

2019年2月						
日	一	二	三	四	五	六
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	1	2
3	4	5	6	7	8	9

公告



昵称: Excalibur

园龄: 5年6个月

粉丝: 10

关注: 16

+加关注

搜索

常用链接

[我的随笔](#)
[我的评论](#)
[我的参与](#)
[最新评论](#)
[我的标签](#)

我的标签

[贪心\(2\)](#)
[字符串\(1\)](#)
[数据结构\(1\)](#)
[数学\(1\)](#)

随笔分类(102)

[Algorithm\(6\)](#)
[Android\(1\)](#)
[c/c++\(15\)](#)
[cf&etc\(5\)](#)
[CSAPP](#)
[JAVA\(7\)](#)
[Linux\(3\)](#)
[Machine Learning\(9\)](#)
[刷题&LeetCode\(55\)](#)
[优化方法\(1\)](#)

随笔档案(99)

[2018年12月 \(1\)](#)
[2018年8月 \(3\)](#)
[2018年7月 \(14\)](#)
[2018年6月 \(3\)](#)
[2018年5月 \(9\)](#)
[2018年4月 \(11\)](#)
[2018年3月 \(8\)](#)
[2018年2月 \(3\)](#)
[2018年1月 \(1\)](#)
[2017年12月 \(6\)](#)
[2017年11月 \(4\)](#)
[2017年9月 \(3\)](#)
[2017年8月 \(2\)](#)
[2017年4月 \(2\)](#)
[2017年3月 \(2\)](#)
[2017年2月 \(2\)](#)
[2016年11月 \(1\)](#)
[2016年10月 \(10\)](#)

机器学习笔记——最小二乘法

一. 简介

首先来看百度百科对最小二乘法的介绍: 最小二乘法 (又称最小方法) 是一种数学优化技术。它通过最小化误差的平方和寻找数据的最佳函数匹配。利用最小二乘法可以简便地求得未知的数据, 并使得这些求得的数据与实际数据之间误差的平方和为最小。最小二乘法还可用于曲线拟合。其他一些优化问题也可通过最小化能量或最大化熵用最小二乘法来表达。

简而言之, 最小二乘法同梯度下降类似, **都是一种求解无约束最优化问题的常用方法**, 并且也可以用于曲线拟合, 来解决回归问题。**最小二乘法实质就是最小化“均方误差”, 而均方误差就是残差平方和的1/m(m为样本数), 同时均方误差也是回归任务中最常用的性能度量。**

二. 对于一元线性模型

如果以最简单的一元线性模型来解释最小二乘法。回归分析中, 如果只包括一个自变量和一个因变量, 且二者的关系可用一条直线近似表示, 这种回归分析称为一元线性回归分析。如果回归分析中包括两个或两个以上的自变量, 且因变量和自变量之间是线性关系, 则称为多元线性回归分析。对于二维空间线性是一条直线; 对于三维空间线性是一个平面, 对于多维空间线性是一个超平面...

对于一元线性回归模型, 假设从总体中获取了m组观察值 $(X_1, Y_1), (X_2, Y_2), \dots, (X_m, Y_m)$ 。对于平面中的这m个点, 可以使用无数条曲线来拟合。要求样本回归函数尽可能好地拟合这组值。综合起来看, 这条直线处于样本数据的中心位置最合理。选择最佳拟合曲线的标准可以确定为: 使总的拟合误差 (即总残差) 达到最小。有以下三个标准可以选择:

(1) 用“残差和最小”确定直线位置是一个途径。但可能会出现计算“残差和”存在相互抵消的问题。

(2) 用“残差绝对值和最小”确定直线位置也是一个途径。但绝对值的计算比较麻烦。

(3) 最小二乘法的原则是以“残差平方和最小”确定直线位置。用最小二乘法除了计算比较方便外, 得到的估计量还具有优良特性。这种方法对异常值非常敏感。

最常用的是普通最小二乘法 (Ordinary Least Square, OLS): 所选择的回归模型应该使所有观察值的残差平方和达到最小。

在讲最小二乘的详情之前, 首先明确两点: 1. 我们假设在测量系统中不存在有系统误差, 只存在有纯偶然误差。比如体重计或者身高计本身有问题, 测量出来的数据都偏大或者都偏小, 这种误差是绝对不存在的。(或者说这不能叫误差, 这叫错误) 2. 误差是符合正态分布的, 因此最后误差的均值为0 (这一点很重要)。

