1. 引言
2. 数据中的知识发现包括哪几个步骤？

数据的聚集-数据选择与预处理-数据挖掘-解释说明与评价

1. 数据挖掘应用

数据分析与决策支持（市场分析与管理、风险分析与管理、诈骗识别及识别异常事件）

文本挖掘与网页挖掘

流数据挖掘

生物信息分析

-目标识别

-文本分类

-语音识别

-传感数据建模

-自动驾驶

-推荐系统

-医疗诊断

1. 数据和数据预处理
2. 有哪四种不同的属性类型？分别可以进行什么操作？

|  |  |
| --- | --- |
| 标称属性（nominal attribute） | Mode 熵 contingency correlation列联相关 X2 |
| 序数属性（ordinal attribute） | 中位数、百分数、秩相关Rank correlation、运行测Run test、符号验证Sign tests |
| 区间属性（Interval attribute） | 平均数、标准差、皮尔逊相关、t and F测试 |
| 数值属性（Ratio attribute） | 几何平均值、调和中项、百分比变化 |

1. 非对称属性？

只有非零属性值是有意义的。

1. **数据对象之间相似度、相异度计算**

欧几里得Euclidean Distance



Minkowski Distance



Mahalanobis Distance



1. 数据预处理的主要任务

数据清理（增添缺失值、异常数据清理、错误纠正、重复数据清除）数据整合（多合一、统一标识、识别与处理数据值冲突，处理重复数据） 数据转换（标准化） 数据减少（数据压缩，回归、直方图、聚类、取样） 数据离散

1. 处理缺失值的方法？

减少数据集：排除有缺失值的样本

找到缺失值：专家填写合理的缺失值，根据约束条件自动填补缺失值（均值等）

1. 概念学习

<https://www.cnblogs.com/lufangtao/archive/2013/05/24/3086935.html>

1. 概念学习的定义

概念学习就是学习把具有共同属性的事物集合在一起并冠以一个名称，把不具有此类属性的事物排除出去。

1. Find-S算法

Find-S算法是概念学习中最简单的一个算法，目的是给定训练集，学习出训练集中的正例所代表的概念。算法的步骤是：  
1、将h初始化为假设空间H中最特殊的假设；  
2、对于每一个训练数据：（1）如果是反例，h不变；（2）如果是正例，h变为和正例一致的最特殊的假设  
3、输出h  
  
因此我们可以看出h是和所有正例都一致的最特殊的假设。它的优点是简单，缺点是如果训练集有噪声的话，会很大的影响性能，并且对反例提供的信息并没有利用上，并且h可能是不唯一的。

1. 什么是归纳偏置

机器学习试图去建造一个可以学习的算法，用来预测某个目标的结果。要达到此目的，要给于学习算法一些训练样本，样本说明输入与输出之间的预期关系。然后假设学习器在预测中逼近正确的结果，其中包括在训练中未出现的样本。既然未知状况可以是任意的结果，若没有其它额外的假设，这任务就无法解决。这种关于目标函数的必要假设就称为归纳偏置

1. 变型空间

候选消除算法能够表示与训练样例一致的所有假设。在假设空间中的这一子集被称为关于假设空间H和训练样例D的变型空间(version space)，因为它包含的是目标概念的所有合理的变型。

1. 决策树学习
2. 决策树学习的基本思想

决策树的算法框架：

决策树的主函数：各种决策树的主函数大同小异，本质上是个递归函数，该函数主要功能是根据某种规则生长出决策树的各个分支节点，并根据终止条件结束算法。

输入需要分类的数据集和类别标签

根据某种分类规则得到最优的划分特征，并创建特征的划分节点——计算最优特征子函数

按照该特征的每个取值划分数据集为若干部分——划分数据集子函数

根据划分子函数的计算结果构建出新的节点，作为树生长出的新分支

检验是否符合递归终止条件

将划分的新节点包含的数据集和类别标签作为输入，递归执行上述步骤

计算最优特征子函数：不同标准导致不同类型的决策树，如ID3的最优特征选择标准是信息增益，C4.5是信息增益率，CART是节点方差的大小等等。算法逻辑上，一般选择最优特征需要遍历整个数据集，评估每个特征，找到最优的那个返回。

