线性回归学习笔记

李向阳 d1142845997@gmail.com

目录

1 基本线性回归 **3**
1.1 引入 3
1.2 数据集 4
1.3 参数估计 4

2 正则化 **8**

3 统计角度 VS 机器学习 **11**

4 贝叶斯线性回归 **11**

5 变分贝叶斯线性回归 **13**
5.1 变分贝叶斯一般框架 13
5.2 变分贝叶斯线性回归 14

6 总结 **14**

1 基本线性回归

1.1 引入

不管是统计还是机器学习, 线性回归 (Linear Regression) 都是非常基础而重要的. 因此, 机器学习笔记的这个系列文章, 还是从线性回归开始.

在数据分析方法这门课中, 我们已经详细的从经典统计学的角度学习了线性回归, 包括模型的参数估计、估计量的统计性质、回归方程及回归系数的显著性检验以及预测及统计推断, 不仅如此, 我们还学了残差分析和如何选取回归方程等. 如有遗忘或者想对比学习以使自己的知识体系融会贯通而不冲突, 可以复习课本.

下面从机器学习的角度研究线性回归. 不过由于已有统计的基础, 因此会稍微简略一些.

回归问题的变量一般是连续变化的, 这是它与分类问题的重要区别. 一个变量 y 的主要部分可能线性的依赖于一些特征变量 $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)^T$, 即我们可以用 $\boldsymbol{x}$ 的一个线性函数 $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ 去近似 y 的值

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n = \sum_{i=0}^{n} \theta_i x_i = \boldsymbol{\theta}^T \boldsymbol{x}$$

这里 $\boldsymbol{\theta} = (\theta_0, \theta_1, \cdots, \theta_n)^T$ 为未知参数, 或者称为回归系数, 而且为了表示的方便, 我们引入了特征 $x_0 = 1$, 即 $\boldsymbol{x} = (x_0, x_1, \cdots, x_n)^T$.

我们的目的是估计出参数 $\boldsymbol{\theta}$, 当得到估计量 $\hat{\boldsymbol{\theta}}$ 后, 我们可以得到经验回归方程

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \hat{\boldsymbol{\theta}}^T \boldsymbol{x}$$

当得到一个新的样本 $\boldsymbol{x}^{(0)}$ 后, 我们把它代入上式计算即可得到对应 $y^{(0)}$ 的预测值.

这里先直接声明一个重要的东西, 即看似函数 $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ 是关于特征 $x_1, x_2, \cdots, x_n$ 是线性的, 其实它的本质只要求关于参数 $\theta_1, \theta_2, \cdots, \theta_n$ 是线性的即可, 实际上, 一个最一般的线性回归模型为

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=0}^{N} \theta_j f_j(x_1, x_2, \cdots, x_n) \tag{1}$$

其中 $f_j(x_1, x_2, \cdots, x_n), j = 0, 1, \cdots, p$ 是 p 个线性无关的已知函数, 这时只要设置新的变量

$$z_j = f_j(x_1, x_2, \cdots, x_n), j = 0, 1, \cdots, p$$

便可将模型化为普通的线性回归模型, 通过假定 $f_j(x_1, x_2, \cdots, x_n), j = 0, 1, \cdots, p$ 的不同的形式 (相当于将原样本数据映射到不同的空间中), 模型 (1) 实际上包含了十分广泛的回归模型.

1.2 数据集

在线性回归中, 我们的样本是 $\{\boldsymbol{x}, y\}$, 其中 y 是要预测的变量, 而 $\boldsymbol{x}$ 表示样本的特征, 设有 n 个特征, 即 $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)^T$, 因此我们也可以将样本表示为 $\{x_1, \cdots, x_n, y\}$, 我们的期待是由这 n 个样本特征来预测因变量 y 的值. 设训练集有 m 个样本, 即 $\{\boldsymbol{x}^{(i)}, y^{(i)}\}, i = 1, 2, \cdots, m$, 其中

$$\boldsymbol{x}^{(i)} = (x_1^{(i)}, x_2^{(i)}, \cdots, x_n^{(i)})^T, i = 1, 2, \cdots, m$$

这里要注意记号的不同, 也有些文献中把粗体的 $\boldsymbol{x}$ 用 $\mathbf{x}$ 来表示, 这里我们仍将粗体倾斜.

关于记号, 这里要多说一点, 这篇文章中我采用机器学习大纲中的第一套记号, 即用粗体的 $\boldsymbol{x}$ 表示样本, 它有 n 个分量, 分量用下标表示, 即有 $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)^T$, 用上标表示具体的样本, 即 $\boldsymbol{x}^{(i)}$ 表示第 i 个样本, 并设总共有 m 个样本. 采用这样的记号, 个人觉得比采用双下标要好一些, 但推导时还是略显麻烦, 因此有些文献中会采用第二套记号: 用粗体的 $\boldsymbol{x}$ 表示样本, 同时用带下标 i 的 $\boldsymbol{x}_i$ 表示第 i 个样本, 注意 $\boldsymbol{x}_i$ 实际上有 n 个分量, 但为了推导方便并不写出, 直接理解为带 n 个分量的向量即可 (不过要加粗, 有些文档中甚至不加粗, 这很不好), 然后设样本个数为 N. 这也是一种常见的记号.

