**SVM（支持向量机）**

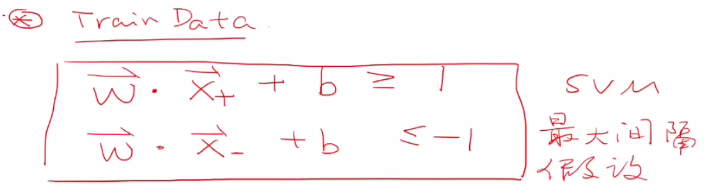
**1、support vector machine**

主要目的：如何合理的找到最好的分类边界，即找到分类超平面。

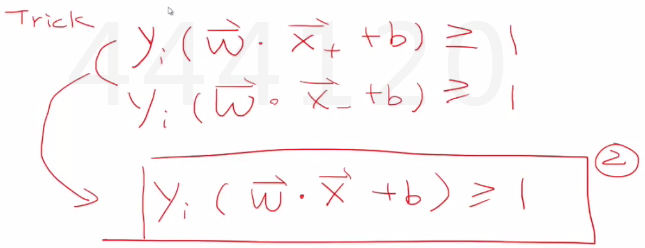
最好的边界：最大间隔，margin:离正负样本，决策边界。

预测：得到参数w，直接获得w\*u+b>=0，为+。

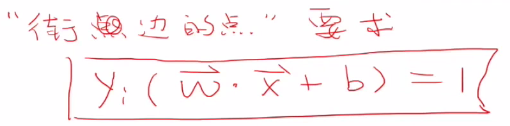
最大间隔假设：



约束条件：

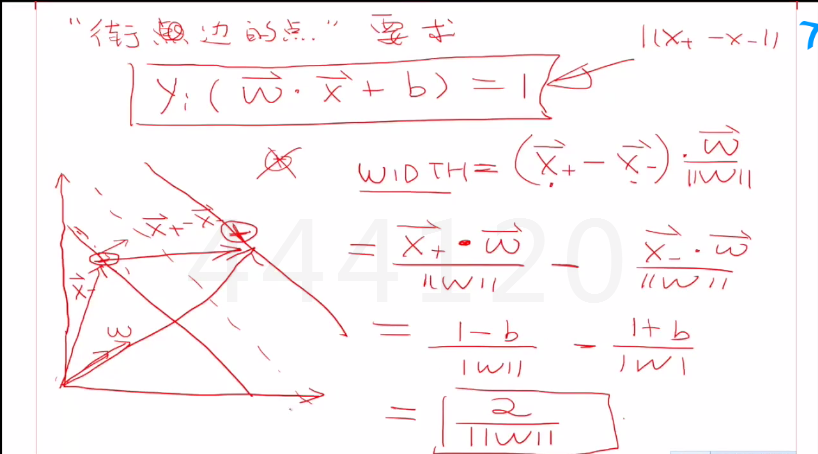


支持向量：



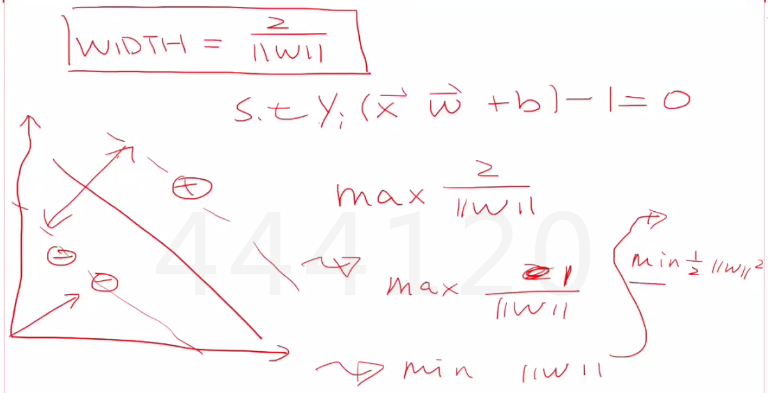
街边宽度：

单位向量，正负样本在w上投影的长度。

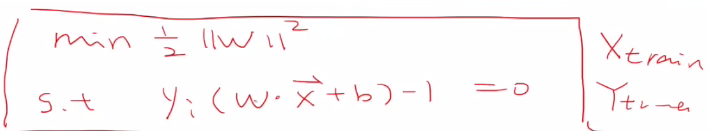


Margin长度为：

目标函数为：为了数学的方便得到最后的min||w||2。

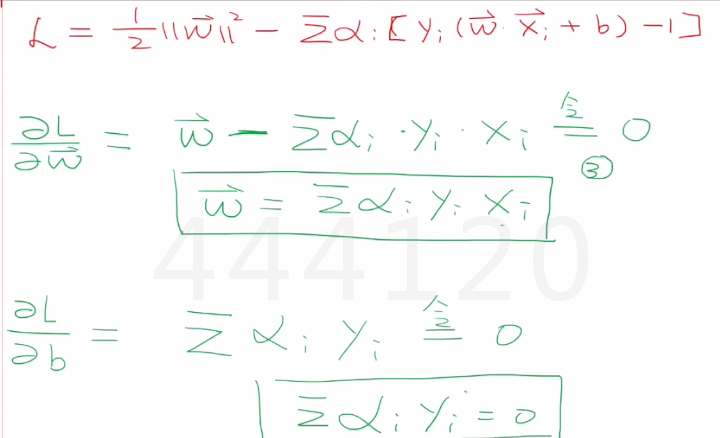


即最后可以写成：

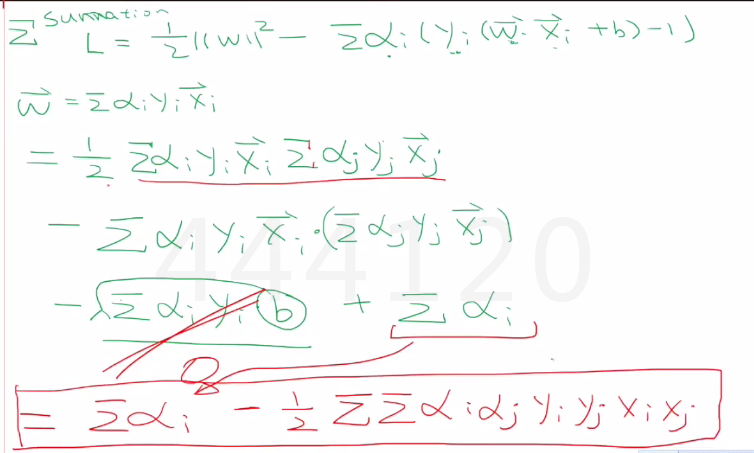


利用拉格朗日定力求解：

分别对w和b求导数：

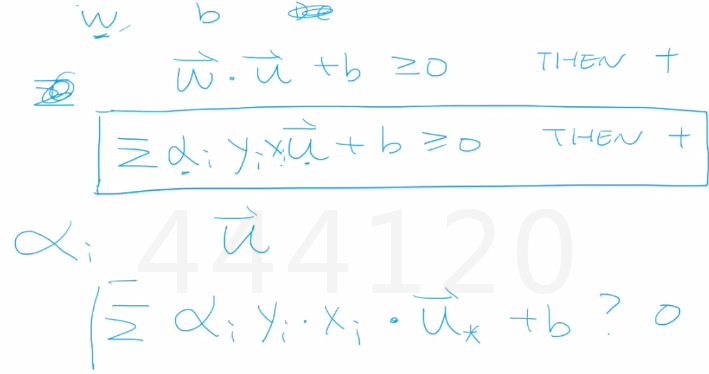


再将导数带入到原来的L式子中去。

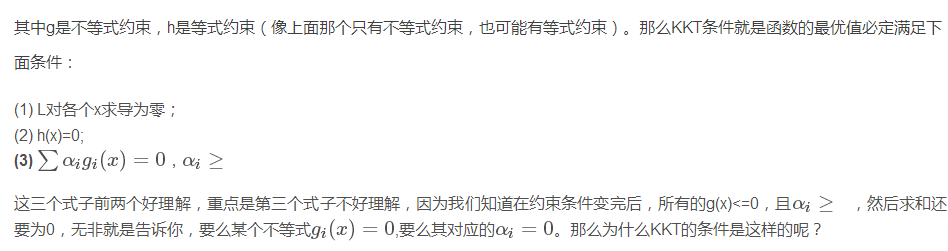


这个式子值取决于：训练集中数据点两两之间的点乘，于是便转化为求解该式子中的阿尔法的值。

于是预测结果值：



**KKT条件：**

****

**2、线性回归和逻辑回归的不同点：**

logistic回归就是一个线性分类模型，它与线性回归的不同点在于：为了将线性回归输出的很大范围的数，例如从负无穷到正无穷，压缩到0和1之间，这样的输出值表达为“可能性”才能说服广大民众。当然了，把大值压缩到这个范围还有个很好的好处，就是可以消除特别冒尖的变量的影响（不知道理解的是否正确）

