**目录**

目录

[1.梯度](#_Toc2930_WPSOffice_Level1) [2](#_Toc2930_WPSOffice_Level1)

[1.1.梯度的概念](#_Toc6055_WPSOffice_Level2) [2](#_Toc6055_WPSOffice_Level2)

[1.2.梯度下降算法的实现](#_Toc16427_WPSOffice_Level2) [2](#_Toc16427_WPSOffice_Level2)

[1.3.梯度下降树](#_Toc21947_WPSOffice_Level2) [4](#_Toc21947_WPSOffice_Level2)

[2.决策树](#_Toc6055_WPSOffice_Level1) [5](#_Toc6055_WPSOffice_Level1)

[2.1.ID3决策树学习算法](#_Toc24483_WPSOffice_Level2) [5](#_Toc24483_WPSOffice_Level2)

[2.2.C4.5决策树算法](#_Toc14932_WPSOffice_Level2) [5](#_Toc14932_WPSOffice_Level2)

[2.3.基尼指数CART决策树算法](#_Toc30843_WPSOffice_Level2) [6](#_Toc30843_WPSOffice_Level2)

[2.4.剪枝处理](#_Toc27909_WPSOffice_Level2) [6](#_Toc27909_WPSOffice_Level2)

[2.4.1.预剪枝](#_Toc6055_WPSOffice_Level3) [6](#_Toc6055_WPSOffice_Level3)

[2.4.2.后剪枝](#_Toc16427_WPSOffice_Level3) [6](#_Toc16427_WPSOffice_Level3)

[3.偏差与方差](#_Toc16427_WPSOffice_Level1) [6](#_Toc16427_WPSOffice_Level1)

[3.1.导致偏差和方差的原因](#_Toc30040_WPSOffice_Level2) [7](#_Toc30040_WPSOffice_Level2)

[3.2.深度学习中的偏差与方差](#_Toc4543_WPSOffice_Level2) [7](#_Toc4543_WPSOffice_Level2)

[4.生成模型与判别模型](#_Toc21947_WPSOffice_Level1) [7](#_Toc21947_WPSOffice_Level1)

[4.1.两者之间的联系](#_Toc9945_WPSOffice_Level2) [8](#_Toc9945_WPSOffice_Level2)

[4.2.常见模型](#_Toc18763_WPSOffice_Level2) [8](#_Toc18763_WPSOffice_Level2)

[5.先验概率与后验概率](#_Toc24483_WPSOffice_Level1) [8](#_Toc24483_WPSOffice_Level1)

[6.Bagging vs Boosting](#_Toc14932_WPSOffice_Level1) [9](#_Toc14932_WPSOffice_Level1)

[7.信息熵、KL 散度（相对熵）与交叉熵](#_Toc30843_WPSOffice_Level1) [9](#_Toc30843_WPSOffice_Level1)

[8. 一个完整的机器学习项目流程](#_Toc27909_WPSOffice_Level1) [10](#_Toc27909_WPSOffice_Level1)

**1.梯度**

**1.1.梯度的概念**

梯度是微积分中一个很重要的概念，梯度的意义如下：在单变量的函数中，梯度其实就是函数的微分，代表着函数在某个给定点的切线的斜率, 在多变量函数中，梯度是一个向量，向量有方向，梯度的方向就指出了函数在给定点的上升最快的方向