1.3 参数估计

关于线性回归的参数估计, 实在是接触太多了, 高中时便学过一元线性回归, 那时便推导出了回归方程 $y = a + bx$ 中回归系数的计算公式, 之后再物理、化学的多次实验中还用到了它 (当然, 主要是计算斜率, 想当年先是用计算器算后来用 Excel 算), 在大学的高代课、数分课还有数值分析课程上都直接或者间接的推导过多元线性回归的正规方程组的解形式, 不过那时的焦点多在于多项式拟合, 或者说散点数据的拟合. 计算方法上, 从开始的对单个变量求偏导到后来的对矩阵求导. 这个过程真的是经历了好久.

下面将会稍微总结一下线性回归的参数估计.

之前的过程虽然曲折, 但估计方法却是一个, 那就是最小二乘估计. 当然, 在学习概率论与数理统计的时候, 接触到了最小一乘法. 在学习数据分析方法的时候, 也学习到了回归系数的最大似然估计.

最小二乘估计无疑是最基础而且重要的, 而且这个标准容易理解, 既然是用 $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ 去近似 y 的值, 那我们当然希望这个误差越小越好, 也就是选取参数 $\boldsymbol{\theta}$ 使下式 (2) 达到最小

$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)})^2 \tag{2}$$

在机器学习的框架中, 一般称 $J(\boldsymbol{\theta})$ 为损失函数 (Loss Function) 或者称为代价函数 (Cost Function), 然后对代价函数求极小得到参数估计, 我们看到, 这同最小二乘估计是一样的, 毕竟优化目标都是同一个, 此外, 我们也知道, 在线性回归中, 它们还和最大似然估计是等价的.

注意到这里 $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ 是 $\boldsymbol{x}$ 的线性函数, 因此可以把式 (2) 进一步改写为

$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{m}(\boldsymbol{\theta}^T\boldsymbol{x}^{(i)} - y^{(i)})^2 = \frac{1}{2}||\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}||_2^2 = \frac{1}{2}(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})^T(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}) \quad (3)$$

其中 $\boldsymbol{X}$ 是把 m 个样本值以行的形式并起来的矩阵, 在数据分析方法中我们称为设计矩阵, 而 $\boldsymbol{y}$ 是预报变量.

$$\boldsymbol{X} = \begin{pmatrix} (\boldsymbol{x}^{(1)})^T \\ (\boldsymbol{x}^{(2)})^T \\ \vdots \\ (\boldsymbol{x}^{(m)})^T \end{pmatrix} = \begin{pmatrix} x_0^{(1)} & x_1^{(1)} & \cdots & x_n^{(1)} \\ x_0^{(2)} & x_1^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_0^{(m)} & x_1^{(m)} & \cdots & x_n^{(m)} \end{pmatrix}, \boldsymbol{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{pmatrix}$$

这样就有

$$\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y} = \begin{pmatrix} (\boldsymbol{x}^{(1)})^T\boldsymbol{\theta} \\ (\boldsymbol{x}^{(2)})^T\boldsymbol{\theta} \\ \vdots \\ (\boldsymbol{x}^{(m)})^T\boldsymbol{\theta} \end{pmatrix} - \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{pmatrix} = \begin{pmatrix} h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(1)}) - y^{(1)} \\ h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(2)}) - y^{(2)} \\ \vdots \\ h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(m)}) - y^{(m)} \end{pmatrix}$$

写成了式 (3) 的形式后, 现在学习了数值分析和矩阵分析的我们熟知

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{X}^T\boldsymbol{y}$$

因此, 令梯度为 0, 我们就可以得到正规方程组

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta} = \boldsymbol{X}^T\boldsymbol{y}$$

我们知道正规方程组一定是有解的 (在高代中先是用矩阵的秩证明过一次, 后来又知道矩阵 $\boldsymbol{X}^T\boldsymbol{X}$ 是半正定的), 因此 (不妨假设 $\boldsymbol{X}$ 列满秩, 从而 $\boldsymbol{X}^T\boldsymbol{X}$ 正定) 参数估计为

$$\boldsymbol{\theta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

而且在数值优化课上我们讲过一个思想, 即如果 $\boldsymbol{X}^T\boldsymbol{X}$ 不正定, 也就是不可逆, 考虑到对于充分大的数 α, 矩阵 $\boldsymbol{X}^T\boldsymbol{X} + \alpha I$ 必然是正定的, 因此我们可以给上式变为

$$\boldsymbol{\theta} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda I)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

我们当时说这只是为了保证 $\boldsymbol{X}^T\boldsymbol{X}$ 的正定性, 接下来我们会看到这还是有很深的道理的 (此处先挖个坑).