**3、欧氏距离来计算最近的邻居之间的距离，有时也用曼哈顿距离，请对比下这两种距离的差别**

欧氏距离，最常见的两点之间或多点之间的距离表示法，又称之为欧几里得度量，它定义于欧几里得空间中，如点 x = (x1,...,xn) 和 y = (y1,...,yn) 之间的距离为：

https://pic1.zhimg.com/80/v2-5c52b3e1a005c1b088a4877445523f37_hd.jpg

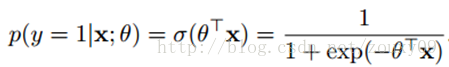
曼哈顿距离，我们可以定义曼哈顿距离的正式意义为L1-距离或城市区块距离，也就是在欧几里得空间的固定直角坐标系上两点所形成的线段对轴产生的投影的距离总和。例如在平面上，坐标（x1, y1）的点P1与坐标（x2, y2）的点P2的曼哈顿距离为

https://pic4.zhimg.com/80/v2-2b7100736e6870c9ce4e2a51037dbae1_hd.jpg

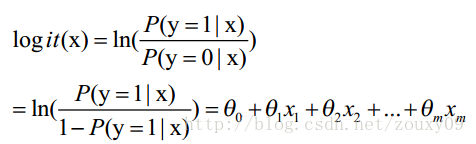
通俗来讲，想象你在曼哈顿要从一个十字路口开车到另外一个十字路口，驾驶距离是两点间的直线距离吗？显然不是，除非你能穿越大楼。而实际驾驶距离就是这个“曼哈顿距离”，这也是曼哈顿距离名称的来源， 同时，曼哈顿距离也称为城市街区距离(City Block distance)。

**4、逻辑回归LR**

假设我们的样本是{x, y}，y是0或者1，表示正类或者负类，x是我们的m维的样本特征向量。那么这个样本x属于正类，也就是y=1的“概率”可以通过下面的逻辑函数来表示：



其中表示要学习的模型参数，即回归系数，表示激活函数sigmoid.



logistic回归就是一个线性分类模型，它与线性回归的不同点在于：为了将线性回归输出的很大范围的数，例如从负无穷到正无穷，压缩到0和1之间，这样的输出值表达为“可能性”才能说服广大民众。当然了，把大值压缩到这个范围还有个很好的好处，就是可以消除特别冒尖的变量的影响（不知道理解的是否正确）。而实现这个伟大的功能其实就只需要平凡一举，也就是在输出加一个logistic函数。另外，对于二分类来说，可以简单的认为：如果样本x属于正类的概率大于0.5，那么就判定它是正类，否则就是负类。实际上，SVM的类概率就是样本到边界的距离，这个活实际上就让logistic regression给干了.