明确了上面两点以后, 重点来了: 为了计算 β_0, β_1 的值, 我们采取如下规则: β_0, β_1 应该使计算出来的函数曲线与观察值的差的平方和最小。用数学公式描述就是:

$$Q = \min \sum_{i=1}^n (y_{ie} - y_i)^2$$

其中, y_{ie} 表示根据 $y = \beta_0 + \beta_1 x$ 估算出来的值, y_i 是观察得到的真实值。

为什么要用残差的平方和最小? 用差的绝对值不行么?

以下是一个相对靠谱的解释:

我们假设直线对于坐标 X_i 给出的预测 $f(X_i)$ 是最靠谱的预测, 所有纵坐标偏离 $f(X_i)$ 的那些数据点都含有噪音, 是噪音使得它们偏离了完美的一条直线, 一个合理的假设就是偏离路线越远的概率越小, 具体小多少, 可以用一个正态分布曲线来模拟, 这个分布曲线以直线对 X_i 给出的预测 $f(X_i)$ 为中心, 实际纵坐标为 Y_i 的点 (X_i, Y_i) 发生的概率就正比于 $\exp[-(\Delta Y_i)^2]$ 。 ($\exp(\dots)$ 代表以常数 e 为底的多少次方)。

所以我们在前面的两点里提到, 假设误差的分布要为一个正态分布, 原因就在这里了。

另外说一点我自己的理解: 从数学处理的角度来说, 绝对值的数学处理过程, 比平方和的处理要复杂很多。搞过机器学习的同学都知道, L1正则就是绝对值的方式, 而L2正则则是平方和的形式。L1能产生稀疏的特征, 这对大规模的机器学习灰常灰常重要。但是L1的求解过程, 实在是太过蛋疼。所以即使L1能产生稀疏特征, 不到万不得已, 我们也还是宁可L2正则, 因为L2正则计算起来方便得多。。。

明确了前面的cost function以后, 后面的优化求解过程反倒变得容易了。

样本的回归模型很容易得出:

$$Q = \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x)^2$$

2016年9月 (2) 2015年6月 (1) 2015年4月 (1) 2015年3月 (1) 2015年1月 (1) 2014年6月 (1) 2014年5月 (2) 2014年4月 (2) 2013年12月 (1) 2013年11月 (1) 2013年10月 (1)	
文章分类 (30) Alogrithm(1) c#(1) java(7) linux(7) other(3) python(8) sql(2) 计算机网络(1)	
相册 (4) ML(1) python3.x(3)	
积分与排名 积分 - 58707 排名 - 8573	
最新评论 1. Re:机器学习笔记——测试集和验证集的区别 简而言之就是: 验证集 核对的是 模型可训练参数的泛化能力 测试集 核对的是 模型超参数的泛化能力 --CrazyNong 2. Re:寻找数组中第K大的数 简单清晰的思路和描述。 --柳树人 3. Re:2018年美团春招(第二批)题解 写的很好。膜拜大佬 --flyrainkey 4. Re:并查集的简介 不错,写的挺清晰! --dookjen 5. Re:c++中ifstream读文件的问题(关于eof()) 缺了个unget();不然第一个的name会少一个字符(被c=infile.get()吃掉了) 后面的c=infile.get()没出问题是 因为吃了 \n --long_ao_tian	
阅读排行榜 1. 机器学习笔记——测试集和验证集的区别(18465) 2. 机器学习笔记——最小二乘法(10446) 3. java中向JTextArea中添加滚动条(垂直的和水平的)(8737) 4. C++STL中的unique函数解析(7169) 5. c++中ifstream读文件的问题(关于eof())(3627) 6. java.util.ConcurrentModification的解决办法(2856) 7. java中对HashMap遍历的方式(2645) 8. 浅谈对java中传参问题的理解(2446) 9. 寻找数组中第K大的数(1897) 10. 机器学习笔记——简述坐标下降法(1564)	
评论排行榜 1. c++中ifstream读文件的问题(关于eof())(1)	