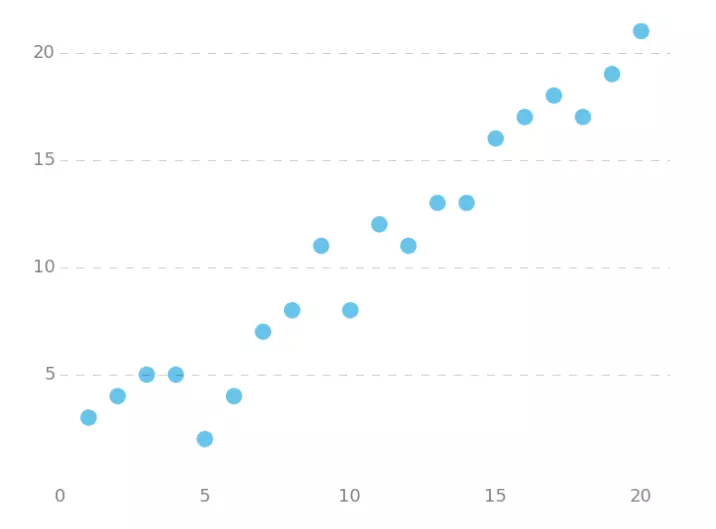


此公式的意义是：J是关于Θ的一个函数，我们当前所处的位置为Θ0点，要从这个点走到J的最小值点，也就是山底。首先我们先确定前进的方向，也就是梯度的反向，然后走一段距离的步长，也就是α，走完这个段步长，就到达了Θ1这个点！

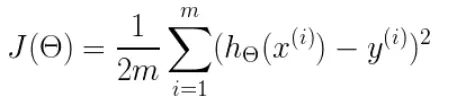
为什么要梯度要乘以一个负号？  
梯度前加一个负号，就意味着朝着梯度相反的方向前进！我们在前文提到，梯度的方向实际就是函数在此点上升最快的方向！而我们需要朝着下降最快的方向走，自然就是负的梯度的方向，所以此处需要加上负号

# 1.2.梯度下降算法的实现

下面我们将用python实现一个简单的梯度下降算法。场景是一个简单的[线性回归](https://link.jianshu.com/?t=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FLinear_regression" \t "_blank)的例子：假设现在我们有一系列的点，如下图所示

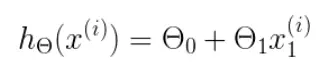


我们将用梯度下降法来拟合出这条直线！首先，我们需要定义一个代价函数，在此我们选用[均方误差代价函数](https://link.jianshu.com/?t=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FLeast_squares" \t "_blank)

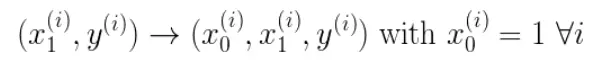


此公式中

* m是数据集中点的个数
* ½是一个常量，这样是为了在求梯度的时候，二次方乘下来就和这里的½抵消了，自然就没有多余的常数系数，方便后续的计算，同时对结果不会有影响
* y 是数据集中每个点的真实y坐标的值
* h 是我们的预测函数，根据每一个输入x，根据Θ 计算得到预测的y值，即



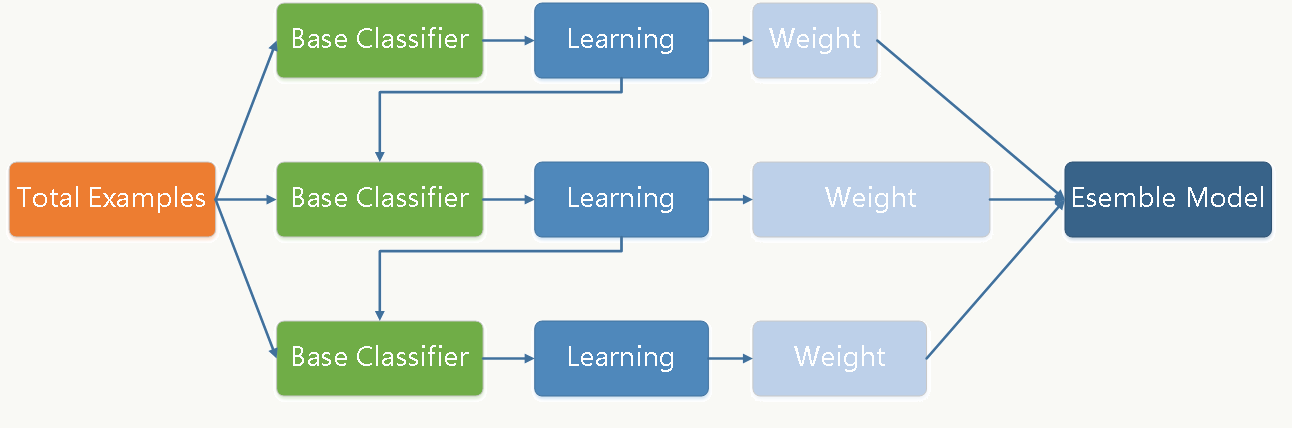
我们有两个变量，为了对这个公式进行矩阵化，我们可以给每一个点x增加一维，这一维的值固定为1，这一维将会乘到Θ0上。这样就方便我们统一矩阵化的计算



