 **卡方检验**就是统计样本的实际观测值与理论推断值之间的偏离程度，实际观测值与理论推断值之间的偏离程度就决定卡方值的大小，卡方值越大，越不符合；卡方值越小，偏差越小，越趋于符合，若两个值完全相等时，卡方值就为0，表明理论值完全符合。

为什么还要使用ROC和AUC呢？因为ROC曲线有个很好的特性：**当测试集中的正负样本的分布变化的时候，ROC曲线能够保持不变。**

在实际的数据集中经常会出现类不平衡（class imbalance）现象，即负样本比正样本多很多（或者相反），而且测试数据中的 正负样本的分布也可能随着时间变化。

**准确率(accuracy)**,其定义是: 对于给定的测试数据集，分类器正确分类的样本数与总样本数之比。

是最常见的evaluation metric但在正反例不均衡的情况下尤其是我们对少数的类更感兴趣的时候，如fraud detection(欺诈监测)，癌症检测。

在测试集里，有100个sample，99个反例，只有1个正例。如果我的模型不分青红皂白对任意一个sample都预测是反例，那么我的模型的accuracy是 正确的个数／总个数 = 99/100 = 99%。你拿着这个accuracy高达99%的模型屁颠儿屁颠儿的去预测新sample了，而它一个正例都分不出来，有意思么。。。

也有人管这叫accuracy paradox。

**精确率(precision)**的公式是P = {TP}/{TP+FP} 它计算的是所有"正确被检索的item(TP)"占所有"实际被检索到的(TP+FP)"的比例

**召回率(recall)**的公式是 R = TP}/{TP+FN},它计算的是所有"正确被检索的item(TP)"占所有"应该检索到的item(TP+FN)"的比例。

**F1值:**是精确率和召回率的调和均值，即F1=2PR/(P+R)，相当于精确率和召回率的综合评价指标。

**Fα值**:为F1值的变体， (α²+1）PR/(α²P+R)，利用α给P和R赋予不同的权重，若α=1则为F1值。

也有人会用Gini系数来评价模型，其实Gini系数与AUC所表示的意义相同，只是计算方式不同。Gini系数指ROC曲线与中线（上图红线）围成的面积和中线（上图红线）之上的面积（0.5）的比例，两者之间换算公式为Gini=2\*AUC-1。

在评价模型时还会用到KS（Kolmogorov-Smirnov）值，KS=max(TPR-FPR)，即为TPR与FPR的差的最大值，KS值可以反映模型的最优区分效果，此时所取的阈值一般作为定义好坏用户的最优阈值。上图ROC曲线的KS值为0.45，此时TPR=0.79，FPR=0.34。

当然，阈值的选取还要考虑应用场景及业务要求，对于FPR不敏感而对TPR敏感的场景，可以适当减少阈值以增加TPR。

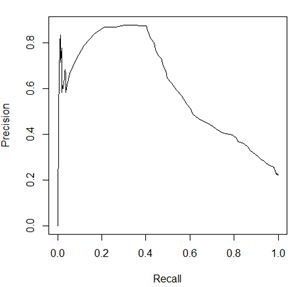
如精准营销领域的商品推荐模型，模型目的是尽量将商品推荐给感兴趣的用户，若用户对推荐的商品不感兴趣，也不会有很大损失，因此此时TPR相对FPR更重要。

再比如反欺诈领域的欺诈预测模型，由于模型结果会对识别的坏人进行一定的处置措施，FPR过高会对好人有一定干扰，造成误杀，影响客户体验，因此模型需保证在低于一定FPR的基础上尽量增加TPR。

**PRC， precision recall curve:**

PR曲线（Precision-Recall curve）和ROC曲线类似，ROC曲线是FPR和TPR的点连成的线，PR曲线是准确率和召回率的点连成的线，如下图所示。

