# 数据挖掘/数据研发面经

loc是根据dataframe的具体标签选取列，而iloc是根据标签所在的位置，从0开始计数。

作者：dimaiyou  
链接：<https://www.nowcoder.com/discuss/8328>  
来源：牛客网

1. **数据库的ACID？常用操作？**

[**http://blog.csdn.net/shuaihj/article/details/14163713**](http://blog.csdn.net/shuaihj/article/details/14163713)

事务的ACID属性

原子性（Atomicity）

原子性是指事务是一个不可分割的工作单位，事务中的操作要么都发生，要么都不发生。

一致性（Consistency）

事务必须使数据库从一个一致性状态变换到另外一个一致性状态。 一致性是指在事务开始之前和事务结束以后，数据库的完整性约束没有被破坏。这是说数据库事务不能破坏关系数据的完整性以及业务逻辑上的一致性

隔离性（Isolation）

事务的隔离性是指一个事务的执行不能被其他事务干扰，即一个事务内部的操作及使用的数据对并发的其他事务是隔离的，并发执行的各个事务之间不能互相干扰。

持久性（Durability）

持久性是指一个事务一旦被提交，它对数据库中数据的改变就是永久性的，接下来的其他操作和数据库故障不应该对其有任何影响

1. **操作系统中的死锁？**

**死锁 (deallocks)： 是指两个或两个以上的进程（线程）在执行过程中，因争夺互斥资源而造成的一种互相等待的现象。**

[**https://www.cnblogs.com/changyaohua/p/4642526.html**](https://www.cnblogs.com/changyaohua/p/4642526.html)

1. 继承和多态？

4.并发编程？

**5.SVM有哪些核函数？如何选择？**

[**http://blog.csdn.net/leonis\_v/article/details/50688766**](http://blog.csdn.net/leonis_v/article/details/50688766)

**SVM核函数通常有四种：**

**1. Linear**

**2. Polynomial**

**3. Gaussian (RBF)**

**4. Sigmoid/Logistic**

**1） Linear核：主要用于线性可分的情形。参数少，速度快，对于一般数据，分类效果已经很理想了。**

**2）RBF核：主要用于线性不可分的情形。参数多，分类结果非常依赖于参数。有很多人是通过训练数据的交叉验证来寻找合适的参数，不过这个过程比较耗时。我个人的体会是：使用libsvm，默认参数，RBF核比Linear核效果稍差。通过进行大量参数的尝试，一般能找到比linear核更好的效果。至于到底该采用哪种核，要根据具体问题，有的数据是线性可分的，有的不可分，需要多尝试不同核不同参数。如果特征的提取的好，包含的信息量足够大，很多问题都是线性可分的。当然，如果有足够的时间去寻找RBF核参数，应该能达到更好的效果。**

**我一般是用控制变量法，先调整一个参数，固定其它所有参数，挑选某值使结果相对最优。再固定该参数，改变其它某个参数，观察结果选择最优参数。如果训练测试一次花费的时间是T，那么估计要花费10T或者20T的时间来选择参数。**

**怎样判断线性可分还是线性不可分呢？简单地说，就是能找到到一个线性函数能将样本完全分开，找不到就是线性不可分**

**作者：Jason Gu**

**链接：https://www.zhihu.com/question/21883548/answer/19693213**

**来源：知乎**

**一般用线性核和高斯核，也就是Linear核与RBF核，需要注意的是需要对数据归一化处理，很多使用者忘了这个小细节，然后一般情况下RBF效果是不会差于Linear，但是时间上RBF会耗费更多，其他同学也解释过了。**

**下面是吴恩达的见解：**

**1. 如果Feature的数量很大，跟样本数量差不多，这时候选用LR或者是Linear Kernel的SVM**

**2. 如果Feature的数量比较小，样本数量一般，不算大也不算小，选用SVM+Gaussian Kernel**

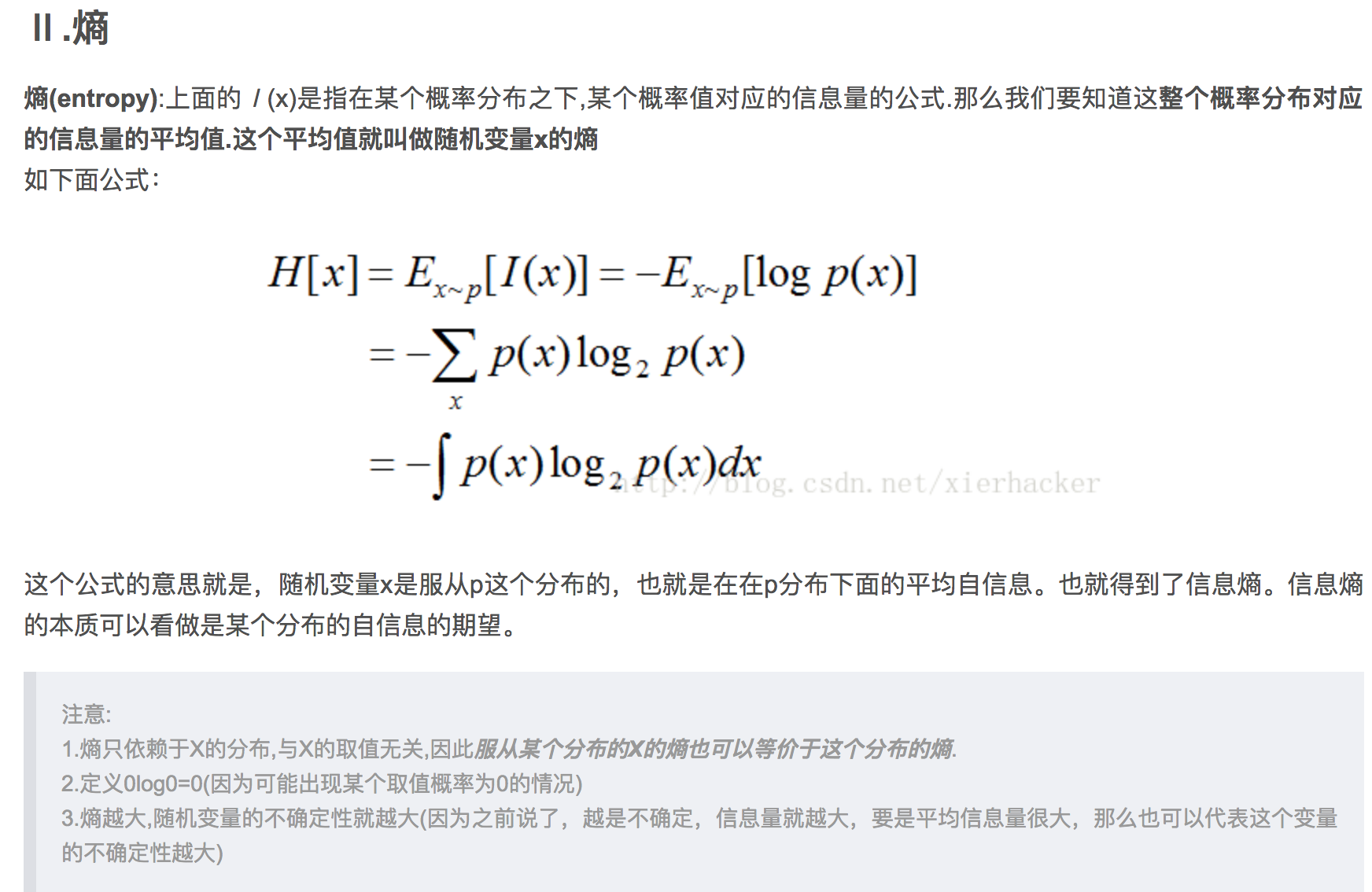
**3. 如果Feature的数量比较小，而样本数量很多，需要手工添加一些feature变成第一种情况**

**1）如果特征维数很高，往往线性可分（SVM解决非线性分类问题的思路就是将样本映射到更高维的特征空间中），可以采用LR或者线性核的SVM；（2）如果样本数量很多，由于求解最优化问题的时候，目标函数涉及两两样本计算内积，使用高斯核明显计算量会大于线性核，所以手动添加一些特征，使得线性可分，然后可以用LR或者线性核的SVM；（3）如果不满足上述两点，即特征维数少，样本数量正常，可以使用高斯核的SVM。**

6.决策树算法？什么是熵？

决策树的建立:

基本思想是以信息熵为度量构造一棵熵值下降最快的树，到叶子节点处的熵值为零, 需要遍历所有特征，选择信息增益最大的特征作为当前的分裂特征，一个特征的信息增益越大，表明属性对样本的熵减少的能力更强，这个属性使得数据由不确定性变成确定性的能力越强。



7．随机森林算法？优缺点？

一种Bagging集成算法，

某竞赛，如果让你继续优化，你打算怎么做？开放题

对大数据怎么理解？

对数据研发岗位的理解？

职业规划？

数据挖掘的应用的了解？

优缺点？

给你一项业务，你将如果开展工作？开放题

<http://blog.echen.me/2011/04/27/choosing-a-machine-learning-classifier/>

实际上在面试过程中，懂这些算法的基本思想和大概流程是远远不够的，那些面试官往往问的都是一些公司内部业务中的课题，往往要求你不仅要懂得这些算法的理论过程，而且要非常熟悉怎样使用它，什么场合用它，算法的优缺点，以及调参经验等等。说白了，就是既要会点理论，也要会点应用，既要有点深度，也要有点广度，否则运气不好的话很容易就被刷掉，因为每个面试官爱好不同。

# 如何寻找机器学习分类器

## 训练集多大

训练集很小：高偏差/低方差（如朴素贝叶斯）优于低偏差/高方差（如KNN），后者会过拟合，但是随着训练集增大，后者会更显优势。

可以把这个当做是生成模型和判别模型的区别。

## 一些特殊算法的优点

朴素贝叶斯：简单，如果朴素贝叶斯的条件独立假设成立，朴素贝叶斯分类器会收敛得比判别模型比如逻辑回归更快，所以你需要更少的训练数据。即使朴素贝叶斯假设条件没成立，在实践中这个分类器通常也表现得很好

