liblinear, sklearn

<https://www.csie.ntu.edu.tw/~cjlin/libsvm/>

<https://www.csie.ntu.edu.tw/~cjlin/liblinear/>

<https://www.ics.uci.edu/~mlearn/pythonlibs/>

```
from sklearn import svm
```

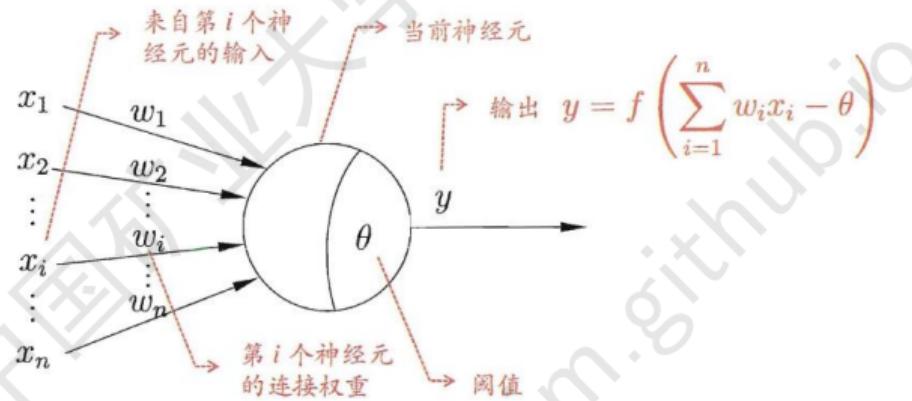
支持向量机

实例

- ① 机器学习概述
- ② 逻辑斯蒂回归、k近邻和贝叶斯分类器
- ③ 决策树与随机森林
- ④ 支持向量机
- ⑤ 神经网络
- ⑥ 聚类分析

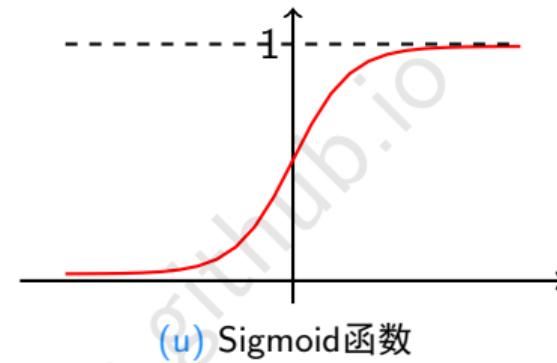
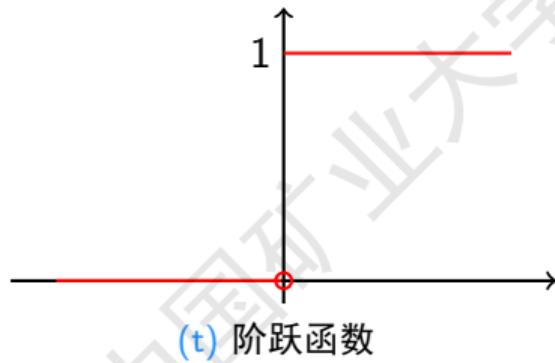
神经网络

神经元模型：神经元互相连接，当其电位超过了“一个”阈值” (threshold)，那么它就会被激活，即“兴奋”起来，向其他神经元发送化学物质。



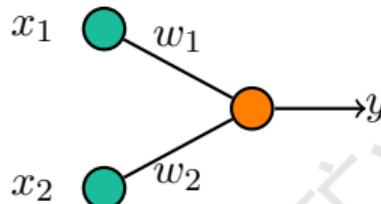
把许多个这样的神经元按一定的层次结构连接起来，就得到了**神经网络**(neural networks)。
人工神经网络(artifical neural networks, ANN)：由具有适应性的简单**单元**组成的广泛并行互连的**网络**，它的组织能够**模拟**生物神经系统对真实世界物体所作出的交互反应。

常见激活函数



神经网络-感知机

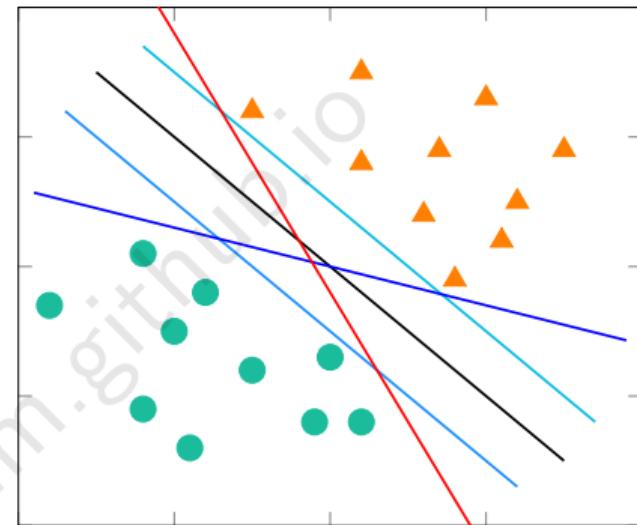
感知机(Perceptron): 一种线性分类模型,
由两层神经元组成, 包含输入层与输出层



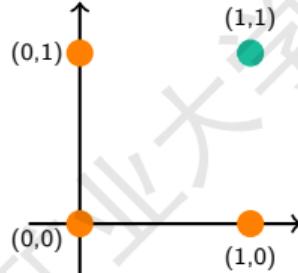
给定数据集, 学习从输入空间到输出空间的如下函数

$$f(x) = \text{sign}(w^\top x + b)$$

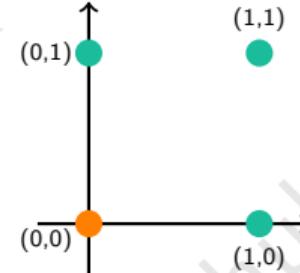
其中 $\text{sign}(\cdot)$ 是符号函数。



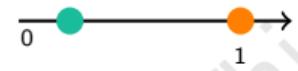
神经网络-感知机



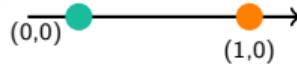
(a) 与



(b) 或



(c) 非



(d) 异或

神经网络-感知机

假设训练数据集是线性可分的，感知机寻找一个能够将正负类点完全正确分开的分离超平面。

数学表达: 定义损失函数并最小化。

损失函数: 误分类点到超平面的总距离。

对于样本 x_0 , 其到超平面的距离为

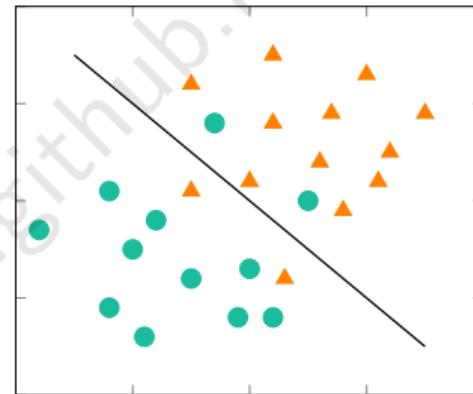
$$\frac{1}{\|w\|}(w^\top x_0 + b)$$

由于误分类点满足 $-y_i(w^\top x_i + b) > 0$, 所以误分类点到分割平面的总距离为

$$D = -\frac{1}{\|w\|} \sum_{x_i \in M} y_i(w^\top x_i + b)$$

最终损失函数为

$$L(w, b) = - \sum_{x_i \in M} y_i(w^\top x_i + b)$$



误分类点越少，误分类点离超平面越近，损失函数值就越小

随机梯度下降法(stochastic gradient descent)

首先初始化 w, b , 然后计算损失函数关于 w, b 的梯度

$$\nabla_w L(w, b) = - \sum_{x_i \in M} y_i x_i$$

$$\nabla_b L(w, b) = - \sum_{x_i \in M} y_i$$

随机选取一个误分类点, 对 w, b 进行更新

$$w \leftarrow w + \eta y_i x_i$$

$$b \leftarrow b + \eta y_i$$

当一个实例点被误分类, 即位于分离超平面的错误一侧时, 则调整 w, b 的值, 使分离超平面向该误分类点的一侧移动, 以减少该误分类点与超平面间的距离, 直至超平面越过该误分类点使其被正确分类.

