基于Litemall的数据挖掘及数据分析预测

鲲鹏创新实践课：鲲鹏应用数据分析与管理实战

前言

- 本课程主要介绍数据挖掘的相关理论知识，包括数据挖掘的定义和特点，数据挖掘的基本流程，数据预处理和特征工程简介，以及数据挖掘常用算法与最后的评估标准，帮助学员更好地掌握数据挖掘的基本特点。

目标

- 学完本课程后，您将能够：
  - 了解什么是数据挖掘，数据挖掘与数据分析的区别
  - 掌握数据挖掘的流程
  - 掌握数据预处理和特征工程
  - 掌握数据挖掘常用算法
  - 掌握模型评估的标准

目录

数据挖掘概述

- 数据挖掘是通过对大量的数据进行分析，以发现和提取隐含在其中的具有价值的信息和知识的过程。
- 与数据挖掘相关联的其他名称：
  - 数据库内知识发现（KDD- Knowledge discovery in databases）
  - 数据/模式分析
  - 商业智能
  - 人工智能
  - ……

数据挖掘可以解决哪些问题

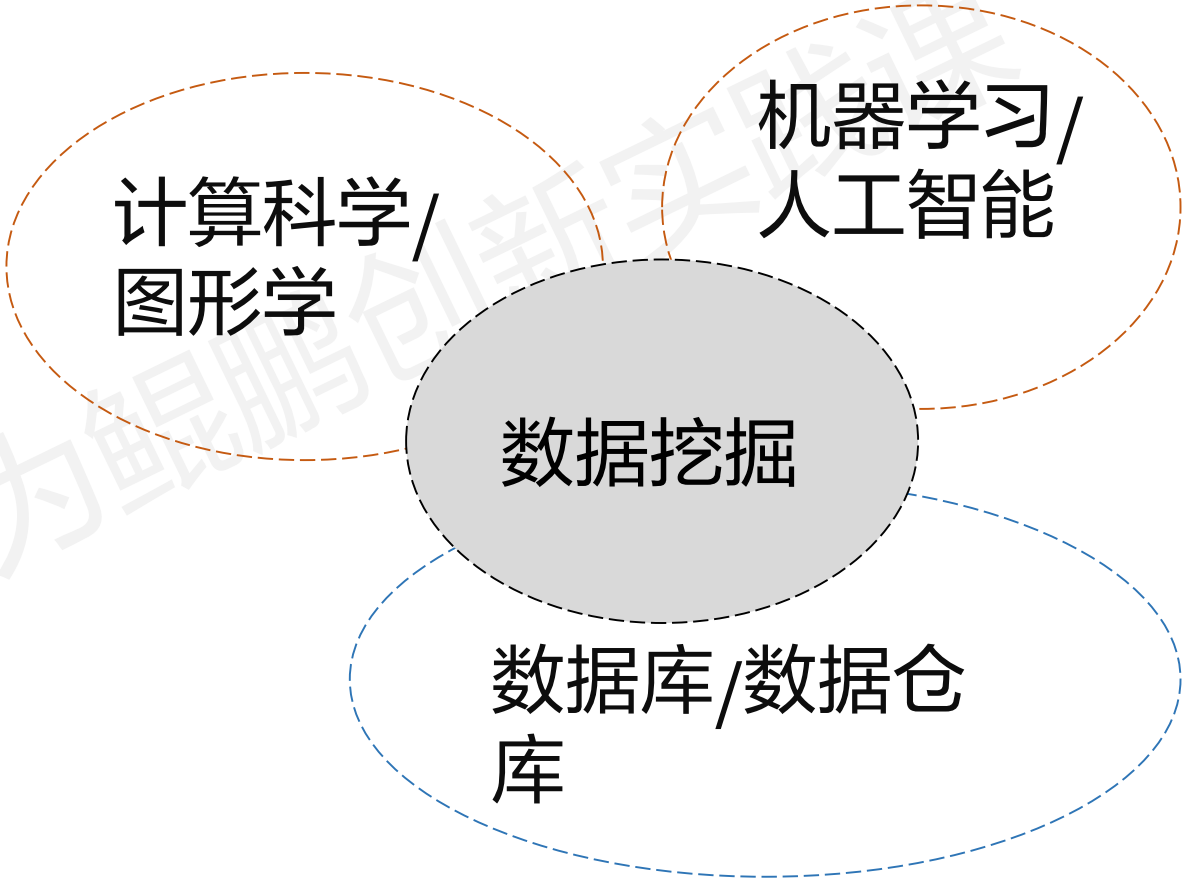
- 如何进行金融行业客户分群?
- 如何能降低用户流失率?
- 如何细分现有目标市场?
- 如何对企业及法人进行风险预警，维稳防范?
- 如何打造政府办公自动化?
- 如何制定交叉销售策略以提升销售额?

数据挖掘与数据分析的关系

	数据分析	数据挖掘
概念	对数据进行分析，重点是观察数据	从大量的数据中，挖掘出未知的、且有价值的信息和知识的过程。重点是从数据中发现“知识规则”。
分析目的	对历史数据进行统计学上的一些分析	数据挖掘更侧重于机器对未来的预测
分析过程	侧重于统计学上面的一些方法，经过人的推理演译得到结论	侧重由机器进行自学习，直接得到结论
分析结果	准确的统计量	一般是模糊的结果
使用工具	用到成熟的的分析工具，比如EXCEL、SPSS、SAS等	数据挖掘则需要有编程基础
联系	都跟数据打交道 知识技能有很多交叉点 在职业上他们没有很明显的界限 数据分析与数据挖掘的本质都是一样的，都是从数据里面发现关于业务的知识（有价值的信息），从而帮助业务运营、改进产品以及帮助企业做更好的决策。 狭义的数据分析与数据挖掘构成广义的数据分析。	

数据挖掘覆盖的学科

- 数据挖掘是多个领域的融合：
  - 人工智能
  - 数据库
  - 统计学
  - 并行计算
  - 图形学
  - ……

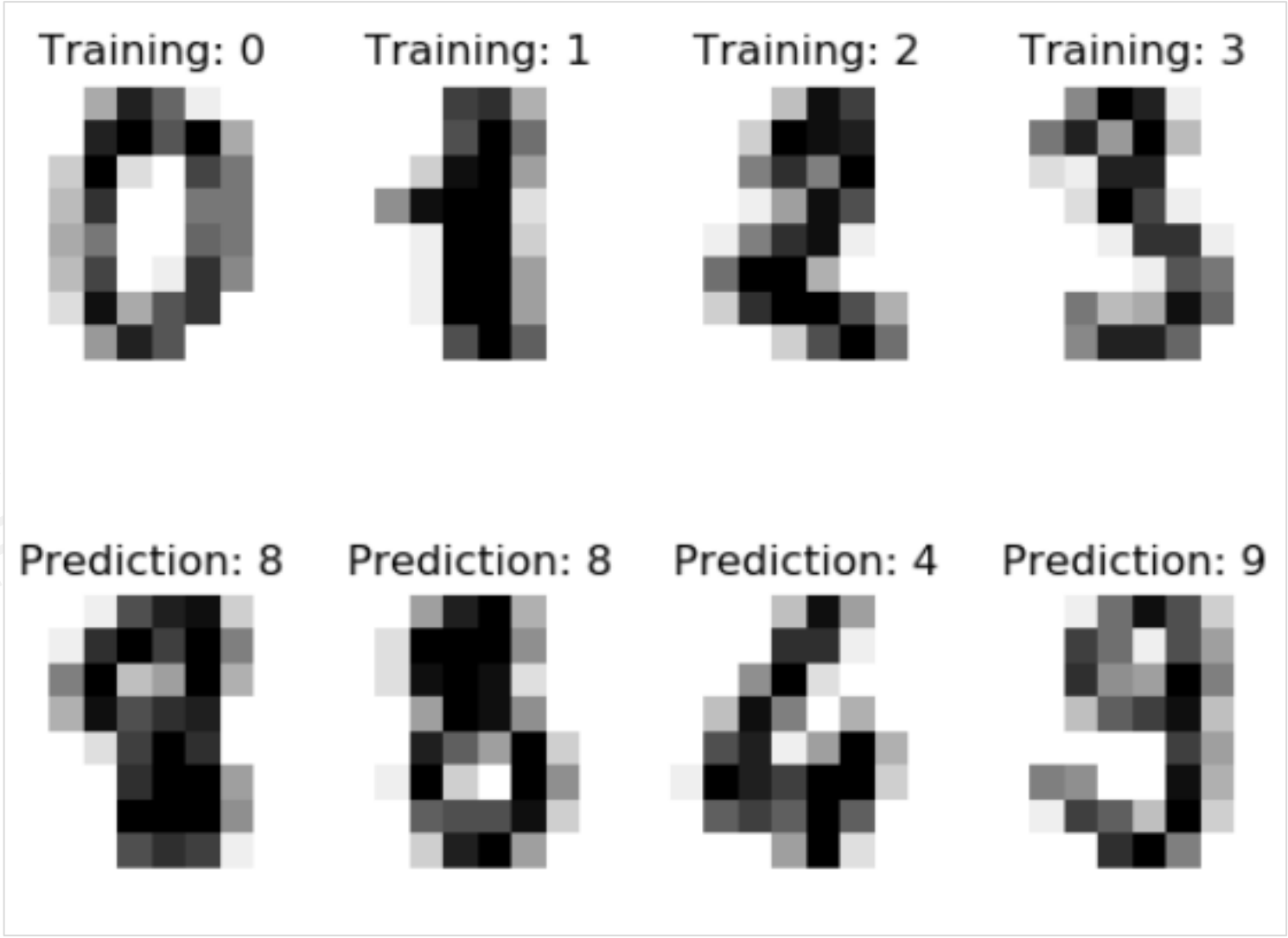


数据挖掘模式分类

- 根据训练数据是否拥有标记信息
  - 监督学习
  - 半监督学习
  - 非监督学习
- 根据应用角度
  - 分类
  - 回归
  - 聚类
  - 关联分析

