newGLMNET （L1 regularized Logistic Regression）

目标问题

带L1正则化项的逻辑回归问题形式化为($y_i \in \{-1,1\}$):

$$\min_{w} \quad \|w\|_1 + C\sum_{i=1}^{l}\log(1+e^{-y_i w^T x_i}).$$

在liblinear中，为了避免计算$e^{-y_i w^T x_i}$,其改写为：

$$f(w) = \|w\|_1 + C\left(\sum_{i=1}^{l}\log(1+e^{-w^T x_i}) + \sum_{i:y_i=-1} w^T x_i\right)$$

可以将优化目标拆为损失函数和正则化项如下：

$$\min_{\mathbf{w}} \quad f(\mathbf{w}),$$

$$f(\mathbf{w}) \equiv \|\mathbf{w}\|_1 + L(\mathbf{w}),$$

Coordinate Desent（CD) 方法

CDN算法

CDN算法是一种经典的CD算法，其在每轮循环中针对参数向量的每一维去做如下的优化：

$$\min_{d} \quad f(\mathbf{w}^{k,j} + d\mathbf{e}_j) - f(\mathbf{w}^{k,j}),$$

$$f(\mathbf{w}^{k,j} + d\mathbf{e}_j) - f(\mathbf{w}^{k,j}) = \|\mathbf{w}^{k,j} + d\mathbf{e}_j\|_1 - \|\mathbf{w}^{k,j}\|_1 + L(\mathbf{w}^{k,j} + d\mathbf{e}_j) - L(\mathbf{w}^{k,j}).$$

其中：

$$\mathbf{w}^{k,j} \equiv [w_1^{k+1}, \ldots, w_{j-1}^{k+1}, w_j^k, \ldots, w_n^k]^T$$

每次的更新公式为：

$$w_t^{k,j+1} = \begin{cases} w_t^{k,j} + d & \text{if } t = j, \\ w_t^{k,j} & \text{otherwise.} \end{cases}$$

而在逻辑回归中，上述优化问题得不到闭式解，因此可以考虑对损失函数的差$L(w^{k,j}+de_j)-L(w^{k,j})$用其二阶近似：

$$\min_{d} \quad \nabla_j L(\mathbf{w}^{k,j})d+\frac{1}{2}\nabla^2_{jj}L(\mathbf{w}^{k,j})d^2+|w^k_j+d|-|w^k_j|.$$

对于该二次规划问题求得最优解即可求得d,之后只需在此方向上做线搜索确定最优步长$\bar{\lambda}$即可得到更新公式：

$$e^{\mathbf{w}^T\mathbf{x}_i} \leftarrow e^{\mathbf{w}^T\mathbf{x}_i}\cdot e^{\bar{\lambda}dx_{ij}}, \forall i,$$

缺点

Sigmoid中的指数和对数运算的计算时间开销很大。尤其在解决逻辑回归时，对于一般的CD方法会每轮针对每一维计算一次损失函数的梯度和Hessian阵$\nabla L(w)$, $\nabla^2 L(w)$,而后再执行一次线搜索的过程中也需要计算目标函数值$f(x)$,导致这种方法需要执行大量次数的指数/对数运算，每一轮迭代至少需要$O(nl)$的指数运算。实验也证明传统CD方法中指数运算的时间占据了大头，因此如何缩减指数运算的次数也就是newGLMNET方法提出的出发点。

Data set	exp/log	Total
epsilon	64.25 (73.0%)	88.18
webspam	72.89 (66.6%)	109.39

Table 1: Timing analysis of the first CD cycle of CDN. Time is in seconds.

GLMNET

GLMNET方法是另一种迭代方法，其每轮的优化目标为：

$$\min_{\mathbf{d}} \quad q_k(\mathbf{d}),$$

$$q_k(\mathbf{d}) \equiv \nabla L(\mathbf{w}^k)^T\mathbf{d}+\frac{1}{2}\mathbf{d}^T H^k\mathbf{d}+\|\mathbf{w}^k+\mathbf{d}\|_1-\|\mathbf{w}^k\|_1,$$