其最基本的学习算法最大似然算法：

假设我们有n个独立的训练样本{(x1, y1) ,(x2, y2),…, (xn, yn)}，y={0, 1}。那每一个观察到的样本(xi, yi)出现的概率是：

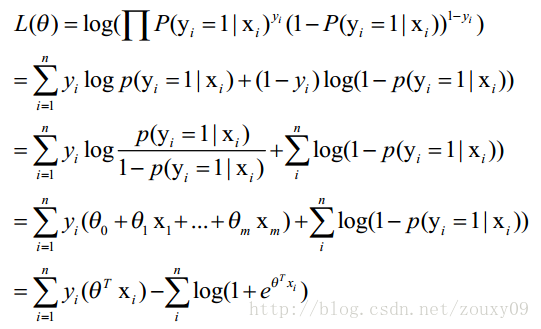
http://img.blog.csdn.net/20140302234309468?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvem91eHkwOQ==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/SouthEast

上面为什么是这样呢？当y=1的时候，后面那一项是不是没有了，那就只剩下x属于1类的概率，当y=0的时候，第一项是不是没有了，那就只剩下后面那个x属于0的概率（1减去x属于1的概率）。所以不管y是0还是1，上面得到的数，都是(x, y)出现的概率。那我们的整个样本集，也就是n个独立的样本出现的似然函数为（因为每个样本都是独立的，所以n个样本出现的概率就是他们各自出现的概率相乘）。

http://img.blog.csdn.net/20140302234332921?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvem91eHkwOQ==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/SouthEast

目的是：,求解其中的模型参数。这个似然函数就是代价函数。

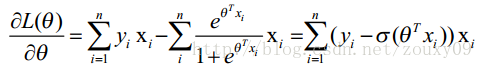
接下来就是优化cost函数了：



首先：求自然对数log：

对数相乘->对数相加：

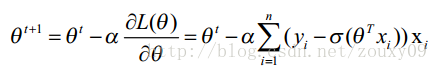
然后再对求导：



得不到解析解：

所以需要用到梯度下降算法来解决：

对于逻辑回归来讲，其梯度下降算法如下所示：



其中是步长，也称为学习率。

备注：因为本文中是求解的Logit回归的代价函数是似然函数，需要最大化似然函数。所以我们要用的是梯度上升算法。但因为其和梯度下降的原理是一样的，只是一个是找最大值，一个是找最小值。找最大值的方向就是梯度的方向，最小值的方向就是梯度的负方向。

梯度下降算法在每次更新回归系数的时候都需要遍历整个数据集（计算整个数据集的回归误差），该方法对小数据集尚可。但当遇到有数十亿样本和成千上万的特征时，就有点力不从心了，它的计算复杂度太高。改进的方法是一次仅用一个样本点（的回归误差）来更新回归系数。这个方法叫随机梯度下降算法。

**5、Expectation Maximization**

EM算法就是这样，假设我们想估计知道A和B两个参数，在开始状态下二者都是未知的，但如果知道了A的信息就可以得到B的信息，反过来知道了B也就得到了A。可以考虑首先赋予A某种初值，以此得到B的估计值，然后从B的当前值出发，重新估计A的取值，这个过程一直持续到收敛为止。

**6、LR和SVM的联系和区别**

（1）LR和SVM都是处理分类问题的，且一般都用来处理线性二分类问题，

（2）两个方法都可以增加不同的正则项，例如L1和L2.

（3）从目标函数来看，逻辑回归采用的是logistical loss，SVM采用的hige loss.这两个损失函数的目的都是：增加对分类影响较大的数据点的权重，减少与分类关系较小的数据点的权重。

（4）SVM的处理方法只考虑support vectors,采用与分类最相关的少数点学习分类器；逻辑回归通过非线性映射，大大减小了离分类平面较远的点的权重，相对提升了与分类最相关的数据点的权重。

（5）逻辑回归模型简单、比较好理解、对大规模的线性分类很方便。而SVM则相对复杂一些，在转化为对偶问题后，分类只需要计算少数几个支持向量的距离。

**7、过拟合的原因与解决方案**

原因：数据和算法

数据：数据不规范，数据量小、统计特征不规范、模型复杂度过高、训练数据过少、训练误差过少，测试误差大。

算法：算法太过于复杂

解决：

1. 数据规范化、处理缺失值、增加数据量、添加噪声数据
2. 正则化（加上正则惩罚项）、控制模型的复杂程度
3. early stopping、减少迭代次数、
4. 调整学习率，包括调大和调小
5. 融合几个模型

**8、L1和L2区别**

(1)L1是Lasso Regression，表示向量中每个元素绝对值的和：L1范数的解通常是稀疏性的，倾向于选择数目较少的一些非常大的值或者数目较多的insignificant的小值

(2) L2是岭回归，Ridge Regression，是欧氏距离也就是平方和的平方根。L2范数越小，可以使得w的每个元素都很小，接近于0，但L1范数不同的是他不会让它等于0而是接近于0

（3）L1正则化的w可取的值是转置的方形，L2对应的是圆形。这样损失函数l（w）的最小值更容易在L1对应的边角上取得，从而这些维度变成0了。L1在w=0处不可微，可以分解为一个求和的形式。