神经网络-感知机

例题：给定训练集，正类 $x_1 = (3, 3)^\top$, $x_2 = (4, 3)^\top$, 负类 $x_3 = (1, 1)^\top$, 学习一个感知机模型 $f(x) = \text{sign}(w^\top x + b)$.

求解：

(1) 取初始值 $w_0 = 0, b_0 = 0$;

(2) 对 $x_1 = (3, 3)^\top$, 有 $y_1(w_0^\top x_1 + b_0) = 0$, 未能被正确分类, 更新 w, b

$$w_1 = w_0 + y_1 x_1 = (3, 3)^\top, b_1 = b_0 + y_1 = 1$$

得到线性模型

$$w_1^\top x + b_1 = 3x^{(1)} + 3x^{(2)} + 1$$

(3) 显然 x_1, x_2 正确分类, x_3 误分类, 更新 w, b

$$w_2 = w_1 + y_3 x_3 = (2, 2)^\top, b_2 = b_1 + y_3 = 0$$

得到线性模型

$$w_2^\top x + b_2 = 2x^{(1)} + 2x^{(2)}$$

神经网络-感知机

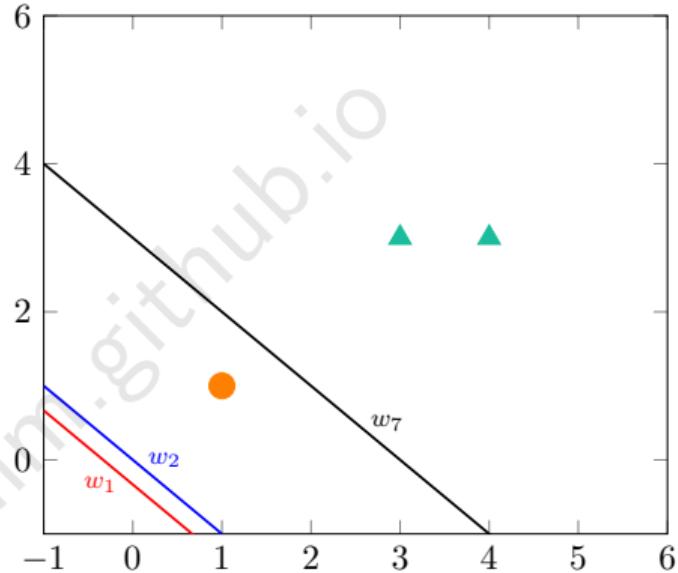
如此继续下去，直到

$$w_7 = (1, 1)^\top, b_7 = -3$$

$$w_7^\top x + b_7 = x^{(1)} + x^{(2)} - 3$$

最终的分割直线为

$$x^{(1)} + x^{(2)} - 3 = 0$$



问题的对偶形式
对 w, b 进行更新

$$w \leftarrow w + \eta y_i x_i$$

$$b \leftarrow b + \eta y_i$$

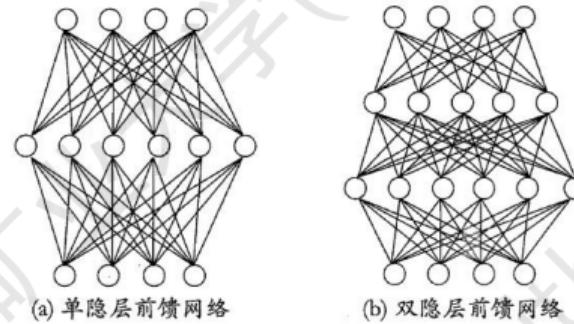
最后学习到的 w, b 分别表示为

$$w = \sum_{i=1}^m \alpha_i y_i x_i$$

$$b = \sum_{i=1}^m \alpha_i y_i$$

神经网络

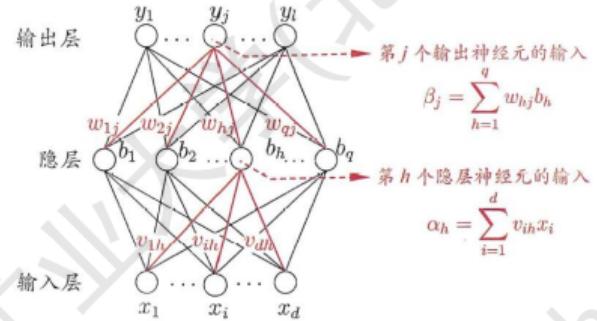
要解决非线性可分问题，需要多层感知机-多层前馈神经网络



结构：输入层、隐藏层、输出层

前馈：不存在同层连接、跨层连接

神经网络



对于某个训练样本 (x_k, y_k) , 网络输出为 $\hat{y}_k = (\hat{y}_1^k, \hat{y}_2^k, \dots, \hat{y}_l^k)$, 即

$$\hat{y}_j^k = f(\beta_j + b^j)$$

均方误差为

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

神经网络

使用误差反向传播(error BackPropagation, 简称BP)算法求解网络权重: 基于梯度下降法, 将梯度信息从后向前传播, 不断更新网络参数直至收敛

对误差 E_k , 给定学习率 η , 有

$$\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{hj}}$$

使用链式法则

$$\begin{aligned}\frac{\partial E_k}{\partial w_{hj}} &= \frac{\partial E_k}{\partial \hat{y}_j^k} \frac{\partial \hat{y}_j^k}{\partial \beta_j} \frac{\partial \beta_j}{\partial w_{hj}} \\ &= -(\hat{y}_j^k - y_j^k) \cdot \hat{y}_j^k (1 - \hat{y}_j^k) \cdot b_h\end{aligned}$$

同理可求得 Δv_{ih}

标准BP算法与累计BP算法

思考：如何设置层数和每层神经元个数？

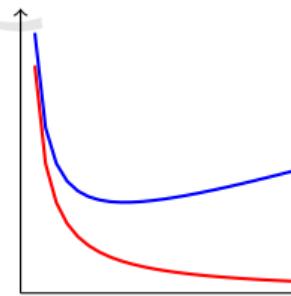
[Hornik et al., 1989]证明只需一个包含足够多神经元的隐层，多层前馈网络就能以任意精度逼近任意复杂度的连续函数。