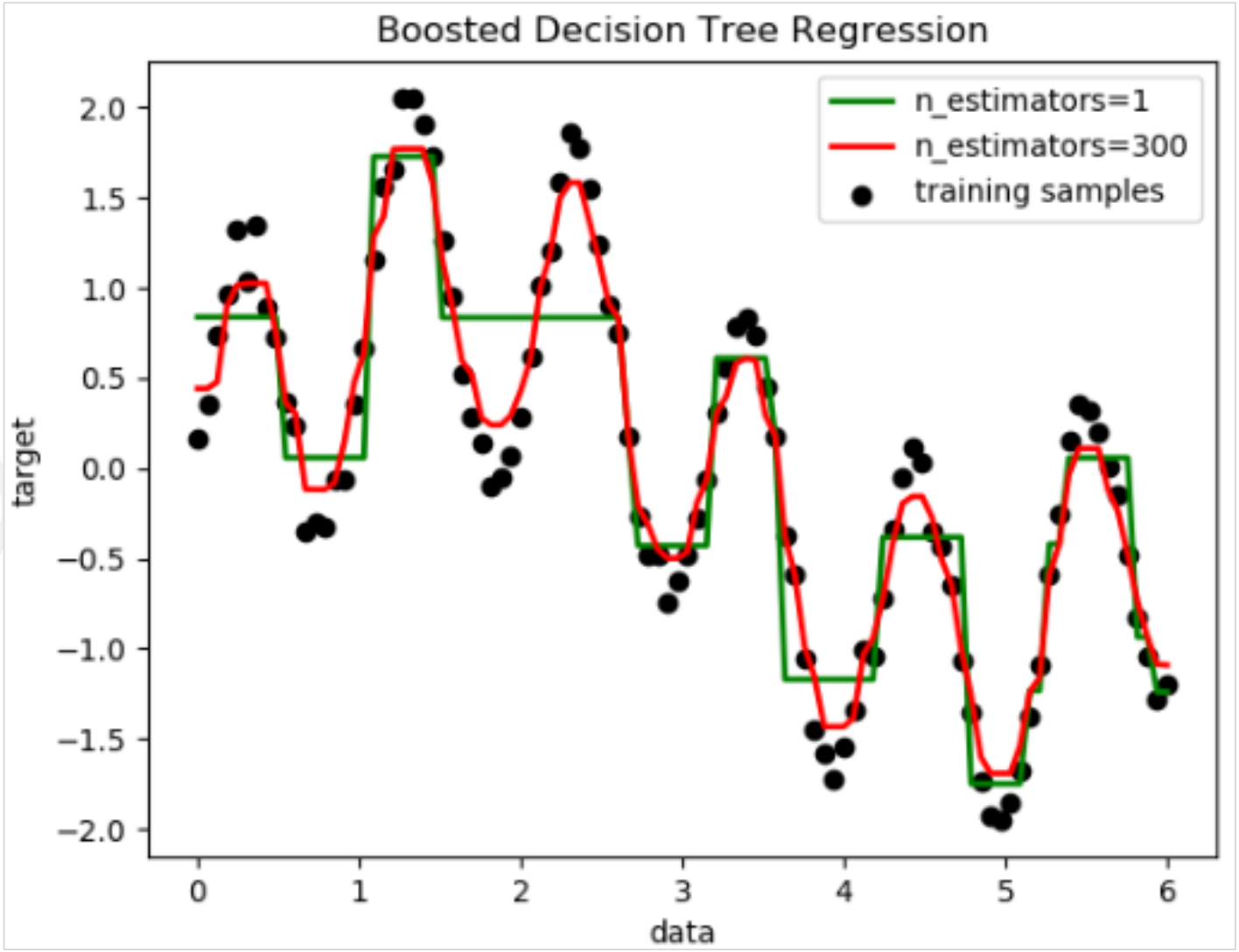
分类

- 对现有的数据进行学习，得到一个目标函数或规则，把每个属性集x映射到一个预先定义的类标号y上。
- 右图的分类案例为识别手写数字图像。



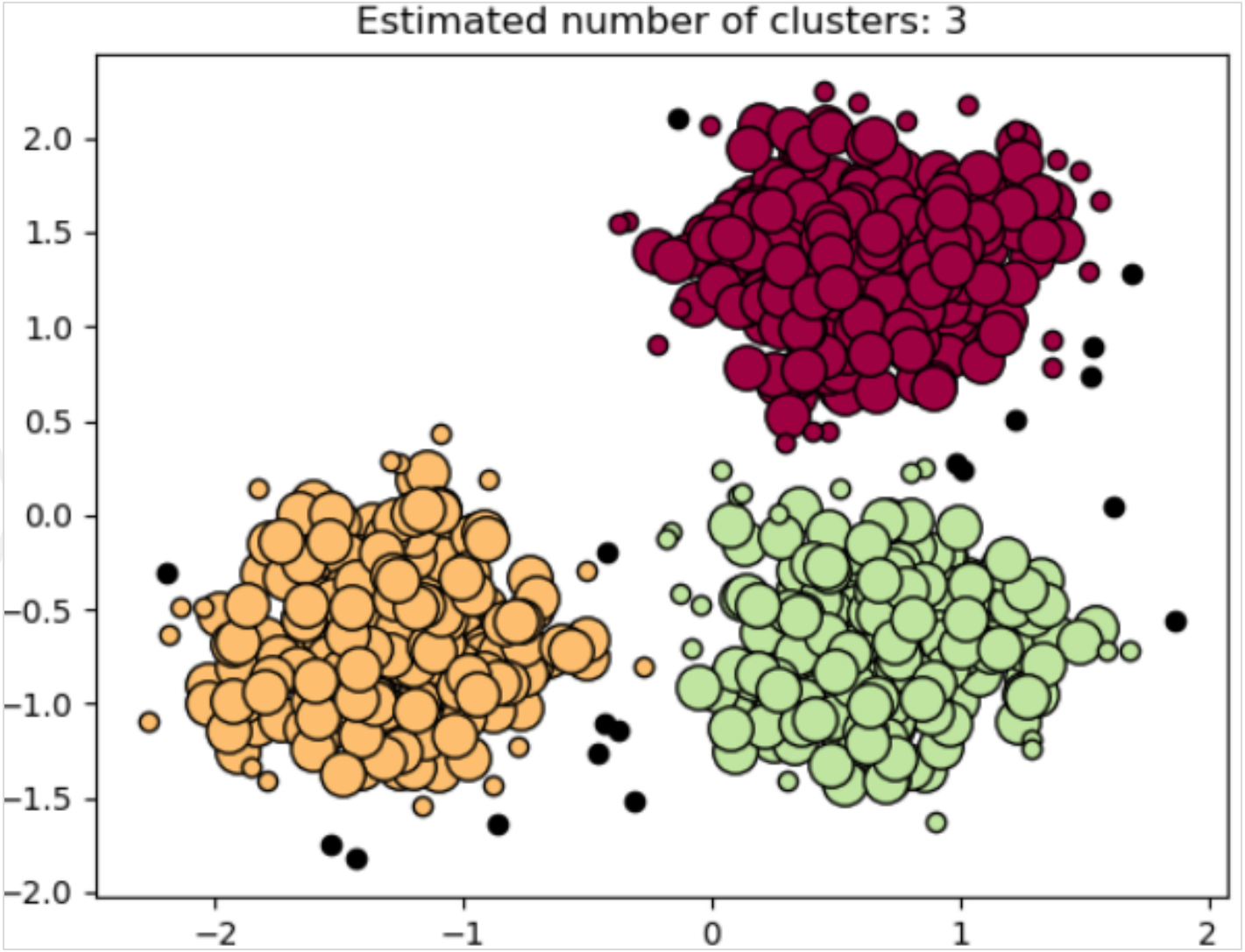
回归

- 回归分析是通过规定因变量和自变量来确定变量之间的因果关系，建立回归模型，并根据实测数据来求解模型的各个参数，然后评价回归模型是否能够很好的拟合实测数据。
- 右图的回归案例为将299个增强（300个决策树）与单个决策树回归器进行比较，属于AdaBoost决策树回归算法。



聚类

- 将数据对象分组成为多个类或者簇，它的目标是：在同一个簇中的对象之间具有较高的相似度，而不同簇中的对象差别较大。
- 右图的聚类案例为查找高密度的核心样本并从中扩展聚类，属于DBSCAN聚类算法。



关联

- 交叉销售问题等属于关联问题，另外运营商的告警压缩也用到了关联分析算法。关联分析也叫购物篮分析，最经典的案例就是“啤酒尿布“的故事。
- 我们要掌握常见的关联分析算法：Aprior算法、Carma算法，序列算法等。

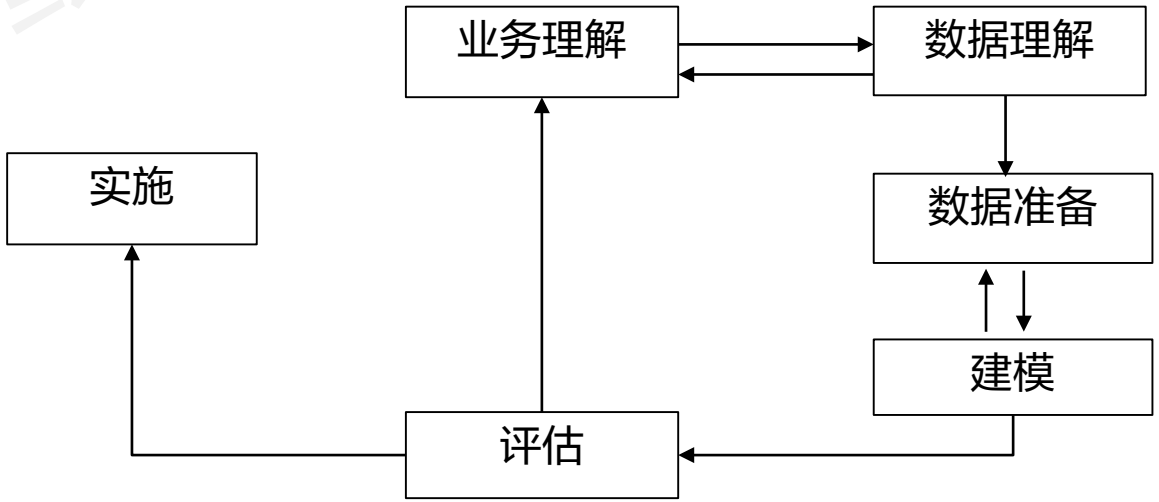
数据挖掘误区

- 数据挖掘是人们处理商业问题的某些方法，通过适量的数据挖掘来获得有价值的结果，最好的数据挖掘工程师往往是那些熟悉和理解业务的人。
- 一个平台不会因为数据挖掘就变成金钥匙，反而一个拥有数据挖掘思维的人员才是关键，而且他还必须对业务数据有深刻的认识，这样才可能从数据中导出模式指引业务的改善。

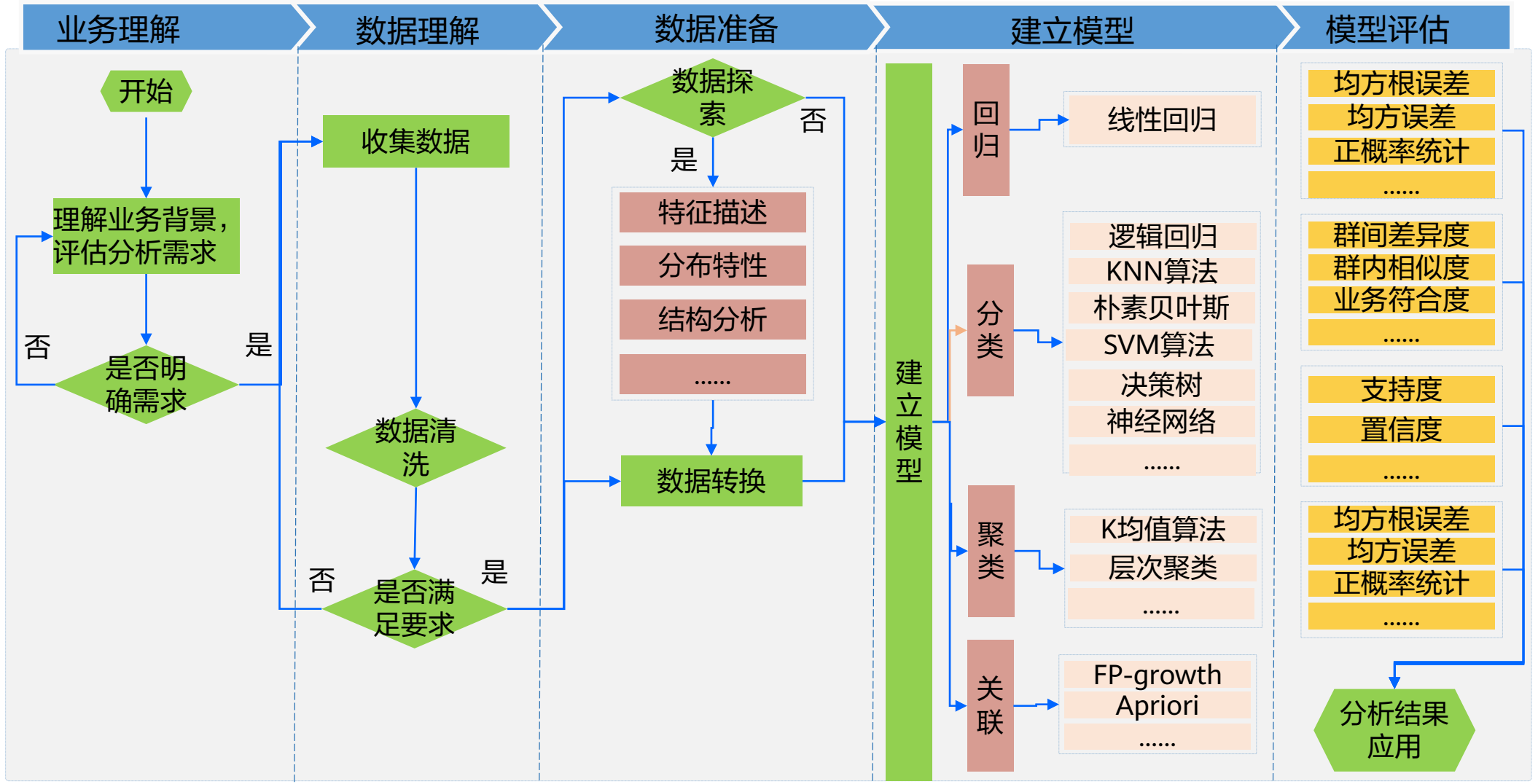
目录

CRISP-DM模型

- CRISP-DM（跨行业数据挖掘标准流程）是Cross Industry Standard Process —Data Mining的缩写，是当今数据挖掘业界通用流行的标准之一。它强调数据挖掘技术在商业中的应用，是用以管理并指导Data Miner有效、准确开展数据挖掘工作以期获得最佳挖掘成果的一系列工作步骤的规范标准。
- CRISP-DM模型的基本步骤包括：
  - 业务理解
  - 数据理解
  - 数据准备
  - 建立模型
  - 模型评估
  - 模型实施



数据挖掘标准流程



目录

数据预处理的重要性

- 在数据挖掘过程中，数据预处理是不可或缺的部分。
  - 大数据应用中数据的典型特点是独立的、不完整、含噪声和不一致。
  - 大部分数据挖掘算法对数据质量以及数据规模有特殊要求，通过数据预处理能有效的提高数据的质量，为数据挖掘过程节约大量时间和空间。

数据清洗

- 数据清洗是指通过删除、转换、组合等方法，处理数据集中的异常样本，为数据建模提供优质数据的过程。
- 一般来说，数据清洗包含以下两大任务：
  - 缺失值处理
  - 异常值处理

缺失值处理 - 数据缺失原因

- 在实际业务中，不可避免的会出现数据缺失的现象，总结下来大致有如下几种情形。
  - 人为疏忽、机器故障等客观因素导致信息缺失。
  - 人为刻意隐瞒部分数据。比如在数据表中，有意将一列属性视为空值，此时缺失值就可看作是一种特殊的特征值。
  - 数据本身不存在，比如银行做用户信息收集时，对学生群体来说工资这一属性不存在，因此在数据表里显示为空值。
  - 系统实时性能要求较高。
  - 历史局限性导致数据收集不完整。