正则化具体问题的解释：

机器学习中的目标函数为：

C:\Users\ADMINI~1\AppData\Local\Temp\1520927939(1).png

其中第一项表示：训练模型预测值和标签值之间的误差。因为我们训练的模型要拟合我们的训练样本，所以要求这一项最小，也就是要求我们的模型尽量的拟合我们的训练数据。不仅要保证训练误差最小，还要希望模型测试的误差要小，所以我们需要加上第二项：也就是参数约束我们的模型尽量简单。

L1范数是指各个元素的绝对值之和，也有一个名字叫做“稀疏规则算子”。L1可以实现稀疏，具有更好地优化求解特性。

**特征选择和可解释性：**

稀疏规则化的关键原因是：可以实现特征的自动选择，一般来说样本x的大部分元素都是和最终的输出y没有关系或者说不提供任何信息，在最小化目标函数的时候x需要考虑这些额外的特征，虽然可以获得更小的训练误差，但是在预测新的样本时，这些没用的信息反而会被考虑，从而干扰了对正确y值的预测。系数规则化算子的引入就是为了完成特征自动选择的任务，学习如何去掉这些没用的信息的特征，也就是将这些对应特征的权重设置为0.

模型更容易解释。例如某人患病的概率是y,然后我们收集的数据x是1000维的，也就是我们需要寻找这1000种元素到底是怎么患上这种病的？假设这个是回归模型：（当然y值的取值范围是：0-1，一般还要加一个逻辑回归函数）。通过模型学习得到最后的w，发现只有5个非零的w,这5个对应的特征在患病分析上提供的信息是巨大的，我们就有理由相信患不患病只和这5个因素有关。那么医生就好分析多了，加入1000个w都非0，那么医生面对这1000个参数也是累觉不爱。

L2范数也称为权值衰减，其强大的功效是：过拟合问题。至于过拟合是：模型训练的误差很小，但是测试时误差却很大，也就是训练模型复杂可以你和所有的训练样本，但在实际预测新的样本的时候效果却糟糕的一塌糊涂。

L2范数是各元素的平方和求平方根，即L2范数的规则项最小，可以使得w的每个元素都很小，都接近于0，却不会等于0，参数越小，模型越简单，模型越简单，越不容易产生过拟合现象。

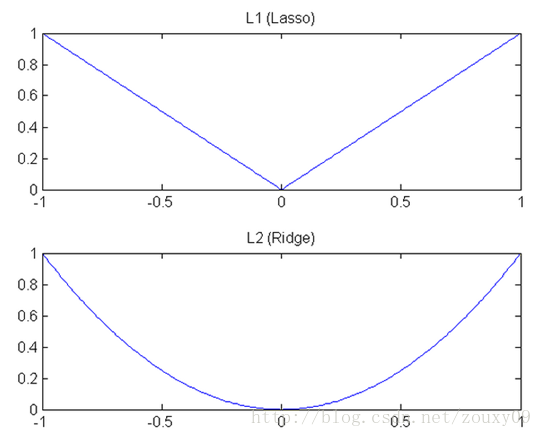
**好处：**

（1）L2范数可以防止过拟合，提升模型的泛化能力

（2）L2范数有助于处理condition number不好的情况矩阵求逆很困难的问题。

对于L1约束和L2约束都是规则化的方式：

将权系数以L1和L2的方式放到代价函数中去，然后模型就会尝试着最小化这些权值参数，L1和L2的差别就在于：他们的坡度不同。



**规则化参数的选择：**

机器学习中的目标函数除了loss和规则项之外，还有一个参数，叫做超参数。用来平衡loss和规则项这两项内容，其中越大，表示规则项要比模型训练误差更重要。当等于0的时候，目标函数全部取决于第一项。但是我们希望的模型是：既可以拟合我们的数据，又可以约束它的特性。这样才使得训练模型发挥强大的功能。

C:\Users\ADMINI~1\AppData\Local\Temp\1520927939(1).png

那么具体的怎么来选择那？一是：在训练之前大概计算一下这时候的loss值是多少，Ω(w)是多少？针对他们的比例来确定；二是：交叉验证cross validation.首先将训练集分为几份，取一部分做训练集，一部分做测试集，然后选择不同的训练这个模型；用测试集测试训练好的N个模型，选择N个模型中测试误差最小的作为最终的。