我们又知道，Recall=TPR，因此PRC的横坐标为ROC的纵坐标。



****这个面试的时候也经常会问，让你比较某几个算法的适用条件，数据、特征量到什么规模时选用哪种算法。****

****（1）决策树****

适用条件：数据不同类边界是非线性的，并且通过不断将特征空间切分为矩阵来模拟。特征之间有一定的相关性。特征取值的数目应该差不多，因为信息增益偏向于更多数值的特征。

优点：1. 直观的决策规则；

1. 可以处理非线性特征；
2. 考虑了变量之间的相互作用。

缺点：1.容易过拟合（随机森林）；

2.处理缺失数据时的困难。

****（2）   SVM****

适用条件：特征空间大，可以处理非线性的特征。

优点：

1. 可以处理高维特征；
2. 使用和函数应对非线性特征空间；
3. 分类面不依赖所有数据；
4. 对缺失的一些数据并不敏感。

缺点：1.对于大量的预测样本，效率会很低；

2.需要找合适的核函数。

****（3）   LR****

适用条件：数据线性分布；

优点：1.模型简单，训练速度快；2.逻辑回归广泛应用与工业问题上。

缺点：1.特征空间大时性能不好；2.对于非线性特征需要转换；3.依赖于全部数据。

****（4）   三者对比：****

模型复杂度：SVM支持核函数，可处理线性非线性问题;LR模型简单，训练速度快，适合处理线性问题;决策树容易过拟合，需要进行剪枝

损失函数：SVM hinge loss; LR L2正则化; DT adaboost 指数损失

数据敏感度：SVM添加容忍度对outlier不敏感，只关心支持向量，且需要先做归一化; LR对远点敏感

数据量：数据量大就用LR，数据量小且特征少就用SVM非线性核

****（5）   神经网络****

适用条件：数据量庞大，参数之间存在内在联系。

优点：1.并行分布处理能力强；2.提取数据特征；3.逼近复杂的非线性关系。

缺点：1.需要大量参数；2.学习时间过长；3.不能观察之间的学习过程，输出结果难以解释。

算法

**概率论方法解决问题：模型（逻辑回归）+策略（损失函数）+算法（梯度下降法）**

梯度下降法：一阶导数，计算简单

随机梯度下降：针对梯度下降的全局迭代，每次迭代一条记录，因为大数据量的情况下可能不需要全局数据就可以迭代到一个稳定值，有可能很快就趋于稳定了

牛顿法：二阶导数，下降率快，求解过程矩阵分解复杂

拟牛顿法：针对牛顿法不好求解问题，使用近似矩阵代替原始矩阵，方便求解

损失函数= 误差 + 正则化部分 表名预测和真值间的差异

机器学习或者统计机器学习常见的损失函数如下：

1.0-1损失函数 （0-1 loss function）

L(Y,f(X))={1,0,Y ≠ f(X)Y = f(X)L(Y,f(X))={1,Y ≠ f(X)0,Y = f(X)

2.平方损失函数（quadratic loss function)

L(Y,f(X))=(Y−f(x))2L(Y,f(X))=(Y−f(x))2

3.绝对值损失函数(absolute loss function)

L(Y,f(x))=|Y−f(X)|L(Y,f(x))=|Y−f(X)|

4.对数损失函数（logarithmic loss function) 或对数似然损失函数(log-likehood loss function)

L(Y,P(Y|X))=−logP(Y|X)L(Y,P(Y|X))=−logP(Y|X)

1. Margin大小

使用这些目标函数的时候有时会过拟合，此时常对模型参数做一定限制，使模型更简单，称为正则化。

L1正则化：目标函数加上||w||1(参数向量w的L1范数)

L2正则化：目标函数加上1/2||w||2(参数向量w的L2范数的平方的一半)

L1正则化的优点是优化后的参数向量往往比较稀疏；L2正则化的优点是其正则化项处处可导。

**1、逻辑回归：**

**逻辑回归中，采用的则是对数损失函数**。如果损失函数越小，表示模型越好。

算法使用的是梯度下降算法。

在逻辑回归的推导中，我们**假设样本是服从伯努利分布(0-1分布)的，然后求得满足该分布的似然函数，最终求该似然函数的极大值。整体的思想就是求极大似然函数的思想。**而取对数，只是为了方便我们的在求MLE(Maximum Likelihood Estimation)过程中采取的一种数学手段而已，因为似然函数的损失函数不好求导，取对数方便计算。