划分数据集子函数：分隔数据集。

分类器：将测试遍历整个生成的决策树，并找到最终的叶子节点的类别标签。这个标签就是返回的结果。

1. 分类错误率，熵，信息增益的概念，如何根据不同度量选择最佳划分

信息熵是一个数学上颇为抽象的概念，在这里不妨把信息熵理解成某种特定信息的出现概率（离散随机事件的出现概率）。一个系统越是有序，信息熵就越低；反之，一个系统越是混乱，信息熵就越高。信息熵也可以说是系统有序化程度的一个度量。

信息增益”（Information Gain）来衡量一个属性区分以上数据样本的能力。信息增益量越大，这个属性作为一棵树的根节点就能使这棵树更简洁，比如说一棵树可以这么读成，如果风力弱，就去玩；风力强，再按天气、温度等分情况讨论，此时用风力作为这棵树的根节点就很有价值。如果说，风力弱，再又天气晴朗，就去玩；如果风力强，再又怎么怎么分情况讨论，这棵树相比就不够简洁了。

1. 缺失值对决策树有何影响？

抛弃极少量的缺失值的样本对决策树的创建影响不是太大。但是如果属性缺失值较多或是关键属性值缺失,创建的决策树将是不完全的,同时可能给用户造成知识上的大量错误信息,所以抛弃缺失值一般不采用。

如果数据库的数据较大,缺失值较多(当然,这样获取的数据库在现实中使用的意义已不大,同时在信息获取方面基本不会出现这样的数据库),这样根据填充后的数据库创建的决策树可能和根据正确值创建的决策树有很大变化。

1. 给定混淆矩阵，分类效果度量不同指标的含义及计算方法。

混淆矩阵就是为了进一步分析性能而对该算法测试结果做出的总结。

查准率 = 精度 = precision =tp/(tp+fp)

查全率 = 召回率 = recall=tp/(tp+fn)

1. 评估分类器性能的留一法和k折交叉验证

k折交叉验证 k-fold cross validation

1、 将全部训练集 S分成 k个不相交的子集，假设 S中的训练样例个数为 m，那么每一个子 集有 m/k 个训练样例，，相应的子集称作 {s1,s2,…,sk}。

2、每次从分好的子集中里面，拿出一个作为测试集，其它k-1个作为训练集

3、根据训练训练出模型或者假设函数。

4、 把这个模型放到测试集上，得到分类率。

5、计算k次求得的分类率的平均值，作为该模型或者假设函数的真实分类率。

这个方法充分利用了所有样本。但计算比较繁琐，需要训练k次，测试k次。

留一法 leave-one-out cross validation

留一法就是每次只留下一个样本做测试集，其它样本做训练集，如果有k个样本，则需要训练k次，测试k次。

留一发计算最繁琐，但样本利用率最高。适合于小样本的情况。

1. 过拟合和欠拟合

当某个模型过度的学习训练数据中的细节和噪音，以至于模型在新的数据上表现很差，我们称过拟合发生了。这意味着训练数据中的噪音或者随机波动也被当做概念被模型学习了。而问题就在于这些概念不适用于新的数据，从而导致模型泛化性能的变差。

简单理解就是训练样本的得到的输出和期望输出基本一致，但是测试样本输出和测试样本的期望输出相差却很大 。为了得到一致假设而使假设变得过度复杂。

欠拟合指的是模型在训练和预测时表现都不好的情况。一个欠拟合的机器学习模型不是一个良好的模型并且由于在训练数据上表现不好这是显然的。

过拟合：在训练数据上表现良好，在未知数据上表现差。   
欠拟合：在训练数据和未知数据上表现都很差。

1. 神经网络
2. 神经网络如何学习？

在神经网络中随机初始化权重

我们将第一组输入值发送给神经网络，使其传播通过网络并得到输出值。

我们将输出值和期望的输出值进行比较，并使用成本函数计算误差。

我们将误差传播回网络，并根据这些信息设置权重。

对于训练集中的每个输入值，重复2至4的步骤。

当整个训练集都发送给了神经网络，我们就完成了一个epoch, 之后重复多次epochs。

所以，这只是神经网络如何学习的一个简化表示。我没有提到的是，在实践中，训练集被分成两部分，第二部分用于验证网络。

1. 神经网络学习有何局限性？

一般来说，任何需要推理的程序设计，或应用科学的方法（长期规划和类似算法的数据操作），无论你投入多少数据，深度学习模型都是无法完成的。即使学习具有深层神经网络的排序算法也是难以实现的。