# 机器学习算法总结

## 监督学习：

### 决策树：

决策树的优点：

　　⑴模型易于理解、结果易于解释;树结构能够可视化。

　　⑵只需少量的数据准备。别的方法经常要求数据标准化，创建虚拟变量，移除空白值。

　　⑶训练决策树的成本随着树的高度呈指数增加。

　　⑷能够处理数值和分类数据，其它方法通常只擅长处理只包含一种类型数据的数据集。

　　⑸能够处理多输出问题。

　　⑹是一个白盒子模型，易于理解和解释，不像人工神经网络模型(黑盒子)，难以解释。

　　⑺可以使用统计测试验证模型，增加模型的可靠性。

⑻即使模型采取的假设与真实数据有些违背，其最终性能也比较好。

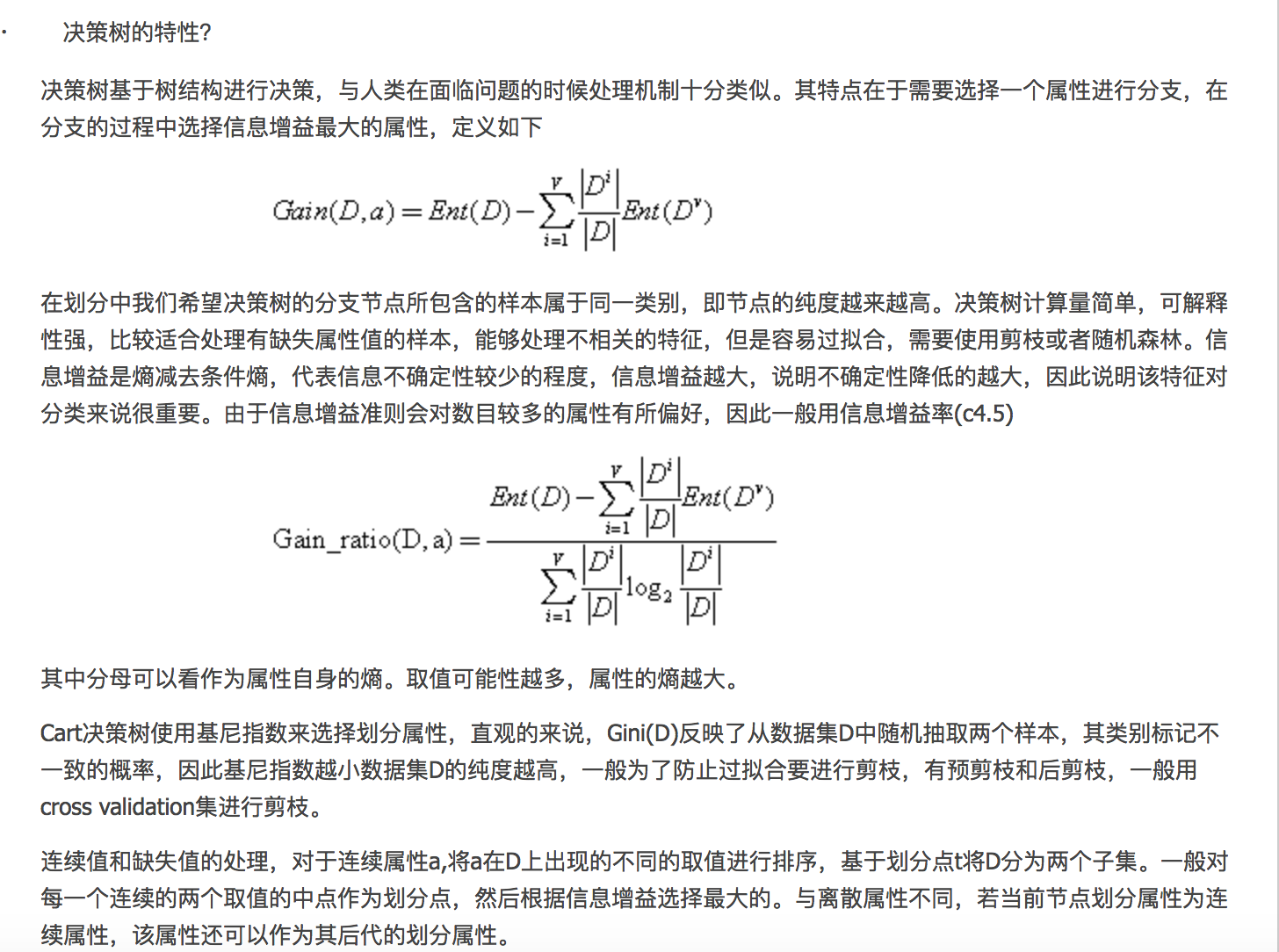
决策树的缺点：

　　⑴不宜创建过度复杂的树，以免造成过拟合。需要诸如修剪(当前不支持)，设置叶节点所需的最小样本数或设置树的最大深度的机制来避免这个问题。

　　⑵决策树可能不稳定，小数据的变化可能导致生成完全不同的树。这个问题可以通过在集成模型中使用决策树来缓解。

　　⑶有些概念很难学习，因为决策树不能轻易表达它们。例如XOR，奇偶校验或多路复用器问题。

　　⑷学习一棵最优决策树相当难的，即使对于相当简单概念。因此，在实际的决策树学习算法中，往往是基于启发式算法(如贪婪算法)来实现。但是贪婪算法并不能得到全局最优结果。这可以通过在集成模型中使用多棵树的方法来缓解面临的问题。



1. ID3
2. C4.5

C4.5算法是机器学习算法中的一种分类决策树算法,其核心算法是ID3算法.  C4.5算法继承了ID3算法的优点，并在以下几方面对ID3算法进行了改进：

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) 能够完成对连续属性的离散化处理；  
 4) 能够对不完整数据进行处理。

