最小二乘法

综述

最小二乘法是在回归分析中用于拟合overdetermined system（即方程数多于未知数的方程组）的标准方法。其基本思想是最小化残差平方和（残差被定义为观测值和拟合值的差）。

最小二乘问题分为两类：线性二乘和非线性二乘，线性二乘中所有的残差均为线性。线性二乘问题存在解析解，而非线性二乘问题则需要通过迭代优化的方法，在每步迭代的过程中依然是使用的线性二乘解法，因此这两类最小二乘问题具有统一的形式。

形式化问题

一个简单的输入可以认为是n个点的集合 $(x_i, y_i), i = 1, \ldots, n$ 这里x是自变量，y是因变量（观测值）。而拟合模型 $f(x, \beta)$ 含有m个参数，即 $\beta_i, i = 1, \ldots, m$ ，那么，最小二乘法的目标便是找到最拟合这组数据的模型参数。下面是残差的定义：$r_i = y_i - f(x_i, \beta)$ ，而最小二乘就是最小化这个残差平方的和：$S = \sum_{i=1}^{n} r_i^2$

限制

相比使用total least squares，最小二乘法的目标函数只考虑了观测值的误差，而total least squares则同时考虑了自变量和因变量，其目标函数可以写为：$S = r_x^T M_x^{-1} r_X + r_y^T M_y^{-1} r_y$ 其中M_x 和M_y 是x和y的协方差矩阵，r_x和r_y则是x和y的残差。更多关于total least squares的讨论：Wiki

解最小二乘问题

解最小二乘的本质方法就是使梯度为0，即：

$$\frac{\partial S}{\partial \beta_j} = 2 \sum_i r_i \frac{\partial r_i}{\partial \beta_j} = 0, j = 1, \ldots, m \tag{1}$$

然后，即然$r_i = y_i - f(x_i, \beta)$，那么上面的梯度可以写为：

$$-2 \sum_i r_i \frac{\partial f(x_i, \beta)}{\partial \beta_j} = 0, j = 1, \ldots, m \tag{2}$$

线性最小二乘法

对于**线性最小二乘（Linear Least Squares）**，我们的回归模型是模型参数的线性组合：

$$f(x, \beta) = \sum_{j=1}^{m} \beta_j \phi_j(x) \tag{3}$$

其中ϕ_j是x的函数，我们令$X_{ij} = \phi_j(x_i)$然后把自变量和因变量放入X和Y矩阵中，而D则是所有数据的集合，我们有：

$$Y = \begin{vmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{vmatrix}, X = \begin{vmatrix} \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_m(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_m(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x_n) & \phi_2(x_n) & \cdots & \phi_m(x_n) \end{vmatrix}, \beta = \begin{vmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{vmatrix} \tag{4}$$

$$L(D, \beta) = ||Y - X\beta||^2 = (Y - X\beta)^T (Y - X\beta) = Y^T Y - Y^T X\beta - \beta^T X^T Y + \beta^T X^T X\beta \tag{5}$$

那么，解线性最小二乘就可以计算L的梯度，并使得其为0。

$$Y^T X\beta = \sum_{i=1}^{n} \sum_{j=1}^{m} Y_i \phi_j(x_i) \beta_j = \beta X^T Y \tag{6}$$

$$\frac{\partial L(D,\beta)}{\partial \beta} = \frac{\partial(Y^TY - Y^TX\beta - \beta^TX^TY + \beta^TX^TX\beta)}{\partial\beta} = \frac{\partial(-2\beta^TX^TY + \beta^TX^TX\beta)}{\partial\beta} = -2X^TY + 2X^TX\beta \quad (7)$$

$$-2XTY + 2X^TX\beta = 0 \Rightarrow X^TY = X^TX\beta \quad (8)$$

$$\hat{\beta} = (X^TX)^{-1}X^TY \quad (9)$$

其中$(X^TX)^{-1}X^T$ 称为Moore-Penrose Generalized Inverse Matrix（穆尔-彭罗斯广义逆矩阵），更多：Wiki