**9、几种核函数**

（1）多项式核函数

https://pic4.zhimg.com/80/v2-70c0901a34bdeea99fd8cbd26c890bbc_hd.jpg

（2）高斯核函数

https://pic2.zhimg.com/80/v2-3fc3123424ab0e1d35965333ed5c36b0_hd.jpg

（3）线性核函数

https://pic4.zhimg.com/80/v2-cdf7ce2173c69b8d4cde39b649e5f302_hd.jpg

（4）几种核函数的区别：

**10、线性回归和逻辑回归的区别与联系**

（1）线性回归优化目标函数是最小二乘，逻辑回归的似然函数；

（2）线性回归在整个实数域中进行，分类范围为[0,1]；逻辑回归减小了预测范围，将预测值定义为[0,1]的一种回归模型。因此，逻辑回归的鲁棒性比线性回归要好。

**11、交叉熵**

**联合熵：**

两个随机变量X,Y的联合分布，可以形成联合熵Joint Entropy，用H(x,y)表示。

**条件熵：**

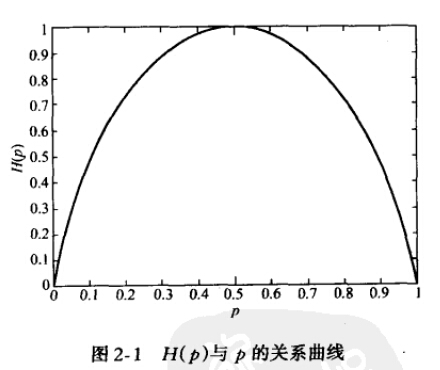
在随机变量x发生的前提下，随机变量y所新带来的熵定义为Y的条件熵。

在信息量中：一个事件发生的概率越大，其携带的信息量就越小。所以定义事件的信息量为：。

熵实际上可以看成是：信息量的期望值，是一个随机变量确定性的度量，熵值越大，变量的取值越不确定，反之就越确定。熵的定义式如下：



当X为0-1分布时，熵与概率p关系如下图所示：

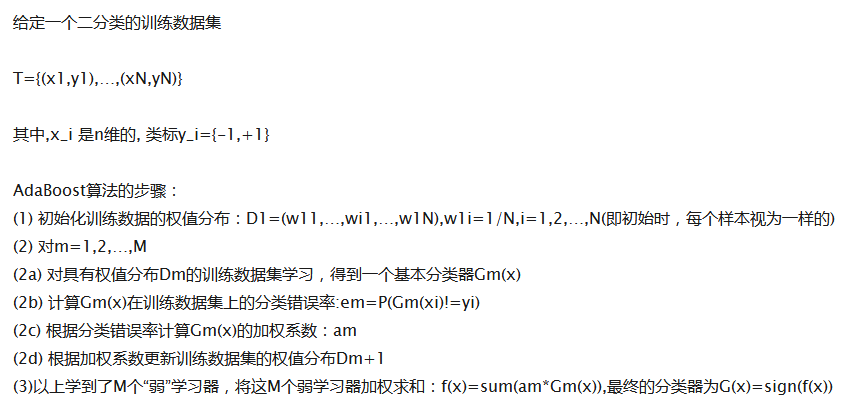


从上图可以看出：当两种取值的可能性相等时，不确定度最大，这个结论可以推广到多种取值情况；在图中也可以看出，当p=0或p=1时，即此时的x完全确定。

**12、集成学习算法总结**

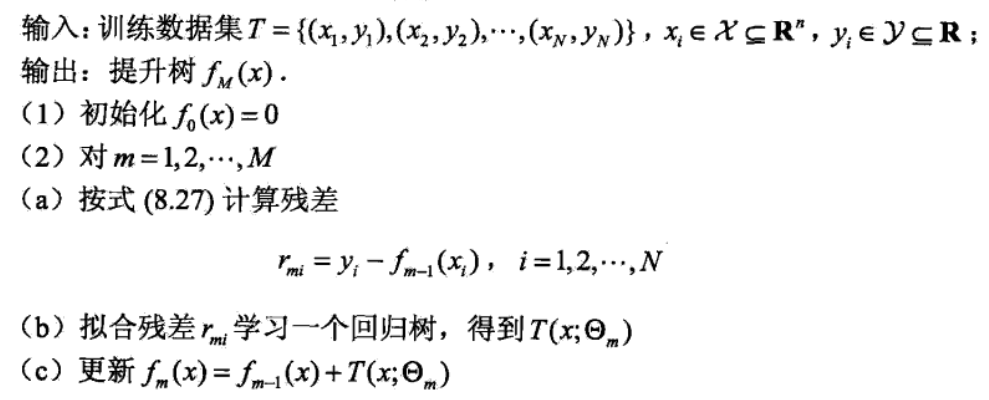
（1）AdaBoost算法

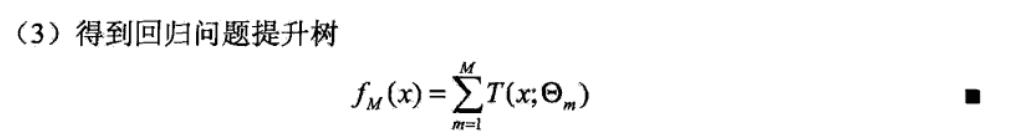
主要思想：不改变训练数据的情况下，通过迭代训练弱分类器，不断提升被错分类样本的权重，不断较少正确分类样本的权重。通过加权线性结合M各弱分类器得到最终的分类器，正确率越高的弱分类器的投票权重越高，正确率低的弱分类器自然投票权重就越低。