缺失值处理 - 数据缺失影响

- 表一中存在数据缺失现象并且没有进行缺失值处理，数据整理后得表二，此时女生概率大于男生概率。
- 表三填充了表一中的缺失数据，数据整理后得表三，此时女生概率等于男生概率。

表一

Name	Gender	Age	Class A or Not
Lucy	F	12	Yes
Lily	F	14	No
Jack	/	12	Yes
Ian	M	/	No

表三

Name	Gender	Age	Class A or Not
Lucy	F	12	Yes
Lily	F	14	No
Jack	M	12	Yes
Ian	M	13	No

表二

性别	数量	概率
F	2	50%
M	1	25%
缺失值	1	25%

表四

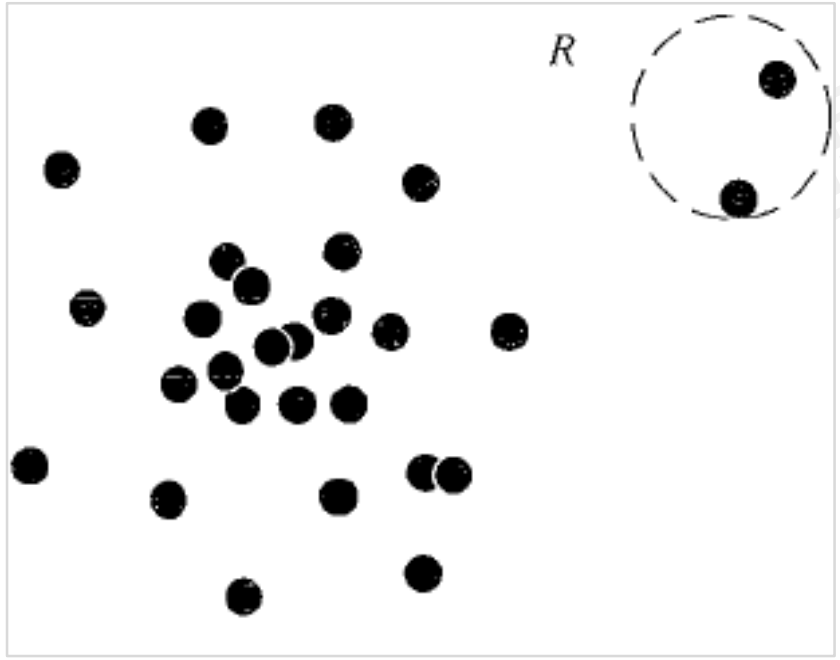
性别	数量	概率
F	2	50%
M	2	50%

缺失值处理 - 数据处理常见处理方式

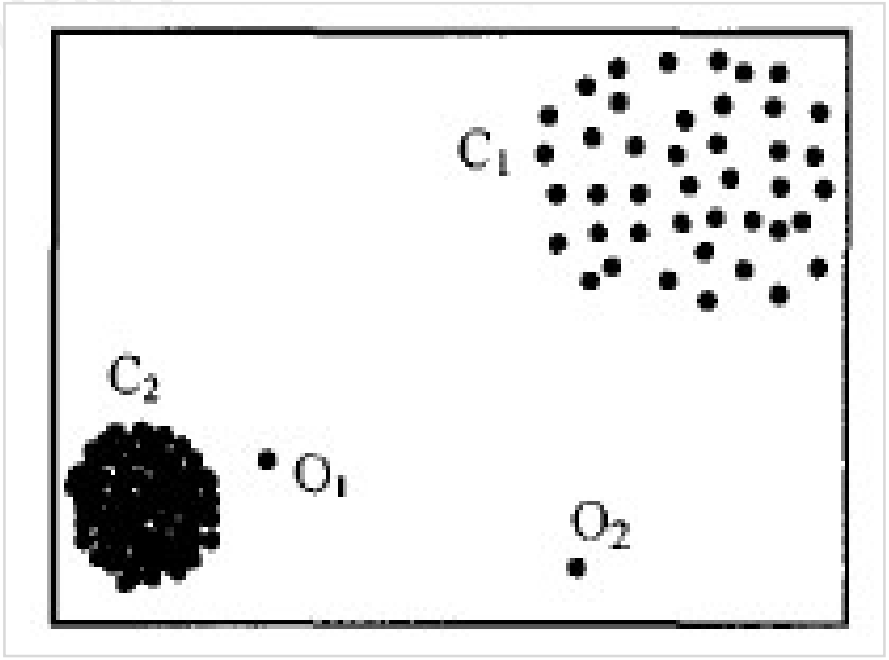
- 数据缺失原因多种多样，针对不同的缺失原因，数据缺失值的处理方式也各不相同。值得注意的是，有时属性缺失并不意味着数据缺失，比如，银行收集客户信息时，学生在“工资”这一栏为空值。缺失本身是包含有价值的信息的。因此要结合具体业务场景、数据场景选择合适的数据缺失值处理方式。
- 缺失值处理方法众多，总结下来为三种：
  - 删除
  - 填充
  - 不处理

异常值处理 - 初识异常值

- 异常值是偏离整体样本的观察值，也叫离群点 (Outlier)，如下图一所示。
  - 异常值会影响数据模型的精准度，因此异常值处理是数据预处理中重要的一步。在实际应用中，研究者可将其用于一些异常检测场景，比如入侵检测、欺诈检测、安全监测等。



图一

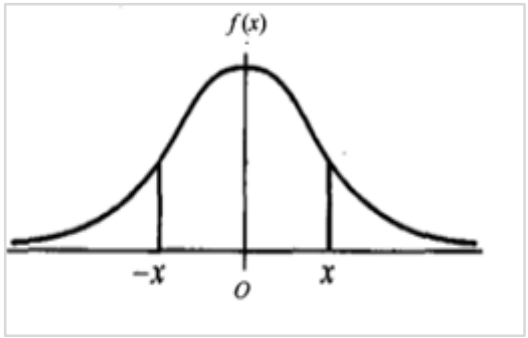


图二

异常值处理 - 异常值检测方法 (1)

- 异常值检测方法较多，本节主要介绍其中四种：散点图、基于分类模型的异常检测、3σ原则及箱型图分析。
  - 散点图。将数据用散点图的形式可视化，可观察到异常值。
  - 基于分类模型的异常检测
    - 根据现有数据建立模型，然后对新数据进行判断从而确实是否偏离，偏离则为异常值。
    - 比如，贝叶斯模型、神经网络、SVM等。
  - 3σ原则
    - 若数据集服从参数为μ，σ的正态分布或高斯分布，记为$X \sim N(\mu, \sigma^2)$。
    - 异常值被定义为，其值与平均值的偏差绝对值超过三倍标准差的值，即

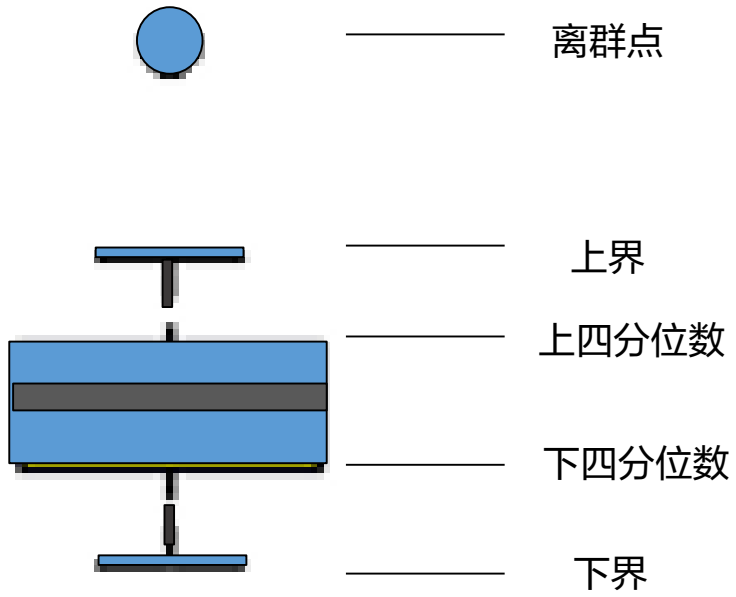
$$P(|x-\mu|>3\sigma) \leq 0.003$$



异常值处理 - 异常值检测方法 (2)

- (续)
  - 箱型图分析 - Tukey‘s test方法

Q1：上四分位数
Q2：下四分位数
IOR：Q1-Q2



最大估计值：Q1+k*IOR
最小估计值：Q2-k*IOR
K代表对异常值的容忍度，一般取K=1.5。

异常值处理 - 异常值处理方法

- 异常值的处理方法多种多样，常用方法有删除异常值、将异常值视为缺失值、估算异常值。具体处理方法需结合实际业务情况综合考虑。
  - 删除异常值。适用于异常值较少的情况。
  - 将异常值视为缺失值，按照缺失值处理方法来处理异常值。
  - 估算异常值。Mean/Mode/Median估计数据填充异常值。

特征工程

- 特征工程同样是对原始数据进行处理和加工的步骤，不过特征工程是将原始数据的属性转换为数据特征的过程。
- 在数据挖掘时，特征来源一般有两种：
  - 来源一：业务已经整理好各种特征数据，即业务指标，有时称为属性，我们需要去找出适合我们问题需要的特征。
    - 该类特征通常是业务专家指出，通常是挖掘遇到的最初数据来源，是已有特征，也称原始特征。
  - 来源二：从业务特征中自己去寻找高级数据特征，如若干原始特征的组合后形成一个新特征，即特征衍生。

特征缩放

- 案例引入
  - 假设住房价格受住房面积和卧室数量的影响。基于此假设,研究人员可用住房面积和卧室数量这两个参数预测房价。已知住房面积范围在[100,400](平方米)，卧室数量范围在[2,4](间)。由此可见，住房面积这一属性值域范围较大，在建立相关模型预测房价时可能会出现仅住房面积这一属性就可预测出房价的现象，弱化了其他属性在数据模型中的作用，造成一定的预测误差。
- 必要性
  - 在实际业务中，当数据的量纲不同，数量级别差距大时，会影响最终的数据模型，因此需用特征缩放来平衡各特征贡献。
  - 特征缩放可提高模型精度和模型收敛速度。它是数据预处理的重要环节之一。特征缩放又叫数据归一化。

特征缩放 - 方法概览

- 方法
  - 标准化 (Standardization)
  - 最小值-最大值归一化 (Min-Max Normalization)
  - 均值归一化 (Mean Normalization)
  - 缩放成单位向量 (Scaling to Unit Length)

特征缩放 - 应用场景

- 应用场景

最小值 - 最大值归一化应用场景	标准化应用场景
1. 不涉及距离度量； 2. 不涉及协方差计算 3. 数据不符合正态分布。	在分类、聚类算法中， 1. 需要使用距离来度量相似性； 2. 需要使用PCA技术进行降维； 3. 需要用到SVM和LR等。
数据较为稳定，不存在极端的最大值和最小值。	数据存在异常值和较多的噪音值。

- 不要在整个数据集上做归一化处理，要区分训练集和测试集；
- 在实际应用中，特征缩放的标准化操作更常用。

数值离散化

- 百度词条把数据离散化定义为把无限空间中有限的个体映射到有限的空间中去，以提高算法的时空效率。简单来说，就是将原始数据按值的大小进行预分类。
- 离散化仅适用于只关注元素之间的大小关系而不关注元素数值本身的情况。

原始数据

Age	11	15	17	30	34	40	67	79	80

数组	{100,300}，{46,66}，{122,9}

离散化

Age	11	15	17	30	34	40	67	79	80
	Young			Mature			Old		

数组	100	300	46	66	122	9
排序	4	6	2	3	5	1

离散化后

Age	Young	Mature	Old

数组	{4,6}，{2,3}，{5,1}

特征编码

- 数据挖掘中，一些算法可以直接计算分类变量，比如决策树模型。但许多机器学习算法不能直接处理分类变量，它们的输入和输出都是数值型数据。因此，把分类变量转换成数值型数据是必要的，可以用独热编码 (One-Hot Encoding) 和哑编码 (Dummy Encoding)实现。
- 比较常用的是对逻辑回归中的连续变量做离散化处理，然后对离散特征进行独热编码 (One-Hot Encoding) 或哑编码 (Dummy Encoding)，这样会使模型具有较强的非线性能力。

分类特征编码 – 独热编码 (1)

- 案例引入

  学校有三家食堂A，B，C，属于无序分类变量。在数据预处理时，很容易把A、B、C映射成数字1、2、3 。本身食堂A，B，C之间是无序的，映射成数字1、2、3则赋予变量顺序的概念。因此，针对此类情况，较为合适的方法是使用独热编码（ One-Hot Encoding ）标识变量。具体如下：

  A，B，C三个食堂，总共三个变量，则用三维表示。A为100，B为010，C为001。

A	B	C
1	0	0
0	1	0
0	0	1

- 无序分类变量的离散化方法较为常用方法，如下：
  - 独热编码 (One-Hot Encoding)。
  - 哑编码 (Dummy Encoding)。

分类特征编码 – 独热编码 (2)

