Notes on L1 Regularized Logistic Regression

Max Leung

November 22, 2022

Disclaimer: This note can contain mistake, typo, error, etc.

1 Subgradient Optimality

If f is a convex function of x (not necessarily differentiable) and $dom(f) = R^n$, we have

$$x^* = argmin_x f(x) \iff 0 \in \partial f(x^*)$$

Proof: $(\Leftarrow)$ $0 \in \partial f(x^*) := \{g \in R^n : f(y) \geq f(x^*) + g'(y-x) \text{ for } \forall y\} \Rightarrow f(y) \geq f(x^*) + 0'(y-x)$ for $\forall y \Rightarrow f(y) \geq f(x^*)$ for $\forall y \Rightarrow x^*$ is minimizer.
$(\Rightarrow)$ $x^* = argmin_x f(x) \Rightarrow f(y) \geq f(x^*)$ for $\forall y \Rightarrow f(y) \geq f(x^*) + 0'(y-x)$ for $\forall y \Rightarrow 0 \in \partial f(x^*)$ Q.E.D.

2 Logistic Regression Without Constraint

2.1 Log Likelihood Function

$$lnL(\boldsymbol{\beta}; \boldsymbol{y}, \boldsymbol{X}) = ln \prod_{i=1}^{N} Pr(y_i|\boldsymbol{x}_i; \boldsymbol{\beta}) = \sum_{i=1}^{N} lnPr(y_i|\boldsymbol{x}_i; \boldsymbol{\beta}) \qquad \text{assume independence of } y_i|\boldsymbol{x}_i$$

$Pr(y_i = 1|\boldsymbol{x}_i; \boldsymbol{\beta}) = logistic(\boldsymbol{x}_i'\boldsymbol{\beta}) := \frac{e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}$ and $Pr(y_i = 0|\boldsymbol{x}_i; \boldsymbol{\beta}) = 1 - Pr(y_i = 1|\boldsymbol{x}_i; \boldsymbol{\beta}) = 1 - \frac{e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}} = \frac{1}{1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}} = \frac{e^0}{1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}$

$$= \sum_{i=1}^{N} ln \frac{e^{\boldsymbol{x}_i'\boldsymbol{\beta} y_i}}{1 + e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}$$
$$= \sum_{i=1}^{N} \{\boldsymbol{x}_i'\boldsymbol{\beta} y_i - ln(1 + e^{\boldsymbol{x}_i'\boldsymbol{\beta}})\}$$

2.2 Gradient Vector

$$\begin{aligned}
\frac{\partial lnL(\boldsymbol{\beta}; \boldsymbol{y}, \boldsymbol{X})}{\partial \boldsymbol{\beta}} &= \frac{\partial \sum_{i=1}^{N} \{\boldsymbol{x}_i'\boldsymbol{\beta} y_i - ln(1 + e^{\boldsymbol{x}_i'\boldsymbol{\beta}})\}}{\partial \boldsymbol{\beta}} \\
&= \sum_{i=1}^{N} \{y_i \frac{\partial \boldsymbol{x}_i'\boldsymbol{\beta}}{\partial \boldsymbol{\beta}} - \frac{\partial ln(1 + e^{\boldsymbol{x}_i'\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}}\} \\
&= \sum_{i=1}^{N} \{y_i \boldsymbol{x}_i - \frac{1}{1 + e^{\boldsymbol{x}_i'\boldsymbol{\beta}}} e^{\boldsymbol{x}_i'\boldsymbol{\beta}} \boldsymbol{x}_i\} \\
&= \sum_{i=1}^{N} \{y_i \boldsymbol{x}_i - logistic(\boldsymbol{x}_i'\boldsymbol{\beta}) \boldsymbol{x}_i\} \\
&= \sum_{i=1}^{N} \{y_i - \underbrace{Pr(y_i = 1|\boldsymbol{x}_i; \boldsymbol{\beta})}_{p_i}\} \boldsymbol{x}_i \qquad \{y_i - Pr(y_i = 1|\boldsymbol{x}_i; \boldsymbol{\beta})\}\boldsymbol{x}_i \text{ is score function} \\
&= \sum_{i=1}^{N} \begin{pmatrix} (y_i - p_i)x_{1i} \\ \vdots \\ (y_i - p_i)x_{pi} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{N} (y_i - p_i)x_{1i} \\ \vdots \\ \sum_{i=1}^{N} (y_i - p_i)x_{pi} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1'(\boldsymbol{y} - \boldsymbol{p}) \\ \vdots \\ \boldsymbol{x}_p'(\boldsymbol{y} - \boldsymbol{p}) \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1' \\ \vdots \\ \boldsymbol{x}_p' \end{pmatrix} (\boldsymbol{y} - \boldsymbol{p}) = \boldsymbol{X}'(\boldsymbol{y} - \boldsymbol{p})
\end{aligned}$$