我们人类自己对图像，声音和语言的理解，建立在我们作为人类的感受体验中，地球上的生物也是如此感受和理解。机器学习模型无法获得这种感受和理解，因此不能以人类的感觉去要求机器学习。通过注释大量的训练示例来填充我们的模型，我们让它们学习一个几何变换，将数据映射到这个特定的一组示例上，但是这个映射只是我们思想中原始模型的简单草图，机器学习像个执行者，执行人类的思路，但没有感情和理解，它们的行为准则都是人类赋予。

1. 梯度下降算法

梯度下降法的计算过程就是沿梯度下降的方向求解极小值（也可以沿梯度上升方向求解极大值）。

将训练集放入神经网络并获得输出。

将输出与期望输出做对比，并使用成本函数计算误差。

基于误差值和使用的成本函数，决定如何改变权重以使误差最小。

重复该过程直到误差值最小。

我刚刚所解释的还有另外一个名字---- 批次梯度下降（Batch Gradient Descent）。这是因为我们把整个训练集放在网络中，然后修改权重。这种方法的问题是，我们可能到达误差函数的一个局部最小值，而不是全局最小值。这也是神经网络中最大的难题之一，不过有多种方式可以解决它。

1. 多层神经网络使用什么算法进行训练？

BP算法

1. 遗传算法
2. 概念：交叉、变异、适应度函数

所谓交叉是指把两个父代个体的部分结构加以替换重组而生成新个体的操作。通过交叉，遗传算法的搜索能力得以飞跃提高。交叉算子根据交叉率将种群中的两个个体随机地交换某些基因，能够产生新的基因组合，期望将有益基因组合在一起。

变异算子的基本内容是对群体中的个体串的某些基因座上的基因值作变动。

遗传算法引入变异的目的有两个:一是使遗传算法具有局部的随机搜索能力。当遗传算法通过交叉算子已接近最优解邻域时，利用变异算子的这种局部随机搜索能力可以加速向最优解收敛。显然，此种情况下的变异概率应取较小值，否则接近最优解的积木块会因变异而遭到破坏。二是使遗传算法可维持群体多样性，以防止出现未成熟收敛现象。此时收敛概率应取较大值。

遗传算法中，交叉算子因其全局搜索能力而作为主要算子，变异算子因其局部搜索能力而作为辅助算子。遗传算法通过交叉和变异这对相互配合又相互竞争的操作而使其具备兼顾全局和局部的均衡搜索能力。所谓相互配合.是指当群体在进化中陷于搜索空间中某个超平面而仅靠交叉不能摆脱时，通过变异操作可有助于这种摆脱。所谓相互竞争，是指当通过交叉已形成所期望的积木块时，变异操作有可能破坏这些积木块。

1. 遗传算法基本思想

遗传算法是从代表问题可能潜在的解集的一个种群开始的，而一个种群则由经过基因编码的一定数目的个体组成。每个个体实际上是染色体带有特征的实体。染色体作为遗传物质的主要载体，即多个基因的集合，其内部表现（即基因型）是某种基因组合，它决定了个体的形状的外部表现，如黑头发的特征是由染色体中控制这一特征的某种基因组合决定的。因此，在一开始需要实现从表现型到基因型的映射即编码工作。由于仿照基因编码的工作很复杂，我们往往进行简化，如二进制编码，初代种群产生之后，按照适者生存和优胜劣汰的原理，逐代演化产生出越来越好的近似解，在每一代，根据问题域中个体的适应度（fitness）大小选择（selection）个体，并借助于自然遗传学的遗传算子（genetic operators）进行组合交叉（crossover）和变异（mutation），产生出代表新的解集的种群。这个过程将导致种群像自然进化一样的后生代种群比前代更加适应于环境，末代种群中的最优个体经过解码（decoding），可以作为问题近似最优解。

1. 遗传算法框架

a)初始化：设置进化代数计数器t=0，设置最大进化代数T，随机生成M个个体作为初始群体P(0)。

b)个体评价：计算群体P(t)中各个个体的适应度。

c)选择运算:将选择算子作用于群体。选择的目的是把优化的个体直接遗传到下一代或通过配对交叉产生新的个体再遗传到下一代。选择操作是建立在群体中个体的适应度评估基础上的。

d)交叉运算：将交叉算子作用于群体。遗传算法中起核心作用的就是交叉算子。

e)变异运算：将变异算子作用于群体。即是对群体中的个体串的某些基因座上的基因值作变动。

群体P(t)经过选择、交叉、变异运算之后得到下一代群体P(t+1)。

f)终止条件判断:若t=T,则以进化过程中所得到的具有最大适应度个体作为最优解输出，终止计算。

1. 贝叶斯学习
2. 根据贝叶斯理论，如何计算一个假设h成立的后验概率？

贝叶斯公式提供了从先验概率P(h)、P(D)和P(D|h)计算后验概率P(h|D)的方法

p(h|D)=P(D|H)\*P(H)/P(D)