**1.3.梯度下降树**

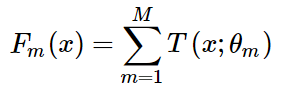
gbdt全称梯度下降树，在传统机器学习算法里面是对真实分布拟合的最好的几种算法之一，在前几年深度学习还没有大行其道之前，gbdt在各种竞赛是大放异彩。原因大概有几个，一是效果确实挺不错。二是即可以用于分类也可以用于回归。三是可以筛选特征。

**gbdt的训练过程**

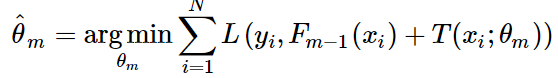


gbdt通过多轮迭代,每轮迭代产生一个弱分类器，每个分类器在上一轮分类器的**残差**基础上进行训练。对弱分类器的要求一般是足够简单，并且是低方差和高偏差的。因为训练的过程是通过降低偏差来不断提高最终分类器的精度

弱分类器一般会选择为CART TREE（也就是分类回归树）。由于上述高偏差和简单的要求 每个分类回归树的深度不会很深。最终的总分类器 是将每轮训练得到的弱分类器加权求和得到的（也就是加法模型）



 模型一共训练M轮，每轮产生一个弱分类器 T(x;θm)。弱分类器的**损失函数**



 Fm−1(x)为当前的模型，gbdt 通过经验风险极小化来确定下一个弱分类器的参数。具体到损失函数本身的选择也就是L的选择，有平方损失函数，0-1损失函数，对数损失函数等等。如果我们选择平方损失函数，那么这个差值其实就是我们平常所说的残差。

下面我们具体来说CART TREE(是一种二叉树) 如何生成。CART TREE 生成的过程其实就是一个选择特征的过程。假设我们目前总共有 M 个特征。第一步我们需要从中选择出一个特征 j，做为二叉树的第一个节点。然后对特征 j 的值选择一个切分点 m. 一个 样本的特征j的值 如果小于m，则分为一类，如果大于m,则分为另外一类。如此便构建了CART 树的一个节点。其他节点的生成过程和这个是一样的。现在的问题是在每轮迭代的时候，如何选择这个特征 j,以及如何选择特征 j 的切分点 m:

原始的gbdt的做法非常的暴力，首先遍历每个特征，然后对每个特征遍历它所有可能的切分点，找到最优特征 m 的最优切分点 j。

**2.决策树**

**2.1.ID3决策树学习算法**

以信息增益为准则来选择划分属性（关于得病与否为例子）在14个样本中，yes=9个，no=5个，此时信息熵=-(9/14\*log(9/14)+5/14\*log(5/14))=0.940286，值得注意的是这里的对数是以2为底，下面随机选择一个属性比如年龄作为划分依据，分成<30,31~40,40>这三组，这三组中yes和no各占不同的数目，计算方式跟上面一样，得到0.97095，0，0.81这三个信息熵，然后成权重后相加得到0.57856，这跟没划分之前的信息熵相差0.361726，这就是信息增益的量。

对其它的属性我们也可以做同样的计算，把得到最大的信息增益的作为最优划分属性，这就是ID3算法选择划分属性的准则。

**2.2.C4.5决策树算法**

信息增益准则对可取值数目较多的属性有所偏好，为了减少这种偏好可能带来的不利影响，C4.5决策树算法不直接使用信息增益，而是使用增益率来选择最优划分属性。

这相当于在上面例子中的信息增益0.57856的基础上再除以-(5/14\*log(5/14)+4/14\*log(4/14)+5/14\*log(5/14))，也就是对属性值d 的权重再求个熵，相除后作为最后的增益率。

