机器学习必备资源 – 常见算法总结及讲解

目录

1.基础概念

(1) 10折交叉验证：英文名是10-fold cross-validation，用来测试算法的准确性。是常用的测试方法。将数据集分成10份。轮流将其中的9份作为训练数据，1分作为测试数据，进行试验。每次试验都会得出相应的正确率（或差错率）。10次的结果的正确率（或差错率）的平均值作为对算法精度的估计，一般还需要进行多次10折交叉验证，在求其平均值，对算法的准确性进行估计。

(2) 极大似然估计：极大似然估计，只是一种概率论在统计学中的应用，它是参数评估的方法之一。说的 已知某个随机样本满足某种概率分布，但是其中具体的参数不清楚，参数估计通过若干次实验，观察其结果，利用结果推出参数的大概值。极大似然估计是建立在这样的思想上的：已知某个参数能使这个样本出现的概率最大。我们当然不会再去选择其他其他小概率的样本，所以干脆就把这个参数作为估计的真实值。

(3)在信息论中，熵表示的是不确定性的量度。信息论的创始人香农在其著作《通信的数学理论》中提出了建立在概率统计模型上的信息度量。他把信息定义为”用来消除不确定性的东西“。熵的定义为信息的期望值。
ps:熵指的是体系的混乱程度，它在控制论，概率论，数论，天体物理，生命科学等领域都有重要的应用，在不同的学科中也有引申出更为具体的定义，是各个领域十分重要的参量。熵由鲁道夫.克劳修斯提出，并应用在热力学中。后来在，克劳德.埃尔伍德.香农 第一次将熵的概念引入到信息论中来。

(4) 后验概率是信息论的基本概念之一。在一个通信系统中，在收到某个消息之后，接收端所了解到的该消息发送的概率称为后验证概率。后验概率是指在得到”结果“的信息后重新修正的概率，如贝叶斯公式中的。是执果寻因的问题。后验概率和先验概率有着不可分割的联系，后验的计算要以先验概率为基础，其实说白了后验概率其实就是条件概率。

(5) PCA 主成分分析:
- 优点：降低数据的复杂性，识别最重要的多个特征。
- 缺点：不一定需要，且可能损失有用信息。
- 适用适用类型：数值型数据。
- 技术类型：降维技术。

简述：在PCA中，数据从原来的坐标系转换到了新的坐标系，新坐标系的选择是由数据本身决定的。第一个新坐标轴选择时原始数据中方差最大的方向，第二个新坐标轴的选择和第一个坐标轴正交且具有最大方差的方向。该过程一直重复，重复次数为原始数据中特征的数目。会发现大部分方差都包含在最前面的几个新坐标轴中。因此，可以忽略余下的坐标轴，即对数据进行了降维处理。除了PCA主成分分析技术，其他降维技术还有ICA(独立成分分析)，因子分析等。

(6) 将不同的分类器组合起来，而这种组合结果则被称为集成方法（ensemble method）或者元算法（meta-algorithm）。

(7) 回归算法和分类算法很像，不过回归算法和分类算法输出标称型类别值不同的是，回归方法会预测出一个连续的值，即回归会预测出具体的数据，而分类只能预测类别。

(8) SVD(singular value decomposition) 奇异值分解:
- 优点：简化数据，去除噪声，提高算法的结果。
- 缺点：数据转换可能难以理解。
- 适用数据类型：数值型数据。
- ps:SVD是矩阵分解的一种类型。

总结：SVD是一种强大的降维工具，我们可以利用SVD来逼近矩阵并从中提取重要特征。通过保留矩阵80%~90%的能量，就可以得到重要的特征并去掉噪声。SVD已经运用到多个应用中，其中一个成功的应用案例就是推荐引擎。推荐引擎将物品推荐给用户，协同过滤则是一种基于用户喜好和行为数据的推荐和实现方法。协同过滤的核心是相似度计算方法，有很多相似度计算方法都可以用于计算物品或用户之间的相似度。通过在低维空间下计算相似度，SVD提高了推荐引擎的效果。