当然, 我们知道, 实际计算中我们是不会直接拿正规方程组去计算参数的, 因为实际数据一般都比较多, 直接求解大型方程组是比较困难的, 回想一下, 在数值分析课程中, 要想快速求解一个大型方程组, 一般要用迭代法 (Jacobi, Gauss-Seidel 等), 甚至构造了一些等价的优化问题借助共轭梯度法求解, 所以, 迭代计算的思想才是非常重要的啊!

扯远了, 如果要求解上面的那个正规方程组, 在数值分析中如果转化为对应的优化问题采用共轭梯度法, 我们熟知对应的优化目标函数不正是式 (3) 或者说式 (2) 吗?

既然如此, 我们还是直接从优化的框架去求参数估计, 即用各种迭代法来求式 (2) 的最小值, 比如梯度下降法、牛顿法、拟牛顿法 (BFGS)、共轭梯度法等等. 在此也说明一下, 在机器学习中, 通常都是得到模型后构造损失函数, 然后用优化的角度来求参数估计.

最简单的莫过于梯度下降法了, 其迭代更新过程如下:

$$\begin{aligned}\boldsymbol{\theta} &:= \boldsymbol{\theta} - \alpha\nabla J(\boldsymbol{\theta}) \\ &:= \boldsymbol{\theta} - \alpha\cdot\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})\end{aligned}$$

如果对矩阵求导不熟悉, 也可以从式 (2) 直接对单个变量求偏导, 再综合起来, 即

$$\theta_j := \theta_j - \alpha\frac{\partial J(\boldsymbol{\theta})}{\partial\theta_j}$$

而单个的偏导数计算也很简单

$$\begin{aligned}\frac{\partial J(\boldsymbol{\theta})}{\partial\theta_j} &= \frac{1}{2}\cdot 2\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)})\cdot\frac{\partial}{\partial\theta_j}(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}) \\ &= \sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)})\cdot\frac{\partial}{\partial\theta_j}\left(\sum_{k=1}^{n}\theta_k x_k^{(i)} - y^{(i)}\right) \\ &= \sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)})\cdot x_j^{(i)}\end{aligned}$$

上面的梯度下降是批处理的, 我们也可以采用随机梯度下降 (Stochastic Gradient Descent) 法.

通过上面的分析, 我们知道, 对于每个 θ_j, 迭代更新公式如下:

$$\theta_j = \theta_j - \alpha\sum_{i=1}^{m}(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)})x_j^{(i)}$$

不断迭代到收敛即可, 这里每一次迭代把 m 个样本的信息全都利用了.

我们也可以这样做: 对于 θ_j, 每一次迭代再划分为 m 步, 即

for i = 1 to m {

$\theta_j = \theta_j - \alpha \left(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}\right) x_j^{(i)}$ }

这样每次迭代也都是遍历整个数据集, 但是每一小步只处理一个样本, 这就叫做 Stochastic Gradient Descent(或 Incremental Gradient Descent), 它不用每次都真的遍历整个数据集, 因此收敛的也会更快些, 关于更多的比较可参考其他文献.

下面我们从统计的角度来估计未知参数, 也就是参数的最大似然估计, 当然, 这在数据分析方法课上已经讲过了, 不过这里还是要提一下.

我们假设 $y = \boldsymbol{\theta}^T \boldsymbol{x} + \varepsilon$, 也就是对每个样本来说,

$$y^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} + \varepsilon^{(i)}$$

其中 $\varepsilon^{(i)}$ 独立同分布于 $\mathcal{N}(0, \sigma^2)$, 也就是

$$p(\varepsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\varepsilon^{(i)})^2}{2\sigma^2}\right)$$

我们知道 $y^{(i)} \sim \mathcal{N}(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)}, \sigma^2)$, 因此每个样本出现的概率为

$$p(y^{(i)}|\boldsymbol{x}^{(i)}; \boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2}{2\sigma^2}\right)$$

于是可得似然函数为

$$\begin{aligned} L(\boldsymbol{\theta}) &= \prod_{i=1}^{m} p(y^{(i)}|\boldsymbol{x}^{(i)}; \boldsymbol{\theta}) \\ &= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2}{2\sigma^2}\right) \end{aligned}$$

进一步可得对数似然函数为

$$\begin{aligned} \ell(\boldsymbol{\theta}) &= \log L(\boldsymbol{\theta}) \\ &= \log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2}{2\sigma^2}\right) \\ &= m \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} \sum_{i=1}^{m} (y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2 \end{aligned}$$

因此, 极大化 $\ell(\boldsymbol{\theta})$ 等价于极小化

$$\frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2$$

而这正是我们之前使用最小二乘方法定义的损失函数 $J(\boldsymbol{\theta})$, 因此这里最大似然估计与最小二乘估计是一样的.