2.3 Hessian Matrix

$$\begin{aligned}
\frac{\partial^2 lnL}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}'} &= \frac{\partial \sum_{i=1}^N \{y_i\boldsymbol{x}_i - \frac{1}{1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}e^{\boldsymbol{x}_i'\boldsymbol{\beta}}\boldsymbol{x}_i\}}{\partial\boldsymbol{\beta}'}\\
&= \sum_{i=1}^N \frac{\partial\{y_i\boldsymbol{x}_i - \frac{1}{1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}e^{\boldsymbol{x}_i'\boldsymbol{\beta}}\boldsymbol{x}_i\}}{\partial\boldsymbol{\beta}'}\\
&= -\sum_{i=1}^N \boldsymbol{x}_i \frac{\partial(1+e^{-\boldsymbol{x}_i'\boldsymbol{\beta}})^{-1}}{\partial\boldsymbol{\beta}'}\\
&= -\sum_{i=1}^N \boldsymbol{x}_i[-(1+e^{-\boldsymbol{x}_i'\boldsymbol{\beta}})^{-2}e^{-\boldsymbol{x}_i'\boldsymbol{\beta}}(-1)\boldsymbol{x}_i']\\
&= -\sum_{i=1}^N \boldsymbol{x}_i(1+e^{-\boldsymbol{x}_i'\boldsymbol{\beta}})^{-1}\frac{e^{-\boldsymbol{x}_i'\boldsymbol{\beta}}}{1+e^{-\boldsymbol{x}_i'\boldsymbol{\beta}}}\boldsymbol{x}_i'\\
&= -\sum_{i=1}^N \boldsymbol{x}_i logistic(\boldsymbol{x}_i'\boldsymbol{\beta})\frac{1}{e^{\boldsymbol{x}_i'\boldsymbol{\beta}}+1}\boldsymbol{x}_i'\\
&= -\sum_{i=1}^N \boldsymbol{x}_i logistic(\boldsymbol{x}_i'\boldsymbol{\beta})\frac{1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}-e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}}\boldsymbol{x}_i'\\
&= -\sum_{i=1}^N \boldsymbol{x}_i logistic(\boldsymbol{x}_i'\boldsymbol{\beta})(1-logistic(\boldsymbol{x}_i'\boldsymbol{\beta}))\boldsymbol{x}_i'\\
&= -\sum_{i=1}^N \boldsymbol{x}_i\boldsymbol{x}_i'\overbrace{Pr(y_i=1|\boldsymbol{x}_i;\boldsymbol{\beta})}^{p_i}(1-Pr(y_i=1|\boldsymbol{x}_i;\boldsymbol{\beta}))\\
&= -\sum_{i=1}^N \begin{pmatrix} x_{1i}x_{1i}p_i(1-p_i) & \cdots & x_{1i}x_{pi}p_i(1-p_i)\\ \vdots & \ddots & \vdots \\ x_{pi}x_{1i}p_i(1-p_i) & \cdots & x_{pi}x_{pi}p_i(1-p_i)\end{pmatrix}\\
&= -\begin{pmatrix} \boldsymbol{x}_1'\boldsymbol{W}\boldsymbol{x}_1 & \cdots & \boldsymbol{x}_1'\boldsymbol{W}\boldsymbol{x}_p\\ \vdots & \ddots & \vdots \\ \boldsymbol{x}_p'\boldsymbol{W}\boldsymbol{x}_1 & \cdots & \boldsymbol{x}_p'\boldsymbol{W}\boldsymbol{x}_p\end{pmatrix} && \text{where } (\boldsymbol{W})_{ii} = p_i(1-p_i) \text{ and } (\boldsymbol{W})_{ij} = 0 \text{ for } \forall i \neq j\\
&= -\begin{pmatrix}\boldsymbol{x}_1'\\ \vdots \\ \boldsymbol{x}_p'\end{pmatrix}\boldsymbol{W}\begin{pmatrix}\boldsymbol{x}_1\cdots\boldsymbol{x}_p\end{pmatrix} = -\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X}
\end{aligned}$$