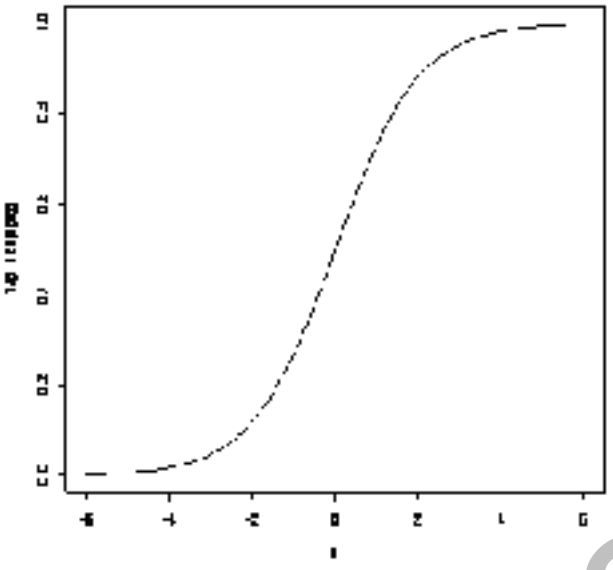
(9)共线性：是指线性回归模型中的解释变量之间由于存在精确的相关关系或高度相关关系而使模型估计失真或难以估计。

2.基本算法

2.1 Logistic回归
- 优点：计算代价不高，易于理解和实现。
- 缺点：容易欠拟合，分类精度可能不高。
- 适用数据类型：数值型和标称型数据。
- 类别：分类算法。
- 试用场景：解决二分类问题。

简述：Logistic回归算法基于Sigmoid函数，或者说Sigmoid就是逻辑回归函数。Sigmoid函数定义如下：1/（1+exp（-z））。函数值域范围(0,1)。可以用来做分类器。

Sigmoid函数的函数曲线如下：



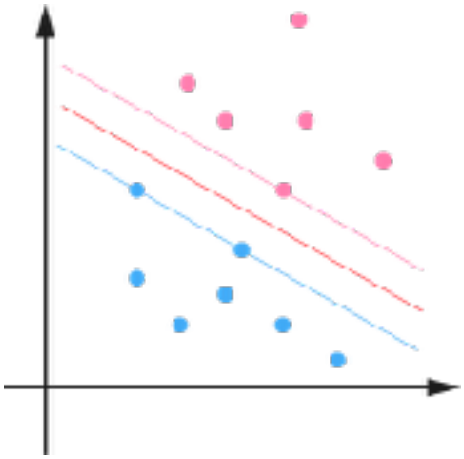
逻辑回归模型分解如下：
(1)首先将不同维度的属性值和对应的一组权重加和:
公式如下： z = w0+w1x1+w2x2+…+wm*xm。（其中x1,x2,…,xm是某样本数据的各个特征，维度为m）
ps：这里就是一个线性回归。W权重值就是需要经过训练学习到的数值，具体W向量的求解，就需要用到极大似然估计和将似然估计函数代入到 优化算法来求解。最常用的最后化算法有 梯度上升算法。
由上面可见：逻辑回归函数虽然是一个非线性的函数，但其实其去除Sigmoid映射函数之后，其他步骤都和线性回归一致。
(2)然后将上述的线性目标函数 z 代入到sigmond逻辑回归函数，可以得到值域为（0,0.5)和（0.5,1）两类值，等于0.5的怎么处理还以自己定。这样其实就得到了2类数据，也就体现了二分类的概念。
总结：Logistic回归的目的是寻找一个非线性函数Sigmoid的最佳拟合参数，参数的求解过程可以由最优化算法来完成。在最优化算法中，最常用的就是梯度上升算法，而梯度上升算法有可以简化为随机梯度上升算法。