- 独热编码 (One-Hot Encoding)
  - 定义
  - 使用M位状态寄存器对M个状态进行编码，每个状态都有独立的寄存器位，这些特征互斥，所以在任意时候只有一位有效。也就是说，这M种状态中只有一个状态位值为1，其他状态位都是0。换句话说，M个变量用M维表示，每个维度的数值或为1，或为0。

分类特征编码 – 独热编码 (3)

- 案例演示

  原始数据如下表，请对其进行独热编码。

	属性1	属性2	属性3
数据1	0	0	3
数据2	1	1	0
数据3	0	2	1
数据4	1	0	2

在Python中可以用OneHotEncoder函数实现独热编码

- 解析

4*3的数据样本，4个数据，每个数据有3个属性。

- 针对属性1，有两个取值 0和1　　[[1,0],[0,1]]
- 针对属性2，有三个取值 0、1和2　　[[1,0,0],[0,1,0],[0,0,1]]
- 针对属性3，有四个取值 0、1、2和3　[[1,0,0,0],[0,1,0,0],[0,0,1,0],[0,0,0,1]]

则针对原数据[0, 1, 1]独热编码后第一列0对应 [1,0]，第二列1对应[0,1,0]，第三列1对应[0,1,0,0]。

最终[0, 1, 1]独热编码为[1,0,0,1,0,0,1,0,0]。

分类特征编码 - 哑编码

- 哑编码 (Dummy Encoding)
  - 哑编码和独热编码很相似，唯一的区别在于哑编码使用M-1位状态寄存器对M个状态进行编码。具体如下：

学校里有三个食堂A，B，C，
使用独热编码 (One-Hot Encoding) 后，A为[1,0,0]，B为[0,1,0]，C为[0,0,1]。
使用哑编码 (Dummy Encoding) 后，A为[1,0]，B为[0,1]，C为[0,0]。
即用（ M-1 ）维就可以表述出A，B，C三个变量。

	A	B	C
独热编码	[1,0,0]	[0,1,0]	[0,0,1]
哑编码	[1.0]	[0,1]	[0,0]

分类特征编码 - Label-Encoding

- 定义
  - 有序分类变量数值之间存在一定的顺序关系，可直接使用划分后的数值进行数据建模。
  - 如分类变量{低年级，中年级，高年级}，可以直接离散化为{0,1,2}。
- 实践
  - 在Python中可以用pandas库中的map函数实现有序分类变量的离散化。

```
A[‘grade’] = A[‘grade’].map({‘low':0,‘medium':1,‘high':2, })
```

分类特征编码 - 优点和缺点

	优点	缺点
独热编码	解决了分类器不好处理分类变量的问题。	分类的变量不易过多，可能会造成稀疏矩阵。
哑编码		
Label-Encoding	解决了分类变量的编码问题。	可解释性差

对于不能处理分类变量的模型，必须要使用独热编码或哑编码，将变量转换成数值型。但若模型可处理分类变量，可无须转换数据，如树模型。

特征选择

- 现实中大数据挖掘任务，往往特征属性过多，而一个普遍存在的事实是，大数据集带来的**关键信息只聚集在部分或少数特征**上，因此需要：
  - 从中选择出重要的特征使得后续的建模过程只在一部分的特征上构建，**减少维数灾难**出现的可能。
  - **去除不相关的特征，留下关键因素，降低学习任务难度**，更容易挖掘数据本身带有的规律；同时在特征选择的过程中，会对数据特征的理解更加充分。

特征选择的原则

- 当数据预处理完成后，需要选择有意义的特征进行模型训练。通常来说，从三个方面考虑：
  - 特征是否发散：如果一个特征不发散，例如方差接近于0，也就是说样本在这个特征上基本上没有差异，这个特征对于样本的区分作用不明显，区分度不高。
  - 特征之间的相关性：特征与特征之间的线性相关性，去除相关性比较高的特征。
  - 特征与目标的相关性：与目标相关性高的特征，应当优先选择。

常见特征选择方法

- 特征选择方法有很多，主要包含特征减少和特征扩增。
- 特征减少
  - 单变量特征选择方法
    - Filter（过滤法）
  - 基于模型的特征选择方法
    - Wrapper（包装法）
    - Embedded（嵌入法）
    - 其他方法：如交叉验证配合Embeded方法
- 特征扩增
  - 在原有基础上构造新的特征

三种常见方法简介

- Filter：过滤法，按照发散性或者相关性对各个特征进行评分，设定阈值或者待选择阈值的个数，选择特征。
- Wrapper：包装法，根据目标函数（通常是预测效果评分），每次选择若干特征，或者排除若干特征。
- Embedded：嵌入法，先使用某些机器学习的算法和模型进行训练，得到各个特征的权值系数，根据系数从大到小选择特征。类似于Filter方法，但是是通过训练来确定特征的优劣。

降维背景

- 维数灾难背景
  - 现实应用中属性维度成千上万，在高维度情况下会带来很多麻烦，而且当维度大的时候，数据样本一般分布的非常稀疏，这是所有学习算法要面对的问题，降维技术应运而生。
- 数据降维
  - 降维是对事物的特征进行压缩和筛选，该项任务相对比较抽象。如果没有特定领域知识，无法预先决定采用哪些数据，比如在人脸识别任务中，如果直接使用图像的原始像素信息，数据的维度会非常高，通常会利用降维技术对图像进行处理，保留下最具有区分度的像素组合。

常见降维方法 (1)

- SVD - 奇异值分解
  - 奇异值分解是一个能适用于任意矩阵的一种分解方法。
  - 奇异值分解发现矩阵中的冗余并提供用于消除它的格式。
  - 奇异值分解就是利用隐藏的特征建立起矩阵行和列之间的联系。
- PCA - 主成分分析
  - 思想：寻找表示数据分布的最优子空间（降维，可以去掉线性相关性）。
  - 数学原理：取协方差矩阵前s个最大特征值对应的特征向量构成映射矩阵，对数据进行降维。

常见降维方法 (2)

- LDA - 线性判别分析
  - 思想：寻找可分性判据最大的子空间。
  - 用到了Fisher的思想，即寻找一个向量，使得降维后类内散度最小，类间散度最大；其实就是取对应的特征向量构成映射矩阵，对数据进行处理。
- LLE - 局部线性嵌入
  - 局部线性嵌入（ Locally Linear Embedding，简称LLE ）也是非常重要的降维方法。和传统的PCA，LDA等关注样本方差的降维方法相比，LLE关注于降维时保持样本局部的线性特征，由于LLE在降维时保持了样本的局部特征，它广泛的用于图像识别，高维数据可视化等领域。

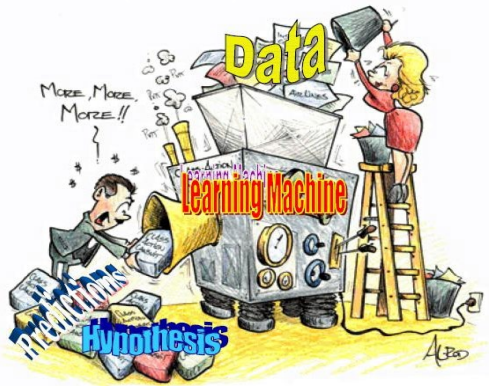
特征选择与降维的区别

- 特征选择与维度归约的异同
  - **解决过拟合的问题**。
  - **降维和特征选择都是为了使数据维度降小**，但实际上两者本质是完全不同。
    - **降维本质上是从一个维度空间映射到另一个维度空间**
    - **特征选择就是单纯地从提取到的所有特征中选择部分特征作为训练集特征**
  - 特征选择是指从已有的特征集合中按某一分类准则选出一组子特征集和作为降维的分类特征使用。经验上一般是先进行特征选择，再进行降维。

目录

大数据挖掘的算法背景

- 大数据的挖掘是从海量、不完全的、有噪声的、模糊的、随机的大型数据库中发现隐含在其中有价值的、潜在有用的信息和知识的过程。而这一过程主要基于人工智能，机器学习，模式学习，统计学等，通过对大数据高度自动化地分析，做出归纳性的推理，从中挖掘出潜在的模式。
- 机器学习（包括深度学习分支）是研究“学习算法”的一门学问。所谓“学习”是指：对于某类任务T和性能度量P，一个计算机程序在T上以P衡量的性能随着经验E而自我完善，那么我们称这个计算机程序在从经验E学习。



机器学习分类

- **有监督学习**：利用已知类别的样本，训练学习得到一个最优模型，使其达到所要求性能，再利用这个训练所得模型，将所有的输入映射为相应的输出，对输出进行简单的判断，从而实现分类的目的，即可以对未知数据进行分类。
- **无监督学习**：对于没有标记的样本，学习算法直接对输入数据集进行建模，例如聚类，即“物以类聚，人以群分”。我们只需要把相似度高的东西放在一起，对于新来的样本，计算相似度后，按照相似程度进行归类。
- **半监督学习**：试图让学习器自动地对大量未标记数据进行利用以辅助少量有标记数据进行学习。
- **强化学习**：学习系统从环境到行为映射的学习，以使奖励信号（强化信号）函数值最大，强化学习不同于连接主义学习中的监督学习，主要表现在教师信号上，强化学习中由环境提供的强化信号是对产生动作的好坏作一种评价（通常为标量信号），而不是告诉强化学习系统如何去产生正确的动作。

基本术语与概念 (回归)

- 监督学习主要的两种类型算法是回归算法和分类算法。
- 回归用于预测输入变量和输出变量之间的关系，即回归模型是表示输入变量到输出变量之间映射的函数。
- 回归问题的学习等价于函数拟合：使用一条函数曲线，使其很好的拟合已知函数且很好的预测未知数据。
- 回归问题分为模型的学习和预测两个过程。基于给定的训练数据集构建一个模型，根据新的输入数据预测相应的输出。**且预测值是连续的，即连续的输出值。**
- 回归问题按照输入变量的个数可以分为一元回归和多元回归；按照输入变量和输出变量之间关系的类型， 可以分为 ：
  - 线性回归
  - 非线性回归

基本术语与概念 (分类)

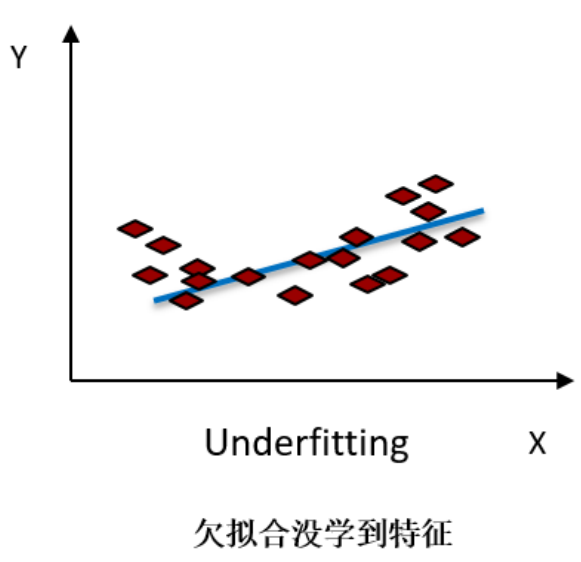
- 分类算法是通过构造一个分类函数或分类器的方法把数据库中的数据项映射到给定类别中的某一个，从而可以用于预测未知数据。
- 通常根据数据情况将分类问题分为线性可分和线性不可分。

基本术语与概念 (偏差/方差)

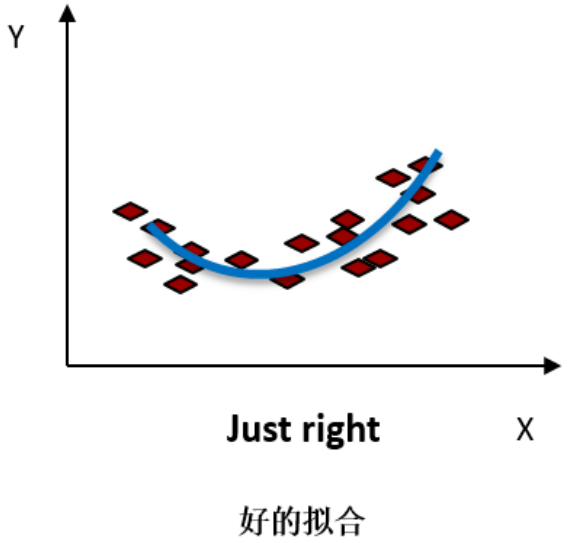
- 偏差指的是算法的期望预测与真实值之间的偏差程度，反映了模型本身的拟合能力。
- 方差度量了同等大小训练集的变动导致学习性能的变化，刻画了数据扰动所导致的影响。
- 当模型越复杂时，拟合能力就越好，模型的偏差就越小。但此时如果换一组数据可能模型的变化就会很大，即模型方差变大，所以复杂的模型容易造成过拟合；当模型简单的时候，即使换一组数据，得到的学习器的分类效果与之前分类器的效果的差别也不会很大，即模型方差很小，但由于模型过于简单，导致偏差会很大。

