梯度类算法

芦星宇

April 2021

目录

1 LASSO 问题

对于 LASSO 问题

$$\min_x \frac{1}{2}\|Ax-b\|_2^2+\mu\|x\|_1.$$

此问题为回归问题，而对于回归问题其实本质就是一个函数拟合的过程，模型不能太过复杂，否则很容易发生过拟合现象，所以我们就要加入正则化项，而对于 LASSO 问题，采用 L1 正则化，会使得部分学习到的特征权值为 0, 从而达到稀疏化和特征选择的目的。

为什么 L1 正则更容易导致稀疏解?

假设只有一个参数 w, 损失函数为 $L(w)$, 加上 L 1 正则项后有:

$$J_{L1}(w)=L(w)+\lambda|w|_1$$

假设 $L(w)$ 在 0 处的导数为 d_0, 即

$$\frac{\partial L(w)}{\partial w}=d_0$$

则可以推导使用 L 1 正则的导数

$$\frac{\partial J_{L1}(w)}{\partial w}|_{w=0^-}=d_0-\lambda$$

$$\frac{\partial J_{L1}(w)}{\partial w}|_{w=0^+}=d_0+\lambda$$

引入 L 1 正则后, 代价函数在 0 处的导数有一个突变, 从 $d_0+\lambda$ 到 $d_0-\lambda$, 若 $d_0+\lambda$ 和 $d_0-\lambda$ 异号，则在 0 处会是一个极小值点。因此，优化时，可能优化到该极小值点上，即 $w=0$ 处。

2 LASSO 问题的梯度下降

2.1 LASSO 问题的连续化策略

目的：寻找一个合适的 μ_t, 求解相应的 LASSO 问题。

方法：从较大的 μ_t 逐渐减小到 μ_0。

代码解析:

1) 更新梯度阈值，他们随着外层迭代的进行逐渐减小，对子问题求解的精度逐渐提高。

2) 当内层循环达到收敛条件退出时，缩减正则化系数 μ_t，并判断收敛。

3) 外层循环收敛的条件：当 μ 已经减小到与 μ_0 相同并且函数值或梯度值满足收敛条件。

2.2 BB 步长梯度下降法

对于可微的目标函数 $f(x)$，梯度下降法通过使用如下重复迭代格式

$$x^{k+1} = x^k - \alpha \bigtriangledown f(x^k)$$

求解 $f(x)$ 的最小值，其中 a_k 为第 k 步的步长。

令 $s^k = x^{k+1} - x^k, y^k = \nabla f(x^{k+1}) - \nabla f(x^k)$，定义两种 BB 步长，$\dfrac{(s^k)^\top s^k}{(s^k)^\top y^k}$ 和 $\dfrac{(s^k)^\top y^k}{(y^k)^\top y^k}$。

理论解释：

如果我们记 $g^k = \bigtriangledown f(x^{(k)})$ 和 $F^k = \bigtriangledown^2 f(x^{(k)})$，那么**一阶方法**就是 $x^{k+1} = x^k - \alpha_k g(x^{(k)})$，其中步长 α_k 是固定的，也可以使线搜索获得的，一阶方法简单但是收敛速度慢，** 牛顿方法 ** 就是 $x^{(k+1)} = x^{(k)} - (F^{(k)})^{-1} g^{(k)}$，其收敛速度更快，但是海森矩阵计算代价较大，而 **BB 方法**就是用 $\alpha_k g^{(k)}$ 来近似 $(F^{(k)})^{-1} g^{(k)}$。