C4.5算法有如下优点：产生的分类规则易于理解，准确率较高。

其缺点是：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

1. CART

***决策树的优点：***

计算量简单，可解释性强，比较适合处理有缺失属性值的样本，能够处理不相关的特征；

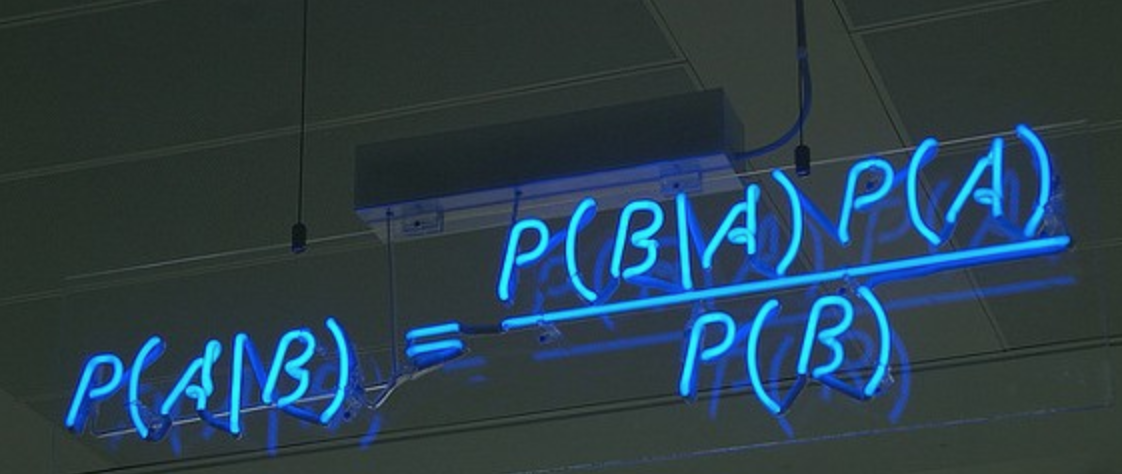
***缺点：***

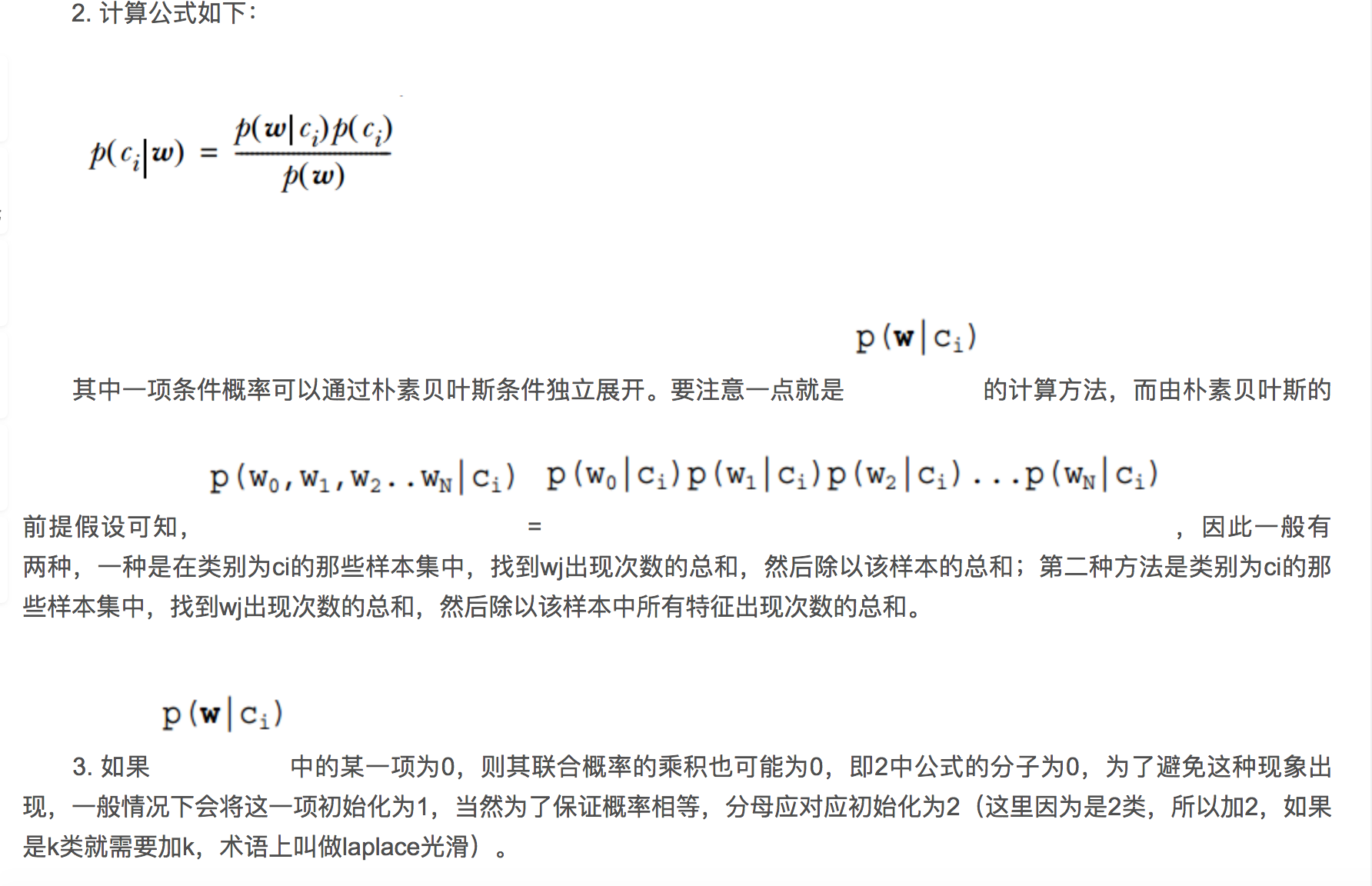
容易过拟合（后续出现了随机森林，减小了过拟合现象）；

### 2.朴素贝叶斯：

**定义：**

朴素贝叶斯分类器是一类基于贝叶斯理论的简单的概率分类器，它假设特征之间是相互独立的。下图所示的就是公式 —— P(A|B)表示后验概率，P(B|A)是似然值，P(A)是类别的先验概率，P(B)代表预测器的先验概率。





***朴素贝叶斯的优点：***

对小规模的数据表现很好，适合多分类任务，适合增量式训练。

缺点**：**

对输入数据的表达形式很敏感。？？？

**使用场景：**

现实场景中的一些例子包括：

检测垃圾电子邮件

将新闻分为科技、政治、体育等类别

判断一段文字表达积极的情绪还是消极的情绪

用于人脸检测软件

### 最小平方回归

**定义：**

最小平方回归是求线性回归的一种方法。你可以把线性回归想成是用一条直线拟合若干个点。拟合的方法有许多种，“最小平方”的策略相当于你画一条直线，然后计算每个点到直线的垂直距离，最后把各个距离求和；最佳拟合的直线就是距离和最小的那一条。线性指的是用于拟合数据的模型，而最小平方指的是待优化的损失函数。

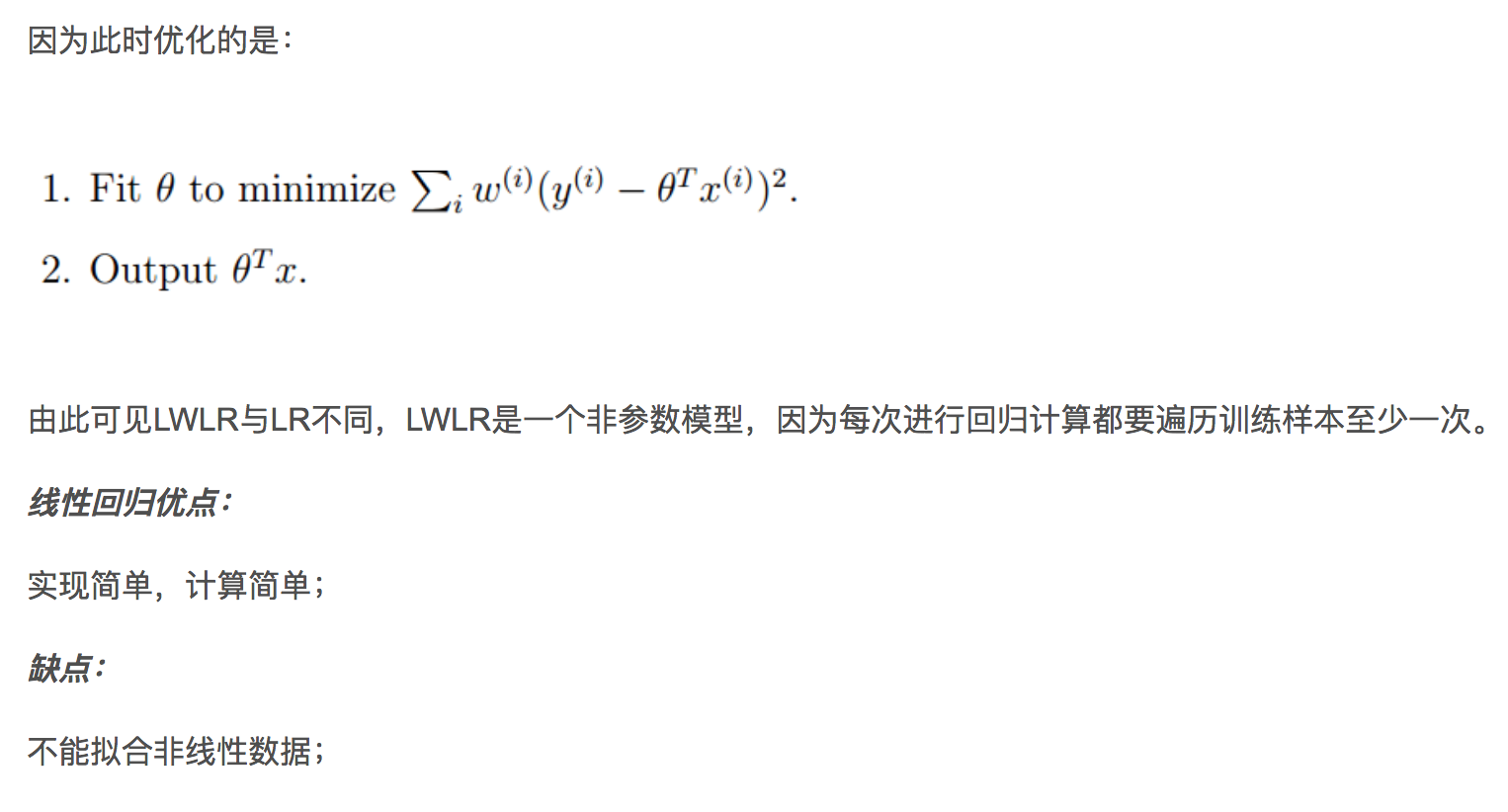
线性回归

其基本思想是用梯度下降法对最小二乘法形式的误差函数进行优化，当然也可以用normal equation直接求得参数的解，结果为：

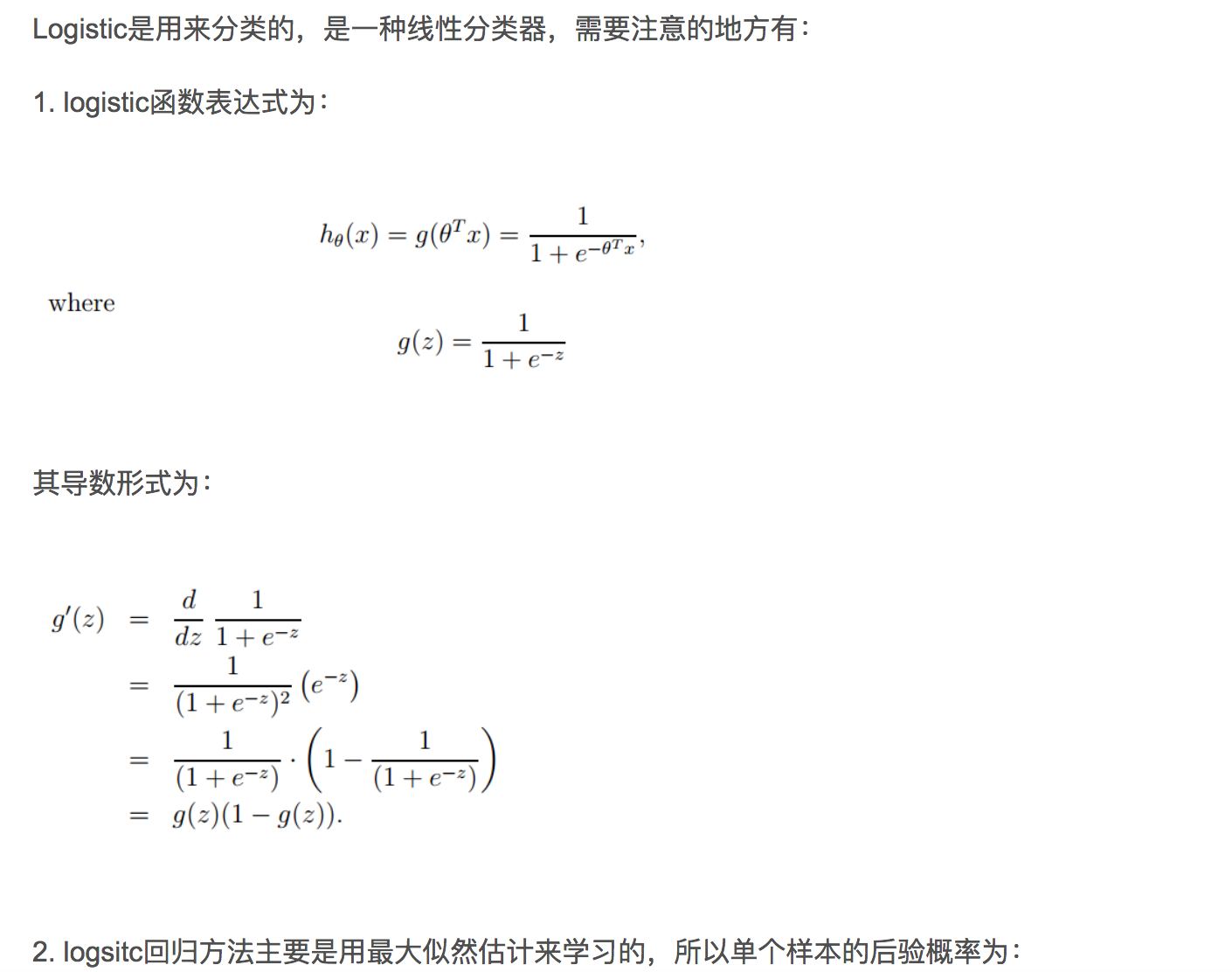
http://images.cnitblog.com/blog/381513/201310/29224913-b105579ae6e84025aed2007d6710bb0e.png