基本术语与概念 (过拟合/欠拟合)

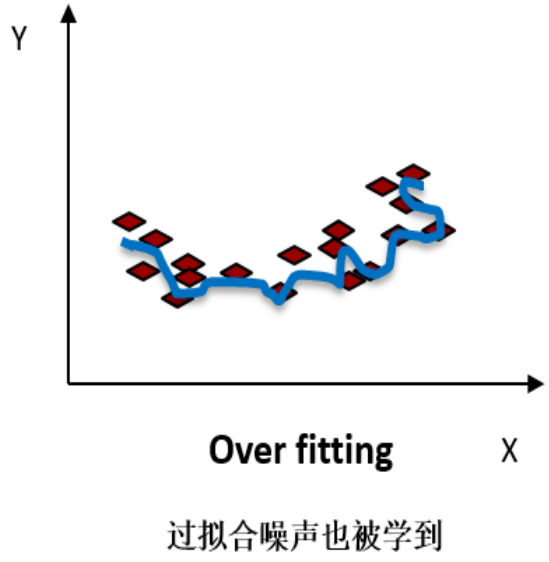
- 欠拟合：模型没有很好地捕捉到数据特征，不能够很好地拟合数据。也就是高偏差，低方差。
- 过拟合：通俗一点地来说过拟合就是模型把数据学习地太彻底，以至于把噪声数据的特征也学习到了，这样就会导致在后期测试的时候不能够很好地识别数据，即不能正确地分类，模型泛化能力太差。也就是高方差，低偏差。



欠拟合没学到特征



好的拟合



过拟合噪声也被学到

基本术语与概念 (性能描述)

- 泛化能力：机器学习的目标是使学得的模型能够很好地适用于新的样本，而不是仅仅在训练样本上工作的很好，学得的模型适用于新样本的能力称为泛化能力。
- 误差：学习到的模型在样本上的预测结果与样本的真实结果之间的差。
  - 训练误差：模型在训练集上的误差。
  - 泛化误差：在新样本上的误差。显然，我们更希望得到泛化误差小的模型。

模型选择

- 在进行完数据预处理和特种工程之后，接下来要进行的就是选择合适的模型来训练我们已经处理好的数据和选择的特征。在这个过程中，一般会从如下几个角度来进行分析，以选择合适的算法：
  - 数据集的大小
  - 特征空间的维度
  - 特征是否独立
  - 是否为线性特征
  - 对拟合程度的要求
  - 其他要求：性能、时间、空间
- Note: 影响选择的因素有很多，如果没有特别的要求，尽量选择简单的模型，越简单的越合适。

分类模型简介

- 在实际的工程项目中，分类问题占据了数据挖掘市场的主导地位。因此分类模型的使用也最为常见。常用的分类模型有：
  - 逻辑回归 (Logistic Regression)
  - K最近邻 (KNN)
  - 朴素贝叶斯 (Naïve Bayes)
  - 决策树 (Decision Tree)
  - 支持向量机 (SVM)
  - 集成算法（随机森林、Adaboost、GBDT、Xgboost）

逻辑回归 (Logistic Regression)

- **逻辑回归(LR)**：适用于特征基本线性相关，并且问题线性可分。当特征为非线性特征时，可使用特征工程将非线性特征转换为线性特征（离散化+one-hot，重排序）。正则化使模型对噪声鲁棒，不容易过拟合训练数据，并且可使用正则化进行特征选择。
- LR的另一个特性是可以输出概率，有效支持排序任务和分类阈值的选择。
- 虽然LR准确率不高，不能百分百的有效，但是在尝试其他分类器之前，通常先使用LR+L2来建立一个不错的**基模型**。

K最近邻 (K Nearest Neighbours)

- **KNN：**是一种基于实例的分类方法。该方法就是找出与未知样本x距离最近的k个训练样本，看这k个样本中多数属于哪一类，就把x归为那一类。
- kNN是一种懒惰学习方法，它存放样本，直到需要分类时才进行分类，如果样本集比较复杂，可能会导致很大的计算开销，**因此无法应用到实时性很强的场合。**
- 它的特点是跟着数据走，没有数据模型可言。
- **适用场景：**需要一个特别容易解释的模型，比如需要向用户解释原因的推荐算法。

朴素贝叶斯 (Naïve Bayes)

- **朴素贝叶斯：**主要利用Bayes定理来预测一个位置类别的样本属于各个类别的可能性，选择其中可能性最大的一个类别作为该样本的最终类别。
- 由于贝叶斯定理的成立本身需要一个很强的条件独立性假设前提，**而此假设在实际情况中经常是不成立的，因而其分类准确性就会下降。**
- **适用场景**：可以高效处理高维数据，但是结果可能不尽如人意。至今依然被垃圾邮件过滤器使用的算法。

支持向量机 (SVM)

- **支持向量机(SVM)**：SVM的核心思想就是找到不同类别之间的分界面，使得两类样本尽量落在面的两边，而且离分界面尽量远。
- 最早的SVM是平面的，局限很大。但是利用核函数(kernel function)，我们可以把平面投射(mapping)成曲面，进而大大提高SVM的适用范围。
- **适用场景：**SVM在很多数据集上都有优秀的表现。预测准确度高，且可适用于特征维度高的数据集（文本分类）。

决策树 (DT) 分类模型

- 决策树：是用于分类和预测的主要技术之一，决策树学习是以实例为基础的归纳学习算法，它着眼于从一组无次序、无规则的实例中推理出以决策树表示的分类规则。
- 构造决策树的目的是找出属性和类别间的关系，用它来预测将来未知类别的记录的类别。它采用自顶向下的递归方式，在决策树的内部节点进行属性的比较，并根据不同属性值判断从该节点向下的分支，在决策树的叶节点得到结论。
- 主要的决策树有：ID3、C4.5、CART。
- 适用场景：因为它能够生成清晰的基于特征(feature)选择不同预测结果的树状结构，数据分析师希望更好的理解手上的数据的时候往往可以使用决策树。受限于它的简单性，决策树更大的用处是作为一些更有用的算法的基石。

集成算法 (1)

- **随机森林：**数据维度相对低（几十维），同时对准确性有较高要求时。因为不需要很多参数调整就可以达到不错的效果，基本上不知道用什么方法的时候都可以先试一下随机森林。
- **Adaboost：**具有很高的精度，不容易发生过拟合。可以将不同的分类算法当做弱分类器，且相对于随机森林，Adaboost考虑了每个分类器的权重。但是数据不均衡会导致分类精度下降，且训练比较耗时，如果对训练时间有要求，慎用。主要应用领域为：模式识别、计算机视觉领域，用于二分类和多分类场景。

集成算法 (2)

- **GBDT**：对于分类问题来说，GBDT一般是适用于二分类问题。单独的使用GBDT模型，容易出现过拟合，在实际应用中往往使用 GBDT + LR的方式做模型训练。
- **Xgboost**：与传统机器学习算法相比，Xgboost具有速度快、准确度高等优势，本质上是GBDT的优化，但是把速度和效率做到了极值。常用做推荐算法。

回归问题的模型选择

- 回归分析是研究自变量和因变量之间关系的一种预测模型技术。这些技术应用于预测，时间序列模型和找到变量之间关系。例如，可以通过回归去研究超速与交通事故发生次数的关系。
- 常用的回归模型有：
  - 线性回归
  - 多项式回归
  - 岭回归
  - LASSO回归
  - 决策树及GBDT

线性回归 (Linear Regression)

- 线性回归是利用线性函数来近似确定两种或两种以上变量间相互依赖的定量关系的一种统计分析方法。
- 如果回归分析是线性回归，并且只包括一个自变量和一个因变量，这种回归分析称为一元线性回归分析。如果回归分析中包括两个或两个以上的自变量，则称为多元线性回归分析。

多项式回归 (Polynomial Regression)

- 利用多项式：

$$y = \sum_{n,m} a_n x_1^n + b_m x_2^m + \cdots$$

作为拟合函数来近似确定两种或两种以上变量间相互依赖的定量关系的一种统计分析方法。

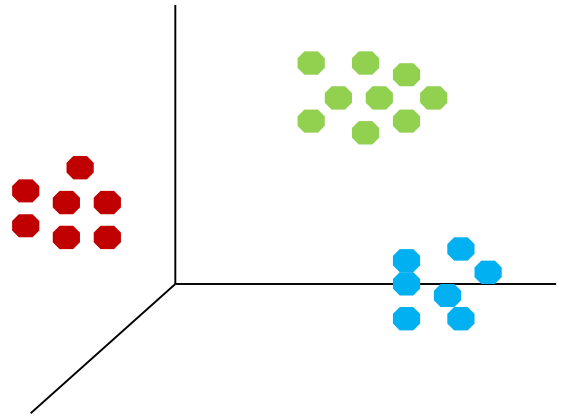
- 同样的，多项式回归也分一元多项式回归分析和多元多项式回归分析，区别就在于自变量x_i 的个数。

回归树 (Regression Tree)

- 决策树除了用于分类模型，还可用于回归分析的场景之中。通常，为了区别两种场景，用于分类模型的决策树被叫做“分类树”，而用于回归模型的被叫做“回归树”。
- 回归树，顾名思义，就是用树模型做回归问题，每一片叶子都输出一个预测值。预测值一般是该片叶子所含训练集元素输出的均值。

聚类分析模型简介

- 聚类分析是分析研究对象（样品或变量）如何按照多个方面的特征进行综合分类的一种多元统计方法，它是根据物以类聚的思想将相似的样品（或变量）归为同一“簇(cluster)”。
  - 把对象分为不同的簇别，簇别是依据数据的特征确定的。
  - 把相似的东西放在一起，簇别内部的差异尽可能小，
  - 簇别之间的差异尽可能的大。



聚类分析两个基本问题

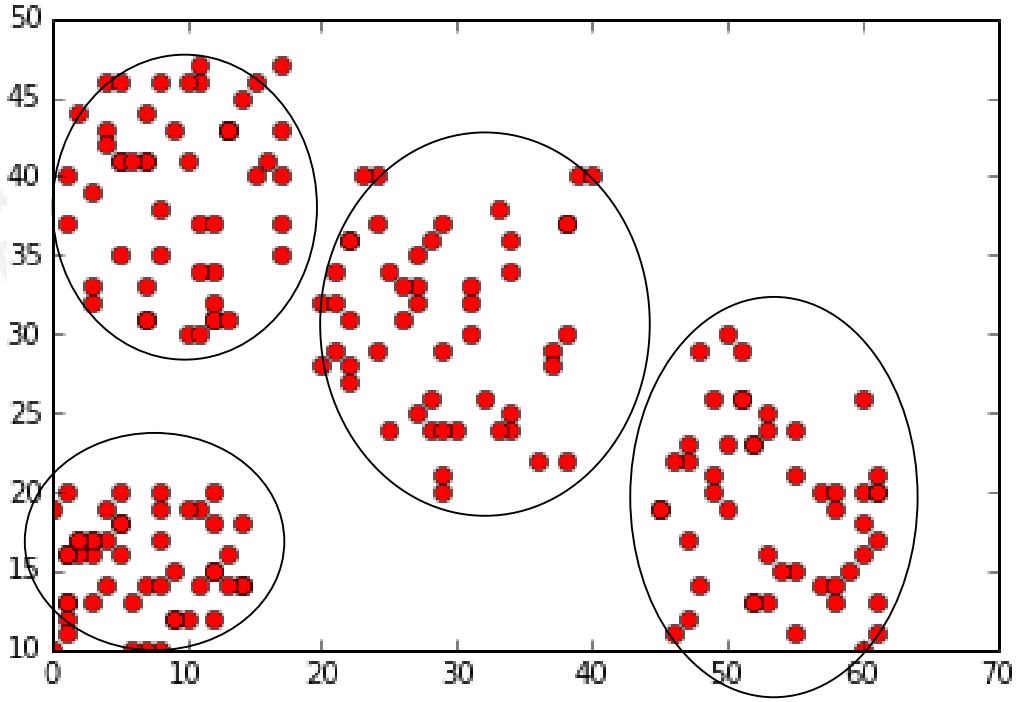
- 距离计算：通常我们是基于某种形式的距离来定义“相似度度量”。距离越大相似度越小，距离越小越相似。
  - 对于距离计算，常用的距离公式包括**欧式距离**和**余弦相似度**：欧氏距离也就是我们常见的两点间距离，而余弦相似度则得名于其计算方式与余弦公式非常相似。
- 性能度量：通过某种性能度量，对聚类结果的好坏进行评估。聚类性能度量一般分两类:
  - 外部指标：将聚类结果与某个“参考模型”进行比较，如将聚类学习结果与业务专家给出的划分结果进行比较。
  - 内部指标：直接考察聚类结果不利用任何参考模型。

常用聚类算法介绍

- 基于原型聚类(partitioning methods)
  - K-Means算法，K-Mediods 算法
- 基于层次聚类(hierarchical methods)
  - Hierarchical Clustering算法、BIRCH算法
- 基于密度聚类(density-based methods)
  - DBSCAN算法