$$\hat{y} = f(x) = f_1(f_2(\cdots(f_n(x))))$$

然而，如何设置隐层神经元的个数仍是个未决问题，实际应用中通常靠“试错法”(trial-by-error) 调整。

过拟合应对策略

- 早停：将数据分成训练集和验证集，若训练集误差降低但验证集误差升高，则停止训练；

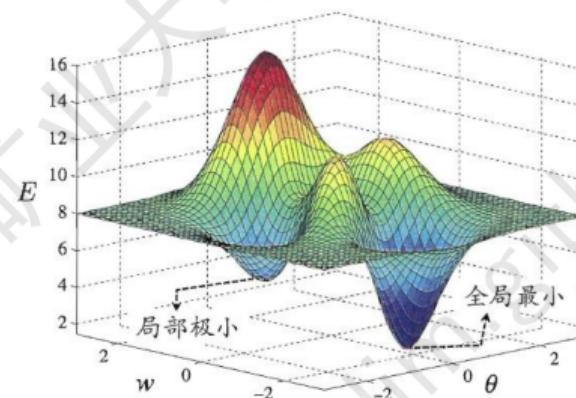


- 正则化：在误差目标函数中增加一个用于描述网络复杂度的部分

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

思考：网络的损失函数是不是凸函数？

全局最小与局部最小



- 以多组不同参数值初始化多个神经网络
- 模拟退火
- 随机梯度下降
- 遗传算法

神经网络

RBF(Radial Basis Function, 径向基函数)网络: 一种单隐层前馈神经网络, 它使用径向基函数作为隐层神经元激活函数, 网络输出可以表示为

$$\phi(x) = \sum_{i=1}^q w_i e^{-\beta_i \|x - c_i\|^2}$$

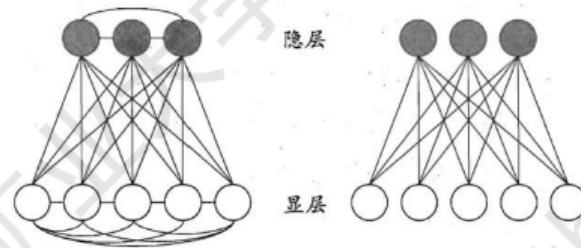
[Park and Sandberg, 1991]证明, 具有足够的隐层神经元的RBF 网络能以任意精度逼近任意连续函数.

训练步骤:

- 确定神经元中心 c_i , 常用的方式包括随机采样、聚类等;
- 利用BP算法等来确定参数 w_i, β_i .

神经网络

Boltzmann机: 一种“基于能量的模型”(energy-based model), 为网络状态定义一个“能量”(energy), 能量最小化时网络达到理想状态, 而网络的训练就是在最小化这个能量函数。



状态向量 s 对应的网络能量定义为

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

状态向量的概率为

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

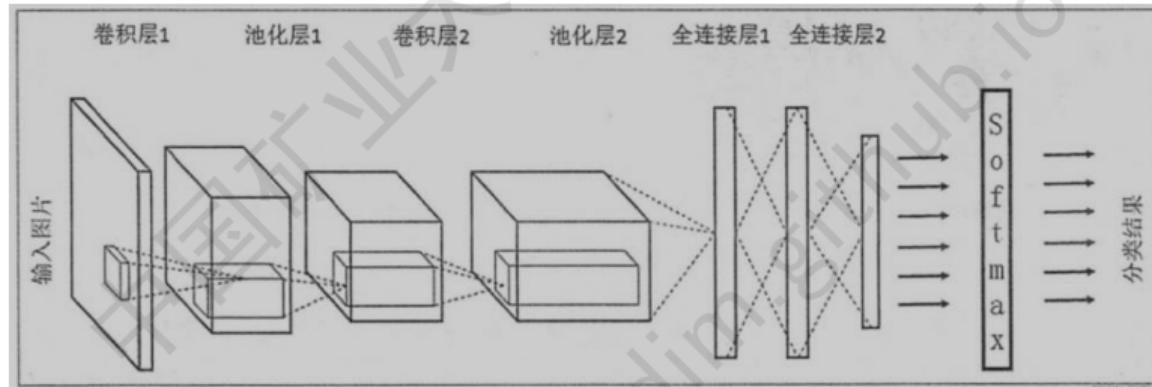
模型缺点

- 黑箱性质：无法得知神经网络如何以及为何会得出一定的输出；
- 试错性：参数的设置缺乏理论指导，主要靠手工调参；
- 网络复杂度高：时间和空间；
- 容易过拟合；

神经网络的发展在20世纪90年代陷入低潮，在21世纪初以深度学习的名义卷土重来，掀起了人工智能领域的革命。

卷积神经网络

卷积神经网络(Convolutional neural network, CNN)是一种专门用来处理具有类似网格结构的数据的神经网络。卷积网络是指那些至少在网络的一层中使用卷积运算来替代一般的矩阵乘法运算的神经网络。



输入层(input layer), 卷积层(convolution layer), 池化层(pooling layer), 激活函数层和全连接层(full-connection layer), softmax层

输入层-图像结构

图像单元

- 像素
- 像素点
- 像素值
- 通道

图像类别

- 二值图像
- 灰度图像
- 彩色图像



(a) 二值图像 (b) 灰度图像



(c) 彩色图像

卷积神经网络

卷积操作：卷积核大小、卷积核个数、步长

0	0	0	0	0	0	0	...
0	156	155	156	158	158	158	...
0	153	154	157	159	159	159	...
0	149	151	155	158	159	159	...
0	146	146	149	153	158	158	...
0	145	143	143	148	158	158	...
...

Input Channel #1 (Red)

0	0	0	0	0	0	0	...
0	167	166	167	169	169	169	...
0	164	165	168	170	170	170	...
0	160	162	166	169	170	170	...
0	156	156	159	163	168	168	...
0	155	153	153	158	168	168	...
...

Input Channel #2 (Green)

0	0	0	0	0	0	0	...
0	163	162	163	165	165	165	...
0	160	161	164	166	166	166	...
0	156	158	162	165	166	166	...
0	155	155	158	162	167	167	...
0	154	152	152	157	167	167	...
...

Input Channel #3 (Blue)

-1	-1	1
0	1	-1
0	1	1

Kernel Channel #1



308

1	0	0
1	-1	-1
1	0	-1

Kernel Channel #2



-498

0	1	1
0	1	0
1	-1	1

Kernel Channel #3



164

$$+ 1 = -25$$

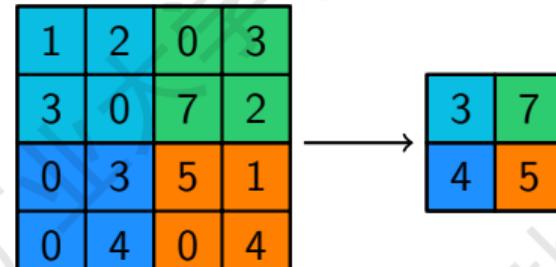
-25				...
				...
				...
				...
...

Output

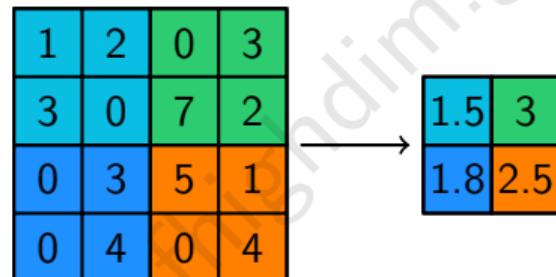
$$\text{Bias} = 1$$

卷积神经网络

池化操作(pooling): 下采样-最大池化、平均池化、随机池化



(a) 最大池化



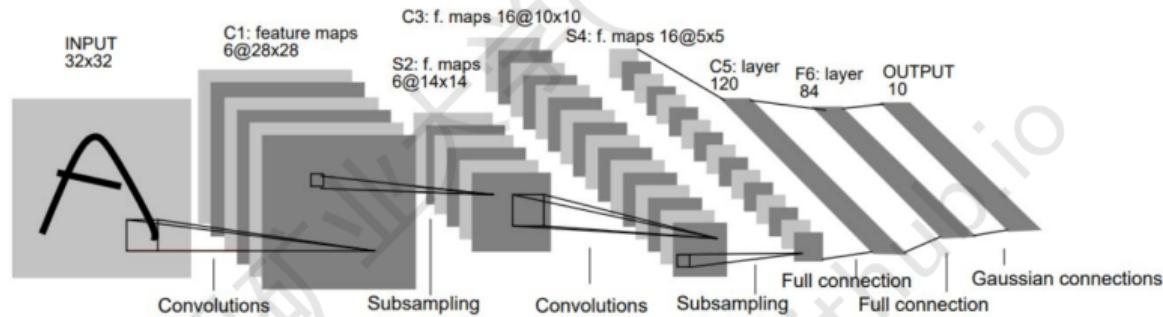
(b) 平均池化

三大特点

- 局部感知：每个神经元只与输入数据的一个局部区域连接，该连接的空间大小叫做神经元的感受野
- 权重共享：每个滤波器的所有局部连接都使用同样的参数
- 多卷积核：每个卷积核提取一种特征，多卷积核提取多个角度的特征