现在需要确定β0、β1，使cost function最小。学过高数的同志们都清楚，求导就OK。对于这种形式的函数求导，根据数学知识我们知道，函数的极值点为偏导为0的点。

$$\frac{\partial Q}{\partial \beta_0} = 2 \sum_i^n (y_i - \beta_0 - \beta_1 x_i)(-1) = 0$$

$$\frac{\partial Q}{\partial \beta_1} = 2 \sum_i^n (y_i - \beta_0 - \beta_1 x_i)(-x_i) = 0$$

将这两个方程稍微整理一下，使用克莱姆法则，很容易求解得出：

$$\beta_0 = \frac{\sum x_i^2 \sum y_i - \sum x_i \sum x_i y_i}{n \sum x_i^2 - (\sum x_i)^2}$$

$$\beta_1 = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{n \sum x_i^2 - (\sum x_i)^2}$$

这就是最小二乘法的解法，就是求得平方损失函数的极值点。**需要注意的一点是β0是常数项对应的系数，此处相当于添加了一个特征值x0且x0恒为1，也就是目标函数中的β0可以看成β0x0,这样的话就不同单独考虑常数项了(在后面的多元线性模型就用到了该性质)。**

三. 对于多元线性模型

如果我们推广到更一般的情况，假如有更多的模型变量x1,x2,⋯,xn，可以用线性函数表示如下：

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + ... + \beta_n x_n$$

对于m个样本来说，可以用如下线性方程组表示：

$$\begin{matrix} \beta_0 + \beta_1 x_{11} + \beta_2 x_{12} + \beta_3 x_{13} + ... + \beta_n x_{1n} = y_1 \\ \beta_0 + \beta_1 x_{21} + \beta_2 x_{22} + \beta_3 x_{23} + ... + \beta_n x_{2n} = y_2 \\ ... \\ \beta_0 + \beta_1 x_{m1} + \beta_2 x_{m2} + \beta_3 x_{m3} + ... + \beta_n x_{mn} = y_m \end{matrix}$$

如果将样本矩阵x_{ij}记为矩阵A,将参数矩阵记为向量β，真实值记为向量Y，上述线性方程组可以表示为：

$$\begin{pmatrix} 1 & x_{11} & x_{12} & x_{13} & \cdots & x_{1n} \\ 1 & x_{21} & x_{22} & x_{23} & \cdots & x_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & x_{m1} & x_{m2} & x_{m3} & \cdots & x_{mn} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$$

$$\text{即} A\beta = Y$$

对于最小二乘来说，最终的矩阵表达形式可以表示为：

$$\min \| A\beta - Y \|_2^2$$

$$A \in R^{m*(n+1)}, \beta \in R^{(n+1)*1}, Y \in R^{m*1}$$

其中m≥n,由于考虑到了常数项，故属性值个数由n变为n+1。

关于这个方程的解法，具体如下：

2. 机器学习笔记——测试集和验证集的区别(1)
3. 并查集的简介(1)
4. 2018年美团春招(第二批)题解(1)
5. 寻找数组中第K大的数(1)

推荐排行榜

1. 机器学习笔记——测试集和验证集的区别(7)
2. 机器学习笔记——最小二乘法(6)
3. 局部变量&&malloc函数&&生命周期的一些见解(2)
4. 对C语言中malloc和free函数的理解(1)
5. 并查集的简介(1)
6. 全排列小结(1)
7. 利用二分法和牛顿法开根号(1)
8. LeetCode——4. Median of Two Sorted Arrays(1)
9. 0-1背包简述(1)
10. 机器学习笔记——逻辑回归(对数几率回归)和朴素贝叶斯分类器的对比(1)

$$\begin{aligned}
 & \|A\beta - Y\|_2^2 \\
 &= (A\beta - Y)^T (A\beta - Y) \\
 &= (\beta^T A^T - Y^T)(A\beta - Y) \\
 &= \beta^T A^T A\beta - \beta^T A^T Y - Y^T A\beta + Y^T Y \\
 &= \beta^T A^T A\beta - 2\beta^T A^T Y + Y^T Y
 \end{aligned}$$