其中H^k可以是Hessian阵，也可以是Hessian阵的近似。这里取Hessian阵，$H^k=\nabla^2 L(w^k)$。每轮的参数更新为：

$$\mathbf{w}^{k+1} \leftarrow \mathbf{w}^k+\mathbf{d}.$$

而对于如何求解上述优化问题，其也使用了CD方法，即每轮最小化如下的目标函数：

$$\begin{aligned}&q_k(\mathbf{d}^{p,j}+z\mathbf{e}_j)-q_k(\mathbf{d}^{p,j})\\ =&|w^k_j+d^p_j+z|-|w^k_j+d^p_j|+\nabla_j\bar{q}_k(\mathbf{d}^{p,j})z+\frac{1}{2}\nabla^2_{jj}\bar{q}_k(\mathbf{d}^{p,j})z^2,\end{aligned}$$

其中$d^{p,j}$同之前的$w^{p,j}$一样：

$$\mathbf{d}^{p,j} \equiv [d_1^{p-1}, d_2^{p-1}, \ldots, d_{j-1}^{p-1}, d_j^p, \ldots, d_n^p]^T,$$

且$\bar{q}_k(d)$为：

$$\bar{q}_k(\mathbf{d}) \equiv \nabla L(\mathbf{w}^k)^T \mathbf{d} + \frac{1}{2}(\mathbf{d})^T \nabla^2 L(\mathbf{w}^k)\mathbf{d}$$

可以看出，其就是一个不再针对内层循环的每一维去依次更新$\nabla L(w^k)$，而是一轮中内层循环结束后一起更新整个参数向量w的CDN算法，因此其相比于CDN算法可以每轮节省依次去计算每一维的n倍时间开销，其指数/对数计算的时间复杂度为$O(l)$。同时，因为上述问题为二次规划问题，也是有闭式最优解的，因此GLMNET算法中没有去做线搜索，节省了线搜索中计算目标函数$f(x)$的时间开销。

然而，因为其没有做线搜索，GLMNET理论上没有收敛的保证；同时在实际测试中比CDN算法更慢。而在实际观察中发现GLMNET算法在前期在全局上收敛慢，而后期在局部收敛快。

newGLMNET

基本思想

因此作者将两者结合，提出了有收敛性理论保证，但又指数/对数计算时间复杂度低的newGLMNET算法。

根据Tseng和Yun(2009)的研究，GLMNET算法只要确保H^k为正定、且加入线搜索，就一定是收敛的。对于确保H^k正定，其采用加入很小的单位阵的形式,其中ν是很小的正数：

$$H^k \equiv \nabla^2 L(\mathbf{w}^k) + \nu I,$$

对于线搜索，其搜索到的步长λ需满足：

$$\begin{aligned} & f(\mathbf{w}^k + \lambda \mathbf{d}) - f(\mathbf{w}^k) \\ \le\ & \sigma\lambda\left(\nabla L(\mathbf{w}^k)^T \mathbf{d} + \gamma \mathbf{d}^T H^k \mathbf{d} + \|\mathbf{w}^k + \mathbf{d}\|_1 - \|\mathbf{w}^k\|_1\right), \end{aligned}$$

其中$0 < \sigma, \gamma < 1$，是给定参数，newGLMNET中直接取$\gamma = 0$.

在liblinear实现中，上述条件为：

$$\begin{aligned} & f(\boldsymbol{w} + \beta^t \boldsymbol{d}) - f(\boldsymbol{w}) \\ =\ & \|\boldsymbol{w} + \beta^t \boldsymbol{d}\|_1 - \|\boldsymbol{w}\|_1 + C\left(\sum_{i=1}^{l} \log\left(\frac{1 + e^{-(\boldsymbol{w}+\beta^t \boldsymbol{d})^T \boldsymbol{x}_i}}{1 + e^{-\boldsymbol{w}^T \boldsymbol{x}_i}}\right) + \beta^t \sum_{i:y_i=-1} \boldsymbol{d}^T \boldsymbol{x}_i\right) \\ =\ & \|\boldsymbol{w} + \beta^t \boldsymbol{d}\|_1 - \|\boldsymbol{w}\|_1 + C\left(\sum_{i=1}^{l} \log\left(\frac{e^{(\boldsymbol{w}+\beta^t \boldsymbol{d})^T \boldsymbol{x}_i} + 1}{e^{(\boldsymbol{w}+\beta^t \boldsymbol{d})^T \boldsymbol{x}_i} + e^{\beta^t \boldsymbol{d}^T \boldsymbol{x}_i}}\right) + \beta^t \sum_{i:y_i=-1} \boldsymbol{d}^T \boldsymbol{x}_i\right) \\ \le\ & \sigma\beta^t \left(\nabla L(\boldsymbol{w})^T \boldsymbol{d} + \|\boldsymbol{w} + \boldsymbol{d}\|_1 - \|\boldsymbol{w}\|_1\right), \end{aligned}$$