卷积神经网络

LeNet



输入层: 32x32

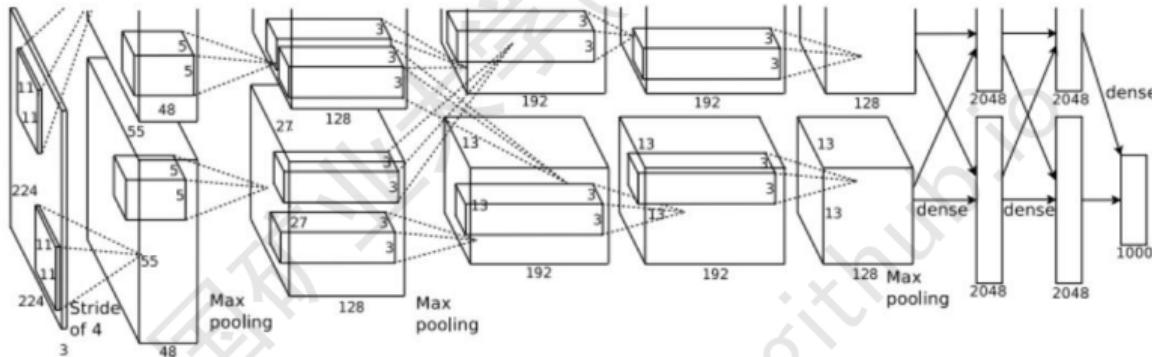
C1层: $6@5 \times 5 \rightarrow 6@28 \times 28$

S2层: $6@2 \times 2 \rightarrow 6@14 \times 14$

C3层: $16@5 \times 5 \rightarrow 16@10 \times 10$

卷积神经网络

AlexNet(8层)



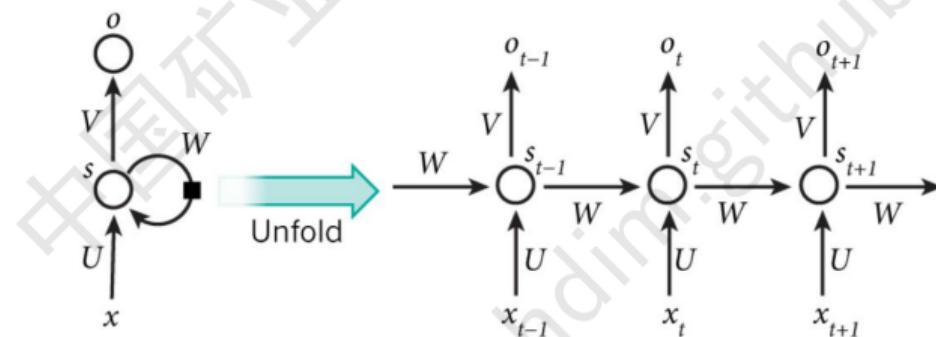
VGG(19层)、GoogleNet(27层)、ResNet(18-152层)

循环神经网络

循环神经网络(recurrent neural network, RNN): 处理序列数据, 挖掘数据中的时序信息以及语义信息, 包括文本、语音、时间序列数据等。