2.2 SVM(Support Vector Machines) 支持向量机

- 优点：泛化错误率低，计算开销不大，结果易解释。
- 缺点：对参数调节和核函数的选择敏感，原始分类器不加修改仅适用于处理二分类问题。
- 适用数据类型：数值型和标称型数据。
- 类别：分类算法。
- 试用场景：解决二分类问题。

简述：通俗的讲，SVM是一种二类分类模型，其基本模型定义为特征空间上的间隔最大的线性分类器，即支持向量机的学习策略便是间隔最大化，最终可转化为一个凸二次规划问题的求解。或者简单的可以理解为就是在高维空间中寻

找一个合理的超平面将数据点分隔开来，其中涉及到非线性数据到高维的映射以达到数据线性可分的目的。
支持向量概念：



上面样本图是一个特殊的二维情况，真实情况当然可能是很多维。先从低纬度简单理解一下什么是支持向量。从图中可以看到3条线，中间那条红色的线到其他两条先的距离相等。这条红色的就是SVM在二维情况下要寻找的超平面，用于二分类数据。而支撑另外两条线上的点就是所谓的支持向量。从图中可以看到，中间的超平面和另外两条线中间是没有样本的。找到这个超平面后，利用超平面的数据数学表示来对样本数据进行二分类，就是SVM的机制了。
ps：《机器学习实战》书中有这么几个概念：

- (1)如果能找到一个直线（或多维的面）将样本点分开，那么这组数据就是线性可分的。将上述数据集分隔开来的直线(或多维的面)称为分隔超平面。分布在超平面一侧的数据属于一个类别，分布在超平面另一侧的数据属于另一个类别
- (2)支持向量（Support vector）就是分离超平面最近的那些点。
- (3)几乎所有分类问题都可以使用SVM，值得一提的是，SVM本身是一个二分类分类器，对多类问题应用SVM需要对代码做一些修改。

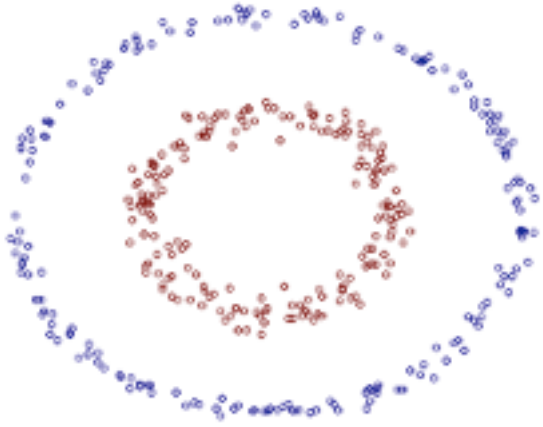
公式：
SVM有很多实现，但是本章值关注其中最流行的一种实现，及序列最小优化（Sequential Minimal Optimization，SMO）算法。
其公式如下：

$$\max_\alpha \quad W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle$$

$$\text{s.t.} \quad 0 \le \alpha_i \le C, \quad i = 1, \ldots, m$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0,$$