2.4 Iteratively Reweighted Least Squares (IRLS)

Note that Newton Method is $x^+ = x - (\nabla^2 f(x))^{-1}\nabla f(x)$. If we apply it here:

$$\begin{aligned}
\boldsymbol{\beta}^+ &= \boldsymbol{\beta} - (\nabla^2 lnL(\boldsymbol{\beta}))^{-1}\nabla lnL(\boldsymbol{\beta})\\
&= \boldsymbol{\beta} - (-\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'(\boldsymbol{y}-\boldsymbol{p}) && \text{using above results}\\
&= \boldsymbol{I}\boldsymbol{\beta} + (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{I}(\boldsymbol{y}-\boldsymbol{p})\\
&= (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}(\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})\boldsymbol{\beta} + (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{W}^{-1}(\boldsymbol{y}-\boldsymbol{p})\\
&= (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{W}\underbrace{(\boldsymbol{X}\boldsymbol{\beta}+\boldsymbol{W}^{-1}(\boldsymbol{y}-\boldsymbol{p}))}_{\boldsymbol{z}}
\end{aligned}$$

Thus, $\boldsymbol{\beta}^+$ is a Weighted Least Squares estimator i.e.,

$$\boldsymbol{\beta}^+ = argmin_{\boldsymbol{\beta}}(\boldsymbol{z}-\boldsymbol{X}\boldsymbol{\beta})'\boldsymbol{W}(\boldsymbol{z}-\boldsymbol{X}\boldsymbol{\beta}) = argmin_{\boldsymbol{\beta}}\sum_{i=1}^N w_i(z_i-\boldsymbol{x}_i'\boldsymbol{\beta})^2 \qquad \text{where } w_i = p_i(1-p_i)$$

We loop $\boldsymbol{\beta}^+ = (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{z}$ until $\boldsymbol{\beta}$ converge. Note that $\boldsymbol{W}$ and $\boldsymbol{z}$ is a function of $\boldsymbol{\beta}$. Thus, minimizing Unpenalized Logistic Regression's negative log likelihood function is the same as repeatedly minimizing weighted least squares functions.

3 L1 Regularized Logistic Regression

3.1 Minimize Negative Log Likelihood Function with L1 Constraint

$$min_{\boldsymbol{\beta}}\{-lnL(\boldsymbol{\beta};\boldsymbol{y},\boldsymbol{X})+\lambda||\boldsymbol{\beta}||_1\}$$

$$\iff min_{\boldsymbol{\beta}}\{-\sum_{i=1}^{N}(\boldsymbol{x}_i'\boldsymbol{\beta}y_i-ln(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}))+\lambda||\boldsymbol{\beta}||_1\}$$