估计先说到这里.

当我们要拟合的数据特征太多或者噪音太大时, 我们的模型就要适当加以调整, 否则可能出现严重的过拟合现象, 这个一会儿再细说. 为了解决数据噪音的问题, 一个简单的方法是给样本点乘以权重, 即 Locally Weighted Linear Regression (LWR), 也就是最小化下式

$$\frac{1}{2}\sum_{i=1}^{m} w_i (y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2$$

至于 $w_i, 1 \leqslant i \leqslant m$ 该如何选取, 这里就不介绍了.

2 正则化

过拟合与正则化是一个很重要的问题, 为了形象的说明这个问题, 我们用线性回归的一个重要应用, 也就是我们熟知的多项式拟合来阐述.

以二维为例 (这里简化记号, 均用下标表示), 对于 m 个样本点 $(x_1, y_1), \cdots, (x_m, y_m)$, 我们希望用一条曲线进行拟合. 比如

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \cdots$$

最开始我们就提到过, 其实这里 $x, x^2, x^3, \cdots$ 就相当于变量的特征, 如果我们令 $z_1 = x, z_2 = x^2, z^3 = x^3, \cdots$, 那么上面本质上还是一个线性模型.

观察下图:

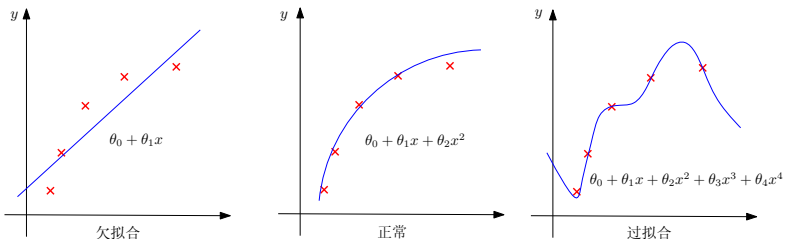

图 1: 拟合情况

一个好的模型不仅应当对已有的训练样本拟合的很好, 还应该对未知数据有很好的预测能力 (这一般称为模型的泛化能力), 否则我们得到这个模型也没有太大实际意义. 如果模型太复杂, 可能对训练样本表现不错, 但预测能力差, 这在机器学习的很多模型中都能见到, 所以我们要限制模型的复杂度, 避免过度拟合数据或者噪音.

在线性回归中, 模型参数 $\boldsymbol{\theta}$ 一定程度上反映了模型的复杂度, 因此我们一般是对参数 $\boldsymbol{\theta}$ 施加惩罚, 这种思想我们在数值优化中也多次接触过.

常见的惩罚有 L_1 和 L_2 惩罚, 分别称为 Lasso Regression 和 Ridge Regression.

所谓 Ridge Regression, 就是使得下式最小

$$\frac{1}{2}\sum_{i=1}^{m}(y^{(i)}-\boldsymbol{\theta}^T\boldsymbol{x}^{(i)})^2+\alpha\sum_{j=1}^{n}\theta_j^2=\frac{1}{2}||\boldsymbol{X}\boldsymbol{\theta}-\boldsymbol{y}||_2^2+\alpha||\boldsymbol{\theta}||_2^2 \tag{4}$$

注意我们不需要对 θ_0 做惩罚, 所以上式右边的求和中 j 是从 1 到 n, 而不是从 0 到 n(不过最后实际计算时差别影响不大, 所以简写为范数时也就写为了 $\boldsymbol{\theta}$). 为什么这样能够达到正则化的目的呢? 其实在优化中已经多次提到过这种思想, 这里再说明一下.

比如上面的散点模型可以用二次曲线拟合, 但是我们加了两个特征 x^3 和 x^4, 那为了让这两个特征几乎不起作用, 我们对它们的系数 θ_3 和 θ_4 施加大的惩罚, 即乘以一个非常大的数, 比如 10000, 也就是极小化下式

$$\frac{1}{2}\sum_{i=1}^{m}(y^{(i)}-\boldsymbol{\theta}^T\boldsymbol{x}^{(i)})^2+10000\theta_3^2+10000\theta_4^2$$

那么为了求得上式的最小值, θ_3 和 θ_4 只能"牺牲"一下, 或者说"被逼"的, 近似等于 0 了, 这样模型就近似是二次的, 也就达到了避免过拟合的目的.