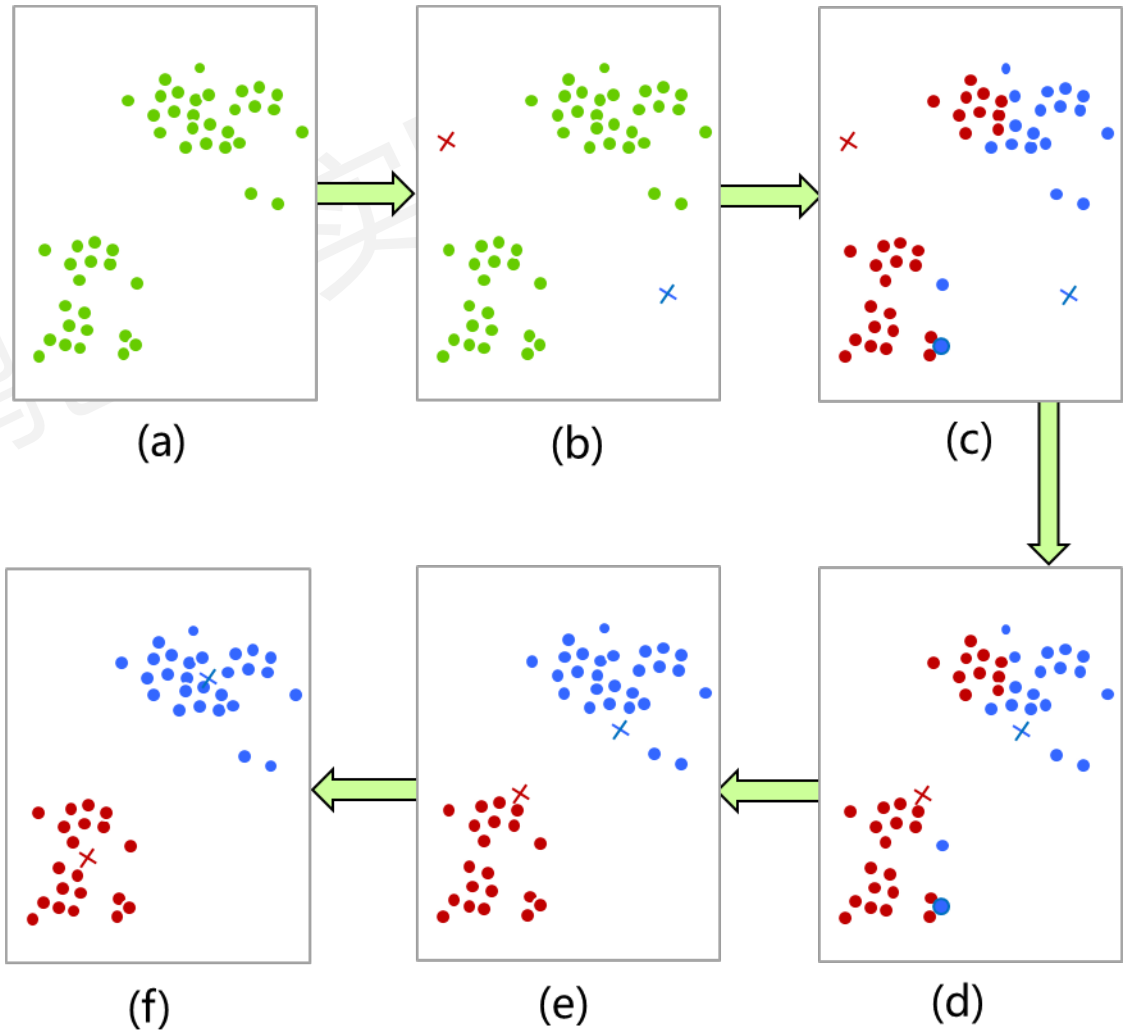
K-Means聚类算法

- 算法思想
  - 输入聚类个数k，以及包含n个数据对象的数据集，输出标准的k个聚类的一种算法。
  - 然后将n个数据对象划分为k个聚类，而最终所获得的聚类满足: (1)同一聚类中的对象相似度较高；(2)而不同聚类中的对象相似度较小。



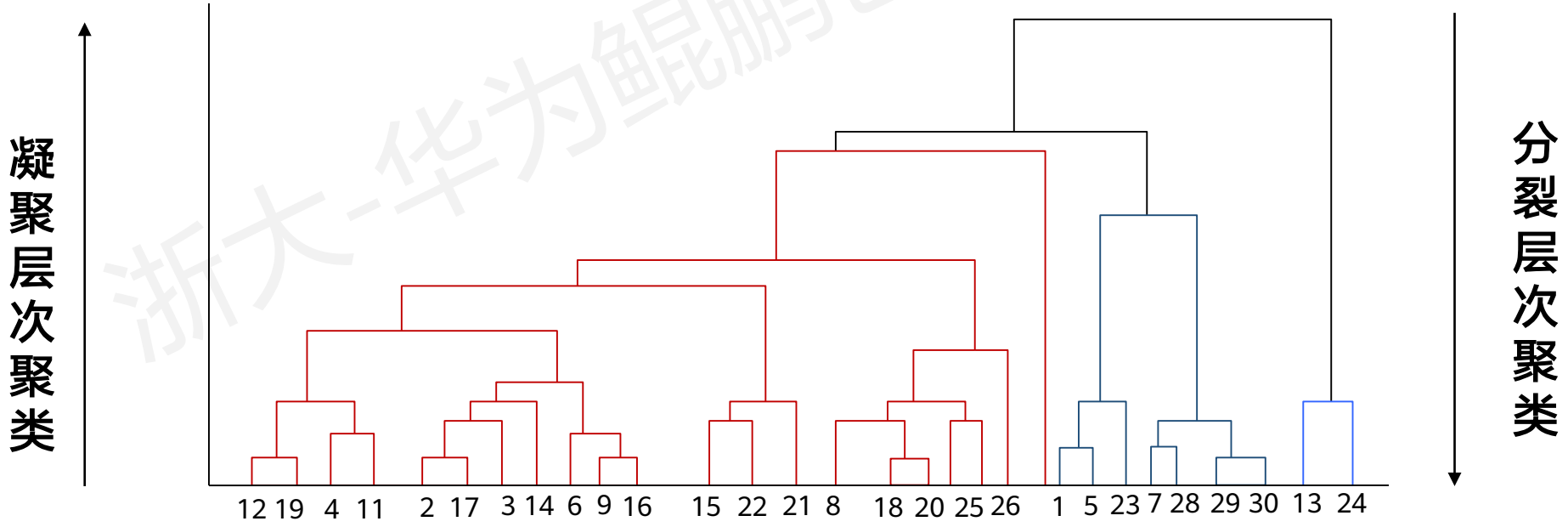
K-Means聚类算法步骤

- 执行步骤
  - 1）选取K个对象作为初始中心（叫质心），作为聚类中心；
  - 2）对每个样本数据，计算它们与中心的欧氏距离，按距离最近的准则将它们分到距离最近的聚类中心所对应的类；
  - 3）更新聚类中心：将每个类别中所有对象所对应的均值作为该类别的新中心（新的质心），计算目标函数；
  - 4）判断聚类中心和目标函数的值是否改变，若不变，则输出结果，若改变，则返回 2）。



层次聚类法

- 层次聚类法试图在不同层次对数据集进行划分，从而形成树形的聚类结构，数据集的划分可采用“自下向上”的聚合策略，也可以采用“自顶向下”的分拆策略。聚类的层次被表示成树形图。树根拥有所有样本的唯一聚类，叶子是仅有一个样本的聚类。



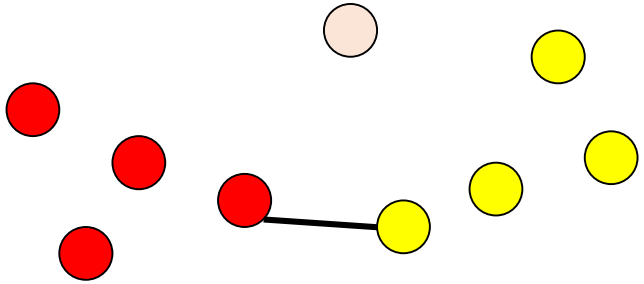
基于层次聚类常用算法

- 层次聚类由不同层次的分割聚类组成，层次之间的分割具有嵌套的关系。它不需要输入参数，这是它的一个明显的优点，其缺点是终止条件必须具体指定。
- 与原型聚类和密度聚类不同，层次聚类试图在不同的“层次”上对样本数据集进行划分，一层一层地进行聚类。
- 典型的分层聚具体有：Hierarchical Clustering算法、 BIRCH算法等。

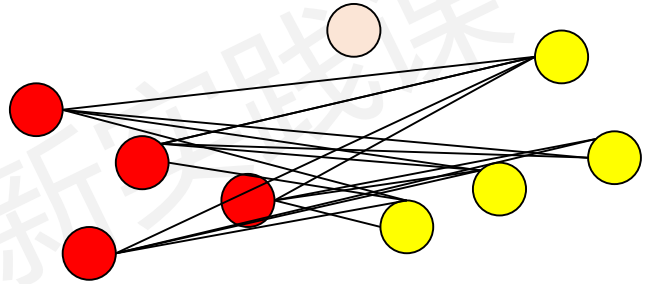
基本概念：判断两个簇之间的距离（1）

距离	描述	公式	参数(linkage)描述
最小距离	定义簇的邻近度为不同两个簇的两个最近的点之间的距离。	$d_{min}(c_i, c_j) = \min_{x \in c_i, y \in c_j} dist(x, y)$	单链接（single linkage）
最大距离	定义簇的邻近度为不同两个簇的两个最远的点之间的距离。	$d_{max}(c_i, c_j) = \max_{x \in c_i, y \in c_j} dist(x, y)$	全链接（complete/Maximum linkage）
平均距离	定义簇的邻近度为取自两个不同簇的所有点对邻近度的平均值。	$d_{avg}(c_i, c_j) = \text{ave}_{x \in c_i, y \in c_j} dist(x, y)$	均链接（average linkage）
方差	所有聚类内的平方差总和。	最小化所有聚类内的平方差总和。	Ward

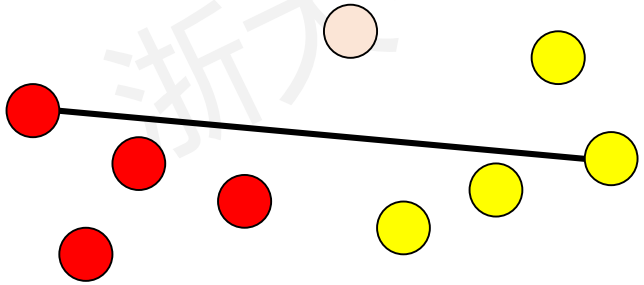
基本概念：判断两个簇之间的距离 (2)



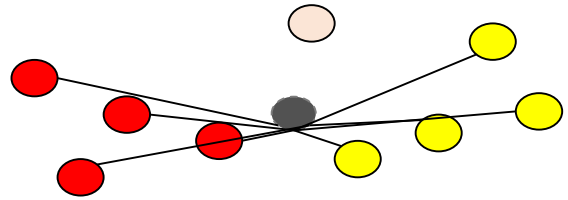
(1)



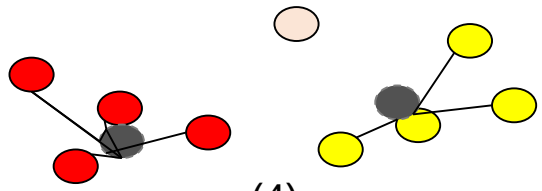
(2)



(3)



VS



(4)

