Optimizer Algorithm

coorty

2018 年 5 月 6 日

梯度下降(Gradient descent)和最速下降(deepest descent)是同一个求解无约束最优化问题的方法. 已知第k次的参数值$\vec{\theta_k}$, 求下次迭代的参数$\vec{\theta}$时, 可以对其进行一阶泰勒展开, 得到下式:

$$f(\vec{\theta}) \approx f(\vec{\theta_k}) + \vec{g_k^T}(\vec{\theta} - \vec{\theta_k}) \tag{1}$$

其中, $\vec{g_k^T}$表示函数$f(x)$在$\vec{\theta_k}$处的导数, 是个向量哦, 转置成为了行向量, 与后面的列向量相乘后, 变成一个标量.

我们的目的是使得$f(\vec{\theta})$每次变得更小, 因此有: $f(\vec{\theta}) - f(\vec{\theta_k}) = \vec{g_k^T}(\vec{\theta} - \vec{\theta_k}) \leq 0$. 对于两个向量而言, 如果两向量的夹角是180度时(线性, 相反方向), 乘积是最小的, 因此有:

$$\vec{\theta} - \vec{\theta_k} = -\lambda \vec{g}_k \tag{2}$$

因此有:

$$\vec{\theta} = \vec{\theta_k} - \lambda \vec{g}_k \tag{3}$$

牛顿法和拟牛顿法也是求解无约束优化问题的典型方法, 有收敛速度块等优点. 牛顿法和梯度下降法都是迭代求解算法. 和上面类似, 对目标函数进行二阶泰勒展开:

$$f(\vec{\theta}) \approx f(\vec{\theta_k}) + \vec{g_k^T}(\vec{\theta} - \vec{\theta_k}) + \frac{1}{2}(\vec{\theta} - \vec{\theta_k})^T H_k(\vec{\theta} - \vec{\theta_k}) \tag{4}$$

注意, 这里的H_k不是个向量了, 而是一个海森矩阵! 后面整个这项算下来也是一个标量! 有:

$$H(\theta) = [\frac{\partial f}{\partial \theta_i \partial \theta_j}]_{n\times n} \tag{5}$$

牛顿法使用了极小值的必要条件:

$$\triangle f(x) = 0 \tag{6}$$

对$\vec{\theta}$进行求导(注意是对$\vec{\theta}$, 而不是$\vec{\theta_k}$):

$$\frac{\partial f(\vec{\theta})}{\partial \vec{\theta}} = \vec{g}_k + H_k(\vec{\theta} - \vec{\theta_k}) = 0 \tag{7}$$

因此得到梯度更新公式:

$$\vec{\theta} = -\frac{\vec{g}_k}{H_k} + \vec{\theta_k} = \vec{\theta_k} - \frac{\vec{g}_k}{H_k} = \vec{\theta_k} - H_k^{-1}\vec{g}_k \tag{8}$$

公式推导出来了, 是不是还挺直观的, 梯度更新时利用了一阶和二阶导信息. 但是这个H_K^{-1}这项不太好求啊! 牛顿法的优缺点如下:

- 同时利用了一阶和二阶导数信息, 收敛速度更快;c
- 但是牛顿法需要计算海森矩阵的逆, 这点是很慢的!!!

拟牛顿法使用一个n阶的矩阵G_k来对H_k^{-1}进行近似, 令$\vec{\theta} = \vec{\theta}_{k+1}$, 则有:

$$\vec{g}_{k+1} - \vec{g}_k = H_k(\vec{\theta_{k+1}} - \vec{\theta_k}) \tag{9}$$

计$\vec{y_k} = \vec{g}_{k+1} - \vec{g_k}$, $\vec{\delta_k} = \vec{\theta_{k+1}} - \vec{\theta_k}$, 则有:

$$\vec{y_k} = H_k \vec{\delta_k} \tag{10}$$

或:

$$H_k^{-1} \vec{y_k} = \vec{\delta_k} \tag{11}$$

上面这个式子称为"**拟牛顿条件**". 对于我们想选择的替代矩阵G_k, 必须要满足两个条件:

- 满足拟牛顿条件: $G_{k+1}\vec{y_k} = \vec{\delta_k}$
- 所有的G_k都要是正定矩阵(只有正定才能保证牛顿法的搜索方向是往下的)

正定矩阵是一种实对称矩阵. 广义定义是: 设M为n阶方阵, 对于任何非零向量$\vec{z}$, 都有$\vec{z}^T M \vec{z} > 0$, 则称M是正定矩阵.**下面证明满足这两个条件后, 搜索的方向一定是向下的:**

将(8)式的梯度更新公式加上学习率λ之后带回到式(1)中, 注意是式(1)而不是式(4), 一阶展开式便能证明我们的结论了, 有:

$$f(\vec{\theta}) \approx f(\vec{\theta_k}) - \lambda \vec{g}_k^T H_k^{-1} \vec{g}_k \tag{12}$$

因此要求$G_{k+1} = H_k^{-1}$(注意: 这里表示时, 使用G_{k+1}来表示H_k^{-1})是个正定矩阵, 因此$\vec{g}_k^T H_k^{-1} \vec{g}_k > 0$, 当$\lambda$是个充分小的数时, 总有$f(\vec{\theta}) < f(\vec{\theta_k})$, 因此这里保证$G_k$是正定矩阵也就保证了搜索的方向一定是向下的!

如果得到每次迭代的G_k呢? 而且每次迭代G_k都必须是正定矩阵, 而且都要满足(11)式, 可以使用DFP算法.

DFP算法选择G_{k+1}的方法是: 假设每次迭代中矩阵G_{k+1}都是由G_k加上两个附加项:

$$G_{k+1} = G_k + P_k + Q_k \tag{13}$$

参考(11)式, 右边都乘以一个$\vec{y_k}$, 则有:

$$G_{k+1}\vec{y_k} = G_k\vec{y_k} + P_k\vec{y_k} + Q_k\vec{y_k} \tag{14}$$

为了满足(11)式, 令$P_k\vec{y_k} = \vec{\delta_k}$, 则对于剩下的两项, 有:

$$G_k\vec{y_k} = -Q_k\vec{y_k} \tag{15}$$

找到P_k和Q_k不难, 直接令:

$$P_k = \frac{\vec{\delta_k}\vec{\delta_k}^T}{\vec{\delta_k}^T\vec{y_k}} \tag{16}$$

$$Q_k = \frac{G_k\vec{y_k}\vec{y_k}^T G_k}{\vec{y_k}^T G_k \vec{y_k}} \tag{17}$$

可以证明, 只要初始的G_0是正定矩阵, 迭代过程中每个G_k都是正定的!

如果使用(10)式, 利用B_{k+1}来逼近H_k(注意, 不是H_k^{-1}), 就是**BFGS算法**了, 推导起来和DFS算法相似:

$$\vec{y_k} = B_{k+1}\vec{\delta_k} \tag{18}$$

$$B_{k+1} = B_k + P_k + Q_k \tag{19}$$

$$B_{k+1}\vec{\delta}_k = B_k\vec{\delta}_k + P_k\vec{\delta}_k + Q_k\vec{\delta}_k \tag{20}$$

用的套路都是一样的!

总结

- 梯度下降是对代价函数进行一阶泰勒展开, 牛顿法是进行二阶泰勒展开;
- 牛顿法每次迭代的时候需要求海森矩阵H_k的逆, 这点比较逆天, 比较耗时, 因此出现了拟牛顿法;
- 对于拟牛顿法, 都是想用各种方式来对H_k进行近似, DFP算法使用G_{k+1}来近似H_k^{-1}; BFGS算法使用B_{k+1}来近似H_k

References

[1] 统计学习方法　　附录B

[2] https://blog.csdn.net/batuwuhanpei/article/details/51979831