然而，线搜索中需对不同步长λ计算目标函数值$f(w^k + \lambda d)$,直接计算的复杂度显然是$O(nl)$的；然而我们可以提前维护一个保存Xd的缓存，并每次更新：

$$(X\mathbf{d}^{p,j+1})_i \leftarrow (X\mathbf{d}^{p,j})_i + X_{ij}z, \ \forall i,$$

这样，假设我们也记录了之前的$e^{w^T x_i}$,在计算$f(w^k + \lambda d)$时只需：

$$e^{(\mathbf{w}^k+\lambda\mathbf{d})^T\mathbf{x}_i} = e^{(\mathbf{w}^k)^T\mathbf{x}_i} \cdot e^{\lambda(X\mathbf{d})_i}.$$

因此上述开销为$O(l)$，线搜索的加入并不会导致复杂度的增高，其开销是相对较小的。

可以说，newGLMNET基本上就是GLMNET+确保H^k正定+线搜索+算法细节优化。

终止条件

GLMNET算法本身终止条件的设置比较粗糙，因此会导致两层迭代的次数过多，在算法运行前期收敛速度较慢。因此，为了使得运行早期时更倾向于CDN算法在全局快速收敛，后期倾向于Newton-like的方法，使用二阶信息在局部快速收敛，newGLMNET的内层终止条件为：

$$\sum_{j=1}^{n} |\nabla_j^S q_k(\mathbf{d}^{p,j})| \leq \varepsilon_{\text{in}},$$

其中$\nabla^S q(d)$是一个最小范数的次梯度：

$$\nabla_j^S q(\mathbf{d}) \equiv \begin{cases} \nabla_j \bar{q}(\mathbf{d}) + 1 & \text{if } w_j + d_j > 0, \\ \nabla_j \bar{q}(\mathbf{d}) - 1 & \text{if } w_j + d_j < 0, \\ \operatorname{sgn}(\nabla_j \bar{q}(\mathbf{d})) \max(|\nabla_j \bar{q}(\mathbf{d})| - 1, 0) & \text{if } w_j + d_j = 0. \end{cases}$$

其依然能确保$\nabla^S q(d) = 0$ 当且仅当 d是最优解。

而如果只经过一轮，条件就被触发，则会对ϵ_{in}做指数衰减：

$$\varepsilon_{\text{in}} \leftarrow \varepsilon_{\text{in}}/4.$$

这使得模型可以自己调整ϵ_{in},也即根据轮数逐渐提升精度，也使得我们可以不在意初始ϵ_{in}的设置，可以直接赋一个大值。同时也可达到前期内层循环数量多，倾向于CDN算法快速在全局收敛，后期稳定在内层执行一轮循环上下，趋向于用外层循环的Hessian阵信息局部快速收敛的效果。

对于外层循环，停止条件类似地为：

$$\sum_{j=1}^{n} |\nabla_j^S f(\mathbf{w}^k)| \leq \varepsilon_{\text{out}}.$$

Shrinking

为了提高计算效率，newGLMNET借鉴CDN也加入了shrinking,也即假设对于某些等于0同时满足最优解条件的参数已经达到最优，在之后迭代中不会改变，因此提前固定这些参数为0不变，从而缩小问题规模，提升计算效率。newGLMNET是一个两层循环的算法，其shrinking也是两层的：

1. 在外层，针对w优化$||w||_1 + L(w)$,w的最优条件为：

$$-1 < \nabla_j L(\mathbf{w}^*) < 1 \quad \text{implies} \quad w_j^* = 0.$$

其选择满足如下条件的w中的元素停止更新:

$$w_j^k = 0 \quad \text{and} \quad -1 + \frac{M^{\text{out}}}{l} < \nabla_j L(\mathbf{w}^k) < 1 - \frac{M^{\text{out}}}{l},$$

$$M^{\text{out}} \equiv \max\left(\left|\nabla_1^S f(\mathbf{w}^{k-1})\right|, \ldots, \left|\nabla_n^S f(\mathbf{w}^{k-1})\right|\right).$$