**2、线性回归**

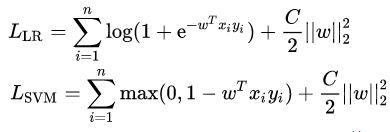
采用平房损失，即最小二乘法。

梯度下降求解

**3、SVM**

它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，**使用拉格朗日函数求解**。

分类函数f(x) = w.x + b（w.x表示w与x的内积）中的两个参数w和b，通俗理解的话w是法向量，b是截距

我们再来比较一下logistic regression和SVM的（第二种）目标函数：

可以看出，二者的唯一区别就是对于单个数据点，计算损失的方法不同。  
Logistic regression中的称为log loss + L2正则；  
而SVM中的称为合页损失hinge loss + 容错项。

多分类：svm最初为二分类设计，处理多类需合理构造多类分类器

1. 直接法：修改目标函数，将多个分类面的参数求解合并到一个最优化问题中，通过求解最优化问题一次实现多分类，但是计算复杂度高，实现困难，只适用小型问题。
2. One vs rest，某个类别归于一类，其余样本另一类，k个类别构造k个svm，分类时将未知样本分类为具最大分类函数值的那个类。 这种方法有种缺陷,因为训练集是1:M,这种情况下存在biased.因而不是很实用.
3. One vs one，任意两类样本之间设计一个svm，k个类别需要k(k-1)/2个分类器，类别多时，代价很高。