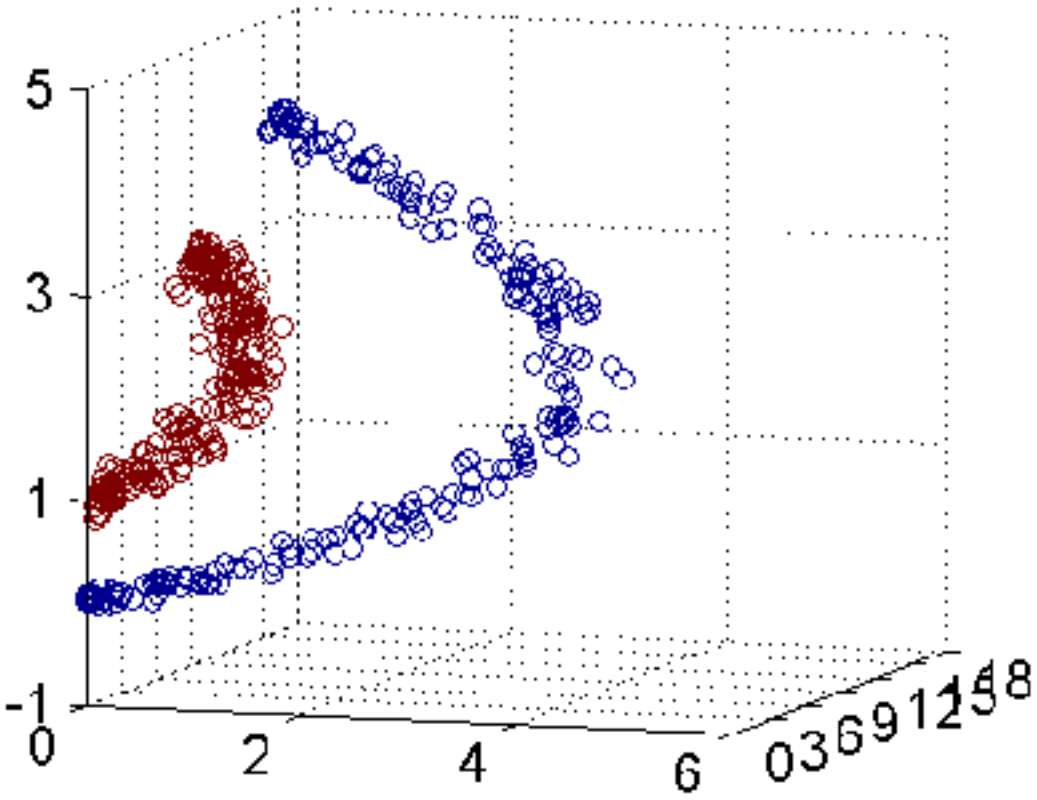
SMO算法的目标是求出一些列的alpha，一旦求出了alpha，就很容易计算出权重向量w并得到分隔超平面。
SMO算法的工作原理是：每次循环中选择两个alpha进行优化处理。一旦找到一对合适的alpha，那么就增大其中一个同时减小另一个。这里所谓的“合适”就是指两个alpha必须符合一定的条件，条件之一就是这两个alpha必须要在间隔边界之外，而其第二个条件则是这两个alpha还没有进行过区间化处理或者不在边界上。
核函数将数据从低维度映射到高维：
SVM是通过寻找超平面将数据进行分类的，但是当数据不是线性可分的时候就需要利用核函数将数据从低维映射到高维使其线性可分后，在应用SVM理论。



示例：
这个二维数据分布不是线性可分的，其方程为：

$$a_1X_1 + a_2X_1^2 + a_3X_2 - a_4X_2^2 + a_5X_1X_2 + a_6 = 0$$

但是通过核函数维度映射后，其变为：



对应的方程为：

$$\sum_{i=1}^{5} a_i Z_i + a_6 = 0$$

这样映射后的数据就变成了线性可分的，就可以应用SVM理论了。
总结：支持向量机是一种分类器。之所以成为“机”是因为他会产生一个二值决策结果，即它是一种‘决策’机。核方法或者说核技巧会将数据（有时是非线性数据）从一个低维空间映射到一个高维空间，可以将一个在低维空间中的非线性问题转换为高维空间下的线性问题来求解。

2.3 决策树

- 优点：计算复杂度不高，输出结果易于理解，对中间值的缺失不敏感，可以处理不相关特征数据。
- 缺点：可能会产生匹配过度问题。
- 适用数据类型：数值型和标称型。
- 算法类型：分类算法。
- 数据要求：树的构造只适用于标称型的数据，因此数值型数据必须离散化。

简述：在构造决策树时，我们需要解决的第一个问题就是，当前数据集上哪个特征在划分数据分类时起决定性作用。为了找到决定性特征，划分出最好的结果，我们必须评估每个特征。完成测试后，原始数据就被划分为几个数据子集。