P(h|D)随着P(h)和P(D|h)的增长而增长，随着P(D)的增长而减少，即如果D独立于h时被观察到的可能性越大，那么D对h的支持度越小。

1. 极大后验概率假设和极大似然假设有何区别？

极大后验概率假设

学习器在候选假设集合H中寻找给定数据D时可能性最大的假设h，h被称为极大后验假设（MAP）确定MAP的方法是用贝叶斯公式计算每个候选假设的后验概率，计算式如下:

h\_map=argmax P(h|D)=argmax (P(D|h)\*P(h))/P(D)=argmax P(D|h)\*p(h) (h属于集合H)最后一步，去掉了P(D)，因为它是不依赖于h的常量。

极大似然假设有

在某些情况下，可假定H中每个假设有相同的先验概率，这样式子可以进一步简化，只需考虑P(D|h)来寻找极大可能假设。

h\_ml = argmax p(D|h) h属于集合H

P(D|h)常被称为给定h时数据D的似然度，而使P(D|h)最大的假设被称为极大似然假设。

1. 最小描述长度的基本思想

其基本原理是对于一组给定的实例数据 D ， 如果要对其进行保存 ，为了节省存储空间， 一般采用某种模型对其进行编码压缩，然后再保存压缩后的数据。同时， 为了以后正确恢复这些实例数据，将所用的模型也保存起来。所以需要保存的数据长度( 比特数) 等于这些实例数据进行编码压缩后的长度加上保存模型所需的数据长度，将该数据长度称为总描述长度。最小描述长度( MDL) 原理就是要求选择总描述长度最小的模型。

1. 贝叶斯最优分类器的基本思想

贝叶斯判定准则：为最小化总体风险，只需要在每个样本上选择那个能使条件风险最小的类别标记，即

h∗(x)=argminc∈Y R(c∣x)=argmaxc∈YP(c∣x)

此时，称h∗为贝叶斯最优分类器，与之对应的总体风险R(h∗)为贝叶斯风险，1−R(h∗)反映了分类器能达到的最好的性能。

1. 朴素贝叶斯分类算法

如果需要最小化风险，就必须知道后验概率P(c∣x) . 从这个角度来看，所有的机器学习算法就是基于训练样本来估计后验概率。主要的策略有：

1. 给定x，通过直接建模P(c∣x),然后预测c，这种得到的模型称之为“判别式模型“。

2. 先对联合概率分布P(c,x)进行建模，然后由此获得P(c∣x)

,这种模型称之为 “生成模型“。

对于生成模型来说，有：

P(c∣x)=P(x,c)/P(x)

又结合贝叶斯定理：

P(c∣x)=P(x∣c)P(c)/P(x) (5)

贝叶斯决策的难点在于：类条件概率P(x∣c)

是所有属性上的联合概率分布。朴素贝叶斯分类器假设所有的属性是条件独立的（注意不同于独立），那么公式(5)就可以转化为：

P(c∣x)=P(c)P(x1,x2,⋯,xd|c)/P(x)=P(c)/P(x)∏i=1 P(xi∣c)←(根据各维度的独立性可得)(6)=P(c)∏di=1P(xi∣c)/∑nj=1P(x∣cj)P(cj)←(对分母使用全概率公式可得)=P(c)∏di=1P(xi∣c)/∑nj=1P(cj)∏di=1P(xi∣c)←(对分母中的对应分量的条件独立性可得)

在公式（6）中可知P(x)与x所属的类别没有关系，也就是不论它属于哪个类别，在计算不同类别的后验概率的时候其值都是相等的。那么贝叶斯分类器的问题可以描述为：

y^(i)=argmaxcjP(cj∣x)=argmaxcjP(cj)∏i=1dP(xi∣cj)