动机: 序列数据需要结合上下文才能做出准确的判断

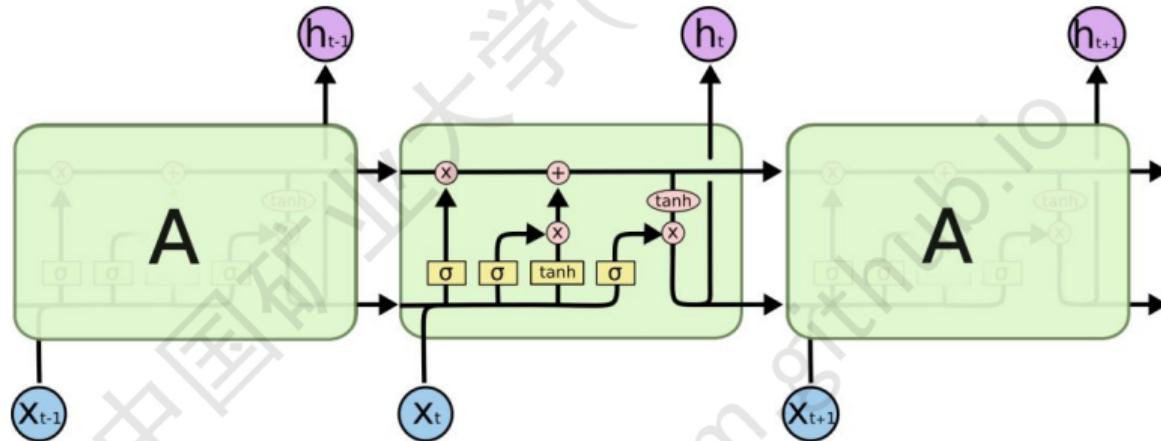
- ①我喜欢吃苹果!
- ②苹果是一家美国公司。



应用: 机器翻译、文本生成、语音识别

循环神经网络

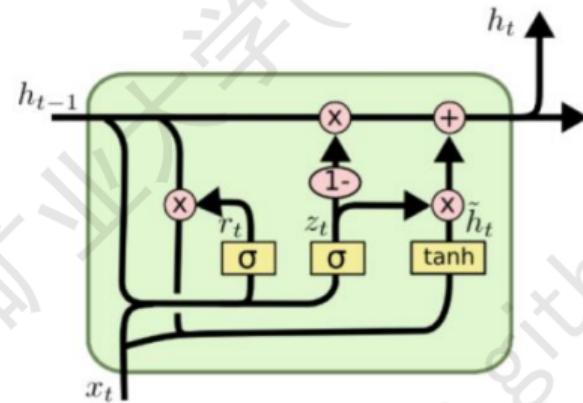
LSTM-长短期记忆网络



关键部分：遗忘门、输入门、输出门

循环神经网络

GRU-门控循环单元



关键部分：重置门、更新门

神经网络

Transformer、Bert、GPT-3、chatGPT、GPT4



神经网络

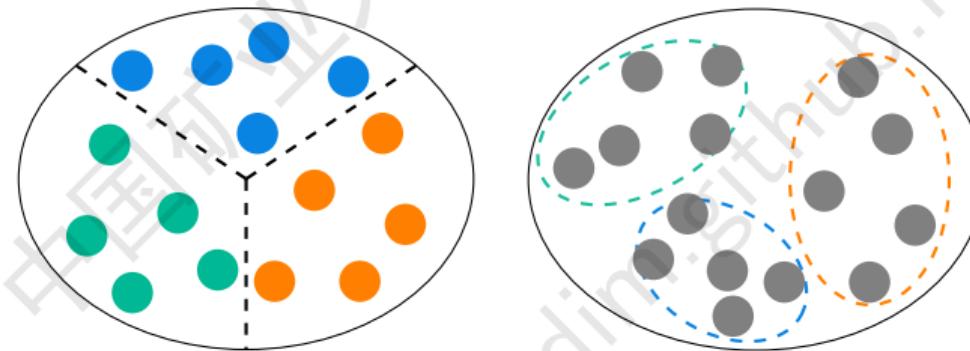
编程框架：



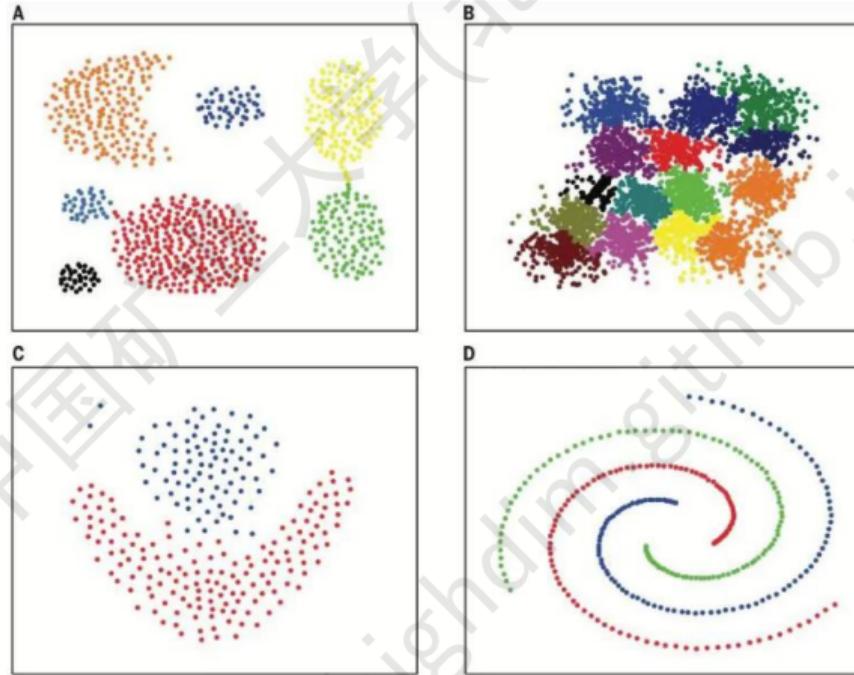
- ① 机器学习概述
- ② 逻辑斯蒂回归、k近邻和贝叶斯分类器
- ③ 决策树与随机森林
- ④ 支持向量机
- ⑤ 神经网络
- ⑥ 聚类分析

聚类分析

问题定义：聚类学习是按照某种特定标准(如距离等)把一个数据集划分为不同的类或簇(子集)，使得同一个簇内的数据对象的相似性尽可能大，不在同一个簇中的数据对象的差异性也尽可能地大(即聚类后同一类的数据尽可能聚集到一起，不同类数据尽量分离)。



聚类分析



聚类分析- k -均值

k 均值(k -means)聚类：利用迭代求解的方式将给定数据集划分为指定个数的簇。

给定样本集 $D = \{x_1, x_2, \dots, x_m\}$, 假设算法将集合划分为 k 个簇 C_1, C_2, \dots, C_k , 定义损失函数为

$$E = \sum_{i=1}^k \sum_{x \in C_i} \|x - \mu_i\|^2$$

其中

$$\mu_i = \frac{1}{m_i} \sum_{x \in C_i} x$$

求解困难，使用贪心策略迭代求解

求解步骤：

- ① 随机选择初始向量作为簇心；
- ② 计算每个样本点到所有簇心的距离，将样本划入到距离最近的簇中；
- ③ 重新计算每个簇的簇心，如果簇心与第1步不同，重复步骤2，否则停止迭代，获得最终的簇；
- ④ 重复上面2,3两步操作，直到达到某个中止条件(迭代次数、最小误差变化等)。

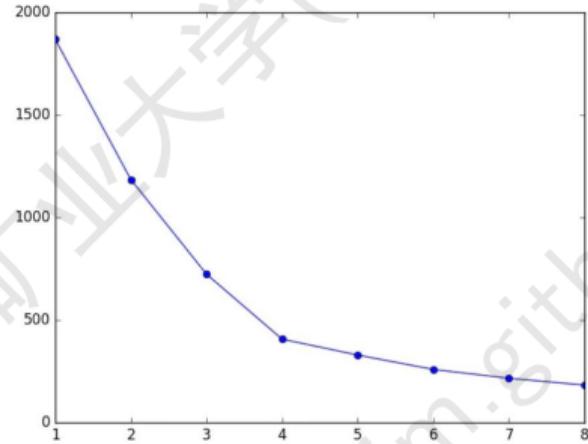
聚类分析- k 均值

牧师-村民模型



聚类分析- k -均值

如何确定 k -**肘法**



如何定义距离：欧氏距离、闵可夫斯基距离、马氏距离...

聚类分析- k 均值

实例-西瓜数据集

编号	密度	含糖率	编号	密度	含糖率	编号	密度	含糖率
1	0.697	0.460	11	0.245	0.057	21	0.748	0.232
2	0.774	0.376	12	0.343	0.099	22	0.714	0.346
3	0.634	0.264	13	0.639	0.161	23	0.483	0.312
4	0.608	0.318	14	0.657	0.198	24	0.478	0.437
5	0.556	0.215	15	0.360	0.370	25	0.525	0.369
6	0.403	0.237	16	0.593	0.042	26	0.751	0.489
7	0.481	0.149	17	0.719	0.103	27	0.532	0.472
8	0.437	0.211	18	0.359	0.188	28	0.473	0.376
9	0.666	0.091	19	0.339	0.241	29	0.725	0.445
10	0.243	0.267	20	0.282	0.257	30	0.446	0.459

假定聚类簇数 $k = 3$, 算法开始时随机选取三个样本 x_6, x_{12}, x_{27} 作为初始均值向量, 那么

$$\mu_1 = (0.403, 0.237)^\top, \mu_2 = (0.343, 0.099)^\top, \mu_3 = (0.532, 0.472)^\top$$