这些数据的子集分布在第一个决策点的所有分支上，如果某个分支下的数据属于同一个类型，则无需进一步对数据集进行切割。反之则需要进一步切割。
创建分支的伪代码如下：

```
检测数据集中的每个子项是否属于同一分类：
  if so return 类标签；
  else
     寻找数据集的最好特征
     划分数据集
     创建分支结点
        for 每个划分的子集
           调用函数createBranch并增加返回结果到分支结点中
        return 分支结点
```

在可以评测哪种数据划分方式是最好的数据划分之前，我们必须学习如何计算信息增益。集合的信息度量方式称为香农熵或者简称为熵。熵在信息论中定义为信息的期望值。
信息熵的计算公式为：
H(信息熵) = -∑ P（xi） log2P（xi） ps:其中p（xi）表示选择该分类的概率。
下面简述一下生成决策树的步骤：

- (1) 根据给定的训练数据，根据熵最大原则根据每一个维度来划分数据集，找到最关键的维度。
- (2) 当某个分支下所有的数据都数据同一分类则终止划分并返回类标签，否则在此分支上重复实施(1)过程。
- (3) 依次计算就将类标签构建成了一棵抉择树。
- (4) 依靠训练数据构造了决策树之后，我们就可以将它用于实际数据的分类。
- ps:当然生成决策树的算法不止这一个，还有其他一些生成决策树的方法，比如：C4.5和CART。

总结：
决策树分类器就像带有终止块的流程图，终止块表示分类结果。开始处理数据集时，我们首先需要测量集合中数据的不一致性，也就是熵，然后寻找最优的方案划分数据集，直到数据集中的所有数据属于同一个分类。

2.4 朴素贝叶斯

- 优点：在数据较少的情况下仍然有效，可以处理多类别问题。
- 缺点：对于输入数据的准备方式较为敏感。
- 适用的数据类型：标称型数据。
- 算法类型：分类算法

简述：朴素贝叶斯是贝叶斯理论的一部分，贝叶斯决策理论的核心思想，即选择具有高概率的决策。朴素贝叶斯之所以冠以朴素开头，是因为其在贝叶斯理论的基础上做出了两点假设：(1)每个特征之间相互独立、(2)每个特征同等重要。

贝叶斯准则是构建在条件概率的基础之上的，其公式如下：
P（HIX）=P（XIH)P（H)/P(X)
ps：P（HIX）是根据X参数值判断其属于类别H的概率，称为后验概率。P（H)是直接判断某个样本属于H的概率，称为先验概率。
P(XIH)是在类别H中观测到X的概率（后验概率），P(X)是在数据库中观测到X的概率。可见贝叶斯准则是基于条件概率并且和观测到样本的先验概率和后验概率是分不开的。
总结：对于分类而言，使用概率有事要比使用硬规则更为有效。贝叶斯概率及贝叶斯准则提供了一种利用已知值来估计未知概率的有效方法。可以通过特征之间的条件独立性假设，降低对数据量的需求。尽管条件独立性的假设并不正确，但是朴素贝叶斯仍然是一种有效的分类器。

2.5 K-近邻算法（KNN）
- 优点：精度高、对异常值不敏感、无数据输入假定。
- 缺点：计算复杂度高，空间复杂度搞。
- 适用数据范围：数值型和标称型。
- 算法类型：分类算法。

简述：算法原理，存在一个样本数据集合，也称作训练样本集，并且样本集中每个数据都存在标签，即我们知道样本集中每一个数据与所属分类的对应关系。输入没有标签的新数据后，将新数据的每个特征和样本集中数据对应的特征进行比较，然后算法提取样本集中特征最相似数据（最近邻）的分类标签。一般来说，我们只选择样本数据集中前k个最相似的数据，这就是k-近邻算法中k的出处，通常k是不大于20的整数。最后选择k个最相似数据中出现次数最多的分类，作为新数据的分类。