## **2.3.[基尼](https://www.baidu.com/s?wd=%E5%9F%BA%E5%B0%BC&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "https://blog.csdn.net/huacha__/article/details/_blank)指数**CART决策树算法

CART决策树算法使用基尼指数来选择划分属性，这个准则不再用熵和增益来衡量数据集的纯度，而是用基尼值来度量。直观来说这个值反映了从数据集中随机抽取两个样本，其类别标记不一致的概率。因此基尼值越小，则数据集的纯度越高。

# 2.4.**剪枝处理**

剪枝是决策树算法对付“过拟合”的主要手段，通过主动去掉一些分支来****降低过拟合的风险****。根据****泛化性能****是否提升来判断和评估要不要留下这个分支。

那么，如何判断决策树的泛化性能是否提升呢？可以采用“留出法”（一种性能评估方法），即预留一部分数据用作“验证集”来进行性能评估。

## **2.4.1.预剪枝**

是否应该进行这个划分？预剪枝要对****划分前和划分后****的泛化性能进行估计。划分后的验证集精度如果比划分前的验证集精度高，那么，就决定划分，否则，预剪枝策略禁止这个结点被划分。

优点：预剪枝使得决策树的很多分支都没有展开，这不仅降低来过拟合的风险，还显著减少了决策树的训练时间开销和测试时间的开销。

缺点：预剪枝基于“贪心”的本质，禁止这些分支展开，给预剪枝决策树带来来欠拟合的风险。

## **2.4.2.后剪枝**

后剪枝先从训练集上生成一棵完整的树，从最后一个分支来判断要不要剪枝。它的训练时间开销比未剪枝决策树和预剪枝决策树都要大的多。

## **3.偏差与方差**

## ****偏差****与****方差****分别是用于衡量一个模型****泛化误差****的两个方面；

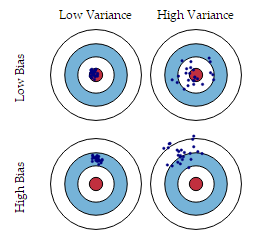
模型的****偏差****，指的是模型预测的****期望值****与****真实值****之间的差；

模型的****方差****，指的是模型预测的****期望值****与****预测值****之间的差平方和；

在****监督学习****中，模型的****泛化误差****可****分解****为偏差、方差与噪声之和



****偏差****用于描述模型的****拟合能力****；  
****方差****用于描述模型的****稳定性****。



**3.1.导致偏差和方差的原因**

* ****偏差****通常是由于我们对学习算法做了****错误的假设****，或者模型的复杂度不够；
* 比如真实模型是一个二次函数，而我们假设模型为一次函数，这就会导致偏差的增大（欠拟合）；****由偏差引起的误差****通常在****训练误差****上就能体现，或者说训练误差主要是由偏差造成的
* ****方差****通常是由于****模型的复杂度相对于训练集过高****导致的；比如真实模型是一个简单的二次函数，而我们假设模型是一个高次函数，这就会导致方差的增大（过拟合）；****由方差引起的误差****通常体现在测试误差相对训练误差的****增量****上。

### **3.2.深度学习中的偏差与方差**

神经网络的拟合能力非常强，因此它的****训练误差****（偏差）通常较小；但是过强的拟合能力会导致较大的方差，使模型的测试误差（****泛化误差****）增大；因此深度学习的核心工作之一就是研究如何降低模型的泛化误差，这类方法统称为****正则化方法****。