（2）提升树算法boosting tree

根据训练数据集，输出提升树。主要根据：残差。

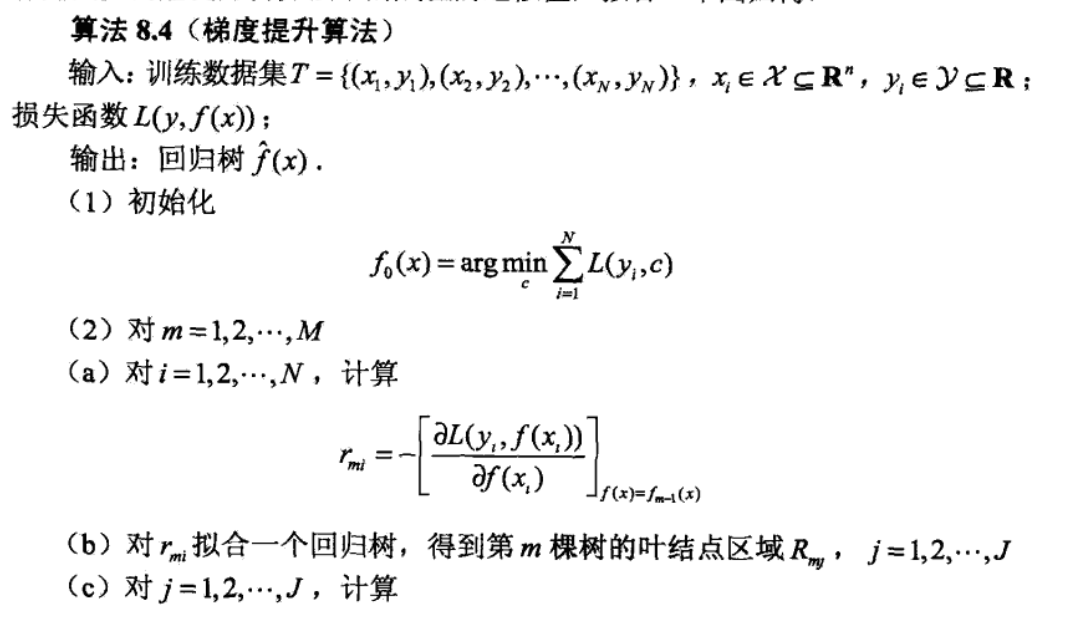


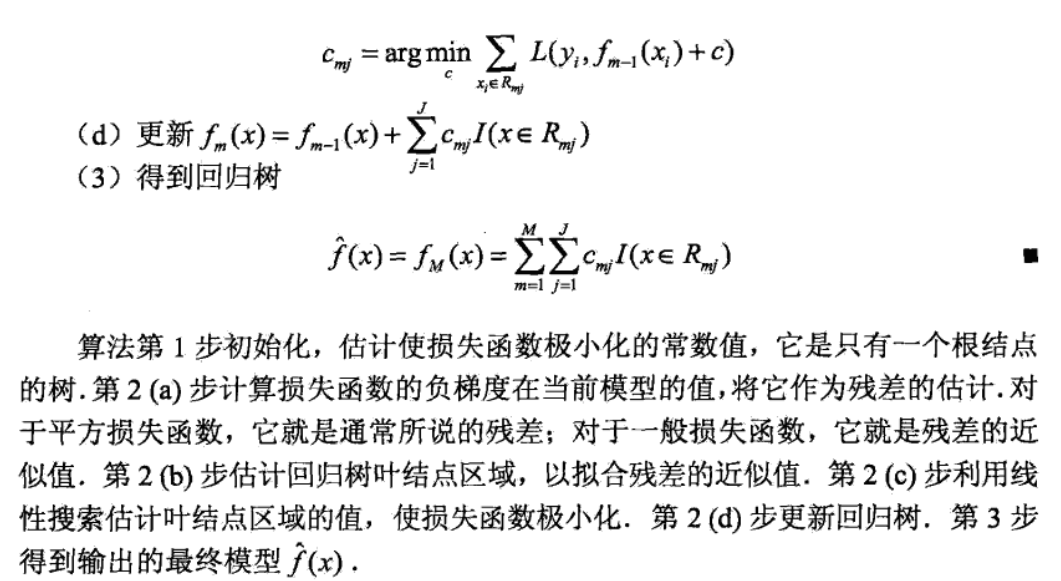


（3）GDBT

（利用损失函数的负梯度在当前模型的值作为回归问题提升树算法中的残差的近似值，拟合一个回归树）

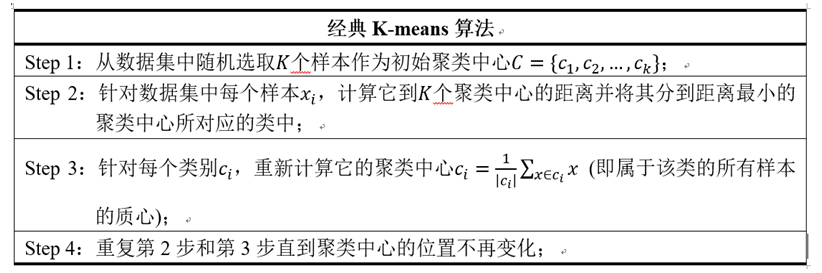
GBDT的基本原理是boosting里面的 boosting tree（提升树），并使用gradient boost。GBDT中的树都是**回归树**，不是**分类树** ，因为gradient boost 需要按照**损失函数**的梯度近似的拟合残差，这样拟合的是连续数值，因此只有回归树。Gradient Boosting是一种Boosting的方法，其与传统的Boosting的区别是，每一次的计算是为了减少上一次的残差(residual)，而为了消除残差，可以在残差减少的梯度(Gradient)方向上建立一个新的模型。所以说，在Gradient Boosting中，每个新的模型的建立是为了使得之前模型的残差往梯度方向减少，与传统Boosting对正确、错误样本进行加权有着很大的区别。这个梯度代表上一轮学习器损失函数对预测值求导。与Boosting Tree的区别：Boosting Tree的适合于损失函数为**平方损失**或者**指数损失**。而Gradient Boosting适合**各类损失函数**（损失函数为：平方损失则相当于Boosting Tree拟合残差、损失函数为：使用指数损失则可以近似于Adaboost，但树是回归树）





**12、聚类**

定义：大量未标注的数据集，按照其内在相似性将数据集划分为多个类别，使类内的数据相似度较大，类间的数据相似度较小。



K-means**与**K-means++**：**原始K-means算法最开始随机选取数据集中K个点作为聚类中心，而K-means++按照如下的思想选取K个聚类中心：假设已经选取了n个初始聚类中心(0<n<K)，则在选取第n+1个聚类中心时：距离当前n个聚类中心越远的点会有更高的概率被选为第n+1个聚类中心。在选取第一个聚类中心(n=1)时同样通过随机的方法。可以说这也符合我们的直觉：聚类中心当然是互相离得越远越好。