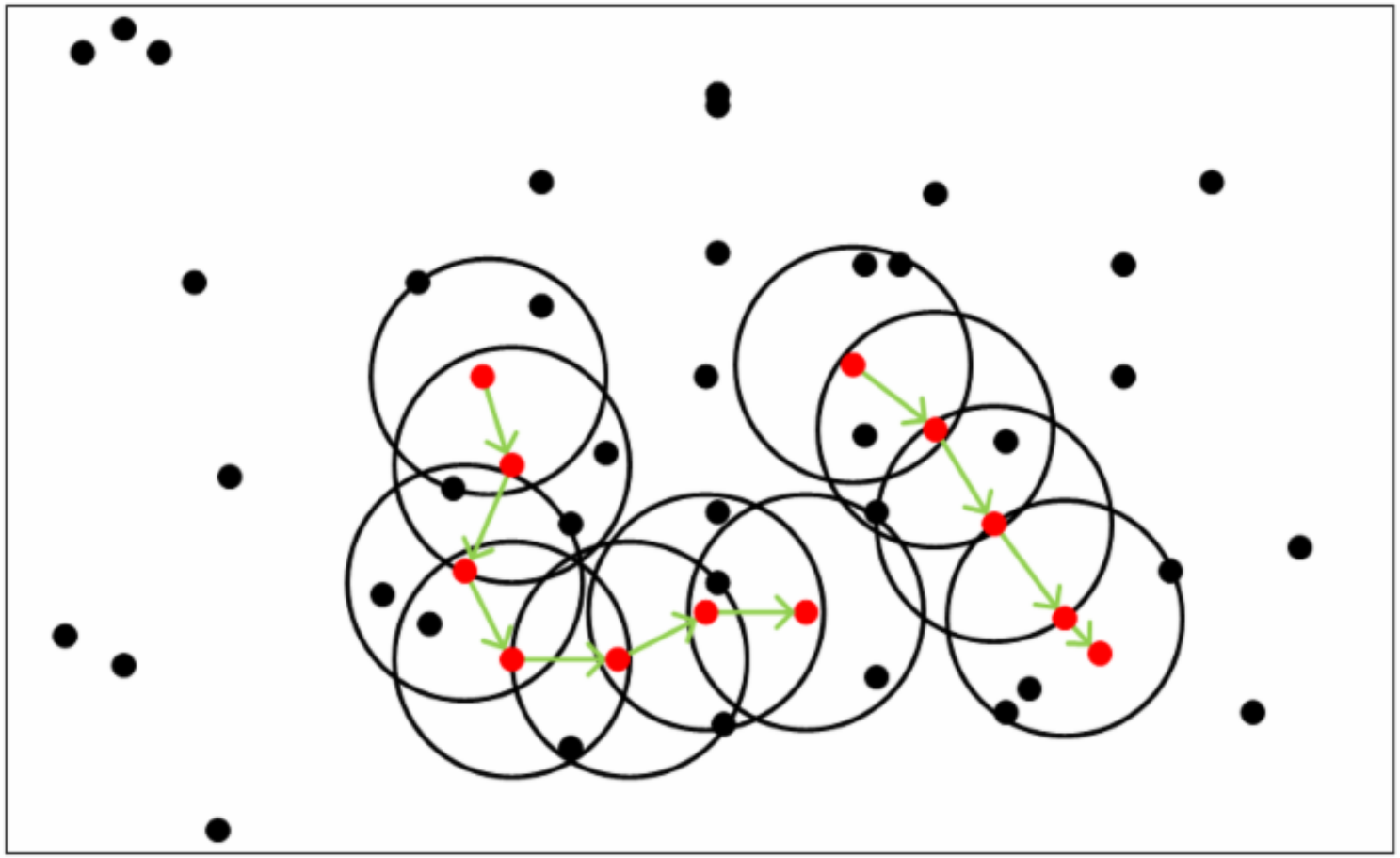
Hierarchical Clustering算法原理

- Hierarchical Clustering算法简单易理解。
- 主要思路：确保距离近的点落在同一个簇(cluster)之中，流程如下：
  - 将每个对象作为一个簇$c_i = \{x_i\}$，形成簇的集合 $C = \{c_i\}$；
  - 迭代以下步骤直至所有对象都在一个族中；
  - 找到一对距离最近的簇：$\min D(c_i, c_j)$；
  - 将这对簇合并为一个新的簇；
  - 从原集合C中移除这对簇；
  - 最终产生层次树形的聚类结构: 树形图。

密度聚类法

- 密度聚类的思想不同于K-Means，它是通过聚类的簇是否紧密相连来判断样本点是否属于一个簇，代表性的算法就是DBSCAN，它基于一组邻域参数来判断某处样本是否是紧密。
- DBSCAN（Density-Based Spatial Clustering of Applications with Noise，具有噪声的基于密度的聚类方法）是一种很典型的密度聚类算法，和K-Means，BIRCH这些一般只适用于凸样本集的聚类相比，DBSCAN还适用于非凸样本集。

DBSCAN算法的基本概念



关联规则

“A事件发生，B事件很可能也会发生。”

Apriori算法简介

- Apriori算法是一种挖掘关联规则的频繁项集算法，其核心思想是通过频繁项集生成和关联规则生成两个阶段来挖掘频繁项集。它的主要任务就是设法发现事物之间的内在联系。
- 比如在常见的超市购物数据集，或者电商的网购数据集中，如果我们找到了频繁出现的数据集，那么对于超市，我们可以优化产品的位置摆放，对于电商，我们可以优化商品所在的仓库位置，达到节约成本，增加经济效益的目的。
- 算法已经被广泛的应用到商业、网络安全，移动通信等各个领域。

关联规则 - 基本概念 (1)

- 基本概念：假设两个不相交的非空集合X、Y（物品集），N为数据记录总数。

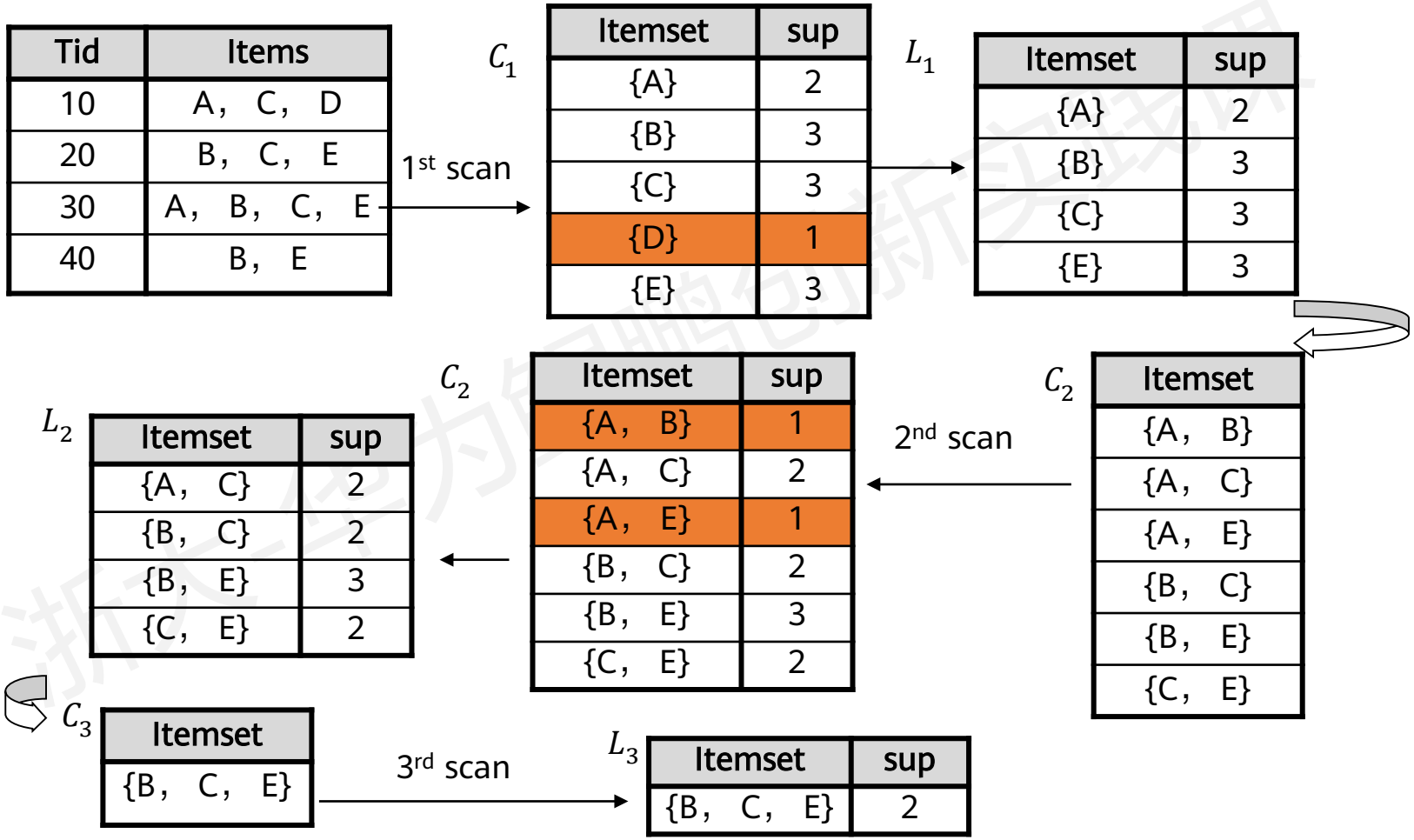
TID	Items
T1	{牛奶，面包}
T2	{面包，尿布，啤酒，鸡蛋}
T3	{牛奶，尿布，啤酒，可乐}
T4	{面包，牛奶，尿布，啤酒}
T5	{面包，牛奶，尿布，可乐}

  - 项集：项的集合，包含k个项的项集称为k项集。
  - 频繁项集：满足规定的最小支持度的项集。
  - 支持度：$support(X \rightarrow Y) = |X \cap Y|/N$，表示物品集X和Y同时出现的次数占总记录数的比例。
  - 置信度：$confidence(X \rightarrow Y) = |X \cap Y|/|X|$，集合X与集合Y同时出现的总次数/集合X出现的记录数 。
  - 提升度：$lift(X \rightarrow Y) = confidence(X \rightarrow Y)/P(Y)$，表示含有X的条件下，同时含有Y的概率，与Y总体发生的概率之比。

关联规则 - 基本概念 (2)

- 最小支持度：专家定义的衡量支持度的阈值，表示项目集在统计意义上的最低重要性。
- 最小置信度：专家定义的衡量置信度的阈值，表示关联规则的最低可靠性。
- 强关联规则：同时满足最小支持度阈值和最小置信度阈值的规则。
- 利用提升度体现了X和Y之间的关联关系:
  - 提升度大于1则$X \rightarrow Y$是有效的强关联规则;
  - 提升度小于等于1则$X \rightarrow Y$是无效的强关联规则;
  - 如果X和Y独立，则有 $Lift(X \rightarrow Y) = 1$ ，因为此时P(X|Y)=P(X)。

Apriori算法举例 (1)



Apriori算法优缺点

- 优点：

  （1）使用先验原理，大大提高了频繁项集逐层产生的效率；

  （2）简单易理解；数据集要求低。

- 缺点：

  （1）每一步产生侯选项目集时循环产生的组合过多，没有排除不应该参与组合的元素;

  （2）每次计算项集的支持度时，都对数据库D中的全部记录进行了一遍扫描比较，如果是一个大型的数据库的话，这种扫描比较会大大增加计算机系统的I/O开销。而这种代价是随着数据库的记录的增加呈现出几何级数的增加。因此人们开始寻求更好性能的算法。

模型调参

- 学习模型中一般有两种参数，一种参数是可以从学习中得到，还有一种无法靠数据里面得到，只能靠人的经验来设定，这类参数就叫做超参数。
- GridSearchCV：最常用的超参数收缩方式是网格搜索，它存在的意义就是自动调参，只要把参数输进去，就能给出最优化的结果和参数。
- 代码展示如下：

```
From sklearn.model_selection import GridSearchCV
#参数包括： estimator（模型的选择）param_grid（要调整参数的选择，传入字典）；cv（设置几折交叉验证）
Model = GridSearchCV()
#输出最好的参数
Print(model.best_params_)
#输出最好的模型
Print(model.best_estimator_)
#输出最好的分数
Print(model.best_score_)
```

目录

最优化模型的概述

- 从某种程度上说，我们的世界是由最优化问题组成的。每一天，我们的生活都面临无数的最优化问题：上班怎么选择乘车路线，才能舒服又快速地到达公司；旅游如何选择航班和宾馆，既省钱又能玩地开心；跳槽应该选择哪家公司，钱多、事少、离家近；买房子应该选在哪里，交通发达有学区，生活便利升值快。
- 可以看出，上面所有的问题都面临无数的选择，我们会根据自己的偏好对每个选择打一个不同的分数，再从所有的选择中找出最优的一个。这个寻求最优解的过程其实就是最优化问题，我们要打的分数就称为目标函数。
- 最优化方法是机器学习中模型训练的基础，机器学习的很大一部分内容就是通过最优化方法找到最合适的参数，使得模型的目标函数最优。

损失函数的概念

- **损失函数(Loss function)**：是用来估量模型的预测值与真实值的不一致程度，是一个非负实值函数。损失函数越小，模型的鲁棒性就越好。损失函数是经验风险函数的核心，也是结构风险函数的重要组成部分。模型的风险结构通常包括风险项和正则项。如下所示：

$$\theta^* = arg\min_{\theta} \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, f(x_i;\theta)\right) + \lambda\Phi(\theta)$$

- 其中，前面的均值函数表示的是经验风险函数，L代表的是**损失函数**，后面Φ是**正则化项**(regularizer)或者叫惩罚项 (penalty term)，它可以是L1，也可以是L2，或者其他的正则函数。整个式子表示的意思是找到使目标函数最小时的θ值。

0-1损失函数

- 0-1损失函数：该损失函数的意义就是，当预测错误时，损失函数值为1，预测正确时，损失函数值不考虑预测值和真实值的误差程度，也就是只要预测错误，预测错误差一点和差很多都一样。