3.2 Apply Subgradient Optimality

As $-\sum_{i=1}^{N}(\boldsymbol{x}_i'\boldsymbol{\beta}y_i-ln(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}))+\lambda||\boldsymbol{\beta}||_1$ is a convex function of $\boldsymbol{\beta}$, Subgradient Optimality implies that

$$\boldsymbol{\beta}^*=argmin_{\boldsymbol{\beta}}\{-\sum_{i=1}^{N}(\boldsymbol{x}_i'\boldsymbol{\beta}y_i-ln(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}}))+\lambda||\boldsymbol{\beta}||_1\} \iff 0\in\partial\{-\sum_{i=1}^{N}(\boldsymbol{x}_i'\boldsymbol{\beta}^*y_i-ln(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}^*}))+\lambda||\boldsymbol{\beta}^*||_1\}$$

$$\begin{aligned}
0 &\in \partial\{-\sum_{i=1}^{N}(\boldsymbol{x}_i'\boldsymbol{\beta}^*y_i-ln(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}^*}))+\lambda||\boldsymbol{\beta}^*||_1\} \\
&\in -\partial\sum_{i=1}^{N}(\boldsymbol{x}_i'\boldsymbol{\beta}^*y_i-ln(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}^*}))+\lambda\partial||\boldsymbol{\beta}^*||_1 \\
&\in -\boldsymbol{X}'(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*))+\lambda\partial||\boldsymbol{\beta}^*||_1 && \text{Apply Gradient Vector result above} \\
\boldsymbol{X}'(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*)) &\in \lambda\partial||\boldsymbol{\beta}^*||_1
\end{aligned}$$

We want to pick $\boldsymbol{v}\in\partial||\boldsymbol{\beta}^*||_1$ such that $\boldsymbol{X}'(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*))=\lambda\boldsymbol{v}$

$$\begin{aligned}
\boldsymbol{X}'(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*)) &= \lambda\boldsymbol{v} \\
\begin{pmatrix}\boldsymbol{x}_1' \\ \vdots \\ \boldsymbol{x}_p'\end{pmatrix}(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*)) &= \lambda\boldsymbol{v} \\
\begin{pmatrix}\boldsymbol{x}_1'(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*)) \\ \vdots \\ \boldsymbol{x}_p'(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*))\end{pmatrix} &= \lambda\begin{pmatrix}v_1 \\ \vdots \\ v_p\end{pmatrix}
\end{aligned}$$

So, we have $\boldsymbol{x}_j'(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*))=\lambda v_j$ for $\forall j\in\{1,\cdots,p\}$
As $\boldsymbol{v}\in\partial||\boldsymbol{\beta}^*||_1$, we have

$$v_j\in\partial|\beta_j^*|=\begin{cases}\{sign(\beta_j^*)\} & \text{if } \beta_j^*\neq 0 \\ [-1,1] & \text{if } \beta_j^*=0\end{cases}$$

if $\beta_j^*\neq 0$

$$v_j\in\partial|\beta_j^*|=\{sign(\beta_j^*)\}$$

that is

$$v_j=sign(\beta_j^*) \qquad \{sign(\beta_j^*)\} \text{ is singleton}$$

Thus, we have

$$\boldsymbol{x}_j'(\boldsymbol{y}-\boldsymbol{p}(\boldsymbol{\beta}^*))=\lambda sign(\beta_j^*)$$

This is Equation 4.32 on page 126 of the famous book The Elements of Statistical Learning (2nd ed.)

4 Algorithm for Solving L1 Regularized Logistic Regression

4.1 Proximal-Newton Iterative Approach in R package glmnet (Friedman et al., 2010)

As mentioned in Section 2.4, Unpenalized Logistic Regression Problem can be solved by repeatedly solving Weighted Least Squares Problems with working response $\boldsymbol{z} = \boldsymbol{X\beta} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{p})$ and weight matrix $\boldsymbol{W}$ where $(\boldsymbol{W})_{ii} = p_i(1-p_i)$ and $(\boldsymbol{W})_{ij} = 0$ for $\forall i \neq j$. Similarly, solving

$$min_{\boldsymbol{\beta}}\{-\sum_{i=1}^{N}(\boldsymbol{x}_i'\boldsymbol{\beta}y_i - ln(1+e^{\boldsymbol{x}_i'\boldsymbol{\beta}})) + \lambda||\boldsymbol{\beta}||_1\} \qquad \text{L1 Regularized Logistic Regression Problem}$$