其中倒数第二行中的中间两项为标量，所以二者相等。然后利用该式对向量 β 求导：

$$\begin{aligned}
 & \frac{\partial(\beta^T A^T A\beta - 2\beta^T A^T Y + Y^T Y)}{\partial\beta} \\
 &= \frac{\partial(\beta^T A^T A\beta - 2\beta^T A^T Y)}{\partial\beta} \\
 &= \frac{\partial(\beta^T A^T A\beta)}{\partial\beta} - 2A^T Y
 \end{aligned} \tag{1}$$

由矩阵的求导法则：

、向量积对列向量 x 求导运算法则

$$\frac{d(\mathbf{u}^T \mathbf{v})}{d\mathbf{x}} = \frac{d(\mathbf{u}^T)}{d\mathbf{x}} \cdot \mathbf{v} + \frac{d(\mathbf{v}^T)}{d\mathbf{x}} \cdot \mathbf{u}$$

结论：

$$\begin{aligned}
 \frac{d(\mathbf{x}^T \mathbf{x})}{d\mathbf{x}} &= \frac{d(\mathbf{x}^T)}{d\mathbf{x}} \cdot \mathbf{x} + \frac{d(\mathbf{x}^T)}{d\mathbf{x}} \cdot \mathbf{x} = 2\mathbf{x} \\
 \frac{d(\mathbf{x}^T \mathbf{A} \mathbf{x})}{d\mathbf{x}} &= \frac{d(\mathbf{x}^T)}{d\mathbf{x}} \cdot \mathbf{A} \mathbf{x} + \frac{d(\mathbf{x}^T \mathbf{A}^T)}{d\mathbf{x}} \cdot \mathbf{x} = (\mathbf{A} + \mathbf{A}^T) \mathbf{x}
 \end{aligned}$$

可知(1)式的结果为：

$$\begin{aligned}
 & \frac{\partial(\beta^T A^T A\beta)}{\partial\beta} - 2A^T Y \\
 &= (A^T A + A^T A)\beta - 2A^T Y \\
 &= 2(A^T A\beta - A^T Y)
 \end{aligned}$$

令上式结果等于0可得：

$$\begin{aligned}
 & (A^T A)\beta = A^T Y \\
 & \Leftrightarrow \beta = (A^T A)^{-1} A^T Y
 \end{aligned} \tag{2}$$

上式就是最小二乘法的解析解，它是一个全局最优解。

四. 其他一些想法

1. 最小二乘法和梯度下降

乍一看 β 的最终结果，感觉很面熟，仔细一看，这不就是NG的ML课程中所讲到的正规方程嘛！实际上，NG所说的正规方程的解法就是最小二乘法求解解析解的解法。

- (1)最小二乘法和梯度下降法在线性回归问题中的目标函数是一样的(或者说本质相同)，都是通过最小化均方差来构建拟合曲线。
- (2)二者的不同点可见下图(正规方程就是最小二乘法)：

梯度下降	正规方程
需要选择学习率 α	不需要
需要多次迭代	一次运算得出
当特征数量 n 大时也能较好适用	需要计算 $(X^T X)^{-1}$ 如果特征数量 n 较大则运算代价大，因为矩阵逆的计算时间复杂度为 $O(n^3)$ ，通常来说当 n 小于 10000 时还是可以接受的
适用于各种类型的模型	只适用于线性模型，不适合逻辑回归模型等其他模型

需要注意的一点是最小二乘法只适用于线性模型(这里一般指线性回归)；而梯度下降适用性极强，一般而言，**只要是凸函数**，都可以通过梯度下降法得到全局最优值(对于非凸函数，能够得到局部最优解)。

梯度下降法只要保证目标函数存在一阶连续偏导，就可以使用。

2.最小二乘法的一些限制和解决方法：

我们由**第三部分(2)式**可知道，要保证最小二乘法有解，就得保证 $A^T A$ 是一个可逆阵(非奇异矩阵)；那如果 $A^T A$ 不可逆怎么办？什么情况下 $A^T A$ 不可逆？

