Linear Regression

Exercise A. Let $y = X\beta + e$, where $y = (y_1, \ldots, y_N)$, X is an $N \times P$ patrix with ith row x_i and e a vector of model residuals. Further let W be the $N \times N$ diagonal vector of weights w_i.

We may then rewrite the equation $\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{N} \frac{w_i}{2}(y_i - x_i^T\beta)^2$ in vector form as

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^p} (y - X\beta)^T W (y - X\beta)$$

. We note that this is equivalent to

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^p} ||y - X\beta||_W$$

.

Let $r = y - X\beta$ We want to show that a vector $\beta \in \mathbb{R}^p$ minimizes the weighted norm $||r||_W = ||y - X\beta||_W$ iff $X^T W y = X^T W X \beta$

We note, however, that

$$\begin{aligned} X^T W y &= X^T W X \beta \Rightarrow \\ X^T W y - X^T W X &= 0 \\ X^T W (y - X\beta) &= 0 \\ X^T W r &= 0 \text{ (note that, because } W \text{ is diagonal, } W = W^T) \end{aligned}$$

Thus, it is equivalent to prove that $\beta \in \mathbb{R}^p$ minimizes the weighted norm $||r||_W = ||y - X\beta||_W$ iff r is orthogonal to $range(X)$ under the inner product induced by the weighted norm (the weighted inner product). This, however, follows immediately from the properties of orthogonality.

This orthogonal approach gives good geometric intuition into the problem.

Another method that can be used here is a differentiation argument, as follows. We wish to minimize

$$(y - X\beta)^T W (y - X\beta)$$

. We wish to find $\hat{\beta}$ s.t. $(y-X\beta)^T W(y-X\beta)$ is minimized. Thus, in order for this quantity to be minimized, it must satisfy:

$$\begin{aligned}
\frac{d}{d\beta}\left((y-X\hat{\beta})^T W(y-X\hat{\beta})\right) &= 0 \\
\frac{d}{d\beta}\left((y^T-(X\hat{\beta})^T)W(y-X\hat{\beta})\right) &= 0 \\
\frac{d}{d\beta}\left((y^TW-\hat{\beta}^TX^TW)(y-X\hat{\beta})\right) &= 0 \\
\frac{d}{d\beta}\left(y^TWy-y^TWX\hat{\beta}-\hat{\beta}^TX^TWy+\hat{\beta}^TX^TWX\hat{\beta}\right) &= 0 \\
\frac{d}{d\beta}\left(y^TWy-(\hat{\beta}^TX^TWy)^T-\hat{\beta}^TX^TWy+\hat{\beta}^TX^TWX\hat{\beta}\right) &= 0 \\
\frac{d}{d\beta}\left(y^TWy-2\hat{\beta}^TX^TWy+\hat{\beta}^TX^TWX\hat{\beta}\right) &= 0 \text{ (note that } \hat{\beta}^TX^TWy \text{ is a scalar)} \\
0-2X^TWy+2X^TWX\hat{\beta} &= 0 \\
-X^TWy+X^TWX\hat{\beta} &= 0
\end{aligned}$$

Thus $\hat{\beta}$ must satisfy $X^TWX\hat{\beta}=X^TWy$. $\square$

Exercise B. Numerically speaking, the inversion method is certainly not a very stable way to actually solve the linear system. Furthermore, when the problem is formulated in terms of computing an explicit inverse (rather than a matrix solve method), it is certainly not a very fast way to solve the linear system.

In numerical linear algebra, there are three fairly standard factorizations used to formulate the least squares problem without any explicit inversions: Cholesky factorization, QR factorization, and singular value decomposition. Each is valuable in different scenarios. Solving least squares via cholesky factorization utilizes the normal equations and is one of the fastest methods to solve for β. However, this method is unstable. On the other hand, solving the least squares problem via the SVD is stable, but is one of the slowest methods. It is the best algorithm for cases when the matrix X is close to begin rank-deficient (the smallest singular value is very small). Generally, solving WLS via a QR factorization is the most appropriate method. Ths method is faster than the SVD and far more stable than the Cholesky method. We will implement WLS via QR factorization as "our method".