继续回到 (4) 式, 要求出它的最小值, 同样可以借助矩阵求导, 直接对 $\boldsymbol{\theta}$ 求梯度, 并令其为 0, 可以得到

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta}-\boldsymbol{X}^T\boldsymbol{y}+\alpha\boldsymbol{\theta}=0$$

于是可得估计为

$$\boldsymbol{\theta}=(\boldsymbol{X}^T\boldsymbol{X}+\alpha I)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

当 α 充分大的时候, 必然能使惩罚参数近似为 0, 相当于矩阵 $\boldsymbol{X}^T\boldsymbol{X}+\alpha I$ 一定正定, 这不正是我们之前讨论过正规方程组系数矩阵不正定的情况吗? 所以, 我们当时给系数矩阵加上了 αI 是有道理的, 它居然还能避免过拟合, 想想真是好!

当然, 真正求解 Ridge Regression, 肯定也不会用解方程组的办法, 而是用优化中的各种迭代法, 也就还是梯度下降法、牛顿法等等, 这里的梯度求法和基本线性回归的一样, 就不再写出了.

所谓 Lasso Regression, 与 Ridge Regression 的区别在于施加的是 L_1 惩罚, 即使得下式最小

$$\frac{1}{2}\sum_{i=1}^{m}(y^{(i)}-\boldsymbol{\theta}^T\boldsymbol{x}^{(i)})^2+\alpha\sum_{j=1}^{n}|\theta_j|=\frac{1}{2}||\boldsymbol{X}\boldsymbol{\theta}-\boldsymbol{y}||_2^2+\alpha||\boldsymbol{\theta}||_1$$

其中 α 也是惩罚参数. Lasso Regression 能够正则化的原因同 Ridge Regression 是一样的, 毕竟只是换了个范数嘛, 那他们有什么区别呢?

首先, 一个大的区别就是 L_1 范数不容易处理, 这我们再很多数学课都非常熟悉了, 所以 Lasso Regression 无法得出解的精确表达式, 只能借助优化办法进行数值计算.

其次, 这涉及到 L_1 范数和 L_2 范数的本质不同. 如果测试过数据你会发现, 用 L_2 范数得到的参数估计值可能都比较小, 但用 L_1 范数的话, 会发现很多参数的估计值为 0, 也就是得到的解比较稀疏. 这样当特征维数比较高的时候非常有用. 这是为什么呢? 直观展示可见下图

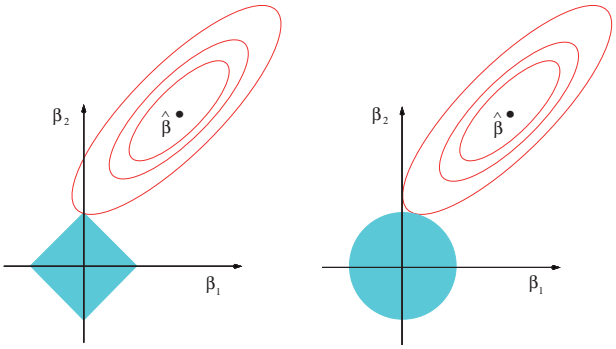

FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \leq t$ *and* $\beta_1^2 + \beta_2^2 \leq t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

图 2: Ridge and Lasso

该图来自 ESL 的第三章, 关于 Ridge VS Lasso 很多文献中都有大篇幅讨论, 这里就不再多述了.

最后提一下 Elastic Net, 它是把 Ridge 和 Lasso 综合到了一块, 即极小化下式

$$\frac{1}{2m}||\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}||_2^2 + \alpha\left[\rho||\boldsymbol{\theta}||_1 + \frac{1-\rho}{2}||\boldsymbol{\theta}||_2^2\right]$$

其中 α 是惩罚参数, 而 ρ 调整 Ridge 和 Lasso 的比重. Python 的 sklearn 包和 R 的 glmnet 包都能提供以上正则化模型的计算方法. 事实上, 正则化还涉及到了变量选择或者说模型选择的问题, 这就是一个更大的问题了, 可参考其他文献.

3 统计角度 VS 机器学习

以上大都是从机器学习的角度来看问题, 主要的目的是估计出参数 $\boldsymbol{\theta}$, 至于预测效果与模型选择等, 可由训练误差与测试误差决定.

这里我们要注意一个问题, 即统计方法和机器学习方法的异同. 从经典统计学的角度看, 参数 $\boldsymbol{\theta}$ 是一个未知的固定常数, 但是估计量 $\hat{\boldsymbol{\theta}}$ 是一个随机变量, 是随机样本的函数, 因此, 经典统计学会进一步分析估计量的统计性质, 比如分布等等, 进而寻求参数的区间估计, 或者对我们的结果进行假设检验, 比如模型系数的显著性检验, 以及利用 AIC 准则或者 BIC 准则进行模型选择等等. 而在机器学习的框架中, 虽然也会借助统计方法构建模型, 但重点不在于那些假设检验, 机器学习有自己的模型选择方法.