2.6 线性回归(Linear Regression)
- 优点：结果易于理解，计算上不复杂。
- 缺点：对非线性数据拟合不好。
- 适用数据类型：数值型和标称型数据。
- 算法类型：回归算法。
- ps:回归于分类的不同，就在于其目标变量时连续数值型。

简述：在统计学中，线性回归（Linear Regression）是利用称为线性回归方程的最小平方函数对一个或多个自变量和因变量之间关系进行建模的一种回归分析。这种函数是一个或多个称为回归系数的模型参数的线性组合（自变量都是一次方）。只有一个自变量的情况称为简单回归，大于一个自变量情况的叫做多元回归。
线性方程的模型函数的向量表示形式为：

$$h_\theta(x) = \theta^T X$$

通过训练数据集寻找向量系数的最优解，即为求解模型参数。其中求解模型系数的优化器方法可以用“最小二乘法”、“梯度下降”算法，来求解损失函数：

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$\min_\theta J_\theta$$

的最优值。

附加：岭回归（ridge regression）：

岭回归是一种专用于共线性数据分析的有偏估计回归方法，实质上是一种改良的最小二乘估计法，通过放弃最小二乘法的无偏性，以损失部分信息、降低精度为代价，获得回归系数更为符合实际、更可靠的回归方法，对病态数据的耐受性远远强于最小二乘法。

岭回归分析法是从根本上消除复共线性影响的统计方法。岭回归模型通过在相关矩阵中引入一个很小的岭参数K（1>K>0），并将它加到主对角线元素上，从而降低参数的最小二乘估计中复共线特征向量的影响，减小复共线变量系数最小二乘估计的方法，以保证参数估计更接近真实情况。岭回归分析将所有的变量引入模型中，比逐步回归分析提供更多的信息。

总结：与分类一样，回归也是预测目标值的过程。回归与分类的不同点在于，前者预测连续型的变量，而后者预测离散型的变量。回归是统计学中最有力的工具之一。在回归方程里，求得特征对应的最佳回归系统的方法是最小化误差的平方和。

2.7 树回归

- 优点：可以对复杂和非线性的数据建模。
- 缺点：结果不易理解。
- 适用数据类型：数值型和标称型数据。
- 算法类型：回归算法。

简述：线性回归方法可以有效的拟合所有样本点(局部加权线性回归除外)。当数据拥有众多特征并且特征之间关系十分复杂时，构建全局模型的回归算法是比较困难的。此外，实际中很多问题为非线性的，例如常见的分段函数，不可能用全局线性模型类进行拟合。树回归将数据集切分成多份易建模的数据，然后利用线性回归进行建模和拟合。较为经典的树回归算法为CART（classification and regreesion trees 分类回归树）。

2.8 K-Means(K 均值算法)

- 优点：容易实现。
- 缺点：可能收敛到局部最小值，在大规模数据集上收敛较慢。
- 适用数据类型：数值型数据。
- 算法类型：聚类算法。

ps:K-Means和上面的分类和回归算法不同，它属于非监督学习算法。类似分类和回归中的目标变量事先并不存在。与前面“对于数据变量X能预测变量Y”不同的是，非监督学习算法要回答的问题是：“从数据X中能发现什么？“，这里需要

回答的X方面可能的问题是：”构成X的最佳6个数据簇都是哪些“或者”X中哪三个特征最频繁共现？“。
K-Means的基本步骤：
(1) 从数据对象中随机的初始化K个初始点作为质心。然后将数据集中的每个点分配到一个簇中，具体来讲每个点找到距其最近的质心，并将其分配给该质心所对应的簇。
(2) 计算每个簇中样本点的均值，然后用均值更新掉该簇的质心。然后划分簇结点。
(3) 迭代重复（2）过程，当簇对象不再发生变化时，或者误差在评测函数预估的范围时，停止迭代。
算法的时间复杂度上界为O(nkt), 其中t是迭代次数。
ps:初始的K个质心的选取以及距离计算公式的好坏，将影响到算法的整体性能。
附加：
二分K-均值算法:为克服K-均值算法收敛于局部最小值的问题，有人提出了另一个称为二分K-均值（bisecting K-Means）的算法。该算法首先将所有点作为一个簇，然后将簇一分为二。之后选择其中一个簇继续划分，选择哪个一簇进行划分取决于对其划分是否可以最大程度降低SSE(Sum of Squared Error，两个簇的总误差平方和)的值。