这就是我们所需要求解的贝叶斯分类器，我们通过训练数据学习获得参数P(cj)和P(xi∣cj)。参数的估计方法可以使用极大似然估计。

1. 基于实例的学习

1. K近邻学习算法

近邻算法(Nearest Neighbors)是一种典型的非参模型，与生成方法(generalizing method)不同的是，在近邻算法中，通过以实例的形式存储所有的训练样本，假设有m个训练样本：

此时需要存储这m个训练样本，因此，近邻算法也称为基于实例的模型。

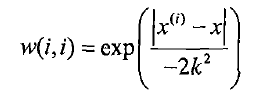
对于一个需要预测的样本，通过与存储好的训练样本比较，以较为相似的样本的标签作为近邻算法的预测结果。

2. K近邻学习计算距离时为何要进行归一化

在处理不同取值范围的特征值时，通常要归一化，以防止某个特征值过大造成距离计算完全取决于该特征值

3. 局部加权线性回归

局部加权回归是基于非参数学习算法的思想，使得特征的选择更好。赋予预测点附近每一个点以一定的权值，在这上面基于波长函数来进行普通的线性回归.可以实现对临近点的精确拟合同时忽略那些距离较远的点的贡献，即近点的权值大，远点的权值小，k为波长参数，控制了权值随距离下降的速度，越大下降的越快。越小越精确并且太小可能出现过拟合的问题。



普通的线性回归函数使用的最小二乘法，即cost函数是

要使得J(Θ)最小化，求得线性回归参数



加入局部加权算法，cost函数变成了加权的cost函数，如下面的形式



其中w(i)是权重

同样可以求得线性回归参数Θ为下面的形式

在使用这个算法训练数据时，我们需要学习两类参数，线性回归的参数以及波长参数。

算法的缺点主要在于对于每一个需要预测的点，都要重新依据整个数据集模拟出一个线性回归模型出来，使得算法的代价极高。

4. 基于案例的学习与k近邻学习的异同

相同点：他们都是懒惰学习方法，当请求到达时再开始泛化。他们通过分析相似的案例来给新的案例分类而忽略那些不同的案例。

不同点：基于案例的学习不会表示案例为真实值点，他们用一种富符号表示。另外，检索相似案例的方法更加细致

5. 懒惰学习和积极学习的区别

Eager Learning考虑到了所有训练样本，说明它是一个全局的近似，虽然它需要耗费训练时间，但它的决策时间基本为0.

Lazy Learning在决策时虽然需要计算所有样本与查询点的距离，但是在真正做决策时却只用了局部的几个训练数据，所以它是一个局部的近似，然而虽然不需要训练，它的复杂度还是需要 O(n),n 是训练样本的个数。由于每次决策都需要与每一个训练样本求距离，这引出了Lazy Learning的缺点：(1)需要的存储空间比较大 (2)决策过程比较慢。

1. 集成学习

1. 集成学习的定义

集成学习通过构建并结合多个学习器来完成学习任务。集成学习通过将多个学习器进行结合，常获得比单一学习器显著优越的泛化性能。

2. 集成学习的两个主要问题

第一是如何得到若干个个体学习器，第二是如何选择一种结合策略

3. Stacking基本思想及其伪代码

Stacking的基本思想是训练一个基本分类器池，然后使用另一个分类器来组合它们的预测，目的是减少泛化误差。

Stacking算法伪代码

1将训练集分成两个不想交的部分；

2在第一部分的训练集上训练若干个基本学习器；

3在第二部分的训练集上测试得到的基本学习器；

4使用步骤3中的预测结果作为输入，将正确的响应（responses）作为输出，训练更高级别的学习器。

4. Bagging基本思想及其伪代码

bagging，对训练集采用有放回采样。通过对原数据集进行有放回的采样，构建出大小和原数据集T一样的新数据集T1，T2，T3…..，然后用这些新的数据集训练多个分类器f1，f2，f3….。最终分类结果根据这些分类器各自结果的投票来决定。

Bagging算法伪代码

给定一个大小为n训练集T;

Bagging通过从T中进行有放回采样，得到n’个新的训练集Ti, 每个Ti的大小为m;

将得到的n’个新训练集，分别用于训练分类器，得到n’个结果；

将n’个结果进行投票，票数多的为最终分类值。

5. Boosting基本思想及其伪代码

Boosting，是一个迭代的过程，每次在新分类器中强调上一个分类器中被错误分类的样本（增加错误分类样本的权重），最后将这些模型组合起来的方法。每次对正确分类的样本降权，对错误分类的样本加权，最后分类器是多个弱分类器的加权组合。Boosting没有对样本进行重采样，而是对样本的分布进行了调整。