**无监督学习和有监督学习的区别**

(1) 有监督学习：对具有标记的训练样本进行学习，尽可能对训练样本集外的数据进行分类预测。（LR、SVM、BP、RF、GDBT）

(2) 对未标记的样本进行训练学习，进而发现这些样本中的结构知识。（KMeans、DL）

**13、卷积神经网络**

对于不同的问题均存在整体与局部的联系，有低层次的特征经过整合，组成高层次的特征，并且得到不同特征之间的空间相似性；CNN可以抓住的共性手段是：局部连接、权值共享、池化操作、多层次结构。局部连接使网络可以提取数据的局部特征，权值共享可以大大降低了网络的训练难度，一个filter只提取一个特征，在整个图片进行卷积；池化操作与多层次结构一起，实现了数据的降维，将低层次的局部特征组合成为较高层次的特征，从而对整个图片进行表示：

14、深度学习的数据集问题

（1）数据集太小

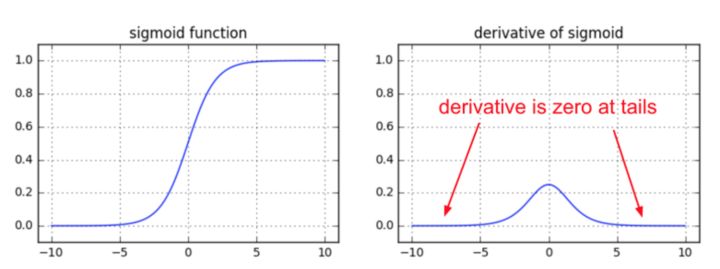
（2）数据集之间没有局部相关性均会导致深度学习效果不好。

15、噪声梯度消失的原因以及影响

神经网络训练过程中，通过改变神经元的权重，使得网路的输出值尽可能的逼近标签以降低误差值，所有普遍采用的算法是BP算法，核心思想：计算输出与标签间的误差损失函数，然后计算每个神经元的梯度，更新权值。

原因：

激活函数将网络的输出值挤压在一个很小的区间内，再激活函数两端交大范围内的梯度为0，从而造成学习停止。

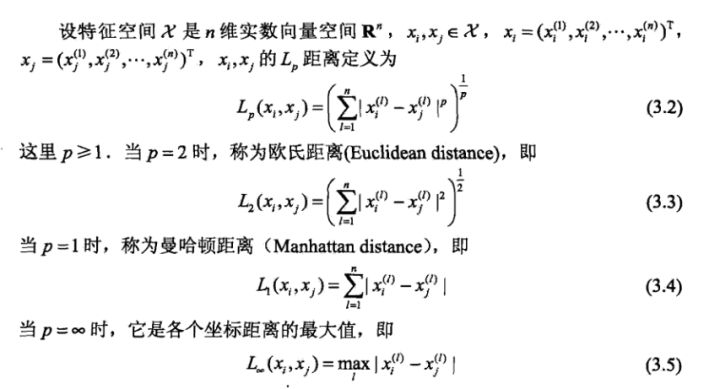


**16、 KNN算法**

K近邻算法是一种基本的分类和回归算法。给定一个训练数据集，训练集中的类别是一定的。分类时，对于新的实例，在训练集中找到该实例最相邻的K个实例，通过多数表决的方式进行预测。

K近邻的三个要素：K值的选择、距离的度量方式、分类决策准则。

距离的话：一般都是欧氏距离、余弦距离、马氏距离。



K的选择：利用交叉验证（Cross Validation）评估一系列不同的k值，选取结果最好的k值作为训练参数。K值要是很小，模型复杂，只有与输入实例相近的实例才起到预测作用，近似误差会减小，估计误差会变大且容易过拟合；K值很大的话，模型简单，减少了估计误差、近似误差会变大，需要较多的实例来确定输入实例的预测值。

KNN算法的伪代码：

1. 计算已知类别数据集中的点与当前点之间的距离；
2. 按照距离递增排序；
3. 选择与当前点距离最小的K个点；
4. 确定前K个点所在类别的出现频率；
5. 返回前K个点出现频率最高的类别作为当前点的预测类别。

备注：

特征归一化的目的：让每个特征都是同等重要的！

实现方式：kd树

**17、决策树学习**

学习目的：从训练数据中归纳出一组分类准则，给定训练数据集构建一个决策树模型，使他能够对实例进行正确的分类。

决策树学习主要用损失函数，正则化的极大似然函数，策略是：最小化损失函数。

选择最优特征，将训练集进行分割成子集，如果这些自己可以被正确分类，那么构造叶节点，并将这些子集分到相应的叶节点中去；如果还有子集为正确分类，继续选择新的最优特征，继续对其进行分割，构建相应的结点，即最后每个自己都被分到每个叶节点上，即都有了明确的类，于是生成了一个决策树。

选择特征的准则：信息增益和信息增益比：

**信息增益：**

熵：随机变量不确定性的度量，熵越大，随机变量的不确定性就越大；

给定某个条件下随机变量Y 的不确定性称为：条件熵。

信息增益：已知特征x的信息，而使得类Y信息的不确定性减少的程度。

经验条件熵与经验熵的差值为信息增益。选择信息增益最大的特征。

**信息增益比**：

定义信息增益g(D,A)与训练数据集H(D)经验熵的比值。

构造决策树的算法：

**ID3**：在决策数各个结点应用信息增益准则选择特征，递归地构建决策树。具体方法是：从根结点开始，计算所有可能特征的信息增益，选择信息增益最大的特征作为结点的特征。由特征的不同取值建立子节点，递归的调用以上方法。构建决策树。直到信息增益很小，或者没有特征可以选择为止。最后便得到决策树。