2. 在内层，针对d优化$q_k(d)$,其选择满足如下条件的d中的元素停止更新:

$$w_j^k + d_j^{p,t} = 0 \quad \text{and} \quad -1 + \frac{M^{\text{in}}}{l} < \nabla_j \bar{q}_k(\mathbf{d}^{p,t}) < 1 - \frac{M^{\text{in}}}{l},$$

$$M^{\text{in}} \equiv \max\left(\left|\nabla_{j_1}^S q_k(\mathbf{d}^{p-1,1})\right|, \ldots, \left|\nabla_{j_{|J^p|}}^S q_k(\mathbf{d}^{p-1,|J^p|})\right|\right)$$

注意到这里只对为0的参数，且使用比$(-1,1)$更小的区间上作为判断标准，是一个相对保守的shrinking策略。

newGLMNET算法

因此newGLMNET算法即为一个两层循环的算法:

Algorithm 3 Overall procedure of newGLMNET

- Given $\mathbf{w}^1$, ε_{in}, and ε_{out}. Choose a small positive number ν. Choose $\beta \in (0,1)$, $\gamma \in [0,1)$, and $\sigma \in (0,1)$.
- Let $M^{\text{out}} \leftarrow \infty$.
- For $k = 1,2,3,\ldots$ // outer iterations
  1. Let $J \leftarrow \{1,\ldots,n\}$, $M \leftarrow 0$, and $\bar{M} \leftarrow 0$.
  2. For $j = 1,\ldots,n$
     - 2.1. Calculate H_{jj}^k, $\nabla_j L(\mathbf{w}^k)$ and $\nabla_j^S f(\mathbf{w}^k)$.
     - 2.2. If $w_j^k = 0$ and $|\nabla_j L(\mathbf{w}^k)| < 1 - M^{\text{out}}/l$ // outer-level shrinking
       $J \leftarrow J \backslash \{j\}$.
       Else
       $M \leftarrow \max(M, |\nabla_j^S f(\mathbf{w}^k)|)$ and $\bar{M} \leftarrow \bar{M} + |\nabla_j^S f(\mathbf{w}^k)|$.
  3. If $\bar{M} \leq \varepsilon_{\text{out}}$
     `return` $\mathbf{w}^k$.
  4. Let $M^{\text{out}} \leftarrow M$.
  5. Get $\mathbf{d}$ and update ε_{in} by solving sub-problem (13) by Algorithm 4.
  6. Compute $\lambda = \max\{1, \beta, \beta^2, \ldots\}$ such that $\lambda\mathbf{d}$ satisfies (20).
  7. $\mathbf{w}^{k+1} = \mathbf{w}^k + \lambda\mathbf{d}$.

Algorithm 4 Inner iterations of newGLMNET with shrinking

- Given working set J, initial solution $\mathbf{d}$, inner stopping condition ε_{in}, and a small positive number ν from the outer problem.
- Let $M^{\text{in}} \leftarrow \infty$, $T \leftarrow J$, and $\mathbf{d} \leftarrow \mathbf{0}$.
- For $p = 1, 2, 3, \ldots, 1000$ // inner iterations
  1. Let $m \leftarrow 0$ and $\bar{m} \leftarrow 0$.
  2. For $j \in T$
     - Let $\nabla^2_{jj}\bar{q}_k(\mathbf{d}) = H^k_{jj}$. Calculate $\nabla_j \bar{q}_k(\mathbf{d})$ and $\nabla^S_j q_k(\mathbf{d})$.
     - If $w^k_j + d_j = 0$ and $|\nabla_j \bar{q}_k(\mathbf{d})| < 1 - M^{\text{in}}/l$ // inner-level shrinking

       $T \leftarrow T \backslash \{j\}$.

       Else

       $m \leftarrow \max(m, |\nabla^S_j q_k(\mathbf{d})|)$ and $\bar{m} \leftarrow \bar{m} + |\nabla^S_j q_k(\mathbf{d})|$.

       $d_j \leftarrow d_j + \arg\min_z q_k(\mathbf{d} + z\mathbf{e}_j) - q_k(\mathbf{d})$.
  3. If $\bar{m} \leq \varepsilon_{\text{in}}$
     - If $T = J$ // inner stopping

       `break`.

       Else // active set reactivation

       $T \leftarrow J$ and $M^{\text{in}} \leftarrow \infty$.

     Else
     - $M^{\text{in}} \leftarrow m$.
- If $p = 1$, then $\varepsilon_{\text{in}} \leftarrow \varepsilon_{\text{in}}/4$.