聚类分析- k -均值

分别计算每个样本到当前均值向量的距离，获得当前簇划分为

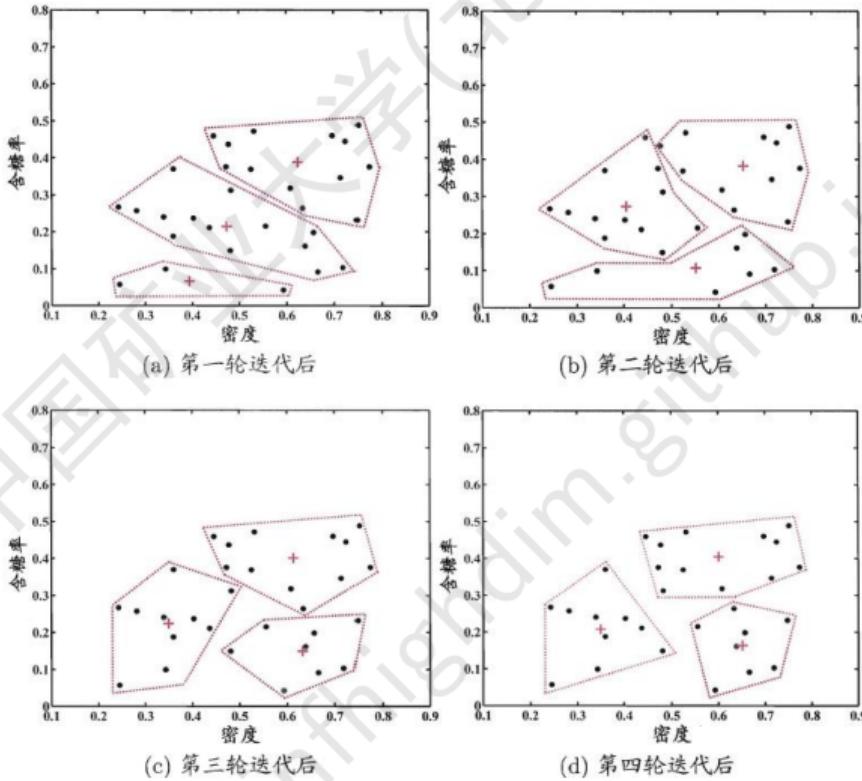
$$C_1 = \{x_5, x_6, \dots, x_{23}\}$$

$$C_2 = \{x_{11}, x_{12}, x_{16}\}$$

$$C_3 = \{x_1, x_2, \dots, x_{30}\}$$

更新当前均值向量后，不断重复上述过程，第五轮迭代产生的结果与第四轮迭代相同，于是算法停止，得到最终的簇划分。

聚类分析- k 均值



聚类分析- k 均值

优点

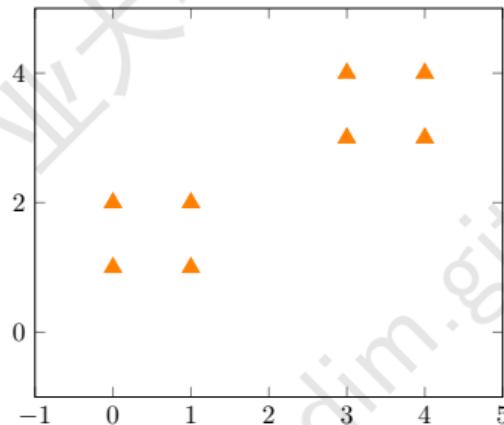
- 容易理解，聚类效果不错；
- 处理大数据集的时候，该算法可以保证较好的伸缩性；
- 当簇近似高斯分布的时候，效果非常不错；
- 算法复杂度低

缺点

- k 值需要人为设定，不同 k 值得到的结果不一样；
- 对**初始的簇中心敏感**，不同选取方式会得到不同结果
- 对异常值敏感；
- 样本只能归为一类，不适合多分类任务；
- 不适合太离散的分类、样本类别不平衡的分类、非凸形状的分类。

聚类分析- k 均值

k -means++: 设计了一种选择初始值（或“种子”）的方法，假设已经选取了 n 个初始聚类中心($0 < n < k$)，则在选取第 $n+1$ 个聚类中心时，距离当前 n 个聚类中心越远的点会有更高的概率被选为第 $n+1$ 个聚类中心。



坐标	(3,4)	(4,4)	(3,3)	(4,3)	(0,2)	(1,2)	(0,1)	(1,1)
距离	8	13	5	10	1	0	2	1
概率	0.2	0.325	0.125	0.25	0.025	0	0.05	0.025

聚类分析- k -均值

k -medoids: 由于 k -means算法对于异常值十分敏感，算法使用集群中位于最中心的对象，而不是将集群中的平均值作为参考点。

算法步骤

- ① 任意选取 k 个点作为medoids
- ② 按照与medoids最近的原则，将剩余点分配到当前最佳的medoids代表的类中
- ③ 在每一类中，计算每个成员点对应的准则函数(当前簇中所有其他点到该中心点的距离之和最小)，选取准则函数最小时对应的点作为新的medoids
- ④ 重复2-3的过程，直到所有的medoids 点不再发生变化，或已达到设定的最大迭代次数

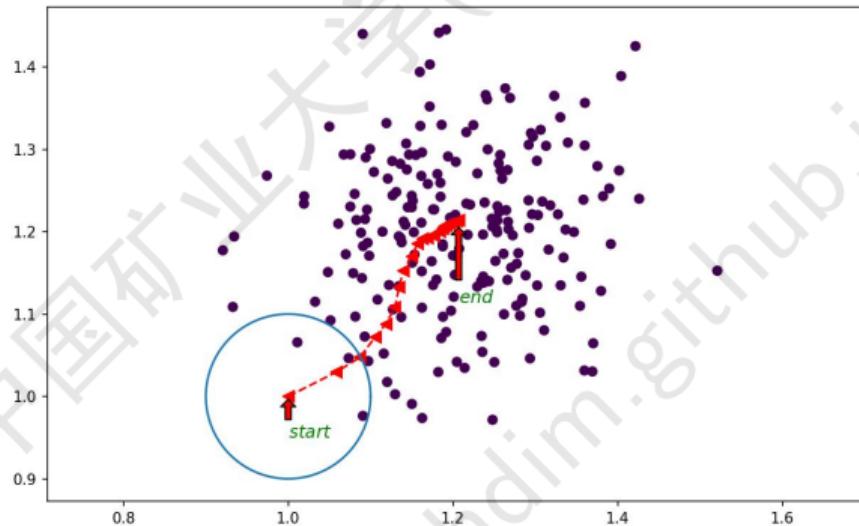
优缺点

优点：当存在噪音和孤立点时， k -medoids比 k -means更健壮。

缺点： k -medoids对于小数据集工作得很好，但不能很好地用于大数据集，计算质心的步骤时间复杂度是 $O(n^2)$ ，运行速度较慢。

聚类分析-均值漂移

均值漂移(mean shift)聚类: 沿着密度上升方向寻找聚簇点。



聚类分析-均值漂移

算法步骤

- ① 在未被分类的数据点中随机选择一个点作为中心点；
- ② 找出离中心点距离在指定半径之内的所有点，记做集合 M ，认为这些点属于簇 c 。
- ③ 计算从中心点开始到集合 M 中每个元素的向量，将这些向量相加，得到偏移向量。

$$S = \frac{1}{k} \sum_{x_i \in M} (x_i - \mu)$$

- ④ 中心点沿着shift的方向移动，移动距离是偏移向量的模。

$$\mu^{t+1} = \mu^t + S$$

- ⑤ 重复步骤2、3、4，直到偏移向量的大小满足设定的阈值要求，记住此时的中心点。
- ⑥ 重复1、2、3、4、5直到所有的点都被归类。

聚类分析-均值漂移

引入核函数，使得计算偏移向量时，越接近中心的点，具有越大的权值。改进后的偏移向量为

$$S = \frac{\sum_{i=1}^n x_i g\left(\frac{\|x-x_i\|^2}{h}\right)}{\sum_{i=1}^n g\left(\frac{\|x-x_i\|^2}{h}\right)} - x$$