**4、决策树**

是一个树状结构，可以是二叉树或非二叉树，树由节点和有向边组成，节点有两种类型：内部节点和叶子节点，内部结点表示一个特征或属性，叶子节点表示一个类

# **决策树**

## **一、  决策树优点**

1、决策树易于理解和解释，可以可视化分析，容易提取出规则。

2、可以同时处理标称型和数值型数据。

3、测试数据集时，运行速度比较快。

4、决策树可以很好的扩展到大型数据库中，同时它的大小独立于数据库大小。

## **二、决策树缺点**

1、对缺失数据处理比较困难。

2、容易出现过拟合问题。

3、忽略数据集中属性的相互关联。

4、ID3算法计算信息增益时结果偏向数值比较多的特征。

## **三、改进措施**

1、对决策树进行剪枝。可以采用交叉验证法和加入正则化的方法。

2、使用基于决策树的combination算法，如bagging算法，randomforest算法，可以解决过拟合的问题

## **三、应用领域**

企业管理实践，企业投资决策，由于决策树很好的分析能力，在决策过程应用较多。

# **KNN算法**

## **一、KNN算法的优点**

1、KNN是一种在线技术，新数据可以直接加入数据集而不必进行重新训练

2、KNN理论简单，容易实现

## **二、KNN算法的缺点**

1、对于样本容量大的数据集计算量比较大。

2、样本不平衡时，预测偏差比较大。如：某一类的样本比较少，而其它类样本比较多。

3、KNN每一次分类都会重新进行一次全局运算。

4、k值大小的选择。

## **三、KNN算法应用领域**

文本分类、模式识别、聚类分析，多分类领域

# **支持向量机（SVM）**

## **一、  SVM优点**

1、解决小样本下机器学习问题。

2、解决非线性问题。

3、无局部极小值问题。（相对于神经网络等算法）

4、可以很好的处理高维数据集。

5、泛化能力比较强。

## **二、SVM缺点**

1、对于核函数的高维映射解释力不强，尤其是径向基函数。

2、对缺失数据敏感。

## **三、SVM应用领域**

文本分类、图像识别、主要二分类领域

# **AdaBoost算法**

## **一、  AdaBoost算法优点**

1、很好的利用了弱分类器进行级联。

2、可以将不同的分类算法作为弱分类器。

3、AdaBoost具有很高的精度。

4、相对于bagging算法和Random Forest算法，AdaBoost充分考虑的每个分类器的权重。

## **二、Adaboost算法缺点**

1、AdaBoost迭代次数也就是弱分类器数目不太好设定，可以使用交叉验证来进行确定。

2、数据不平衡导致分类精度下降。

3、训练比较耗时，每次重新选择当前分类器最好切分点。

## **三、AdaBoost应用领域**

模式识别、计算机视觉领域，用于二分类和多分类场景

# **朴素贝叶斯算法**

## **一、  朴素贝叶斯算法优点**

1、对大数量训练和查询时具有较高的速度。即使使用超大规模的训练集，针对每个项目通常也只会有相对较少的特征数，并且对项目的训练和分类也仅仅是特征概率的数学运算而已。

2、支持增量式运算。即可以实时的对新增的样本进行训练。

3、朴素贝叶斯对结果解释容易理解。

## **二、朴素贝叶斯缺点**

1、由于使用了样本属性独立性的假设，所以如果样本属性有关联时其效果不好。

## **三、朴素贝叶斯应用领域**

文本分类、欺诈检测中使用较多

# **Logistic回归算法**

## **一、logistic回归优点**

1、计算代价不高，易于理解和实现

## **二、logistic回归缺点**

1、容易产生欠拟合。

2、分类精度不高。

## **三、logistic回归应用领域**

用于二分类领域，可以得出概率值，适用于根据分类概率排名的领域，如搜索排名等。

Logistic回归的扩展softmax可以应用于多分类领域，如手写字识别等。

# **人工神经网络**

## **一、  神经网络优点**

1、分类准确度高，学习能力极强。

2、对噪声数据鲁棒性和容错性较强。

3、有联想能力，能逼近任意非线性关系。

## **二、神经网络缺点**

1、神经网络参数较多，权值和阈值。

2、黑盒过程，不能观察中间结果。

3、学习过程比较长，有可能陷入局部极小值。

## **三、人工神经网络应用领域**

目前深度神经网络已经应用与计算机视觉，自然语言处理，语音识别等领域并取得很好的效果。

机器学习方法非常多，也很成熟。下面我挑几个说。

1、首先是SVM。因为我做的文本处理比较多，所以比较熟悉SVM。SVM也叫支持向量机，其把数据映射到多维空间中以点的形式存在，然后找到能够分类的最优超平面，最后根据这个平面来分类。SVM能对训练集之外的数据做很好的预测、泛化错误率低、计算开销小、结果易解释，但其对参数调节和核函数的参数过于敏感。个人感觉SVM是二分类的最好的方法，但也仅限于二分类。如果要使用SVM进行多分类，也是在向量空间中实现多次二分类。  
SVM有一个核心函数SMO，也就是序列最小最优化算法。SMO基本是最快的二次规划优化算法，其核心就是找到最优参数α，计算超平面后进行分类。SMO方法可以将大优化问题分解为多个小优化问题求解，大大简化求解过程。某些条件下，把原始的约束问题通过拉格朗日函数转化为无约束问题，如果原始问题求解棘手，在满足KKT的条件下用求解对偶问题来代替求解原始问题，使得问题求解更加容易。 SVM还有一个重要函数是核函数。核函数的主要作用是将数据从低位空间映射到高维空间。详细的内容我就不说了，因为内容实在太多了。总之，核函数可以很好的解决数据的非线性问题，而无需考虑映射过程。