而在LWLR（局部加权线性回归）中，参数的计算表达式为:

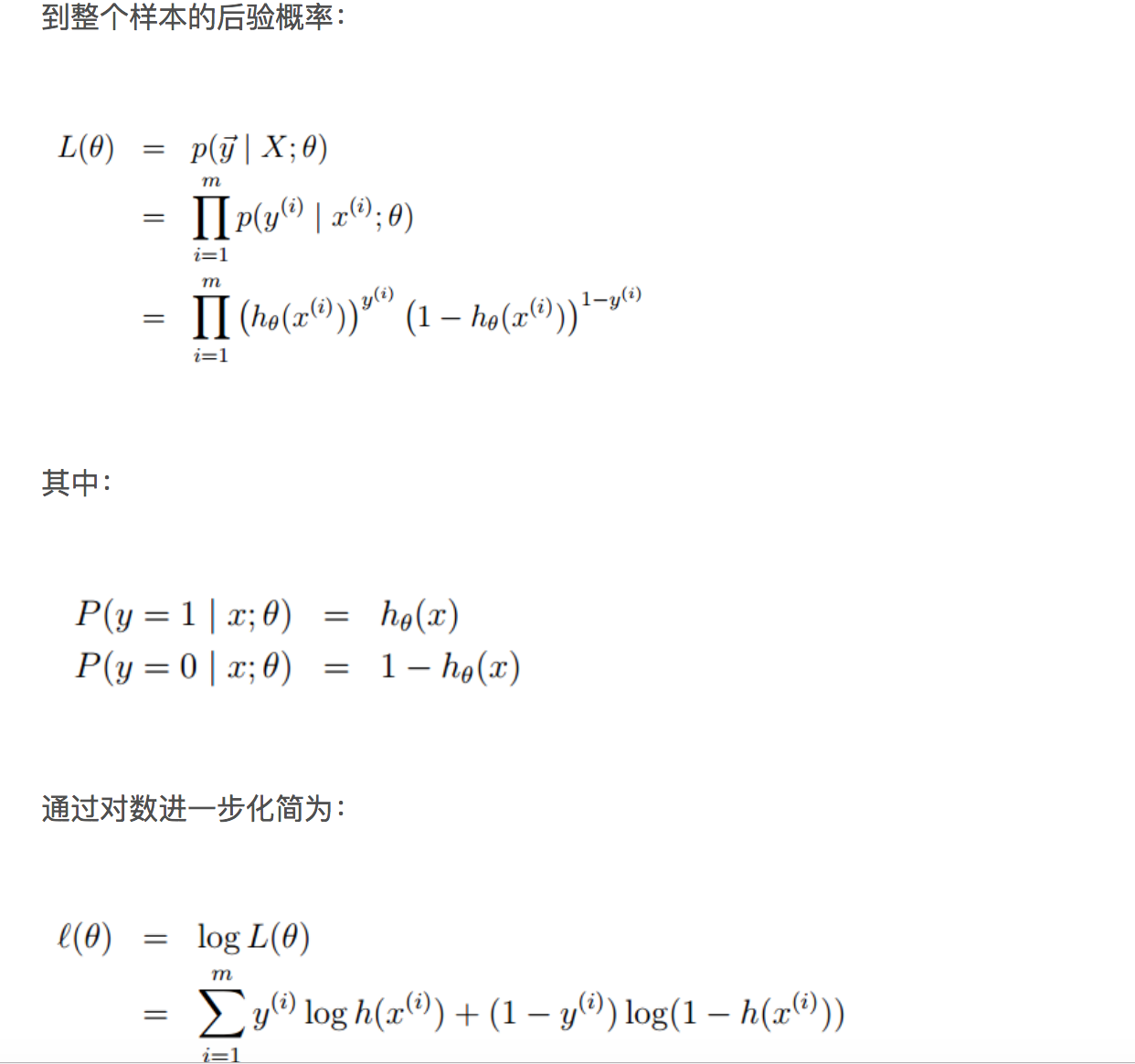
http://images.cnitblog.com/blog/381513/201310/29224932-92fa3e28b8d649adae9b667ad09b9810.png

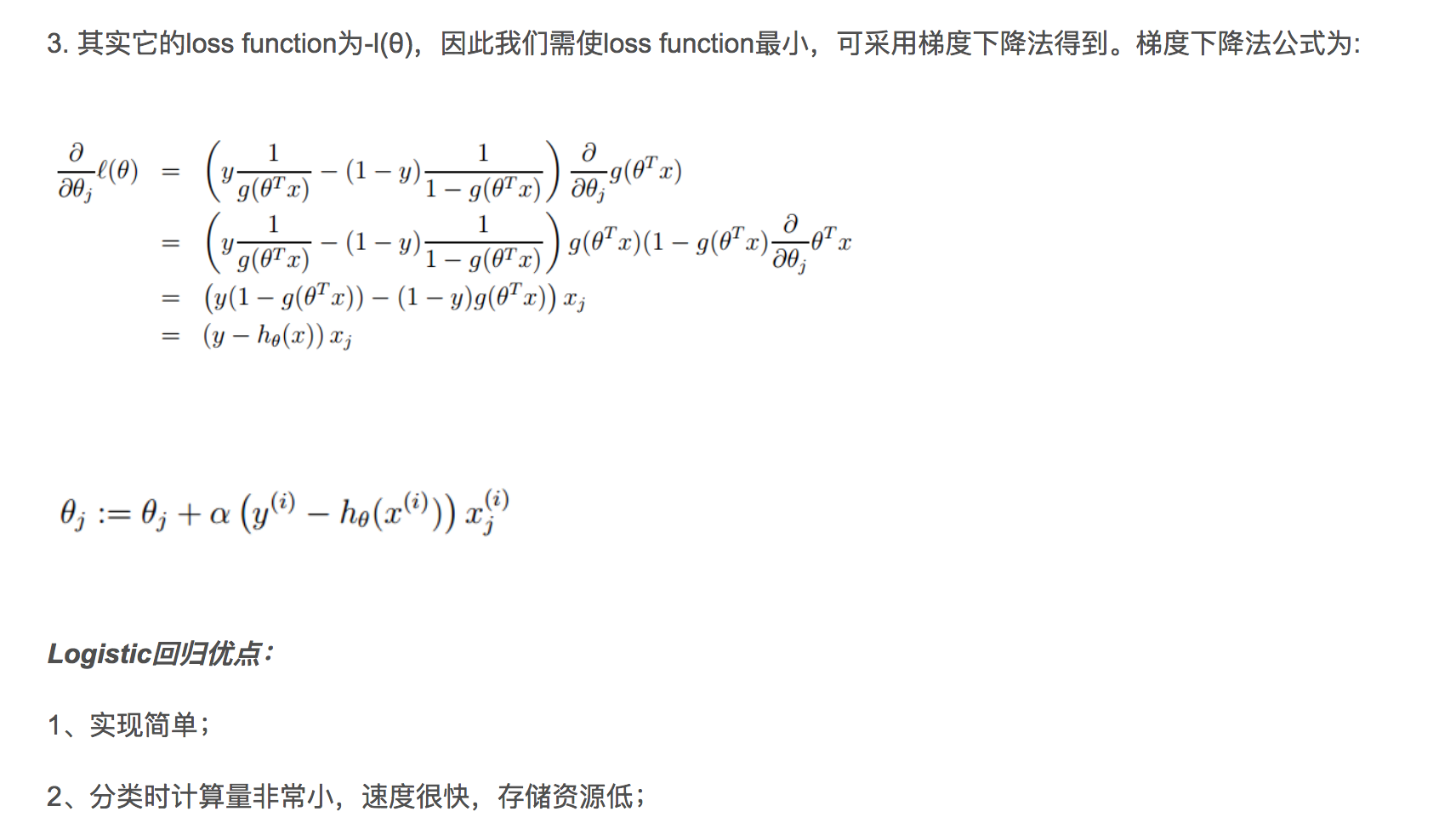


### 4.逻辑回归



http://images.cnitblog.com/blog/381513/201310/29224626-a42a5a2a05a34d82be182d150a625c94.png





***缺点：***

1、容易欠拟合，一般准确度不太高

2、只能处理两分类问题（在此基础上衍生出来的softmax可以用于多分类），且必须线性可分；

**定义：**

逻辑回归模型是一种强大的统计建模方式，它用一个或多个解释性变量对二值输出结果建模。它用逻辑斯蒂函数估计概率值，以此衡量分类依赖变量和一个或多个独立的变量之间的关系，这属于累积的逻辑斯蒂分布。