is equivalent as repeatedly solving

$$min_{\boldsymbol{\beta}}\{\sum_{i=1}^{N} w_i(z_i - \boldsymbol{x}_i'\boldsymbol{\beta})^2 + \lambda||\boldsymbol{\beta}||_1\} \qquad \text{Weighted Lasso Regression Problem}$$

with $\widetilde{\boldsymbol{\beta}}$ from last step, where $z_i = \boldsymbol{x}_i'\widetilde{\boldsymbol{\beta}} + \frac{y_i - p_i}{w_i}$ and $w_i = p_i(1-p_i)$ and $p_i = \widetilde{Pr(y_i = 1}|\boldsymbol{x}_i;\boldsymbol{\beta}) = logistic(\boldsymbol{x}_i'\widetilde{\boldsymbol{\beta}})$. Here, negative log likelihood function is locally and quadratically approximated by weighted least squares function.

The well-known solution for Weighted Lasso Regression is (I may write a note on this if I have time):

$$\begin{aligned}
\beta_j^* &= S_{\lambda/\boldsymbol{x}_j'\boldsymbol{W}\boldsymbol{x}_j}\left(\frac{\boldsymbol{x}_j'\boldsymbol{W}\overbrace{(\boldsymbol{z} - \boldsymbol{X}_{-j}\boldsymbol{\beta}_{-j}^*)}^{\boldsymbol{r}^{(j)}}}{\boldsymbol{x}_j'\boldsymbol{W}\boldsymbol{x}_j}\right) && \boldsymbol{x}_j \text{ is column } j \text{ of } \boldsymbol{X} \\
&= S_{\lambda/<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}\left(\frac{<\boldsymbol{x}_j, \boldsymbol{r}^{(j)}>_w}{<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}\right) && <.,.>_w \text{ is an inner product with weight} \\
&= sign\left(\frac{<\boldsymbol{x}_j, \boldsymbol{r}^{(j)}>_w}{<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}\right)\left(\left|\frac{<\boldsymbol{x}_j, \boldsymbol{r}^{(j)}>_w}{<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}\right| - \lambda/<\boldsymbol{x}_j,\boldsymbol{x}_j>_w\right)_+
\end{aligned}$$

where $S_{\lambda/<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}(.)$ is soft-threshold operator with parameter $\lambda/<\boldsymbol{x}_j,\boldsymbol{x}_j>_w$

As $(z)_+ = 0$ if $z \leq 0$, It is obvious that

$$\beta_j^* = \begin{cases} 0 & \text{if } |\frac{<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w}{<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}| \leq \lambda/<\boldsymbol{x}_j,\boldsymbol{x}_j>_w \\ S_{\lambda/<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}(\frac{<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w}{<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}) & \text{if } |\frac{<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w}{<\boldsymbol{x}_j,\boldsymbol{x}_j>_w}| > \lambda/<\boldsymbol{x}_j,\boldsymbol{x}_j>_w \end{cases}$$

If variables are standardized such that $<\boldsymbol{x}_j,\boldsymbol{x}_j>_w = 1$, we have a special case

$$\beta_j^* = \begin{cases} 0 & \text{if } |<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w| \leq \lambda \\ S_\lambda(<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w) & \text{if } |<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w| > \lambda \end{cases}$$

This is Equation 16.15 on page 315 of the book "Computer Age Statistical Inference".

Given λ and β_l^* for $l \in \{1,\cdots,p\}\backslash\{j\}$, β_j^* is computed by firstly computing the weighted inner product $<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w$ and then put it in soft-threshold operator $S_\lambda(.)$. The process repeats for $\forall j \in \{1,\cdots,p\}$ again and again until convergence. It is called Coordinate Descent, which is fast as for each step it essentially only requires calculating an inner product and performing soft-threshold operation.