2、第二个是KNN。KNN将测试集的数据特征与训练集的数据进行特征比较，然后算法提取样本集中特征最近邻数据的分类标签，即KNN算法采用测量不同特征值之间的距离的方法进行分类。KNN的思路很简单，就是计算测试数据与类别中心的距离。KNN具有精度高、对异常值不敏感、无数据输入假定、简单有效的特点，但其缺点也很明显，计算复杂度太高。要分类一个数据，却要计算所有数据，这在大数据的环境下是很可怕的事情。而且，当类别存在范围重叠时，KNN分类的精度也不太高。所以，KNN比较适合小量数据且精度要求不高的数据。  
KNN有两个影响分类结果较大的函数，一个是数据归一化，一个是距离计算。如果数据不进行归一化，当多个特征的值域差别很大的时候，最终结果就会受到较大影响；第二个是距离计算。这应该算是KNN的核心了。目前用的最多的距离计算公式是欧几里得距离，也就是我们常用的向量距离计算方法。  
个人感觉，KNN最大的作用是可以随时间序列计算，即样本不能一次性获取只能随着时间一个一个得到的时候，KNN能发挥它的价值。至于其他的特点，它能做的，很多方法都能做；其他能做的它却做不了。

3、第三个就是Naive Bayes了。Naive Bayes简称NB（牛X），为啥它牛X呢，因为它是基于Bayes概率的一种分类方法。贝叶斯方法可以追溯到几百年前，具有深厚的概率学基础，可信度非常高。Naive Baye中文名叫朴素贝叶斯，为啥叫“朴素”呢？因为其基于一个给定假设：给定目标值时属性之间相互条件独立。比如我说“我喜欢你”，该假设就会假定“我”、“喜欢”、“你”三者之间毫无关联。仔细想想，这几乎是不可能的。马克思告诉我们：事物之间是有联系的。同一个事物的属性之间就更有联系了。所以，单纯的使用NB算法效率并不高，大都是对该方法进行了一定的改进，以便适应数据的需求。  
NB算法在文本分类中用的非常多，因为文本类别主要取决于关键词，基于词频的文本分类正中NB的下怀。但由于前面提到的假设，该方法对中文的分类效果不好，因为中文顾左右而言他的情况太多，但对直来直去的老美的语言，效果良好。至于核心算法嘛，主要思想全在贝叶斯里面了，没啥可说的。

4、第四个是回归。回归有很多，Logistic回归啊、岭回归啊什么的，根据不同的需求可以分出很多种。这里我主要说说Logistic回归。为啥呢？因为Logistic回归主要是用来分类的，而非预测。回归就是将一些数据点用一条直线对这些点进行拟合。而Logistic回归是指根据现有数据对分类边界线建立回归公式，以此进行分类。该方法计算代价不高，易于理解和实现，而且大部分时间用于训练，训练完成后分类很快；但它容易欠拟合，分类精度也不高。主要原因就是Logistic主要是线性拟合，但现实中很多事物都不满足线性的。即便有二次拟合、三次拟合等曲线拟合，也只能满足小部分数据，而无法适应绝大多数数据，所以回归方法本身就具有局限性。但为什么还要在这里提出来呢？因为回归方法虽然大多数都不合适，但一旦合适，效果就非常好。  
 Logistic回归其实是基于一种曲线的，“线”这种连续的表示方法有一个很大的问题，就是在表示跳变数据时会产生“阶跃”的现象，说白了就是很难表示数据的突然转折。所以用Logistic回归必须使用一个称为“海维塞德阶跃函数”的Sigmoid函数来表示跳变。通过Sigmoid就可以得到分类的结果。  
 为了优化Logistic回归参数，需要使用一种“梯度上升法”的优化方法。该方法的核心是，只要沿着函数的梯度方向搜寻，就可以找到函数的最佳参数。但该方法在每次更新回归系数时都需要遍历整个数据集，对于大数据效果还不理想。所以还需要一个“随机梯度上升算法”对其进行改进。该方法一次仅用一个样本点来更新回归系数，所以效率要高得多。

5、第五个是决策树。据我了解，决策树是最简单，也是曾经最常用的分类方法了。决策树基于树理论实现数据分类，个人感觉就是数据结构中的B+树。决策树是一个预测模型，他代表的是对象属性与对象值之间的一种映射关系。决策树计算复杂度不高、输出结果易于理解、对中间值缺失不敏感、可以处理不相关特征数据。其比KNN好的是可以了解数据的内在含义。但其缺点是容易产生过度匹配的问题，且构建很耗时。决策树还有一个问题就是，如果不绘制树结构，分类细节很难明白。所以，生成决策树，然后再绘制决策树，最后再分类，才能更好的了解数据的分类过程。  
决策树的核心树的分裂。到底该选择什么来决定树的分叉是决策树构建的基础。最好的方法是利用信息熵实现。熵这个概念很头疼，很容易让人迷糊，简单来说就是信息的复杂程度。信息越多，熵越高。所以决策树的核心是通过计算信息熵划分数据集。