**使用场景：**

通常来说，逻辑回归模型在现实场景中的应用包括：

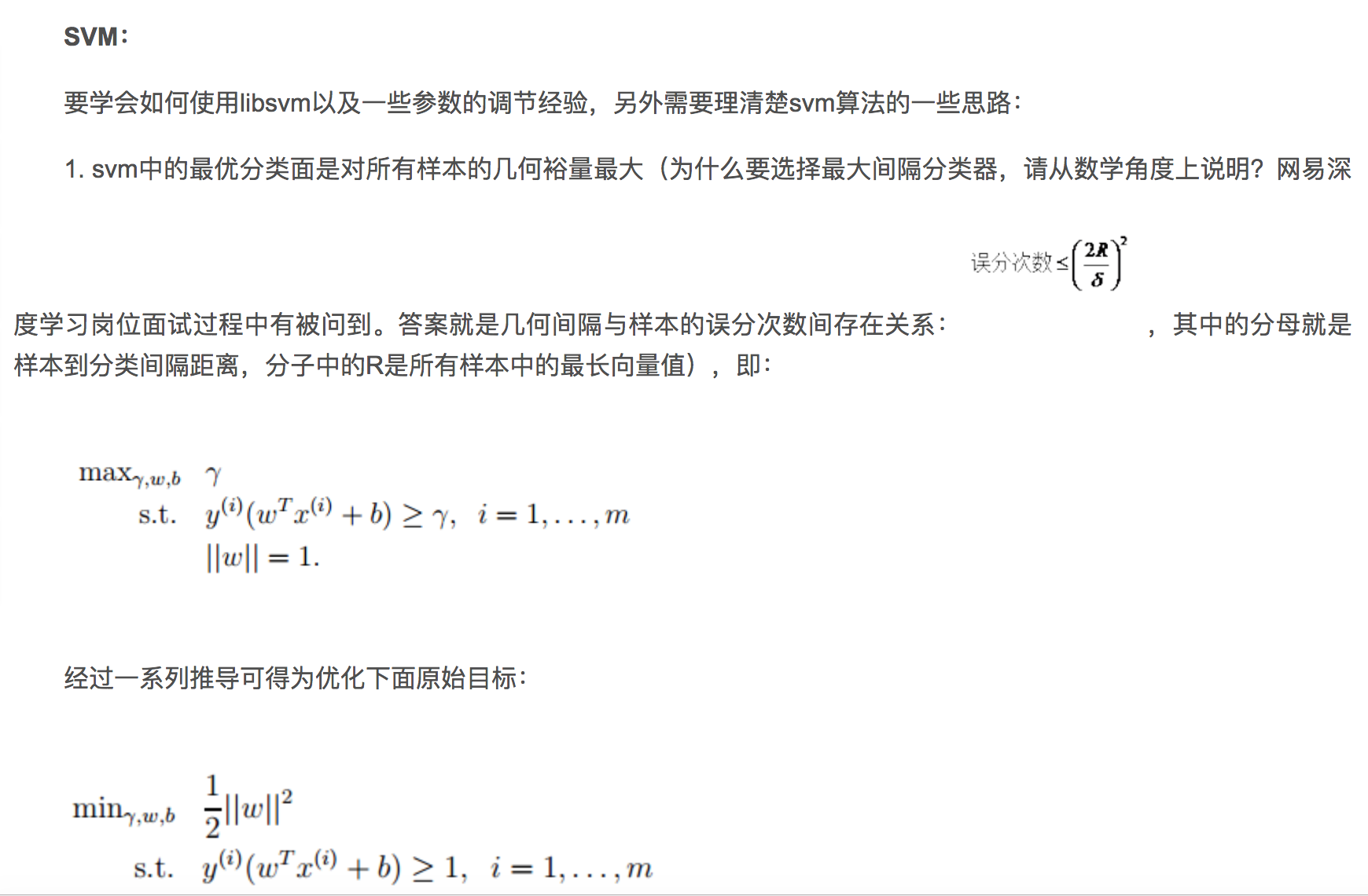
信用评分

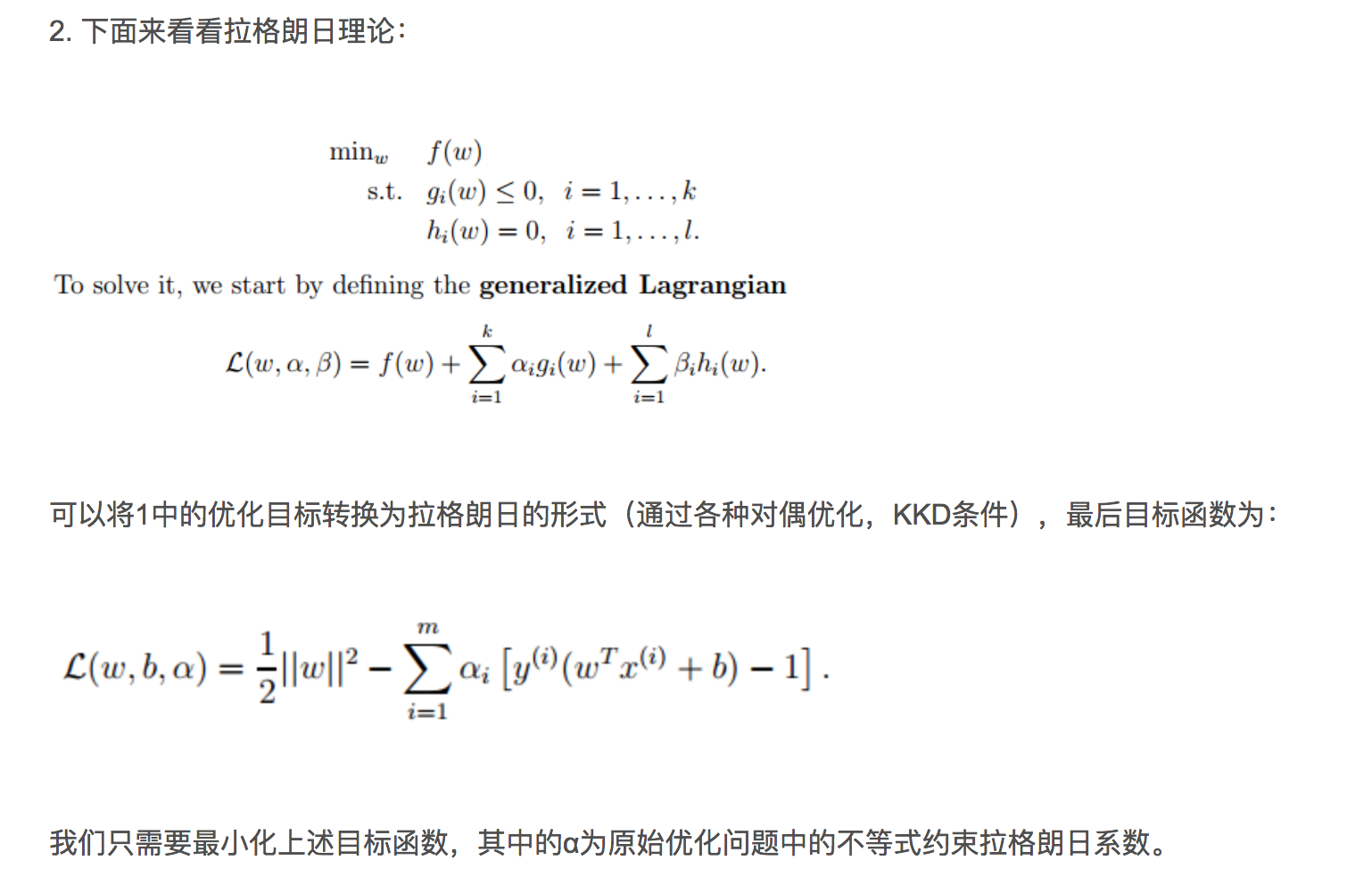
预测商业活动的成功概率

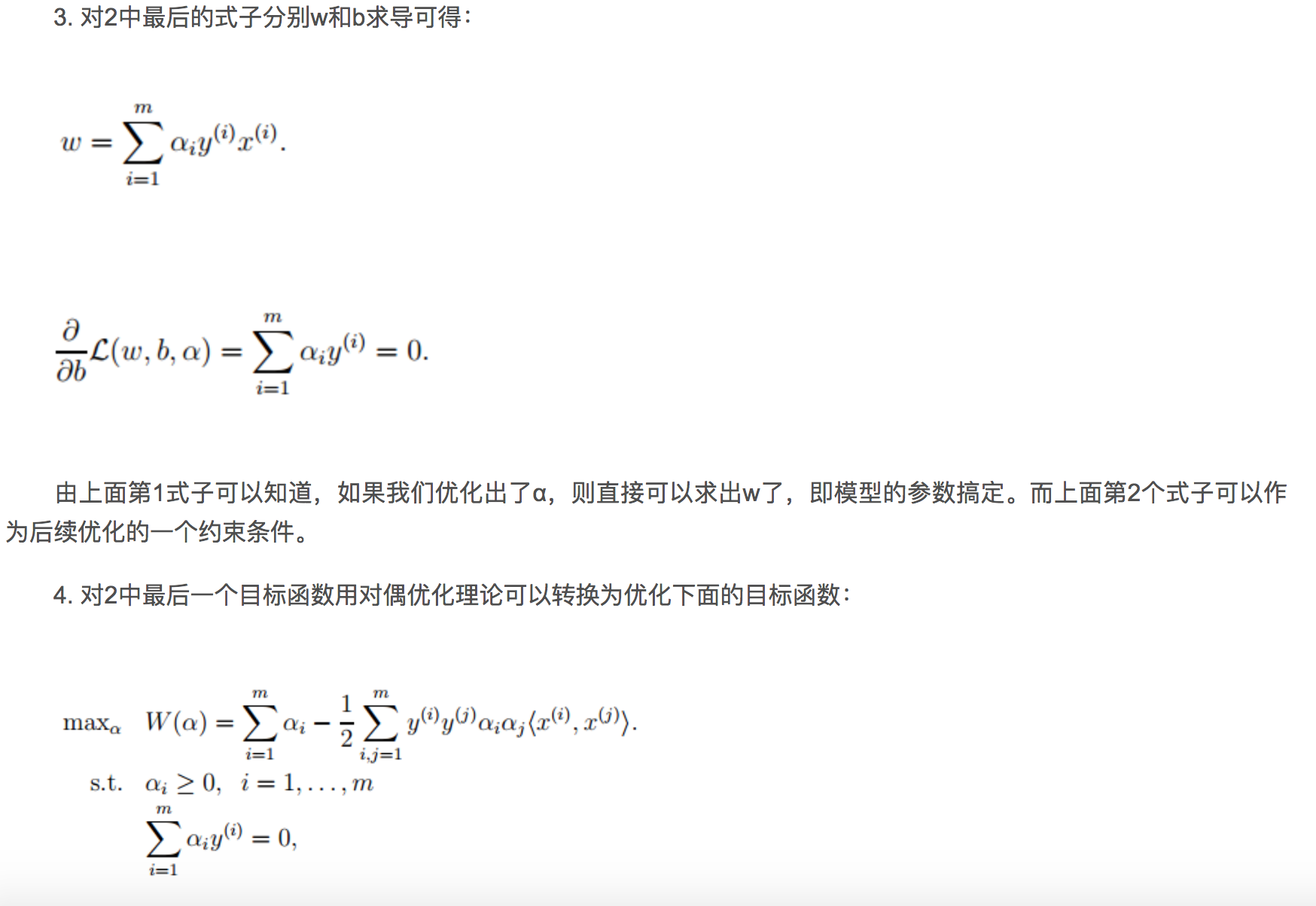
预测某款产品的收益

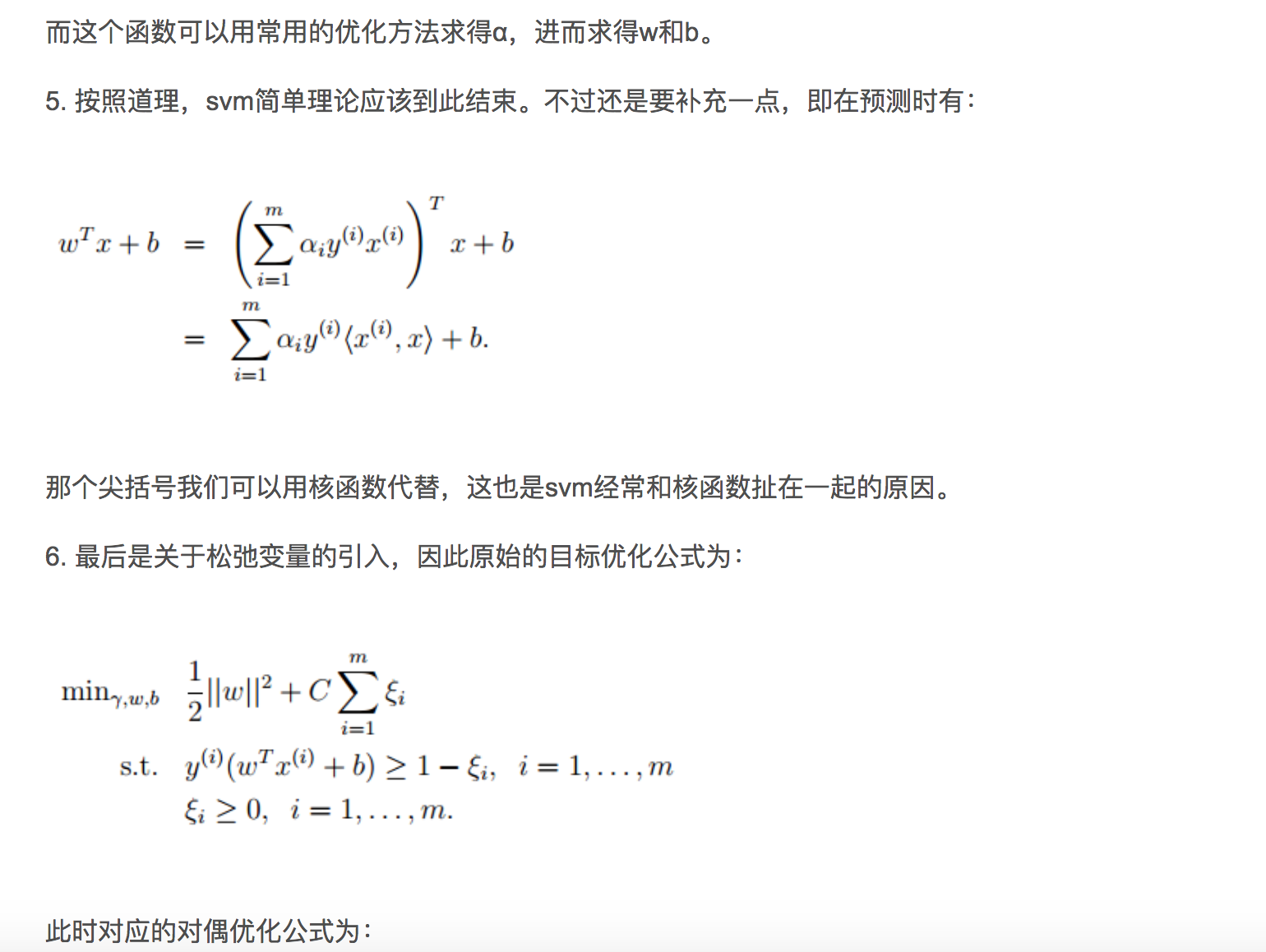
预测某一天发生地震的概率