The remaining question is how to determine λ. In this algorithm, we pre-determine 100 λs:

$$\lambda_{max} := \lambda_1 > \lambda_2 > \cdots > \lambda_{100} = \epsilon\lambda_{max} > 0$$

where $0 < \epsilon < 1$

We choose λ_{max} such that $\beta_j^* = 0$ for $\forall j \in \{1,\cdots,p\}$. This can be achieved by $|<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w| \leq \lambda_{max}$ for $\forall j \in \{1,\cdots,p\}$ i.e., λ_{max} is the upper bound of the set $\{|<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w|\}_{j=1}^p$. One obvious candidate for λ_{max} is the least upper bound $sup_j|<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w|$. If all betas are zero, $\boldsymbol{r}^{(j)} = \boldsymbol{W}^{-1}(\boldsymbol{y} - \bar{y}\boldsymbol{1})$ and $<\boldsymbol{x}_j,\boldsymbol{r}^{(j)}>_w = \boldsymbol{x}_j'\boldsymbol{W}\boldsymbol{r}^{(j)} = \boldsymbol{x}_j'\boldsymbol{W}\boldsymbol{W}^{-1}(\boldsymbol{y} - \bar{y}\boldsymbol{1}) =$

$\boldsymbol{x}_j'(\boldsymbol{y} - \bar{y}\mathbf{1}) = <\boldsymbol{x}_j, \boldsymbol{y} - \bar{y}\mathbf{1}>$. Thus, $\lambda_{max} = sup_j| <\boldsymbol{x}_j, \boldsymbol{y} - \bar{y}\mathbf{1}> |$. The set of non-zero betas (active set) becomes larger and larger when λ decreases from λ_1 to λ_{100}.

To summarize, the Pathwise Coordinate Descent algorithm is:

For $k = 1$ to 100:
  For $m = 1 \cdots$ until $\boldsymbol{\beta}$ converge
    Use $\boldsymbol{\beta}_{m-1}$ or initial value if $m = 1$
    update $p_i = Pr(y_i = 1|\boldsymbol{x}_i; \boldsymbol{\beta}_{m-1}) = logistic(\boldsymbol{x}_i'\boldsymbol{\beta}_{m-1})$
    update $w_i = p_i(1 - p_i)$
    update $z_i = \boldsymbol{x}_i'\boldsymbol{\beta}_{m-1} + \frac{y_i - p_i}{w_i}$
    Find $\boldsymbol{\beta}_m$ by Coordinate Descent:
      Use $\boldsymbol{\beta}_{m-1}$ as initial value, repeat below process for $\forall j \in \{1, \cdots, p\}$ again and again until convergence.

$$\beta_{j,m} = \begin{cases} 0 & \text{if } | <\boldsymbol{x}_j, \boldsymbol{r}^{(j)}>_w | \leq \lambda_k \\ S_{\lambda_k}(<\boldsymbol{x}_j, \boldsymbol{r}^{(j)}>_w) & \text{if } | <\boldsymbol{x}_j, \boldsymbol{r}^{(j)}>_w | > \lambda_k \end{cases}$$

where $\boldsymbol{r}^{(j)} = \boldsymbol{z} - \boldsymbol{X}_{-j}\boldsymbol{\beta}_{-j,m}$

5 References

Efron, B., & Hastie, T. (2016). Computer Age Statistical Inference: Algorithms, Evidence, and Data Science (Institute of Mathematical Statistics Monographs). Cambridge: Cambridge University Press. doi:10.1017/CBO9781316576533

Friedman, J. H., Hastie, T., & Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordin -ate Descent. Journal of Statistical Software, 33(1), 1–22. https://doi.org/10.18637/jss.v033.i01

Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). The Elements of Statistical Learning: Data mining, Inference, and Prediction. 2nd ed. New York: Springer.

Hastie, T., Tibshirani, R., & Wainwright, M. (2015). Statistical Learning with Sparsity: The Lasso and Generalizations (1st ed.). Chapman and Hall/CRC. https://doi.org/10.1201/b18401

Tibshirani, Ryan (2019). Convex Optimization Lecture Notes. https://www.stat.cmu.edu/~ryantibs/convexopt/