**4.生成模型与判别模型**

《统计学习方法》 1.7 生成模型与判别模型

监督学习的任务是学习一个模型，对给定的输入预测相应的输出，这个模型的一般形式为一个****决策函数****或一个****条件概率分布****（后验概率）：

****决策函数****：输入 X 返回 Y；其中 Y 与一个****阈值****比较，然后根据比较结果判定 X 的类别

****条件概率分布****：输入 X 返回 ****X 属于每个类别的概率****；将其中概率最大的作为 X 所属的类别

监督学习模型可分为****生成模型****与****判别模型：****

****判别模型****直接学习决策函数或者条件概率分布，直观来说，****判别模型****学习的是类别之间的最优分隔面，反映的是不同类数据之间的差异

****生成模型****学习的是联合概率分布P(X,Y)，然后根据条件概率公式计算 P(Y|X)

****4.1.两者之间的联系****

由生成模型可以得到判别模型，但由判别模型得不到生成模型。

当存在“****隐变量****”时，只能使用****生成模型****

隐变量：当我们找不到引起某一现象的原因时，就把这个在起作用，但无法确定的因素，叫“隐变量”

****4.2.常见模型****

判别模型 : K 近邻、感知机（神经网络）、决策树、逻辑斯蒂回归、****最大熵模型****、SVM、提升方法、****条件随机场****

生成模型 : 朴素贝叶斯、隐马尔可夫模型、混合高斯模型、贝叶斯网络、马尔可夫随机场

## **5.先验概率与后验概率**

****条件概率****（似然概率）

一个事件发生后另一个事件发生的概率。一般的形式为 P(X|Y)，表示 y 发生的条件下 x 发生的概率。有时为了区分一般意义上的****条件概率****，也称****似然概率****

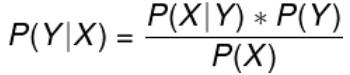
****先验概率****

事件发生前的预判概率,可以是基于历史数据的统计，可以由背景常识得出，也可以是人的主观观点给出。一般都是****单独事件****发生的概率，如 P(A)、P(B)。

****后验概率****

基于先验概率求得的****反向条件概率****，形式上与条件概率相同（若 P(X|Y) 为正向，则 P(Y|X) 为反向）

****贝叶斯公式****



****6.Bagging vs Boosting****

**在Bagging方法中，b个学习器之间彼此是相互独立的，这样的特点使得Bagging方法更容易并行。与Bagging方法不同，在Boosting算法中，学习器之间是存在先后顺序的，同时，每一个样本是有权重的，初始时，每一个样本的权重是相等的。首先，第1个学习器对训练样本进行学习，当学习完成后，增大错误样本的权重，同时减小正确样本的权重，再利用第2个学习器对其进行学习，依次进行下去，最终得到b个学习器，最终，合并这b个学习器的结果。**

# **7.信息熵、KL 散度（相对熵）与交叉熵**

**7.1自信息(self-information)**: H(Xi) = − logP(Xi)

信息论的基本想法是一个不太可能的事件居然发生了，要比一个非常可能的事件发生，能提供更多的信息。

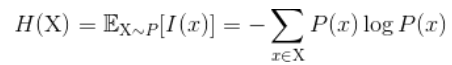
该想法可描述为以下性质：

1. 非常可能发生的事件信息量要比较少，并且极端情况下，确保能够发生的事件应该没有信息量。
2. 比较不可能发生的事件具有更高的信息量。
3. 独立事件应具有增量的信息。例如，投掷的硬币两次正面朝上传递的信息量，应该是投掷一次硬币正面朝上的信息量的两倍。

　　例题：若估计在一次国际象棋比赛中谢军获得冠军的可能性为0.1(记为事件A)，而在另一次国际象棋比赛中她得到冠军的可能性为0.9(记为事件B)。试分别计算当你得知她获得冠军时，从这两个事件中获得的信息量各为多少？

　　　　　　H(A)=-log2 P(0.1)≈3.32(比特)

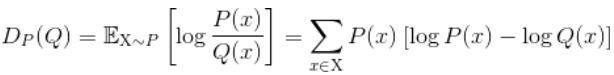
　　　　　　H(B)=-log2 P(0.9)≈0.152(比特)