Boosting算法伪代码

1在训练集T上，训练一个弱分类器；

2根据上一步的结果对训练集T进行权值调整，训练集T中数据被赋予新的权值：对错分的样本数据增加权重，对正确分类的样本数据进行降低权重；得到权值调整后，更新好的训练集T’；

3在权值调整后的训练集T’上，进行弱分类器的学习训练；

4迭代步骤2，直到弱学习器数达到事先指定的数目X

5最终将这X个弱学习器通过结合策略进行整合，得到最终的强学习器。

1. 分类技术
2. 基于规则的分类器有何优点，需要解决什么问题

规则集的表达能力几乎等价于决策树，因为决策树可以用互斥和穷举的规则集表示。基于规则分类器和决策树分类器都对属性空间进行直线划分，并将类指派到每个划分。然而，如果基于规则的分类器允许一条记录触发多条规则的话，就可以构造一个更加复杂的决策边界。

基于规则的分类器通常被用来产生更易于解释的描述性模型，而模型的性能却可与决策树分类器相媲美。

被很多基于规则的分类器（如RIPPER）所采用的基于类的规则定序方法非常适于处理类分布不平衡的数据集。

1. 序列覆盖算法

把一个复杂的问题分成两个或更多的相同或相似的子问题，再把子问题分成更小的子问题……直到最后子问题可以简单的直接求解，原问题的解即子问题的解的合并。

分治法在每一层递归上都有三个步骤：

step1 分解：将原问题分解为若干个规模较小，相互独立，与原问题形式相同的子问题；

step2 解决：若子问题规模较小而容易被解决则直接解，否则递归地解各个子问题

step3 合并：将各个子问题的解合并为原问题的解。

1. 支持向量机基本原理

支持向量机学习的基本思想是求解能够正确划分训练数据集并且几何间隔最大的分离超平面。对线性可分得训练数据集而言，线性可分分离超平面有无穷多个，但是几个间隔最大的分离超平面是唯一的。（这里的间隔最大化也称为硬间隔最大，与训练数据集近似可分时软间隔最大化相对应）。间隔最大化意味着以充分大的确信度对训练数据进行分类，也就是说不仅将正负实例点进行分开，而且对离超平面较近的实例点也有足够大的确信度将它们分开，这样的超平面对未知的新实例也有较好的分类预测能力。

1. 聚类分析
2. 聚类的定义

聚类是将数据对象的集合分成相似的对象类的过程。使得同一个簇（或类）中的对象之间具有较高的相似性，而不同簇中的对象具有较高的相异性。

<https://blog.csdn.net/xzfreewind/article/details/73770327>

聚类可形式描述为：D={o1，o2，…，on}表示n个对象的集合，oi表示第i（i=1，2，…，n）个对象，Cx表示第x（x=1，2，…，k）个簇，CxÍD。用sim(oi，oj)表示对象oi与对象oj之间的相似度。若各簇Cx是刚性聚类结果，则各Cx需满足如下条件：

其中，条件①和②表示所有Cx是D的一个划分，条件③表示簇内任何对象的相似度均大于簇间任何对象的相似度。

1. 聚类的类型

分层聚类、分区聚类、排他性聚类、非排他性聚类、模糊聚类、非模糊聚类、局部聚类、完全聚类。（距离相似度度量、密度相似度度量、连通性相似度度量和概念相似度度量）

1. 簇的类型

完全区分的簇、基于中央的簇、连通性相似度度量簇、密度相似度度量簇、概念相似度度量

1. 层次聚类的两种主要类型

Agglomerative: 凝聚层次聚类

1， 将每个对象作为一个簇，这时就有7个簇。

2， 自底向上合并接近的簇，假设合并成了三个簇：AB，CDE，FG。

3， 重复第二步直到数量达到规定的簇的数量K。

Divisive: 分裂层次聚类

1， 将所有的样本置于一个簇中。

2， 逐渐细分成越来越小的簇。

3， 重复第二步直到数量达到规定的簇的数量K

1. 计算簇间相似性的单链（MIN）和全链（MAX）方法

该方法中两个簇的邻近度定义为两个不同簇中任意两点之间的最短距离。单链技术擅长处理非椭圆形的簇，但是对噪音和离群点很敏感。

两个簇之间的相似度计算公式为：

dist({m1,m2},{m3,m4})=min(dist(m1,m3),dist(m1,m4),dist(m2,m3),dist(m2,m4))