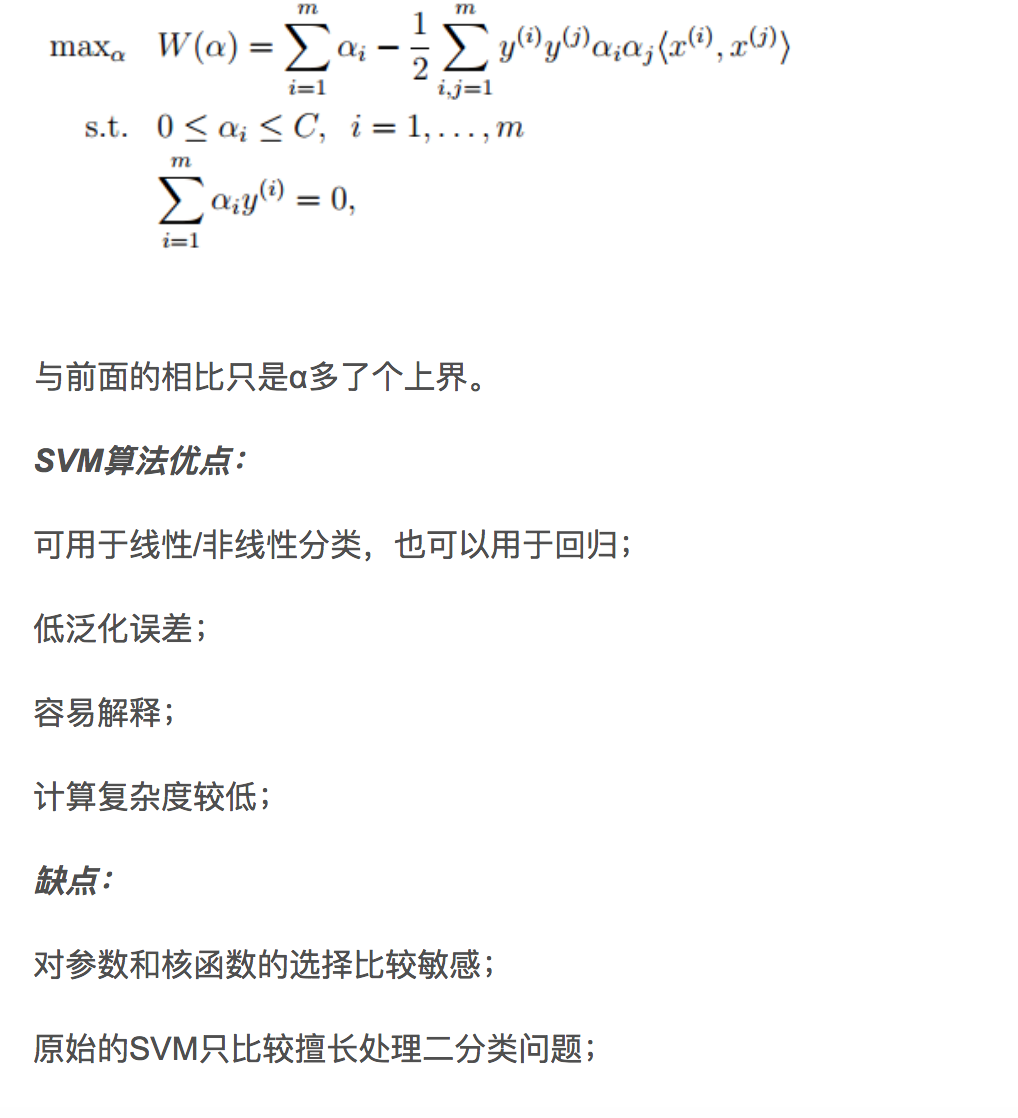
### 5.支持向量机











SVM是最大间隔分类器，几何间隔和样本的误分次数之间存在关系，https://images2015.cnblogs.com/blog/937251/201604/937251-20160419100648351-1060272610.png，其中https://images2015.cnblogs.com/blog/937251/201604/937251-20160419100648804-103630415.png