Essentially, the idea of the QR factorization method is to reduce the problem to a triangular solve. Thus we take $QR = W^{\cdot 5}X$, where QR is the reduced QR factorization of $W^{\cdot}5X$ (where R is a square, triangular matrix and Q has orthonormal columns) and substitute this

to obtain:

$$\begin{aligned} X^T W X \beta &= X^T W y \\ X^T W^{.5} W^{.5} X \beta &= X^T W^{.5} W^{.5} y \\ R^T Q^T Q R \beta &= R^T Q^T W^{.5} y \\ R^T R \beta &= R^T Q^T W^{.5} y \\ R\beta &= Q^T W^{.5} y \end{aligned}$$

Thus, the basic algorithm gives us:

1. Take $W^{.5}X$ (note that this is a simple broadcast)
2. Compute the reduced QR factorization of $W{\cdot}5X$.
3. Compute $Q^T W^{.5} y$
4. Solve the upper triangular system $R\beta = Q^T W^{.5} y$.

Exercise C, D. Please see the attached code, "Exercise 1, Weighted Least Squares.ipynb"

Generalized Linear Models

Exercise A. Let $y_i \sim \text{Binomial}(m_i, w_i)$ where

- y_i is an integer number of successes
- m_i is the number of trials for the ith case.
- w_i is the success probability, a regression on a feature vector x_i given by $w_i = \frac{1}{1+\exp\{-x_i^T \beta\}}$.

Then we may write the negative log likelihood:

$$\begin{aligned} l(\beta) &= -\log\left[\prod_{i=1}^{N} p(y_i|\beta)\right] \\ &= -\sum_{i=1}^{N} \log(p(y_i|\beta)) \\ &= -\sum_{i=1}^{N} \log(\frac{m_i}{y_i(m_i - y_i)} w_i^{y_i}(1-w_i)^{m_i - y_i}) \\ &= -\sum_{i=1}^{N} \log(\frac{m_i}{y_i(m_i - y_i)}) + y_i \log(w_i) + (m_i - y_i)\log(1-w_i) \end{aligned}$$

We now want to derive the gradient for this expression. Note that

$$\nabla_\beta w_i = \frac{-1}{(1+\exp\{-x_i^T\beta\})^2} * -x_i^T \exp\{-x_i^T\beta\}$$

We also note that

$$1 - w_i = \frac{\exp\{-x_i^T\beta\}}{1+\exp\{-x_i^T\beta\}}$$

Thus

$$\nabla_\beta w_i = w_i(1-w_i)x_i^T$$

. We now find

$$\begin{aligned}
\nabla l(\beta) &= -\sum_{i=1}^N \nabla_\beta y_i \log(w_i) + \nabla_\beta(m_i - y_i)\log(1-w_i) \\
&= -\sum_{i=1}^N \frac{y_i}{w_i}\nabla_\beta w_i + \frac{(m_i-y_i)}{1-w_i}\nabla_\beta(1-w_i) \\
&= -\sum_{i=1}^N \frac{y_i}{w_i}\nabla_\beta w_i + \frac{(m_i-y_i)}{1-w_i}\nabla_\beta(1-w_i) \\
&= -\sum_{i=1}^N \frac{y_i}{w_i}w_i(1-w_i)x_i^T - \frac{(m_i-y_i)}{1-w_i}w_i(1-w_i)x_i^T \\
&= -\sum_{i=1}^N y_i(1-w_i)x_i^T - (m_i-y_i)w_i x_i^T \\
&= -\sum_{i=1}^N (y_i - y_i w_i - m_i w_i + y_i w_i)x_i^T \\
&= -\sum_{i=1}^N (y_i - m_i w_i)x_i^T \\
&= -\sum_{i=1}^N (y_i - \hat{y}_i)x_i^T
\end{aligned}$$

Exercise B. See attached Code, "Exercise 1, Conjugate Gradient.ipynb".