我还得说一个比较特殊的分类方法：AdaBoost。AdaBoost是boosting算法的代表分类器。boosting基于元算法（集成算法）。即考虑其他方法的结果作为参考意见，也就是对其他算法进行组合的一种方式。说白了，就是在一个数据集上的随机数据使用一个分类训练多次，每次对分类正确的数据赋权值较小，同时增大分类错误的数据的权重，如此反复迭代，直到达到所需的要求。AdaBoost泛化错误率低、易编码、可以应用在大部分分类器上、无参数调整，但对离群点敏感。该方法其实并不是一个独立的方法，而是必须基于元方法进行效率提升。个人认为，所谓的“AdaBoost是最好的分类方法”这句话是错误的，应该是“AdaBoost是比较好的优化方法”才对。

****简述机器学习十大算法的每个算法的核心思想、工作原理、适用情况及优缺点等。****

****1）C4.5算法：****

ID3算法是以[信息论](http://baike.baidu.com/view/15076.htm" \t "https://blog.csdn.net/mousever/article/details/_blank)为基础，以[信息熵](http://baike.baidu.com/view/401605.htm" \t "https://blog.csdn.net/mousever/article/details/_blank)和信息增益度为衡量标准，从而实现对数据的归纳分类。ID3算法计算每个属性的信息增益，并选取具有最高增益的属性作为给定的测试属性。

C4.5算法核心思想是ID3算法，是ID3算法的改进，改进方面有：

1）用信息增益率来选择属性，克服了用信息增益选择属性时偏向选择取值多的属性的不足；

2）在树构造过程中进行剪枝

3）能处理非离散的数据

4）能处理不完整的数据

 C4.5算法优点：产生的分类规则易于理解，准确率较高。

缺点：

1)在构造树的过程中，需要对数据集进行多次的顺序扫描和排序，因而导致算法的低效。

2)C4.5只适合于能够驻留于内存的数据集，当训练集大得无法在内存容纳时程序无法运行。

****2）K means 算法****：

是一个简单的聚类算法，把n的对象根据他们的属性分为k个分割，k< n。 算法的核心就是要优化失真函数J,使其收敛到局部最小值但不是全局最小值。

其中N为样本数，K是簇数，rnk b表示n属于第k个簇，uk 是第k个中心点的值。然后求出最优的uk

优点：算法速度很快

缺点是，分组的数目k是一个输入参数，不合适的k可能返回较差的结果。

****3）朴素贝叶斯算法：****

朴素贝叶斯法是基于贝叶斯定理与特征条件独立假设的分类方法。算法的基础是概率问题，分类原理是通过某对象的先验概率，利用贝叶斯公式计算出其后验概率，即该对象属于某一类的概率，选择具有最大后验概率的类作为该对象所属的类。朴素贝叶斯假设是约束性很强的假设，假设特征条件独立，但朴素贝叶斯算法简单，快速，具有较小的出错率。

在朴素贝叶斯的应用中，主要研究了电子邮件过滤以及文本分类研究。

****4)K最近邻分类算法（KNN）****

分类思想比较简单，从训练样本中找出K个与其最相近的样本，然后看这k个样本中哪个类别的样本多，则待判定的值（或说抽样）就属于这个类别。

缺点：

1）K值需要预先设定，而不能自适应

2）当样本不平衡时，如一个类的样本容量很大，而其他类样本容量很小时，有可能导致当输入一个新样本时，该样本的K个邻居中大容量类的样本占多数。

该算法适用于对样本容量比较大的类域进行自动分类。

****5)EM最大期望算法****

EM算法是基于模型的聚类方法，是在概率模型中寻找参数最大似然估计的算法，其中概率模型依赖于无法观测的隐藏变量。E步估计隐含变量，M步估计其他参数，交替将极值推向最大。