另外一方面, 从贝叶斯统计的角度看, 一切未知量都是随机变量, 即参数 $\boldsymbol{\theta}$ 不是一个未知的常数, 它也是一个随机变量, 只要我们给它一个合适的先验分布, 然后计算出后验分布, 也可进行各种统计推断, 而且往往比经典统计学更合理一些, 比如上面的最大似然估计会过拟合, 因此需要加上惩罚项来正则化, 而我们很容易证明, 假如 $\boldsymbol{\theta}$ 的先验分布是 Laplace 分布, 即 $p(\boldsymbol{\theta}) = (\alpha/2)^n \exp(-\alpha||\boldsymbol{\theta}||_1)$, 其中常数 $\alpha > 0$, 则 $\boldsymbol{\theta}$ 的最大后验估计由下式给出:

$$\min_{\boldsymbol{\theta}} \left(\sum_{i=1}^{m} -\log p(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta}) + \alpha||\boldsymbol{\theta}||_1 \right)$$

这恰好即为 L_1 正则化. 因此, 运用贝叶斯统计的观点来看问题, 能自动避免过拟合, 接下来我们就来看如何用贝叶斯方法进行线性回归.

4 贝叶斯线性回归

在贝叶斯统计中, 关键是要求出后验分布, 一般利用核方法, 即

$$p(\boldsymbol{\theta}|\boldsymbol{x}) \propto p(\boldsymbol{\theta}) \cdot p(\boldsymbol{x}|\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \cdot L(\boldsymbol{\theta}|\boldsymbol{x}) = K(\boldsymbol{\theta})$$

其中 $L(\boldsymbol{\theta}|\boldsymbol{x})$ 就是样本似然函数 $L(\boldsymbol{\theta})$, 最大后验估计即要使 $K(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \cdot L(\boldsymbol{\theta}|\boldsymbol{x})$ 最大. 注意在线性回归中, 这里的 $p(\boldsymbol{\theta}|\boldsymbol{x})$ 应当理解为 $p(\boldsymbol{\theta}|\boldsymbol{x}, y)$, 因为本质上 $\boldsymbol{x}, y$ 合在一起是一组样本.

前面已经求得

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = L(\boldsymbol{\theta}|\boldsymbol{x}) = L(\boldsymbol{\theta}) = \prod_{i=1}^{m} p(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta})$$

下面来看 $\boldsymbol{\theta}$ 的先验分布 $p(\boldsymbol{\theta})$.

假定 $\boldsymbol{\theta}$ 的先验分布为 Laplace 分布, 即

$$p(\boldsymbol{\theta}) = \left(\frac{\alpha}{2}\right)^n \mathrm{e}^{-\alpha||\boldsymbol{\theta}||_1}$$

其中 $\alpha > 0$ 为常数, 则有

$$K(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \cdot p(\boldsymbol{x}|\boldsymbol{\theta}) = \left(\frac{\alpha}{2}\right)^n \mathrm{e}^{-\alpha||\boldsymbol{\theta}||_1} \cdot \prod_{i=1}^{m} p(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta})$$

对 $K(\boldsymbol{\theta})$ 取负对数, 可得

$$-\log K(\boldsymbol{\theta}) = -n\log\frac{\alpha}{2} + \alpha||\boldsymbol{\theta}||_1 - \sum_{i=1}^{m}\log p(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta})$$

极大化 $K(\boldsymbol{\theta})$ 与极小化 $-\log K(\boldsymbol{\theta})$ 是一样的, 即等价于

$$\min_{\boldsymbol{\theta}} \left(\sum_{i=1}^{m} -\log p(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta}) + \alpha||\boldsymbol{\theta}||_1\right)$$

这就是 L_1 正则化, 于是我们就得到了前面所说的结果.

此外, 如果假定 θ_j 的先验分布为

$$\theta_j \sim N(0, s^2) = \frac{1}{\sqrt{2\pi}s}\mathrm{e}^{-\frac{\theta_j^2}{2s^2}}, j = 1, 2, \cdots n$$

于是可得 $\boldsymbol{\theta}$ 的先验分布为

$$p(\boldsymbol{\theta}) = \prod_{j=1}^{n}\frac{1}{\sqrt{2\pi}s}\mathrm{e}^{-\frac{\theta_j^2}{2s^2}} = \left(\frac{1}{\sqrt{2\pi}s}\right)^n \mathrm{e}^{-\frac{\boldsymbol{\theta}^T\boldsymbol{\theta}}{2s^2}}$$

进一步有

$$K(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \cdot p(\boldsymbol{x}|\boldsymbol{\theta}) = \left(\frac{1}{\sqrt{2\pi}s}\right)^n \mathrm{e}^{-\frac{\boldsymbol{\theta}^T\boldsymbol{\theta}}{2s^2}} \cdot \prod_{i=1}^{m} p(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta})$$

还是对 $K(\boldsymbol{\theta})$ 取负对数, 可得极小化 $-\log K(\boldsymbol{\theta})$ 即为

$$\min_{\boldsymbol{\theta}} \left(\sum_{i=1}^{m} -\log p(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta}) + \frac{\boldsymbol{\theta}^T\boldsymbol{\theta}}{2s^2}\right)$$

注意到 $\boldsymbol{\theta}^T\boldsymbol{\theta} = ||\boldsymbol{\theta}||_2^2$, 因此这相当于 L^2 正则化.