定义 $s^k = x^{k+1} - x^k$ 和 $y^{(k-1)} = g^{(k)} - g^{(k-1)}$，那么海森矩阵实际上就是

$$F^{(k)} s^{(k-1)} = y^{(k-1)}$$

用 $(a_k I)^{-1}$ 来近似 $F^{(k)}$，那么就应该有

$$(\alpha_k I)^{-1} s^{(k-1)} = y^{(k-1)}$$

利用最小二乘法：

$$\alpha_k^{-1} = \arg\min_\beta \frac{1}{2} ||s^{(k-1)}\beta - y^{(k-1)}||^2 => \alpha_k^1 = \frac{(s^{k-1})^\top s^{k-1}}{(s^{k-1})^\top y^{k-1}}$$

$$\alpha_k = \arg\min_\alpha \frac{1}{2} ||s^{(k-1)}\beta - y^{(k-1)}\alpha||^2 => \alpha_k^2 = \frac{(s^{k-1})^\top y^{k-1}}{(y^{k-1})^\top y^{k-1}}$$

BB 方法特点：

1) 几乎不需要额外的计算，但是往往会带来极大的性能收益
2) 实际应用中两个表达式都可以用，甚至可以交换使用，但是优劣需结合具体问题
3) 收敛性很难证明。

2.3 Huber 光滑梯度法

将 LASSO 问题转化为光滑函数，

$$\ell_\sigma(x) = \begin{cases} \frac{1}{2\sigma} x^2, & |x| < \sigma; \\ |x| - \frac{\sigma}{2}, & \text{otherwise.} \end{cases}$$

使用 $L_\sigma(x)=\sum_{i=1}^{n}\ell_\sigma(x_i)$ 替代 $||x||_1$, 得到如下优化问题:

$$\min_x f(x):=\frac{1}{2}\|Ax-b\|_2^2+\mu L_\sigma(x).$$

在 x 点处 f 的梯度为:

$$\nabla f(x)=A^\top(Ax-b)+\mu\nabla L_\sigma(x),$$

其中

$$(\nabla L_\sigma(x))_i=\begin{cases}\text{sign}(x_i), & |x_i|>\sigma;\\ \frac{x_i}{\sigma}, & |x_i|\le\sigma.\end{cases}$$

代码解析:

1) 针对光滑化之后的函数进行梯度下降。

2) 内层循环的收敛条件：当当前梯度小于阈值或者目标函数比那化小于阈值，内层迭代终止

3) 采用线搜索循环选择合适步长并更新 x。在步长不符合线搜索条件的情况下，对当前步长以 η 进行衰减，线搜索次数加 1。

2.4 线搜索方法 (LineSearch)

线搜索的迭代过程是 $x_{k+1}=x_k+\alpha_k p_k$, 其中 α_k 和 p_k 分别表示搜索步长和搜索方向。

p_k 是一个下降方向，满足 $\bigtriangledown f_k p_k\le 0$ ，则 $p_k=-B_k^{-1}\bigtriangledown f_k$, B 为对称非奇异矩阵，根据 B_k 的选择会产生以下几个方向:

1) $B_k=I$ 时，搜索方向为负梯度方向，该方法为最速下降方向。

2) $B_k=\bigtriangledown^2 f_k$ 时，该方法为牛顿方法。

3) 需要满足对称正定矩阵，该方法为拟牛顿方法。

3 总结

对 Matlab 的代码采用 Python 重构，对算法的流程有了比较深入的了解，但是对示例代码进行运行时，生成的迭代次数-函数值图像，相比与网站中给出的同等 Matlab 生成图像更快的收敛。

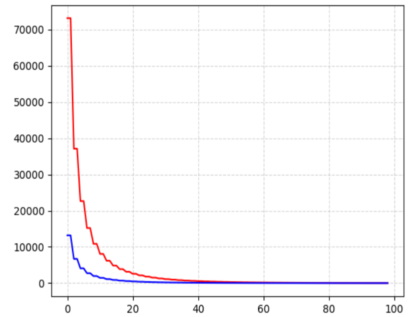

图 1: 利用梯度法解决 LASSO 问题

Github 地址：

https://github.com/Xingyu-Romantic/Machine-learning

参考文献

[1]. 为什么 L 1 正则化导致稀疏解 - CSDN

[2]. 线搜索方法 - CSDN

[3]. 凸优化笔记 15：梯度下降法 - 知乎