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

- Note: 由于相等这个条件太过严格，因此我们可以放宽条件，即满足 |Y−f(X)|<T时，认为相等。

$$L(Y,f(X))=\begin{cases}1, [Y - f(X)] \geq T\\ 0, [Y - f(X)] < T\end{cases}$$

平方损失函数/绝对值损失函数

- 平方损失函数又常被称为最小二乘法：即预测值与真实值差的平方和。

$$L((Y|f(X)) = \sum_{N}(Y - f(X))^2$$

- 绝对值损失函数：意义与平方损失函数相似，只不过取了绝对值，差距不会被平方缩放。

$$L(\mathbf{Y}, \mathbf{f}(\mathbf{X})) = [Y - f(X)]$$

- 最小二乘法的基本原则是：最优拟合曲线应该使得所有点到回归直线的距离和最小。通常使用欧几里得距离进行距离的度量。
- Note：最小二乘法常被用于线性回归中，它将回归的问题转化为了凸优化的问题。

最优化模型的分类

- 最优化模型分类方法有很多，可按变量、约束条件、目标函数个数、目标函数和约束条件的是否线性，是否依赖时间等分类。
- 根据约束条件来分类。首先最优化问题通常是带约束条件，比如对旅行路线的选择，总花费和出发、到达时间就构成了约束条件；对买房子的选择，离公司的路程、总价也可能构成约束条件。我们选择的最优解也必须满足这些约束条件。
- 最优化问题根据约束条件的不同主要分为三类：
  - 无约束优化
  - 等式约束的优化
  - 不等式约束的优化

数据集划分

- 数据集(dataset)：在机器学习任务中使用的一组数据。数据集中每一个数据称为一个样本。反映样本在某方面的表现或性质的事项或属性称为特征。
- 训练集(training data)：训练过程中使用的数据集。数据集中每个训练样本称为训练样本。从数据中学得模型的过程称为学习（训练）。
- 测试集(testing data)：学得模型后，使用其进行预测的过程称为测试，使用的数据集称为测试集，每个样本称为测试样本。
- 交叉验证集(cross validation data)：用于衡量训练过程中模型的好坏。

模型评估概述

- 模型在训练集上的误差通常称为“训练误差”或“经验误差”，而在新样本上的误差称为“泛化误差”。显然，机器学习的目的是得到泛化误差小的学习器。然而，在实际应用中，新样本是未知的，所以只能使训练误差尽量小。所以，为了得到泛化误差小的模型并避免过拟合，在构建模型时，通常将数据集拆分为相互独立的训练数据集，验证数据集和测试数据集等。
- 在训练过程中使用验证数据集来评估模型并据此更新超参数，训练结束中使用测试数据集评估训练好的模型的性能。

数据集划分的方法

- 不断使用测试集和验证集会使其逐渐失去效果。即使用相同数据来设置超参数或其他模型改进的次数越多，结果的泛化效果就越差。验证集的失效速度通常比测试集缓慢。如果可能的话，建议收集更多数据来“刷新”测试集和验证集。重新开始划分是一种很好的重置方式。
- 为了划分这几种数据集，可以选择采用留出法、K-折交叉验证法或者自助法等多种方法。这些方法都对数据集有一些基本的假设，包括：
  - 数据集是随机抽取且独立同分布的
  - 分布是平稳的且不会随时间发生变化
  - 始终从同一分布中抽取样本
  - Note: 请勿对测试数据集进行训练

均方误差

- 在回归问题中，即预测连续值的问题，最常用的性能度量是“均方误差”。
- 均方误差(mean squared error)： 是反映估计量与被估计量之间差异程度的一种度量。因此本质就是预测值与真实值之间误差平方和的均值。公式如下所示：

$$E(f;D)=\frac{1}{m}\sum_{i=1}^{m}(f(x_i)-\mathrm{y}_i)^2$$

混淆矩阵

- 术语：
  - P：正元组，感兴趣的主要类的元组。
  - N：负元组，其他元组。
  - TP：真正例，被分类器正确分类的正元组。
  - TN：真负例，被分类器正确分类的负元组。
  - FP：假正例，被错误地标记为正元组的负元组。
  - FN：假负例，被错误地标记为负元组的正元组。

真实情况	预测情况		合计
	正例 (True)	反例 (False)	
正例 (positive)	TP	FN	P
反例 (negative)	FP	TN	N
合计	P′	N′	P+N

混淆矩阵

- 混淆矩阵：是一个至少为$m \times m$的表。前m行和m列的表目$CM_{i,j}$指出类i的元组被分类器标记为j的个数。
  - 理想地，对于高准确率分类器，大部分元组应该被混淆矩阵从$CM_{1,1}$到$CM_{m,m}$的对角线上的表目表示，而其他表目为0或者接近于0。即FP和FN接近0。

精度/错误率

- 在分类问题中，即预测离散值的问题， 最常用的错误率和精度，错误率是预测错误的样本数占样本总数的比例，又被称为汉明损失(Hamming loss)。精度则是分类正确的样本数占样本总数的比例，又被称为预测准确率(Accuracy)。错误率+精度 = 1
- 错误率定义为：(FP+FN)/(P+N)
- 精度则定义为：(TP+TN)/(P+N)

查准率/查全率

- 查准率 (precision)：在所判别的正例结果中，真正正例的比例。可表示为：TP/(TP+FP)。查准率表示分类算法预测是否分类为1中实际为0的误报成分（真正例样本数/预测结果是正例的样本数）。
- 查全率 (Recall)：又被称为召回率，是指分类器预测为正例的样本占实际正例样本的比例。可表示为：TP/(TP+FN)。查全率则表示算法预测是否漏掉了一些该分为1的，却被分为0的成分，也就是漏报的（真正例样本数/真实是正例的样本数）。
- 宁愿漏掉，不可错杀：一般适用于识别垃圾邮件的场景中。因为我们不希望很多的正常邮件被误杀，这样会造成严重的困扰。因此，在此类场景下Precision 将是一个重要的指标。
- 宁愿错杀，不可漏掉：一般适用于金融风控领域。我们希望系统能够筛选出所有风险的行为或用户，然后进行人工鉴别，如果漏掉一个可能造成灾难性后果。因此，在此类场景下，Recall将是一个重要的指标。

综合评价 (F-Score)

- 其中 β 用于调整权重，当 β = 1 时两者权重相同， 简称为F1 – Score， 此时精确率和召回率都很重要，权重相同。若认为 Precision 更重要，则减小β。若认为Recall更重要，则增大β。

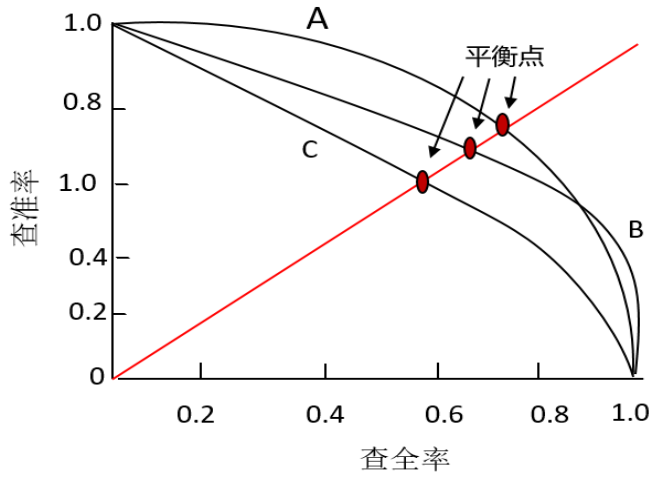
$$F - \text{Score} = (1 + \beta^2).\frac{Precision\ .\ Recall}{\beta^2\ .(Precision + Recall)}$$

- F1-score可表示为：

$$F1\text{–}score = \frac{2}{\left(\frac{1}{precision}\right) + \left(\frac{1}{recall}\right)} = \frac{2.\ Precision\ .\ Recall}{Precision + Recall}$$

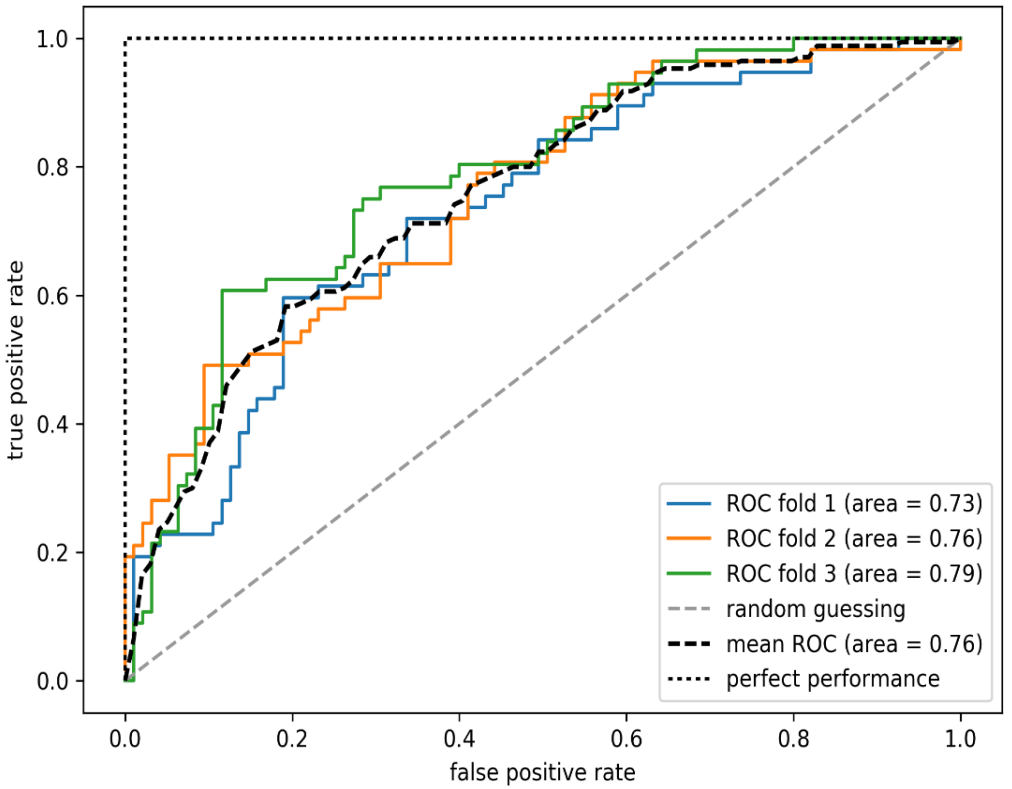
PR曲线

- PR曲线是描述查准率/查全率变化的曲线，以查准率(precision)和查全率(recall)分别作为纵、横轴坐标。
- 根据学习期的预测结果（一般为一个实值或概率）对测试样本进行排序，将最可能是“正例”的样本排在前面，最不可能是“正例”的排在后面，按此顺序逐个把样本作为“正例”进行预测，每次计算出当前的P值和R值。



ROC曲线

- 学习器对测试样本的评估结果一般为一个实值或概率输出，设定一个阈值，大于阈值为正例，小于阈值为负例，因此这个实值/概率输出的好坏直接决定了学习器的泛化性能，若将这些实值/概率输出排序，则排序的好坏决定了学习器的性能高低。 ROC曲线正是从这个角度出发来研究学习器的泛化性能。
- ROC曲线与PR曲线十分类似，都是按照排序的顺序逐一按照正例预测，不同的是ROC曲线以“假正例率”（ False Positive Rate，简称FPR ）为横轴，以 “真正例率”（ True Positive Rate，简称TPR ）为纵轴，**ROC偏重研究 基于测试样本评估值的排序好坏。**



AUC面积

- 进行模型的性能比较时，若一个学习器A的ROC曲线被另外一个学习器B的ROC曲线完全包住，则称B的性能优于A。若A和B的曲线发生了交叉，则谁的曲线与坐标轴围成的面积大，谁的性能更优。ROC曲线下的面积定义为AUC(Area Under ROC Curve)。
- AUC越大，证明分类的质量越好，AUC为1时，证明所有正例排在了负例前面，AUC为0时，所有的负例排在了正例的前面。

聚类模型评估方法

- 聚类的评价方式在大方向上被分成两类，一种是分析外部信息，另一种是分析内部信息。外部信息就是能看得见的直观信息，这里指的是聚类结束后的类别号。还有一种分析内部信息的办法，本质就是聚完类后会通过一些模型生成这个类聚的怎么样的参数，诸如熵和纯度这种数学评价指标。
- 较为常用的分析内部信息的方法，例如互信息评分，兰德系数，轮毂系数等。

Silhonette轮廓系数

- 轮廓系数适用于实际类别信息未知的情况。对于单个样本，设a是其与同类别中其他样本的平均距离，b是这个样本与距离它最近的不同类别中样本的平均距离，其轮廓系数为：

$$S=\frac{b-a}{\max(a,b)}$$

- 对于一个样本集合，它的轮廓系数是所有样本轮廓系数的平均值。轮廓系数的取值范围是[-1,1]，同类别样本距离越相近且不同类别样本距离越远时，轮廓系数越高。

本章总结

- 本章主要介绍数据挖掘的相关概念，包括数据挖掘的定义，数据挖掘与数据分析的对比，数据挖掘基本流程，数据预处理与特征工程，数据挖掘常用算法，以及模型评估的基本标准。

学习推荐

- Scikitlearn 中文学习资料库：
  - http://www.scikitlearn.com.cn/
- Scikitlearn 官方文档：
  - https://scikit-learn.org/stable/


谢 谢
www.huawei.com