以上正态分布的假定相当于 $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, s^2\boldsymbol{I})$. 当然, 我们也可以假定 $\boldsymbol{\theta}$ 的先验分布为其他的形式, 比如 $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{S})$, 只不过处理会稍微复杂些, 这里就先不讨论了. 此外, 这里假定超参数比如 s 是已知的, 如果 s 未知, 我们也可以给 s 一个分布, 只不过讨论会更麻烦些.

这里我们看到, 从贝叶斯统计的观点看, 正则化相当于给参数 $\boldsymbol{\theta}$ 赋予先验信息而已. Ridge Regression 相当于先验分布是正态分布, 而 Lasso Regression 相当于先验分布是 Laplace 分布.

5 变分贝叶斯线性回归

5.1 变分贝叶斯一般框架

在变分贝叶斯中, 我们将所有观测到的样本记为 $\boldsymbol{X}$, 不妨设有 m 个样本, 即有 $\boldsymbol{X}=\{\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)},\cdots,\boldsymbol{x}^{(m)}\}$, 把所有的未知参数和隐变量 (latent variable) 记为 $\boldsymbol{Z}$, 我们的目的是要估计出后验分布 $p(\boldsymbol{Z}|\boldsymbol{X})$, 有了这个后验分布, 我们就可以估计出未知参数了.

但是解析的算出后验分布 $p(\boldsymbol{Z}|\boldsymbol{X})$ 的表达式是几乎不可能的, 我们考虑用一个分布 $q(\boldsymbol{Z})$ 去近似它. 为此先来看一个关系.

$$\begin{aligned}
\ln p(\boldsymbol{X}) &= \ln \frac{p(\boldsymbol{X},\boldsymbol{Z})}{p(\boldsymbol{Z}|\boldsymbol{X})} = \int q(\boldsymbol{Z}) \ln \frac{p(\boldsymbol{X},\boldsymbol{Z})}{p(\boldsymbol{Z}|\boldsymbol{X})}\,\mathrm{d}\boldsymbol{Z} \\
&= \int q(\boldsymbol{Z}) \ln \left(\frac{p(\boldsymbol{X},\boldsymbol{Z})}{p(\boldsymbol{Z}|\boldsymbol{X})} \cdot \frac{q(\boldsymbol{Z})}{q(\boldsymbol{Z})} \right) \mathrm{d}\boldsymbol{Z} \\
&= \int q(\boldsymbol{Z}) \left(\ln \frac{q(\boldsymbol{Z})}{p(\boldsymbol{Z}|\boldsymbol{X})} + \ln \frac{p(\boldsymbol{X},\boldsymbol{Z})}{q(\boldsymbol{Z})} \right) \mathrm{d}\boldsymbol{Z} \\
&= \int q(\boldsymbol{Z}) \cdot \ln \frac{q(\boldsymbol{Z})}{p(\boldsymbol{Z}|\boldsymbol{X})}\,\mathrm{d}\boldsymbol{Z} + \int q(\boldsymbol{Z}) \cdot \ln \frac{p(\boldsymbol{X},\boldsymbol{Z})}{q(\boldsymbol{Z})}\,\mathrm{d}\boldsymbol{Z} \\
&= -\int q(\boldsymbol{Z}) \cdot \ln \frac{p(\boldsymbol{Z}|\boldsymbol{X})}{q(\boldsymbol{Z})}\,\mathrm{d}\boldsymbol{Z} + \int q(\boldsymbol{Z}) \cdot \ln \frac{p(\boldsymbol{X},\boldsymbol{Z})}{q(\boldsymbol{Z})}\,\mathrm{d}\boldsymbol{Z} \\
&= \mathrm{KL}(q||p) + \mathcal{L}(q)
\end{aligned}$$

其中 $\mathrm{KL}(q||p)$ 表示 $q(\boldsymbol{Z})$ 与 $p(\boldsymbol{Z}|\boldsymbol{X})$ 间的 KL 散度.

$$\mathrm{KL}(q||p) = -\int q(\boldsymbol{Z}) \cdot \ln \frac{p(\boldsymbol{Z}|\boldsymbol{X})}{q(\boldsymbol{Z})}\,\mathrm{d}\boldsymbol{Z} \geqslant 0$$

从这个角度看, $\mathrm{KL}(q||p)$ 反映了我们用 $q(\boldsymbol{Z})$ 去近似 $p(\boldsymbol{Z}|\boldsymbol{X})$ 的差距, 这个值越小越好, 而且我们知道 $\mathrm{KL}(q||p)=0 \Longleftrightarrow q(\boldsymbol{Z})=p(\boldsymbol{Z}|\boldsymbol{X})$.