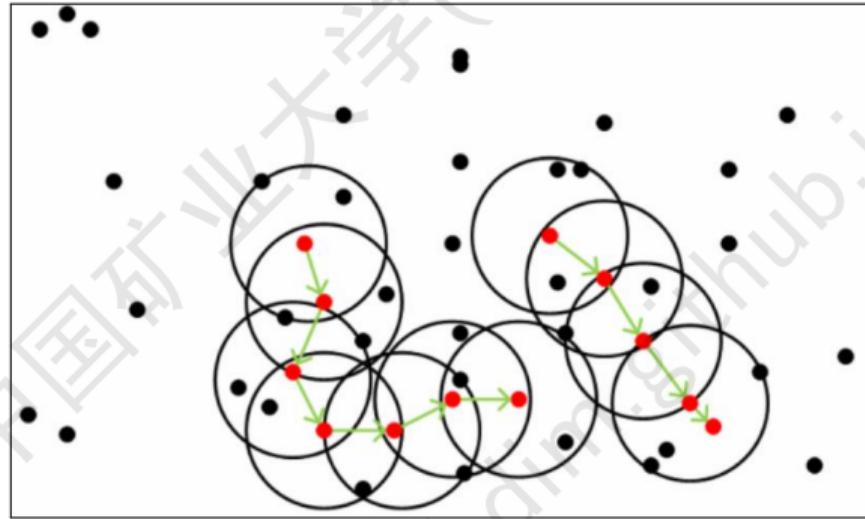
基于密度的聚类(Density-Based Spatial Clustering of Applications with Noise, DBSCAN): 根据样本的分布密度来发现不同的簇，簇定义为密度相连的点的最大集合，能够把具有足够高密度的区域划分为簇，并可在噪声的空间数据库中发现任意形状的聚类。

相关定义：

- **ε -邻域**: 给定对象半径为 ε 内的区域称为该对象的 ε -邻域, 对于 $x_j \in D$, 其 ε -邻域包含样本集 D 中与 x_j 的距离不大于 ε 的子样本集;
- **核心对象**: 如果给定对象 ε -邻域内的样本点数大于等于MinPts, 则称该对象为核心对象; 不属于核心点但在某个核心点的邻域内的点叫做边界点。既不是核心点也不是边界点的是噪声点。
- **密度直达**: 如果 x_i 位于 x_j 的 ε -邻域中, 且 x_j 是核心对象, 则称 x_i 由 x_j 密度直达。注意反之不一定成立, 即此时不能说 x_j 由 x_i 密度直达, 除非且 x_i 也是核心对象。
- **密度可达**: 对于 x_i 和 x_j , 如果存在样本序列 p_1, p_2, \dots, p_T , 满足 $p_1 = x_i, p_T = x_j$, 且 p_{t+1} 由 p_t 密度直达, 则称 x_j 由 x_i 密度可达。也就是说, 密度可达满足传递性。
- **密度相连**: 对于 x_i 和 x_j , 如果存在核心对象样本 x_k , 使 x_i 和 x_j 均由 x_k 密度可达, 则称 x_i 和 x_j 密度相连。

聚类分析-DBSCAN

示意图



算法步骤

输入: 样本集 $D = (x_1, x_2, \dots, x_m)$, 邻域参数(ε , MinPts), 样本距离度量方式

输出: 簇划分 C

- 1) 初始化核心对象集合 $\Omega = \emptyset$, 初始化聚类簇数 $k = 0$, 初始化未访问样本集合 $\Gamma = D$;
 - 2) 对于 $j = 1, 2, \dots, m$, 按下面的步骤找出所有的核心对象:
 - a) 通过距离度量方式, 找到样本 x_j 的 ε -邻域子样本集 $N_\varepsilon(x_j)$
 - b) 如果子样本集样本个数满足 $|N_\varepsilon(x_j)| \geq \text{MinPts}$, 将样本 x_j 加入核心对象样本集合:
$$\Omega = \Omega \cup \{x_j\}$$
 - 3) 如果核心对象集合 $\Omega = \emptyset$, 则算法结束, 否则转入步骤4.
 - 4) 初始化第 $k=1$ 个簇 $C_k = \emptyset$;
 - 5) 在核心对象集合 Ω 中, 随机选择一个核心对象 o , 建立簇 $C_k = \{o\}$, 将 o 的 ε -邻域都加入到簇 C_k 中, 依次遍历 C_k 中的其他核心对象, 不断扩大该簇;
 - 6) 更新 $\Omega = \Omega / \{C_1, C_2, \dots, C_k\}$, 如果 Ω 为空集, 结束算法, 否则 $k=k+1$, 转入步骤5.
- 输出结果:** 簇划分 $C = \{C_1, C_2, \dots, C_k\}$

算法优缺点

优点：

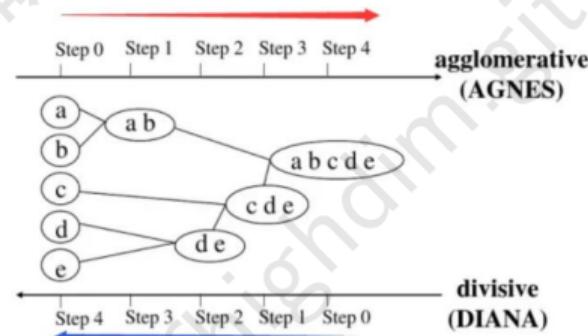
- 可以对任意形状的稠密数据集进行聚类，而K-Means之类的聚类算法一般只适用于凸数据集；
- 可以在聚类的同时发现异常点，对数据集中的异常点不敏感；
- 聚类结果没有偏倚，而K-Means之类的聚类算法初始值对聚类结果有很大影响。

缺点：

- 如果样本集的密度不均匀、聚类间距差相差很大时，这时用DBSCAN一般不适合；
- 如果样本集较大时，聚类收敛时间较长；
- 参数较多，需要对距离阈值 ϵ ，邻域样本数阈值MinPts联合调参，不同的参数组合对最后的聚类效果有较大影响。

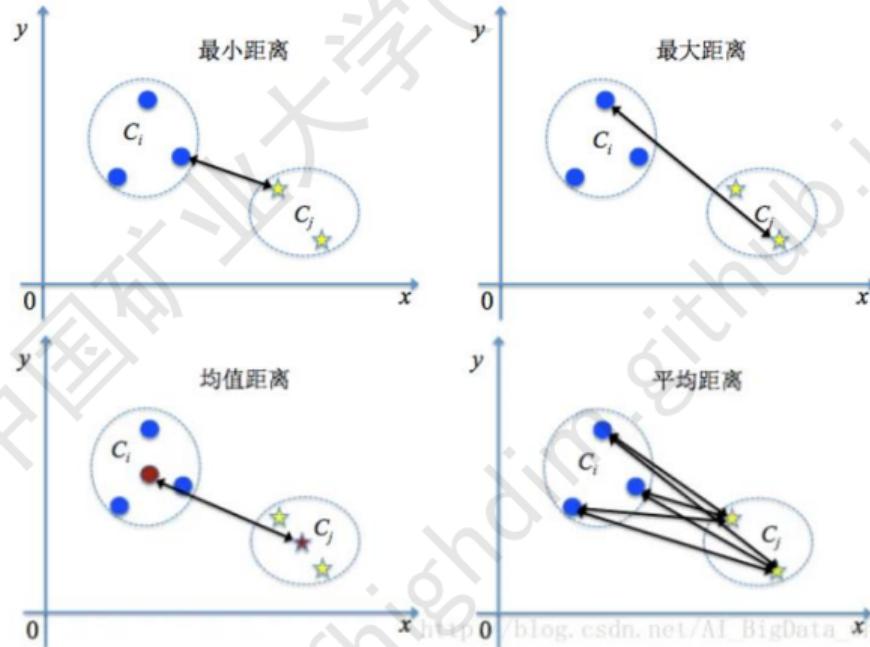
层次聚类(hierarchical clustering): 按照数据之间的相似度，划分层次，构建一颗聚类树。划分层次可以按照自底向上或者自上而下的顺序，分别对应凝聚方法和分裂方法。