Exercise C. Let $\beta_0 \in \mathbb{R}^p$, an intermediate guess for our vector or regression coefficients. We want to show that the second order Taylor approximation of $l(\beta)$ around the point β_0 takes the form

$$q(\beta;\beta_0) = \frac{1}{2}(z - X\beta)^T W(z - X\beta) + c$$

where z is a vector of "working responses" and W is the diagonal matrix of "working weights" and c is a constant that doesnt involve β.

Recall that we have that $\nabla l(\beta) = \sum_{i=1}^N (m_i w_i - y_i)x_i^T$ We note that this could be written $s^T X$, where s is a vector whose ith element is $m_i w_i - y_i$.

Further recall that $\nabla_\beta w_i = w_i(1-w_i)x_i^T$ We want to further consider $\nabla^2 l(\beta)$.

$$\begin{aligned}\nabla^2 l(\beta) &= \nabla_\beta \left(\sum_{i=1}^{N} (m_i w_i - y_i) x_i^T \right) \\ &= \nabla_\beta \left(\sum_{i=1}^{N} m_i w_i x_i^T - y_i x_i^T \right) \\ &= \sum_{i=1}^{N} \nabla_\beta (m_i w_i x_i^T) - \nabla_\beta (y_i x_i) \\ &= \left(\sum_{i=1}^{N} \nabla_\beta (m_i w_i x_i^T) \right)\end{aligned}$$

We consdier $\nabla_\beta(m_i w_i x_i)$ for arbitrary i, where $(m_i w_i x_i^T)$ is a row vector of the form

$$(m_i w_i x_{i1}, m_i w_i x_{i2}, \ldots m_i w_i x_{iP})$$

.

It follows that $\nabla_\beta(m_i w_i x_i^T)$ has form

$$\nabla_\beta(m_i w_i x_i^T) = \begin{bmatrix} m_i \nabla_\beta w_i x_{i1} \\ m_i \nabla_\beta w_i x_{i2} \\ \vdots \\ m_i \nabla_\beta w_i x_{iP} \end{bmatrix}$$

This in turn gives us

$$\nabla_\beta(m_i w_i x_i^T) = \begin{bmatrix} m_i(w_i(1-w_i)) x_{i1} x_i^T \\ m_i(w_i(1-w_i)) x_{i2} x_i^T \\ \vdots \\ m_i(w_i(1-w_i) x_{iP} x_i^T \end{bmatrix}$$

where $\nabla_\beta(m_i w_i x_i^T)$ is a matrix A with j^{th} row $A_j = m_i(w_i(1-w_i))x_{ij}x_i^T$. Noting this gives us that $\nabla_\beta(m_i w_i x_i) = m_i w_i(1-w_i)x_i \otimes x_i$, where $\otimes$ denotes the outer product for a vector v: vv^T.

Thus we write

$$\nabla^2 l(\beta) = \sum_{i=1}^{N} m_i w_i (1-w_i) x_i x_i^T$$

.

Then we can write $X^T W X$ where W is the diagonal matrix with diagonal "weight" elements $m_i w_i(1-w_i)$.

Using the second-order Taylor series expansion centered at the initial point β_0, we have

$$q(\beta) = l(\beta_0) + \nabla f(\beta_0)(\beta - \beta_0) + \frac{1}{2}(\beta - \beta_0)^T \nabla^2 f(\beta_0)(\beta - \beta_0)$$

$$q(\beta) = l(\beta_0) + s^T X(\beta - \beta_0) + \frac{1}{2}(\beta - \beta_0)^T X^T W X(\beta - \beta_0)$$

Exercise D. See attached code, "E1_Newtons.py"

Exercise E. When we compare the convergence rates of Newton's method and gradient descent, it is clear that Newton's method converges it far fewer iterations than gradient descent. However, Newton's method can have a significant overhead cost associated with computing and manipulating the hessian matrix. Sometimes, in light of this cost, it is better to use gradient descent or stochastic gradient descent, which involves calculating the gradient alone, rather than the hession (significantly fewer computations per iteration). On the other hand, various quasi newton methods can also be a good solution to this problem. These methods such as, for example, the BFGS method, approximate the hessian to attain a higher rate of convergence inherent in newton's method, but without the cost of explicit hessian calculation.