在这里, 我们是通过不断最大化 $\mathcal{L}(q)$ 来达到相同的目的, 而 $\mathcal{L}(q)$ 刚好是一个函数 $q(\boldsymbol{Z})$ 的泛函, 变分的名称由此得来.

如同一般的变分法, 这里要对 $q(\boldsymbol{Z})$ 做某些要求, 相当于把 $q(\boldsymbol{Z})$ 限制到某个空间中. 假设 $\boldsymbol{Z}$ 可独立成 M 个部分 $Z_i, i=1,2,\cdots,M$, 那么最常用的一个要求是因子分解分布, 即有

$$q(\boldsymbol{Z}) = \prod_{i=1}^{M} q_i(Z_i)$$

在此限定条件下我们来求 $\mathcal{L}(q)$ 的最大值, 总体思路是把 $\mathcal{L}(q)$ 拆分成独立的 $q_j(Z_j)$, 从而对每一项求最大值, 以下推导中将 $q_j(Z_j)$ 简记为 q_j, 于是

有

$$\begin{aligned}
\mathcal{L}(q) &= \int q(\boldsymbol{Z}) \cdot \ln \frac{p(\boldsymbol{X}, \boldsymbol{Z})}{q(\boldsymbol{Z})} \,\mathrm{d}\boldsymbol{Z} \\
&= \int \prod_i q_i(Z_i) \left(\ln p(\boldsymbol{X}, \boldsymbol{Z}) - \ln \prod_i q_i(Z_i) \right) \mathrm{d}\boldsymbol{Z} \\
&= \int \prod_i q_i \left(\ln p(\boldsymbol{X}, \boldsymbol{Z}) - \sum_i \ln q_i \right) \mathrm{d}\boldsymbol{Z} \\
&= \int \prod_i q_i \ln p(\boldsymbol{X}, \boldsymbol{Z}) \prod_i \mathrm{d}Z_i - \sum_i \int \prod_j q_j \ln q_i \,\mathrm{d}Z_i \\
&= \int q_j \left(\ln p(\boldsymbol{X}, \boldsymbol{Z}) \prod_{i \neq j} (q_i \,\mathrm{d}Z_i) \right) \mathrm{d}Z_j - \int q_j \ln q_j \,\mathrm{d}Z_j - \sum_{i \neq j} \int q_i \ln q_i \,\mathrm{d}Z_i \\
&= \int q_j \ln \tilde{p}(\boldsymbol{X}, Z_j) \,\mathrm{d}Z_j - \int q_j \ln q_j \,\mathrm{d}Z_j - \sum_{i \neq j} \int q_i \ln q_i \,\mathrm{d}Z_i \\
&= -\int q_j \ln \frac{\tilde{p}(\boldsymbol{X}, Z_j)}{q_j} \,\mathrm{d}Z_j - \sum_{i \neq j} \int q_i \ln q_i \,\mathrm{d}Z_i \\
&= -\mathrm{KL}(q_j || \tilde{p}) - \sum_{i \neq j} \int q_i \ln q_i \,\mathrm{d}Z_i
\end{aligned}$$

其中

$$\ln \tilde{p}(\boldsymbol{X}, Z_j) = \int \ln p(\boldsymbol{X}, \boldsymbol{Z}) \prod_{i \neq j} (q_i \,\mathrm{d}Z_i) = \mathbb{E}_{i \neq j}[\ln p(\boldsymbol{X}, \boldsymbol{Z})]$$

显然, 当 $\mathrm{KL}(q_j||\tilde{p}) = 0$ 时上式取得最大值, 也即 $q_j(Z_j) = \tilde{p}(\boldsymbol{X}, Z_j)$, 因此最优分布 $q_j^*(Z_j)$ 即为

$$\ln q_j^*(Z_j) = \ln \tilde{p}(\boldsymbol{X}, Z_j) = \mathbb{E}_{i \neq j}[\ln p(\boldsymbol{X}, \boldsymbol{Z})] + \mathrm{const}$$

这里的常数是为了保证分布 $q_j^*(Z_j)$ 的正则性, 直接进行归一化可得

$$q_j^*(Z_j) = \frac{\exp(\mathbb{E}_{i \neq j}[\ln p(\boldsymbol{X}, \boldsymbol{Z})])}{\int \exp(\mathbb{E}_{i \neq j}[\ln p(\boldsymbol{X}, \boldsymbol{Z})]) \,\mathrm{d}Z_j}$$

这便是最优分布的表达式.

关于变分贝叶斯还有很多东西, 这里就不再讨论了.

5.2 变分贝叶斯线性回归

6 总结

附录