**C4.5**：

使用信息增益比来选择特征。

**剪枝策略**：

极小化整体的损失函数，

**最大熵模型**：原理是学习概率模型时，在所有可能的概率模型分布中，熵最大的模型是最好的模型，在满足约束条件的模型集合中选取熵最大的模型。

**18、机器学习中，为何要经常对数据做归一化？**

主要是两个原因：（1）归一化后，加快了梯度下降求最优解的速度。（2）归一化后有可能提高精度。

因为一些分类器需要计算样本之间的距离，如欧氏距离，例如KNN。但是假如一个特征值范围非常大，那么距离计算主要取决于这个特征，从而这个特征占很大比例，与实际情况相悖。所以归一化之后可以避免这种情况的出现。

**那些机器学习算法不需要归一化？**

但是一些概率模型不需要归一化，因为他们不关心变量的值，而是关心变量分布和变量之间的条件概率。例如：决策树、rf。一般的树形结构的算法也不需要归一化。

**19、BP神经网络算法（梯度下降算法）**

梯度下降算法是一个最优化算法，简单来讲就是沿着梯度下降的方向求出一个函数的极小值。所求的函数即为目标函数cost function.

在进行梯度下降计算的过程中，首先需要确定步长，即learning rate；

然后给定一个初始的值；

确定一个向下的方向，并按照规定的步长，更新；

当下降到某一个预定义的值时，则停止下降。

梯度下降算法的目标函数为：

，

公式推导过程：

**20、简单讲一下贝叶斯定理**

定义以及相关概念：

条件概率：事件A在另外一个事件B已经发生的条件下的发生概率。条件概率可以表示为，可以读作B条件下A的概率。



联合概率是：两个事件共同发生的概率。A和B共同发生的联合概率表示为：



先验概率：在事件B发生之前，对事件A有一个基本的概率判断，称为A的先验概率，用来表示。

后验概率：事件B发生以后，我们对事件A的发生概率重新评估，称为A的后验概率，用表示。

根据推导可以得到：

21、简单说一下 激活函数sigmoid函数？

常用的激活函数有：sigmoid、tanh、relu.函数，前两个激活函数主要用于全连接层，后面的relu函数常见于卷积层。

Sigmoid函数的表达式为：



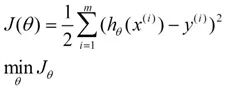
其中z是一个线性组合。输入一个很大的正数或者很小的复数到g(z)函数可知，其结果趋近于1或者0.那么也就是说：sigmoid函数的功能时相当于把一个实数压缩到0到1之间。这样一来，可以把激活函数看成一种分类概率，比如激活函数的输出为0.9的话，便可以解释为90%的概率为正样本。

**21、梯度下降算法**

首先我们输入一个数据，算法会通过一系列的过程得到一个估计函数，这个估计函数可以对输入的新数据进行估计，从而建立了一种新模型。

建立的估计函数为：，其中是要训练的参数，

根据估计函数和数据的标签的差值作为目标函数（误差函数）：



我们的目的就是要最小化目标函数：如何使得目标函数最小，一种常用的方法就是梯度下降法：

其基本过程如下：

（1）首先对赋值，这个值是随机的，也可以让是一个全0的向量。

（2）改变的值，使得按照梯度下降的方向减少。

需要对求偏导：

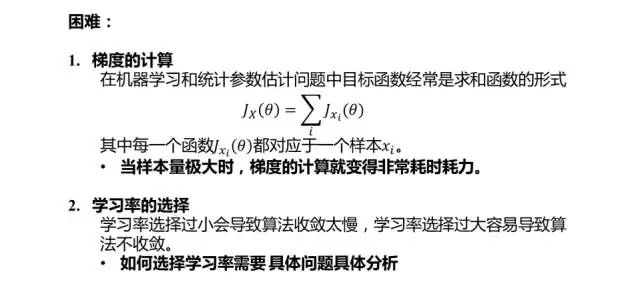
https://pic1.zhimg.com/80/v2-b85b48fcdab8325df0f0f46b509e5985_hd.jpg

向着梯度最小的方向减少，表示更新之后的数值，-号后面的表示按照梯度方向减少的量，表示步长，即每次按照梯度方向减少的方向变化量。

https://pic3.zhimg.com/80/v2-b6c7acb857d391958bfb66622765205c_hd.jpg

梯度下降算法最终点并非是全局嘴有点，极有可能是一个局部最优点。梯度下降算法并不是下降最快的方向，他只是目标函数在当前点的切平面最快的方向。

梯度下降算法需要解决的问题：



22、激活函数的作用与分类

（1）作用：

让神经元非线性化

让神经元的输出变到0-1之间。

（2）激活函数类别：

Sigmoid函数：

Tanh函数：

Relu函数：

22、深度神经网络

DNN是包含多个隐层的神经网络，其中MLP是最简单的DNN，最突出的特点就是：网络有很多层（原则上超过两层就可以成为深层了）。

23、主成分分析

数据降维，用数据中最主要的部分来代替原始数据，假如数据一共有n维，需要降到m维，

主要算法步骤：

（1）计算数据点的均值

（2）计算数据样本的协方差矩阵

（3）对协方差矩阵进行特征值分解

（4）选择最大的m个特征值对应的特征向量，组成特征矩阵w

（5）对数据集中的每个样本进行转化成为新的藤本，得到输出新的数据集。

PCA算法的优点：

1、仅仅需要以方差衡量信息量，不收数据集以外的因素影响；

2、各主成分之间正交，可以消除原始数据成分间的相互影响

3、计算简单，主要运算是特征值分解，易于实现。

缺点：

1、主成分各个维度的含义具有模糊性，不如原始样本特征的解释性强

2、方差小的非主成分可能含有对样本差异的重要信息，因此降维丢弃可能会对后续数据有影响