- **凝聚方法**: 将样本中的每个点都看成一个簇，然后找出距离最小的两个簇进行合并，不断重复到预期簇或者其他终止条件；
- **分裂方法**: 先将所有样本当作一整个簇，然后找出簇中距离最远的两个簇进行分裂，不断重复到预期簇或者其他终止条件。



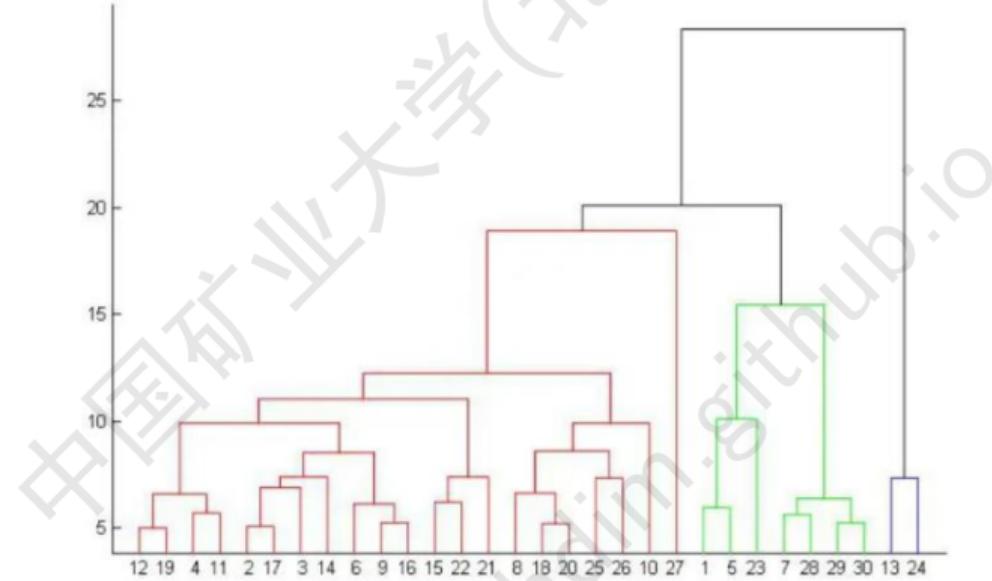
聚类分析-层次聚类

簇间距离: 最小距离, 最大距离, 均值距离, 平均距离



中间距离, 重心距离

聚类分析-层次聚类



优点：

- 距离和规则的相似度容易定义，限制少；
- 不需要预先指定聚类数；
- 可以发现类的层次关系；
- 可以聚类成其它形状

缺点：

- 计算复杂度太高；
- 离群点也能产生很大影响；
- 算法很可能聚类成链状。

聚类分析-高斯混合

高斯混合聚类(Mixture-of Gaussian):

基于概率分布的聚类算法，它是首先假设每个簇符合不同的高斯分布，判断每个样本符合各个分布的概率，将该样本划为概率最大的那个分布簇内。

一般的高斯分布概率密度函数为

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

定义高斯混合分布

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

该分布共由 k 个混合成分组成，每个混合成分对应一个高斯分布。

算法步骤：

- ① 设置 k 的个数，即初始化高斯混合模型的成分个数。（随机初始化每个簇的高斯分布参数（均值和方差）。也可观察数据给出一个相对精确的均值和方差）
- ② 计算每个数据点属于每个高斯模型的概率，即计算后验概率。（点越靠近高斯分布的中心，则概率越大，即属于该簇可能性更高）
- ③ 计算 α, μ, Σ 参数使得数据点的概率最大化，使用数据点概率的加权来计算这些新的参数，权重就是数据点属于该簇的概率。
- ④ 重复迭代2和3直到收敛。

优点：

- (1)GMM使用均值和标准差，簇可以呈现出椭圆形，优于k-means的圆形；
- (2)GMM是使用概率，故一个数据点可以属于多个簇。

缺点：

容易收敛到局部最优解。对于 k -means，通常是重复多次然后取最好的结果，但GMM每次迭代的计算量比 k -means要大很多，一般是先用 k -means(重复并取最优值)，然后将聚类中心点作为GMM的初始值进行训练。

谱聚类(Spectral Clustering): 把所有数据看作空间中的点，这些点之间用带权重的边相连，距离较远的点之间的边权重较低，距离较近的点之间边权重较高，通过对所有数据点和边组成的图进行切图，让切图后不同子图间边权重和尽可能低，而子图内边权重和尽可能高来达到聚类的目的。

定义邻接矩阵 W 和度矩阵 D :

$$W = (w_{ij})_{ij=1}^n$$

构建图拉普拉斯矩阵

$$L = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$$

基于该矩阵对图进行切割，使得切割后的各个组之间相似性很小，而组内数据之间相似性很大。

聚类分析-谱聚类

优点：

- 对于处理稀疏数据的聚类效果很有效;
- 使用了降维，在处理高维数据聚类时比传统聚类好;
- 当聚类的类别个数较小的时候，谱聚类的效果会很好;
- 谱聚类算法建立在谱图理论上，与传统的聚类算法相比，具有能在任意形状的样本空间上聚类且收敛于全局最优解。

缺点：

- 谱聚类对相似图和聚类参数的选择非常敏感;
- 谱聚类适用于均衡分类问题，即簇之间点的个数差别不大，对于簇之间点的个数相差悬殊的问题不适用。

性能度量指标-聚类”有效性指标” (validity index): “簇内相似度”(intra-cluster similarity)高且“簇间相似度”(inter-cluster similarity)低.

- 外部指标: 将聚类结果与某个参考基准进行比较, 常见的指标有纯度、Jaccard系数、FM指数、Rand指数;
- 内部指标: 直接考察聚类结果而不利用任何参考模型, 常见的指标有轮廓系数、CH指数、DB指数。

聚类分析-评估指标

外部指标:

聚类算法给出的簇划分为 C^* , 参考模型给出的簇划分 C , 令 λ^*, λ 分别表示与 C^*, C 对应的簇标记向量, 定义

$$a = |SS|, SS = \{(x_i, x_j) | \lambda_i = \lambda_j, \lambda_i^* = \lambda_j^*, i < j\}$$

$$b = |SD|, SD = \{(x_i, x_j) | \lambda_i = \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\}$$

$$c = |DS|, DS = \{(x_i, x_j) | \lambda_i \neq \lambda_j, \lambda_i^* = \lambda_j^*, i < j\}$$

$$d = |DD|, DD = \{(x_i, x_j) | \lambda_i \neq \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\}$$

$$\text{纯度 } P = \frac{1}{N} \sum_k \max_j |c_k^* \cap c_j|$$

$$\text{Jaccard系数 } JC = \frac{a}{a+b+c}$$

$$\text{FM指数 } FMI = \sqrt{\frac{a}{a+b} \frac{a}{a+c}}$$

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

内部指标

轮廓系数(Silhouette Coefficient)

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

Calinski-Harbasz Score

$$s_k = \frac{D_b}{k-1} / \frac{D_w}{m-k}$$

戴维森堡丁指数(Davies-Bouldin index, DBI)

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{i \neq j} \frac{S_i + S_j}{\|w_i - w_j\|^2}$$

聚类分析-应用

聚类的应用

- 社交网络：用户特征，社交圈子
- 电子商务：客户信息，分群
- 医疗诊断：患者信息，疾病分类



(a)



(b)



(c)

聚类分析

代码调用

```
from sklearn.cluster import KMeans  
from sklearn.cluster import kmeans_plusplus  
from sklearn.cluster import DBSCAN  
from sklearn.cluster import AgglomerativeClustering  
from sklearn.cluster import spectral_clustering
```

谢谢观看!