EM算法比K-means算法计算复杂，收敛也较慢，不适于大规模数据集和高维数据，但比K-means算法计算结果稳定、准确。EM经常用在机器学习和计算机视觉的数据集聚（Data Clustering）领域。

****6）PageRank算法****

是google的页面排序算法，是基于从许多优质的网页链接过来的网页，必定还是优质网页的回归关系，来判定所有网页的重要性。（也就是说，一个人有着越多牛X朋友的人，他是牛X的概率就越大。）

优点：

完全独立于查询，只依赖于网页链接结构，可以离线计算。

缺点：

1）PageRank算法忽略了网页搜索的时效性。

2）旧网页排序很高，存在时间长，积累了大量的in-links，拥有最新资讯的新网页排名却很低，因为它们几乎没有in-links。

****7)AdaBoost****

Adaboost是一种迭代算法，其核心思想是针对同一个训练集训练不同的分类器(弱分类器)，然后把这些弱分类器集合起来，构成一个更强的最终分类器(强分类器)。其算法本身是通过改变数据分布来实现的，它根据每次训练集之中每个样本的分类是否正确，以及上次的总体分类的准确率，来确定每个样本的权值。将修改过权值的新数据集送给下层分类器进行训练，最后将每次训练得到的分类器最后融合起来，作为最后的决策分类器。

整个过程如下所示：  
1. 先通过对N个训练样本的学习得到第一个弱分类器；  
2. 将分错的样本和其他的新数据一起构成一个新的N个的训练样本，通过对这个样本的学习得到第二个弱分类器；  
3. 将和都分错了的样本加上其他的新样本构成另一个新的N个的训练样本，通过对这个样本的学习得到第三个弱分类器；  
4. 如此反复，最终得到经过提升的强分类器。

目前AdaBoost算法广泛的应用于人脸检测、目标识别等领域。

****8）Apriori算法****

Apriori算法是一种挖掘关联规则的算法，用于挖掘其内含的、未知的却又实际存在的数据关系，其核心是基于两阶段频集思想的递推算法 。

Apriori算法分为两个阶段：

1）寻找频繁项集

2）由频繁项集找关联规则

算法缺点：

1. 在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，都对数据库中    的全部记录进行了一遍扫描比较，需要很大的I/O负载。

****9）SVM支持向量机****

支持向量机是一种基于分类边界的方法。其基本原理是（以二维数据为例）：如果训练数据分布在二维平面上的点，它们按照其分类聚集在不同的区域。基于分类边界的分类算法的目标是，通过训练，找到这些分类之间的边界（直线的――称为线性划分，曲线的――称为非线性划分）。对于多维数据（如N维），可以将它们视为N维空间中的点，而分类边界就是N维空间中的面，称为超面（超面比N维空间少一维）。线性分类器使用超平面类型的边界，非线性分类器使用超曲面。

支持向量机的原理是将低维空间的点映射到高维空间，使它们成为线性可分，再使用线性划分的原理来判断分类边界。在高维空间中是一种线性划分，而在原有的数据空间中，是一种非线性划分。

SVM在解决小样本、非线性及高维模式识别问题中表现出许多特有的优势，并能够推广应用到函数拟合等其他机器学习问题中。

****10）CART分类与回归树****

是一种决策树分类方法，采用基于最小距离的基尼指数估计函数，用来决定由该子数据集生成的决策树的拓展形。如果目标变量是标称的，称为分类树；如果目标变量是连续的，称为回归树。分类树是使用树结构算法将数据分成离散类的方法。

优点

1）非常灵活，可以允许有部分错分成本，还可指定先验概率分布，可使用自动的成本复杂性剪枝来得到归纳性更强的树。

2）在面对诸如存在缺失值、变量数多等问题时CART 显得非常稳健。