2.9 算法关联分析
首先了两个概念：
- 频繁项集（frequent item sets）:经常出现在一块的物品的集合。
- 关联规则（association rules）：暗示两种物品间可能存在很强的关系。
- 项集的支持度（support）：数据集中包含该项集记录所占的比例。
- 关联分析的目标包括两项：发现频繁项集合发现关联规则。首先找到频繁项集，然后才能获得关联规则。

2.10 Apriori算法：
- 优点：易编码实现。
- 缺点：在大型数据集上可能较慢。
- 适用数据类型：数值型或标称型数据。
- 原理：如果某个项集时频繁的，那么他的所有子集也是频繁的。
- Apriori运用的DEMO示例参见博客：http://blog.csdn.net/lantian0802/article/details/38331463

简述：
Apriori算法是发现频繁项集的一种方法。Apriori算法的两个输入参数分别是最小支持度和数据集。该算法首先会生成所有单个item的项集列表。然后扫描列表计算每个item的项集支持度，将低于最小支持度的item排除掉，然后将每个item两两组合，然后重新计算整合后的item列表的支持度并且和最小支持度比较。重复这一过程，直至所有项集都被去掉。

总结：
关联分析是用于发现大数据集中元素间有趣关系的一个工具集，可以采用两种方式来量化这些有趣的关系。发现元素间不同的组合是个十分耗时的任务，不可避免需要大量昂贵的计算资源，这就需要一些更智能的方法在合理的时间范围内找到频繁项集。能够实现这一目标的一个方法是Apriori算法，它使用Apriori原理来减少在数据库上进行检查的集合的数目。Apriori原理是说如果一个元素是不频繁的，那么那些包含该元素的超集也是不频繁的。Apriori算法从单元素项集开始，通过组合满足最小支持度要求的项集来形成更大的集合。支持度用来度量一个集合在原始数据中出现的频率。

2.11 FP-growth算法

简述：FP-growth也是用于发现频繁项集的算法，他以FP树的结构存储构建元素，其他Apriori算法的性能要好很多。通常性能要好2个数量级以上。其发现频繁项集的过程如下：(1)构建FP树。(2)从FP树中挖掘频繁项集。

- 优点：一般要快于Apriori。
- 缺点：实现比较困难，在某些数据集上性能会下降。
- 适用数据类型：标称型数据。

总结：FP-growth算法是一种用于发现数据集中频繁模式的有效方法。FP-growth算法利用Apriori原则，执行更快。Apriori算法产生候选项集，然后扫描数据集来检查他们是否频繁。由于只对数据集扫描两次，因此FP-growth算法执行更快。在FP-growth算法中，数据集存储在一个称为FP树的结构中。FP树构建完成后，可以通过查找元素项的条件及FP树来发现频繁项集。该过程不断以更多元素作为条件重复进行，直到FP树只包含一个元素为止。

参考文章：

1.机器学习算法基础概念学习总结

资源推荐：

1.优质在线课程 -《机器学习项目实战班》，网址：http://www.jiuzhang.com/course/19/

学习 KNN, Naive Bayes, Logistic Regression, Neural Network, Deep

Learning，Decision Tree, Bagging, Boosting 等机器学习算法；实战垃圾邮件分类、员工跳槽概率预测、商品类别判断等10余个实战项目。

2.优质公众号 - 机器学习与人工智能，公众号ID: machinelearningai
专注机器学习和人工智能，关注前沿技术和业界实践，旨在提供一线资源和消息。这里有最热门的新闻，这里有最专业的文章，这里有最具有价值的干货。