从线性可分情况下，原问题，特征转换后的dual问题，引入kernel(线性kernel,多项式，高斯),最后是soft margin。

线性：简单，速度快，但是需要线性可分

多项式：比线性核拟合程度更强，知道具体的维度，但是高次容易出现数值不稳定，参数选择比较多。

高斯：拟合能力最强，但是要注意过拟合问题。不过只有一个参数需要调整。

多分类问题，一般将二分类推广到多分类的方式有三种，一对一，一对多，多对多。

一对一：将N个类别两两配对，产生N(N-1)/2个二分类任务，测试阶段新样本同时交给所有的分类器，最终结果通过投票产生。

一对多：每一次将一个例作为正例，其他的作为反例，训练N个分类器，测试时如果只有一个分类器预测为正类，则对应类别为最终结果，如果有多个，则一般选择置信度最大的。从分类器角度一对一更多，但是每一次都只用了2个类别，因此当类别数很多的时候一对一开销通常更小(只要训练复杂度高于O(N)即可得到此结果)。

多对多：若干各类作为正类，若干个类作为反类。注意正反类必须特殊的设计。

**定义：**

支持向量机是一种二分类算法。在N维空间中给定两类点，支持向量机生成一个（N-1）维的超平面将这些点分为两类。举个例子，比如在纸上有两类线性可分的点。支持向量机会寻找一条直线将这两类点区分开来，并且与这些点的距离都尽可能远

**使用场景：**

利用支持向量机（结合具体应用场景做了改进）解决的大规模问题包括展示广告、人体结合部位识别、基于图像的性别检查、大规模图像分类等

### 6.集成方法

***Boosting算法的优点：***

低泛化误差；

容易实现，分类准确率较高，没有太多参数可以调；

***缺点：***

对outlier比较敏感；

**定义：**

集成方法是先构建一组分类器，然后用各个分类器带权重的投票来预测新数据的算法。最初的集成方法是贝叶斯平均，但最新的算法包括误差纠正输出编码和提升算法。

那么集成模型的原理是什么，以及它为什么比独立模型的效果好呢？

它们消除了偏置的影响：比如把民主党的问卷和共和党的问卷混合，从中得到的将是一个不伦不类的偏中立的信息。

它们能减小预测的方差：多个模型聚合后的预测结果比单一模型的预测结果更稳定。在金融界，这被称为是多样化 —— 多个股票的混合产品波动总是远小于单个股票的波动。这也解释了为何增加训练数据，模型的效果会变得更好。

它们不容易产生过拟合：如果单个模型不会产生过拟合，那么将每个模型的预测结果简单地组合（取均值、加权平均、逻辑回归），没有理由产生过拟合。

### 7.BP神经网络

### 8.K近邻KNN

1. 计算训练样本和测试样本中每个样本点的距离（常见的距离度量有欧式距离，马氏距离等）；

2. 对上面所有的距离值进行排序；

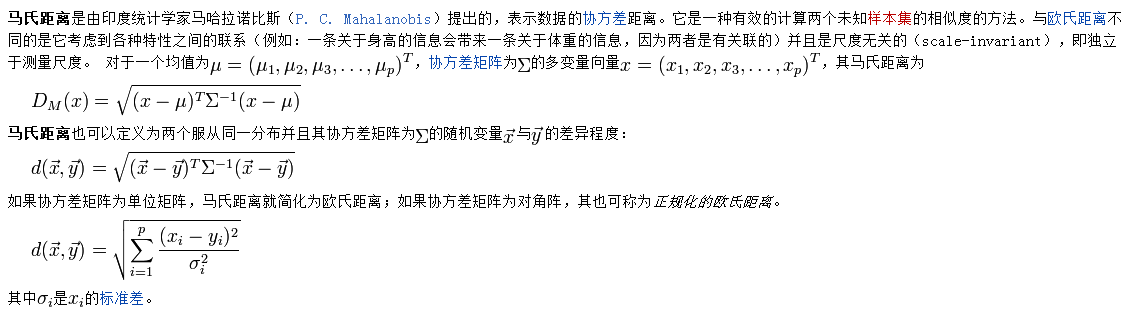
3. 选前k个最小距离的样本；

4. 根据这k个样本的标签进行投票，得到最后的分类类别；

如何选择一个最佳的K值，这取决于数据。一般情况下，在分类时较大的K值能够减小噪声的影响。但会使类别之间的界限变得模糊。一个较好的K值可通过各种启发式技术来获取，比如，交叉验证。另外噪声和非相关性特征向量的存在会使K近邻算法的准确性减小。

近邻算法具有较强的一致性结果。随着数据趋于无限，算法保证错误率不会超过贝叶斯算法错误率的两倍。对于一些好的K值，K近邻保证错误率不会超过贝叶斯理论误差率。

注：马氏距离一定要先给出样本集的统计性质，比如均值向量，协方差矩阵等。关于马氏距离的介绍如下：



***KNN算法的优点：***

1. 思想简单，理论成熟，既可以用来做分类也可以用来做回归；

2. 可用于非线性分类；

3. 训练时间复杂度为O(n)；

4. 准确度高，对数据没有假设，对outlier不敏感；

***缺点：***

1. 计算量大；

2. 样本不平衡问题（即有些类别的样本数量很多，而其它样本的数量很少）；

3. 需要大量的内存；

## 无监督学习：输入数据没有被标记，也没有确定的结果。样本数据类别未知，需要根据样本间的相似性对样本集进行分类（聚类，clustering）试图使类内差距最小化，类间差距最大化。

### 聚类算法

***k-means算法的优点：***

　　（1）k-means算法是解决聚类问题的一种经典算法，算法简单、快速。

　　（2）对处理[大数据](http://lib.csdn.net/base/hadoop)集，该算法是相对可伸缩的和高效率的，因为它的复杂度大约是O(nkt)，其中n是所有对象的数目，k是簇的数目,t是迭代的次数。通常k<<n。这个算法通常局部收敛。

　　（3）算法尝试找出使平方误差函数值最小的k个划分。当簇是密集的、球状或团状的，且簇与簇之间区别明显时，聚类效果较好。

***缺点：***

　　（1）k-平均方法只有在簇的平均值被定义的情况下才能使用，且对有些分类属性的数据不适合。

　　（2）要求用户必须事先给出要生成的簇的数目k。

　　（3）对初值敏感，对于不同的初始值，可能会导致不同的聚类结果。

　　（4）不适合于发现非凸面形状的簇，或者大小差别很大的簇。

　　（5）对于”噪声”和孤立点数据敏感，少量的该类数据能够对平均值产生极大影响。

　　2. 基于层次的聚类：

　　自底向上的凝聚方法，比如AGNES。

　　自上向下的分裂方法，比如DIANA。

　　3. 基于密度的聚类：

　　DBSACN,OPTICS,BIRCH(CF-Tree),CURE.

　　4. 基于网格的方法：

　　STING, WaveCluster.

　　5. 基于模型的聚类：

　　EM,SOM,COBWEB.

定义：

聚类算法的任务是将一群物体聚成多个组，分到同一个组（簇）的物体比其它组的物体更相似。

每种聚类算法都各不相同，这里列举了几种：

基于类心的聚类算法

基于连接的聚类算法

基于密度的聚类算法

概率型算法

降维算法

神经网络/深度学习

(1)K-Means算法

### 2.主成分分析

**PCA算法流程**

算法输入：数据集XmxnXmxn   
\* 按列计算数据集XX的均值XmeanXmean，然后令Xnew=X−XmeanXnew=X−Xmean；   
\* 求解矩阵XnewXnew的协方差矩阵，并将其记为CovCov；   
\* 计算协方差矩阵COvCOv的特征值和相应的特征向量；   
\* 将特征值按照从大到小的排序，选择其中最大的kk个，然后将其对应的kk个特征向量分别作为列向量组成特征向量矩阵WnxkWnxk;   
\* 计算XnewWXnewW，即将数据集XnewXnew投影到选取的特征向量上，这样就得到了我们需要的已经降维的数据集XnewWXnewW。

注意，计算一个nxnnxn矩阵的完整的特征向量分解的时间复杂度为 O(n3)O(n3) 。如果我们将数据集投影到前 kk 个主成分中，那么我们只需寻找前 kk 个特征值和特征向量。这可以使用更高效的方法得到,例如幂方法(power method) (Golub and Van Loan, 1996)，它的时间复杂度为 O(kn2)O(kn2)，或者我们也可以使用 EM 算法。

**2. PCA算法优缺点：**

优点：

* 它是无监督学习，完全无参数限制的。在PCA的计算过程中完全不需要人为的设定参数或是根据任何经验模型对计算进行干预，最后的结果只与数据相关，与用户是独立的。
* 用PCA技术可以对数据进行降维，同时对新求出的“主元”向量的重要性进行排序，根据需要取前面最重要的部分，将后面的维数省去，可以达到降维从而简化模型或是对数据进行压缩的效果。同时最大程度的保持了原有数据的信息。
* 各主成分之间正交，可消除原始数据成分间的相互影响。
* 计算方法简单，易于在计算机上实现。

缺点：

* 如果用户对观测对象有一定的先验知识，掌握了数据的一些特征，却无法通过参数化等方法对处理过程进行干预，可能会得不到预期的效果，效率也不高。
* 贡献率小的主成分往往可能含有对样本差异的重要信息。
* 特征值矩阵的正交向量空间是否唯一有待讨论。
* 在非高斯分布的情况下，PCA方法得出的主元可能并不是最优的，此时在寻找主元时不能将方差作为衡量重要性的标准。