全链:

该方法中两个簇的邻近度定义为两个不同簇中任意两点之间的最长距离。单链技术擅长处理圆形的簇，但是对噪音和离群点不太敏感。

两个簇之间的相似度计算公式为：

dist({m1,m2},{m3,m4})=max(dist(m1,m3),dist(m1,m4),dist(m2,m3),dist(m2,m4))

1. K均值和k中心点算法

k-means算法执行过程，首先需要随机选择初始质心，只有第一次随机选择的初始质心才是实际待聚类点集中的点，而后续将非质心点指派到对应的质心点后，重新计算得到的质心并非是待聚类点集中的点，而且如果某些非质心点是离群点的话，导致重新计算得到的质心可能偏离整个簇，为了解决这个问题，提出了改进的k-medoids聚类算法。

k-means是每次选簇的均值作为新的中心，迭代直到簇中对象分布不再变化。其缺点是对于离群点是敏感的，因为一个具有很大极端值的对象会扭曲数据分布。那么我们可以考虑新的簇中心不选择均值而是选择簇内的某个对象，只要使总的代价降低就可以。

PAM（partitioning around medoid，围绕中心点的划分）是具有代表性的k-medoids算法。

它最初随机选择k个对象作为中心点，该算法反复的用非代表对象（非中心点）代替代表对象，试图找出更好的中心点，以改进聚类的质量。

1. DBScan算法

它将簇定义为密度相连的点的最大集合，于是能够把具有足够高密度的区域划分为簇，并可在有“噪声”的数据中发现任意形状的聚类。

1，如果一个点p的ε-邻域包含多于m个对象，则创建一个p作为核心对象的新簇；

2，寻找并合并核心对象直接密度可达的对象；

4， 没有新点可以更新簇时，算法结束。

1. 关联分析
2. 概念：项集，频繁项集，支持度，置信度，极大频繁项集，闭频繁项集

项的集合称为项集

S的支持度：sup(S)=(包含S的事务数量/D中总的事务数量)×100%

置信度confidence（X−>Y）=同时购买{X,Y}的人数/购买X的人数

项集的出项频率是包含项集的事务数，简称为项集的频率，支持度计数或计数。注意，定义项集的支持度有时称为相对支持度，而出现的频率称为绝对支持度。如果项集I的相对支持度满足预定义的最小支持度阈值，则I是频繁项集。

最大频繁项集是各频繁k项集中符合无超集条件的频繁项集。

所谓闭项集，就是指一个项集X，它的直接超集的支持度计数都不等于它本身的支持度计数。如果闭项集同时是频繁的，也就是它的支持度大于等于最小支持度阈值，那它就称为闭频繁项集。

1. Apriori算法

Apriori算法使用频繁项集的先验知识，使用一种称作逐层搜索的迭代方法，k项集用于探索(k+1)项集。首先，通过扫描事务（交易）记录，找出所有的频繁1项集，该集合记做L1，然后利用L1找频繁2项集的集合L2，L2找L3，如此下去，直到不能再找到任何频繁k项集。最后再在所有的频繁集中找出强规则，即产生用户感兴趣的关联规则。

1. FP增长算法

<https://blog.csdn.net/Bone_ACE/article/details/46669699>

首先，FP-Growth算法的任务是找出数据集中的频繁项集。

然后，FP-Growth算法的步骤，大体上可以分成两步：(1)FP-Tree的构建； (2)FP-Tree上频繁项集的挖掘。

1. 维度约减
2. 过滤方法和包裹方法有何区别及优劣

过滤方法其主要思想是：对每一维的特征“打分”，即给每一维的特征赋予权重，这样的权重就代表着该维特征的重要性，然后依据权重排序。

主要的方法有：

Chi-squared test(卡方检验)

information gain(信息增益)，详细可见“简单易学的机器学习算法——决策树之ID3算法”

correlation coefficient scores(相关系数)

Wrapper方法

其主要思想是：将子集的选择看作是一个搜索寻优问题，生成不同的组合，对组合进行评价，再与其他的组合进行比较。这样就将子集的选择看作是一个是一个优化问题，这里有很多的优化算法可以解决，尤其是一些启发式的优化算法，如GA，PSO，DE，ABC等，详见“优化算法——人工蜂群算法(ABC)”，“优化算法——粒子群算法(PSO)”。