**7.2信息熵（Information-entropy）**对整个概率分布中的不确定性总量进行量化: 

信息熵也称香农熵（Shannon entropy）

## 7.3**相对熵（KL 散度）与交叉熵**

P 对 Q 的 ****KL散度**(**Kullback-Leibler divergence)：



****交叉熵**（cross-entropy）：**

IMG_256

## **一个完整的机器学习项目流程**

Source: https://github.com/geekcircle/machine-learning-interview-qa/blob/master/questions/2.md

**① 数学抽象：**

明确问题是进行机器学习的第一步。机器学习的训练过程通常都是一件非常耗时的事情，胡乱尝试时间成本是非常高的。这里的抽象成数学问题，指的是根据数据明确任务目标，是分类、还是回归，或者是聚类。

**②数据获取：**

数据决定了机器学习结果的上限，而算法只是尽可能逼近这个上限。数据要有代表性，否则必然会过拟合。对于分类问题，数据偏斜不能过于严重（平衡），不同类别的数据数量不要有数个数量级的差距。

对数据的量级要有一个评估，多少个样本，多少个特征，据此估算出内存需求。如果放不下就得考虑改进算法或者使用一些降维技巧，或者采用分布式计算。

**③预处理与特征选择：**

良好的数据要能够提取出良好的特征才能真正发挥效力。

预处理/数据清洗是很关键的步骤，往往能够使得算法的效果和性能得到显著提高。归一化、离散化、因子化、缺失值处理、去除共线性等，数据挖掘过程中很多时间就花在它们上面。这些工作简单可复制，收益稳定可预期，是机器学习的基础必备步骤。

筛选出显著特征、摒弃非显著特征，需要机器学习工程师反复理解业务。这对很多结果有决定性的影响。特征选择好了，非常简单的算法也能得出良好、稳定的结果。这需要运用特征有效性分析的相关技术，如相关系数、卡方检验、平均互信息、条件熵、后验概率、逻辑回归权重等方法。

**④模型训练与调优：**

直到这一步才用到算法进行训练。

现在很多算法都能够封装成黑盒使用。但是真正考验水平的是调整这些算法的（超）参数，使得结果变得更加优良。这需要我们对算法的原理有深入的理解。理解越深入，就越能发现问题的症结，提出良好的调优方案。

**⑤模型诊断：**

如何确定模型调优的方向与思路呢？这就需要对模型进行诊断的技术。

过拟合、欠拟合 判断是模型诊断中至关重要的一步。常见的方法如交叉验证，绘制学习曲线等。过拟合的基本调优思路是增加数据量，降低模型复杂度。欠拟合的基本调优思路是提高特征数量和质量，增加模型复杂度。

误差分析也是机器学习至关重要的步骤。通过观察误差样本，全面分析误差产生误差的原因:是参数的问题还是算法选择的问题，是特征的问题还是数据本身的问题......

诊断后的模型需要进行调优，调优后的新模型需要重新进行诊断，这是一个反复迭代不断逼近的过程，需要不断地尝试， 进而达到最优状态。

**⑥模型融合/集成：**

一般来说，模型融合后都能使得效果有一定提升。而且效果很好。

工程上，主要提升算法准确度的方法是分别在模型的前端（特征清洗和预处理，不同的采样模式）与后端（模型融合）上下功夫。因为他们比较标准可复制，效果比较稳定。而直接调参的工作不会很多，毕竟大量数据训练起来太慢了，而且效果难以保证。

**⑦上线运行：**

这一部分内容主要跟工程实现的相关性更大。工程上是结果导向，模型在线上运行的效果直接决定模型的成败。不单纯包括其准确程度、误差等情况，还包括其运行的速度(时间复杂度)、资源消耗程度（空间复杂度）、稳定性是否可接受。

这些工作流程主要是工程实践上总结出的一些经验。并不是每个项目都包含完整的一个流程。这里的部分只是一个指导性的说明，只有多实践，多积累项目经验，才会有自己更深刻的认识。