关于 $A^T A$ 在什么情况下不可逆：

- (1)当样本的数量小于参数向量(即 β)的维度时，此时 $A^T A$ 一定是不可逆的。例如：你有1000个特征，但你的样本数目小于1000的话，那么构造出的 $A^T A$ 就是不可逆的。
- (2)在所有特征中若存在一个特征与另一个特征线性相关或一个特征与若干个特征线性相关时，此时 $A^T A$ 也是不可逆的。为什么呢？

具体来说假设 A 是 $m \times n$ 维的矩阵，若存在线性相关的特征，则 $R(A) < n, R(A^T) < n, R(A^T A) < n$,所以 $A^T A$ 不可逆。

如果 $A^T A$ 不可逆，应该怎样解决？

- (1)筛选出线性无关的特征，不保留相同的特征，保证不存在线性相关的特征。
- (2)增加样本量。

(3)采用正则化的方法。对于正则化的方法，常见的是L1正则项和L2正则项，L1项有助于从很多特征中筛选出重要的特征，而使得不重要的特征为0(所以L1正则项是个不错的特征选择方法)；如果采用L2正则项的话，实际上解析解就变成了如下的形式：

$$\beta = (A^T A + \lambda \begin{pmatrix} 0 & \cdots & \cdots & 0 \\ \vdots & 1 & 0 & \vdots \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & \cdots & 1 \end{pmatrix})^{-1} A^T Y$$

λ 即正则参数(**是一种超参数**)后面的矩阵为 $(n+1) \times (n+1)$ 维，如果不考虑常数项的话，就是一个单位阵；此时括号中的矩阵一定是可逆的。

3.最小二乘法的改进

最小二乘法由于是最小化均方差，所以它考虑了每个样本的贡献，也就是每个样本具有相同的权重；由于它采用距离作为度量，使得他对噪声比较敏感(最小二乘法假设噪声服从高斯分布)，即使得它对异常点比较敏感。因此，人们提出了加权最小二乘法，

相当于给每个样本设置了一个权重，以此来反应样本的重要程度或者对解的影响程度。

参考: NG《机器学习》

《矩阵分析与应用》

<http://www.cnblogs.com/iamccme/archive/2013/05/15/3080737.html>

<http://blog.csdn.net/bitcarmanlee/article/details/51589143>

分类: Machine Learning

好文要顶

关注我

收藏该文



Excalibur

关注 - 16

粉丝 - 10

+加关注

« 上一篇: c语言中printf()函数中的参数计算顺序

» 下一篇: LeetCode——1. Two Sum

posted on 2017-09-11 23:22 Excalibur 阅读(10446) 评论(0) 编辑 收藏

6

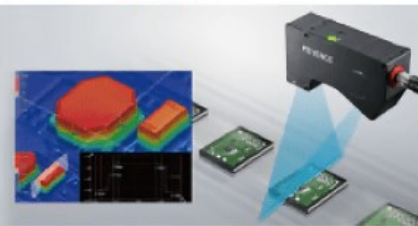

0

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论, 请 [登录](#) 或 [注册](#), [访问网站首页](#)。

【推荐】超50万C++/C#源码: 大型实时仿真HMI组态CAD\GIS图形源码!

【推荐】专业便捷的企业级代码托管服务 - Gitee 码云



3维图像处理

将以往的不可能化为可能。

[» 了解更多](#)

相关博文:

- 机器学习笔记——最小二乘法
- 机器学习经典算法之-----最小二乘法
- 机器学习-最小二乘法
- 机器学习-最小二乘法
- 机器学习笔记 (二) ——多变量最小二乘法



3维图像处理

将以往的不可能化为可能。

[» 了解更多](#)

最新新闻:

- 网联为何选择与万事达合作?
- 宣布“末位淘汰”又扩招1.5万人 刘强东打的什么算盘
- 对巴菲特最大误解, 是错把他当做“股神”
- 5G的爆发与焦虑
- 出门问问发布TicWatch C2, 可主动识别运动姿态
- » 更多新闻...

Powered by:

博客园

Copyright © Excalibur