主要方法有：recursive feature elimination algorithm(递归特征消除算法)

过滤方法vs包装方法

过滤方法：执行速度快，普适强，但倾向于选择大子集

包装方法：准确性高，有泛化能力，减少过度拟合，但执行速度慢，普适度低

1. 五种不同的特征搜索方法，基本思想及其伪代码
2. Naïve Sequential 特征搜索

算法描述： 将 M个特征逐个送入评价函数， 选择得分最高的 N个特征组成特征子集。

算法评价：简单，但没有考虑特征间的相关性，因此通常性能不好。例如，有

{1,2,3,4,5} 这五类样本，一共有 A,B,C 三个特征，现要从中选出 2 个 特征来区分这 5 类。特征 A能将其分为 1 ， 2， 3， {4 ， 5} 这 4 类，特征 B能将其分为 1， {2 ，3} ， {4 ， 5} 这 3 类，特征 C只能将其分为 {1 ， 2， 3， 4} ， {5} 这 2 类。那么显然最优特征是 A，然后是 B，最后是 C，朴素序列特征选择算法会选择特征 A和 B，但是特征 A和 B并不能区分 4 和 5 类。 其实最优选择应该是 A和 C，只有特征 A和 C能将 5 类区分开来。

2. 序列前向选择 ( SFS , Sequential Forward Selection )

算法描述：每次选择一个特征 x 加入特征子集 Y，使得特征函数 J( Y+x ) 最大。

简单说就是，每次都选择一个使得特征函数的取值达到最优的特征加入，其实就是一种简单的贪心算法。 “前向”的意思就是这个算法只能加入特征而不能去除特征。

算法评价：缺点是只能加入不能去除，例如：特征 A完全依赖于特征 B与 C，可以认为如果加入了特征 B与 C则 A就是多余的。假设序列前向选择算法首先将 A加入特征集，然后又将 B与 C加入，那么特征子集中就包含了多余的特征 A。

3. 序列后向选择 ( SBS , Sequential Backward Selection )

算法描述：首先将全部特征加入特征集合 Y，然后每次从特征集 Y中去除一个特征 x，使得 J(Y-x) 最优。“后向”的意思就是特征只能去除而不能加入。

算法评价： 序列后向选择与序列前向选择正好相反， 它的缺点是特征只能去除不能加入。

4.双向搜索 ( BDS , Bidirectional Search )

算法描述：使用序列前向选择 (SFS)与序列后向选择 (SBS)分别从两端开始搜索，

两者搜索到一个相同的特征子集 Y才停止搜索。

双向搜索的出发点是 O(2\*N^(k/2)) < O(N^k)， 如下图所示， O点代表搜索起点， A点代表搜索目标。灰色的圆代表单向搜索可能的搜索范围，绿色的 2 个圆表示某次双向搜索的搜索范围，容易证明绿色的面积必定要比灰色的要小。

为了确保序列前向选择与序列后向选择会搜索到相同的子集，需要确保：

(1) 被 SFS选中的特征 SBS就不能去除

(2) 被 SBS去除的特征 SFS就不能选择

算法评价： BDS结合了 SFS与 SBS，其时间复杂度比 SFS与 SBS小，但是兼有

SFS与 SBS的缺点。

5. 序列浮动选择 ( Sequential Floating Selection )

算法描述：序列浮动选择由增 L 去 R选择算法发展而来，该算法与增 L 去 R

选择算法的不同之处在于 L 与 R不是固定的，而是“浮动”的，也就是变化的。

序列浮动选择同样有以下两种变种。

(1) 序列浮动前向选择 ( SFFS , Sequential Floating Forward Selection )

算法描述：从空集开始，每轮在未加入的特征中选择一个集合 x，使得 J(Y+x)

达到最优，将 x 加入 Y，然后在已选择特征集中选择集合 z，使得 J(Y-z) 达到最优，然后再 Y中剔除 z。

(2) 序列浮动后向选择 ( SFBS , Sequential Floating Backward Selection )

算法描述： 与 SFFS类似， 不同之处在于 SFBS是从全集开始， 每轮先去除特征，然后加入特征。

考试题型

概念解析 3分 \* 10 = 30分

简答 5分 \* 6 = 30分

计算 8分 \* 5 = 40分

6月20日下午2:30-5:00考试，主楼B204。