**定义：**

主成分分析属于统计学的方法，通过正交变换将一组可能存在相关性的变量转换为一组线性不相关的变量，转换后的这组变量叫主成分。

**使用场景：**

主成分分析的一些实际应用包括数据压缩，简化数据表示，数据可视化等。值得一提的是需要领域知识来判断是否适合使用主成分分析算法。如果数据的噪声太大（即各个成分的方差都很大），就不适合使用主成分分析算法。

### 3.奇异值分解

**定义：**

奇异值分解是线性代数中一种重要的矩阵分解，是矩阵分析中正规矩阵对角化的推广。对于给定的m\*n矩阵M，可以将其分解为M = UΣV，其中U和V是m×m阶矩阵，Σ是半正定m×n阶对角矩阵。

**使用场景：**

主成分分析其实就是一种简单的奇异值分解算法。在计算机视觉领域中，第一例人脸识别算法使用了主成分分析和奇异值分解将人脸表示为一组“特征脸（eigenfaces）”的线性组合，经过降维，然后利用简单的方法匹配候选人脸。尽管现代的方法更加精细，许多技术还是于此很相似。

### 4.独立成分分析

**定义：**

独立成分分析是一种利用统计原理进行计算来揭示随机变量、测量值或者信号背后的隐藏因素的方法。独立成分分析算法给所观察到的多变量数据定义了一个生成模型，通常这些变量是大批量的样本。在该模型中，数据变量被假定为一些未知的潜变量的线性混合，而且混合系统也未知。潜变量被假定是非高斯和相互独立的，它们被称为所观察到的数据的独立分量。

**使用场景：**

独立成分分析与主成分分析有关联，但它是一个更强大的技术。它能够在这些经典方法失效时仍旧找到数据源的潜在因素。它的应用包括数字图像、文档数据库、经济指标和心理测量。

### 5. Apriori算法

6.

## 半监督学习

**定义：**

在此学习方式下，输入数据部分被标识，部分没有被标识，这种学习模型可以用来进行预测，但是模型首先需要学习数据的内在结构以便合理的组织数据来进行预测。

**应用场景：**

包括分类和回归，算法包括一些对常用监督式学习算法的延伸，这些算法首先试图对未标识数据进行建模，在此基础上再对标识的数据进行预测。如图论推理算法（Graph Inference）或者拉普拉斯支持向量机（Laplacian SVM）等。

## 强化学习

**定义：**

在这种学习模式下，输入数据作为对模型的反馈，不像监督模型那样，输入数据仅仅是作为一个检查模型对错的方式，在强化学习下，输入数据直接反馈到模型，模型必须对此立刻作出调整。

**使用场景：**

包括动态系统以及机器人控制等。常见算法包括Q-Learning以及时间差学习（Temporal difference learning）。在企业数据应用的场景下，人们最常用的可能就是监督式学习和非监督式学习的模型。在图像识别等领域，由于存在大量的非标识的数据和少量的可标识数据， 目前半监督式学习是一个很热的话题。 而强化学习更多的应用在机器人控制及其他需要进行系统控制的领域。

### **GBDT：**

GBDT(Gradient Boosting Decision Tree) 又叫 MART（Multiple Additive Regression Tree)，好像在阿里内部用得比较多（所以阿里算法岗位面试时可能会问到），它是一种迭代的决策树算法，该算法由多棵决策树组成，所有树的输出结果累加起来就是最终答案。它在被提出之初就和SVM一起被认为是泛化能力（generalization)较强的算法。近些年更因为被用于搜索排序的机器学习模型而引起大家关注。

GBDT是回归树，不是分类树。其核心就在于，每一棵树是从之前所有树的残差中来学习的。为了防止过拟合，和Adaboosting一样，也加入了boosting这一项。

### Regularization:

　　作用是（网易电话面试时有问到）：

　　1. 数值上更容易求解；

　　2. 特征数目太大时更稳定；

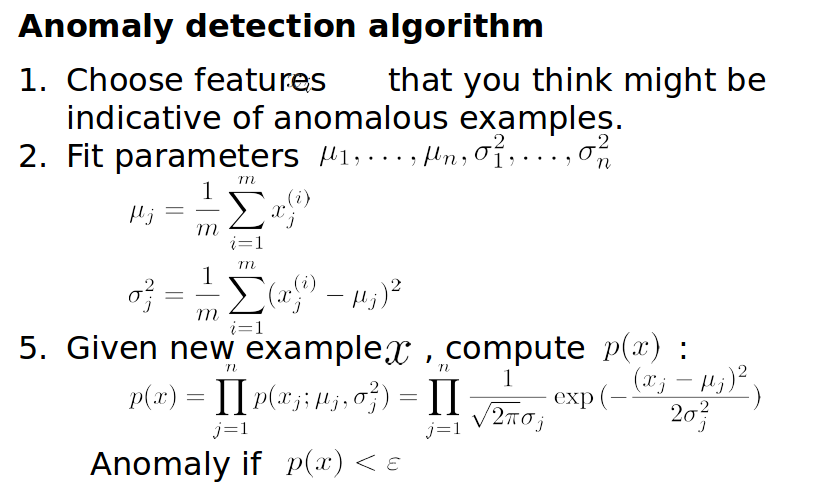
　　3. 控制模型的复杂度，光滑性。复杂性越小且越光滑的目标函数泛化能力越强。而加入规则项能使目标函数复杂度减小，且更光滑。

　　4. 减小参数空间；参数空间越小，复杂度越低。

　　5. 系数越小，模型越简单，而模型越简单则泛化能力越强（Ng宏观上给出的解释）。

### **异常检测：**

　　可以估计样本的密度函数，对于新样本直接计算其密度，如果密度值小于某一阈值，则表示该样本异常。而密度函数一般采用多维的高斯分布。如果样本有n维，则每一维的特征都可以看作是符合高斯分布的，即使这些特征可视化出来不太符合高斯分布，也可以对该特征进行数学转换让其看起来像高斯分布，比如说x=log(x+c),x=x^(1/c)等。异常检测的算法流程如下：



Q:1.机器学习模型按照可使用的数据类型，可以分为哪些类别

1. 各个模型分别基于哪些数学假设
2. 每个模型适合处理哪类数据
3. 每个模型在使用方面有哪些优缺点
4. 常见的评估各个模型的性能指标以及计算方法是怎样的

### SVM、LR、决策树的对比？

SVM既可以用于分类问题，也可以用于回归问题，并且可以通过核函数快速的计算，LR实现简单，训练速度非常快，但是模型较为简单，决策树容易过拟合，需要进行剪枝等。从优化函数上看，soft margin的SVM用的是hinge loss,而带L2正则化的LR对应的是cross entropy loss，另外adaboost对应的是exponential loss。所以LR对远点敏感，但是SVM对outlier不太敏感，因为只关心support vector，SVM可以将特征映射到无穷维空间，但是LR不可以，一般小数据中SVM比LR更优一点，但是LR可以预测概率，而SVM不可以，SVM依赖于数据测度，需要先做归一化，LR一般不需要，对于大量的数据LR使用更加广泛，LR向多分类的扩展更加直接，对于类别不平衡SVM一般用权重解决，即目标函数中对正负样本代价函数不同，LR可以用一般的方法，也可以直接对最后结果调整(通过阈值)，一般小数据下样本维度比较高的时候SVM效果要更优一些。

### GBDT 和随机森林的区别？

随机森林采用的是bagging的思想，bagging又称为bootstrap aggreagation，通过在训练样本集中进行有放回的采样得到多个采样集，基于每个采样集训练出一个基学习器，再将基学习器结合。随机森林在对决策树进行bagging的基础上，在决策树的训练过程中引入了随机属性选择。传统决策树在选择划分属性的时候是在当前节点属性集合中选择最优属性，而随机森林则是对结点先随机选择包含k个属性的子集，再选择最有属性，k作为一个参数控制了随机性的引入程度。

另外，GBDT训练是基于Boosting思想，每一迭代中根据错误更新样本权重，因此是串行生成的序列化方法，而随机森林是bagging的思想，因此是并行化方法。

### GBDT与Adboost

最主要的区别在于两者如何识别模型的问题。Adaboost用错分数据点来识别问题，通过调整错分数据点的权重来改进模型。GBDT通过负梯度来识别问题，通过计算负梯度来改进模型。

### GBDT与xgboost

GBDT算法只利用了一阶的导数信息，xgboost对损失函数做了二阶的泰勒展开，并在目标函数之外加入了正则项对整体求最优解，用以权衡目标函数的下降和模型的复杂程度，避免过拟合。所以不考虑细节方面，两者最大的不同就是目标函数的定义，接下来就着重从xgboost的目标函数定义上来进行介绍

过渡拟合的原因有以下几点：   
•噪音数据：训练数据中存在噪音数据，决策树的某些节点有噪音数据作为分割标准，导致决策树无法代表真实数据。   
•缺少代表性数据：训练数据没有包含所有具有代表性的数据，导致某一类数据无法很好的匹配，这一点可以通过观察混淆矩阵（Confusion Matrix）分析得出。   
•多重比较（Mulitple Comparision）：举个列子，股票分析师预测股票涨或跌。假设分析师都是靠随机猜测，也就是他们正确的概率是0.5。每一个人预测10次，那么预测正确的次数在8次或8次以上的概率为 ，C810∗(0.5)10+C910∗(0.5)10+C1010∗(0.5)10C108∗(0.5)10+C109∗(0.5)10+C1010∗(0.5)10只有5%左右，比较低。但是如果50个分析师，每个人预测10次，选择至少一个人得到8次或以上的人作为代表，那么概率为 1−(1−0.0547)50=0.93991−(1−0.0547)50=0.9399，概率十分大，随着分析师人数的增加，概率无限接近1。但是，选出来的分析师其实是打酱油的，他对未来的预测不能做任何保证。上面这个例子就是多重比较。这一情况和决策树选取分割点类似，需要在每个变量的每一个值中选取一个作为分割的代表，所以选出一个噪音分割标准的概率是很大的。