非线性最小二乘法

对于大多数非线性最小二乘问题，并不存在一个形式固定的数值解，因此我们在这里介绍求解非线性最小二乘问题的迭代法。其一般形式为，即使用迭代结果 β_j 来作为参数 β 的拟合值：

$$\beta_j \approx \beta_j^{k+1} = \beta_j^k + \Delta\beta_j \quad (10)$$

而每一步迭代可以看做是使用泰勒展开的第一项的线性拟合步骤：

$$\begin{aligned} f(x_i,\beta) &\approx f(x_i,\beta^k) + \sum_j \frac{\partial f(x_i,\beta)}{\partial\beta_j}(\beta_j - \beta_j^k) \\ &= f(x_i,\beta^k) + \sum_j J_{ij}(\beta_j - \beta_j^k) \end{aligned} \quad (11)$$

而如前面所说，最小二乘法的目标是最小化残差平方，在非线性最小二乘中，残差的表示与上文一致，为 $r_i = y_i - f(x_i,\beta), i = 1,2,\ldots,m$ 以及残差平方 $S = \sum_{i=1}^m r_i^2$ 则，令残差平方的梯度为0，我们有：

$$\frac{\partial S}{\partial\beta_j} = 2\sum_i r_i \frac{\partial r_i}{\partial\beta_j} = 0 \quad (12)$$

我们可以使用雅可比矩阵表示残差对参数的偏导：

$$\frac{\partial r_i}{\partial\beta_j} = \frac{\partial(y_i - f(x_i,\beta))}{\partial\beta_j} = -\frac{\partial f(x_i,\beta)}{\partial\beta_j} = -J_{ij} \quad (13)$$

而，对于残差，我们有：

$$\begin{aligned} r_i = y_i - f(x_i,\beta) &= (y_i - f(x_i,\beta^k)) + (f(x_i,\beta^k) - f(x_i,\beta)) \\ &\approx \Delta y_i - \sum_{s=1}^n J_{is}\Delta\beta_s \end{aligned} \quad (14)$$

将（13）（14）代入（12）得：

$$\begin{aligned} &-2\sum_{i=1}^m J_{ij}(\Delta y_i - \sum_{s=1}^n J_{is}\Delta\beta_s) \\ &\Rightarrow \sum_{i=1}^m\sum_{s=1}^n J_{ij}J_{is}\Delta\beta_s = \sum_{i=1}^m J_{ij}\Delta y_i, j = 1,2,\ldots,n \end{aligned} \quad (15)$$

使用矩阵改写（15）为，也就是高斯-牛顿迭代法的形式化描述：

$$(J^TJ)\Delta\beta = J^T\Delta y \quad (16)$$

$$\beta^{k+1} = \beta^k + \Delta\beta = \beta^k + (J^TJ)^{-1}J^T\Delta y \quad (17)$$

这里 Δy_i 也可以表示为在第k次迭代时的残差 $r_i^k = y_i - f(x_i,\beta^k)$

Levenberg–Marquardt迭代法

LM法将（16）式改为了带阻尼系数的版本：

$$(J^TJ + \lambda I)\Delta\beta = J^T\Delta y \quad (18)$$

λ 是一个非负系数，用于控制梯度下降的速度。当残差平方下降的速度足够快的时候，取值较小，此时方法趋近高斯-牛顿法；当下降速度较慢的时候，取值较大，此时迭代在梯度的反方向具有更大的步长。

对LM的一种scale-invariant的改进，即对每个梯度元素使用不同的系数 λ，于是我们有：

$$(J^TJ+\lambda diag(J^TJ))\Delta\beta=J^T\Delta y \tag{19}$$

参考：

[1] https://en.wikipedia.org/wiki/Least_squares

[2] http://www.gatsby.ucl.ac.uk/teaching/courses/sntn/sntn-2017/resources/Matrix_derivatives_cribsheet.pdf

[3] https://atmos.washington.edu/~dennis/MatrixCalculus.pdf

[4] https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf

[5] https://en.wikipedia.org/wiki/Levenberg-Marquardt_algorithm