群組分類
Logistic Regression

last modified April 13, 2006

迴歸分析會因爲資料的型態的不同而採取不同的模式。前面提過的簡單線性迴歸與多項式迴歸都是觀察自變數與因變數資料間的關係, 採取的適當模式。這個單元要探討當因變數爲類別資料時,Logistic Regression 是一個不錯的迴歸模式。這個模式在參數的估計上可以配合 Maximum likelihood 的方式, 進行非線性的參數估計。這種非線性的估計在研究領域很常見, 本單元提供最簡單的解決方案, 作爲非線性領域的開端。就此目的而言,Logistic Regression 只是作爲開場白。

1 背景介紹

1.1 Logistic Regression

迴歸模型用來描述自變數與因變數間的關係。當因變數爲群組性的類別資料且其發生的機率如圖1的趨勢時, 前面單元提過的 LDA(Linear Duscriminant Analysis) 的概念結合簡單線性迴歸模式, 適合處理這類的問題。LDA對群組的區別以「後驗機率的 logit transformation 等於0」作爲分野, 即

$$\log \frac{Pr(G=k|X=\mathbf{x})}{Pr(G=l|X=\mathbf{x})}$$

LDA巧妙的利用貝氏定理與群組資料的常態分配假設 (且群組間的共變異矩陣相同), 上式簡化爲一線性函數

$$\log \frac{Pr(G=k|X=\mathbf{x})}{Pr(G=l|X=\mathbf{x})} = \lambda_0 + \lambda^T \mathbf{x}$$

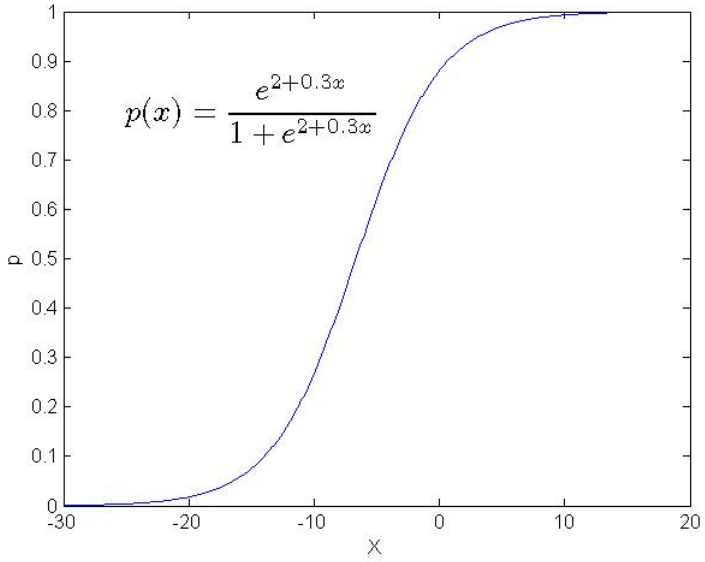

圖 1: sigmoid function

Logistic regression 維持群組分野如上式的簡單線性關係, 並免除對群組資料的常態分配假設, 採用如圖1的函數做爲群組的後驗機率。以兩個群組爲例, 假設

$$Pr(G = 1|X = x) = \frac{e^{f(x)}}{1 + e^{f(x)}} \tag{1}$$

其中 $f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_k x^k = \beta_0 + \beta^T \mathbf{x}$

則後驗機率的logit transformation 寫成

$$z = \log \frac{Pr(G = 1|X = x)}{Pr(G = 2|X = x)} = \beta_0 + \beta^T \mathbf{x} \tag{2}$$

也是個簡單的迴歸模型。本單元的重點將是探討如何利用最大概似估計法估計未知參數 $\beta_0, \beta_1, \cdots$。

關於式 (1) 對後驗機率的假設並非全無道理, 它其實符合自然界某些運作法則。隨著自變數變大, 因變數的反應機率從一個極端 (0) 走向另一個極端 (1)。中間轉換的幅

度可能是和緩的, 也有些是比較陡峭的。例如生物細胞受電壓的激發, 其電壓大小與激發與否的關係也是如此, 不過中間轉換曲線比較直聳, 有點像 step function, 一般也叫做 sigmoid function。式 (2) 亦稱爲 log odds ratio(對數優勢比)。將優勢比取對數後對 x 作多項式迴歸稱爲 Logistic Regression。當 $k = 1$ 時,x 與 z 的關係變成簡單的線性模式。這種透過變數的轉換將較複雜的模式 (1), 變爲簡單的模式是迴歸分析常見的手段, 所謂「山不轉, 路轉。」不管對因變數或自變數都可以。

式 (1) 的後驗機率也可以解讀爲「成功的比例,」是一個遞增函數, 兩端是近乎穩定平緩的水平線, 中間部分變化較大。譬如 x 與 $Pr(G = 1|X = x)$ 代表年齡與罹患心臟病比例, 雖兩者的關係是一平滑連續的漸增曲線, 即年齡越大罹患心臟病的比例越高, 不過當進行實際資料的蒐集時, 原始資料僅呈現年齡 (X) 與罹患心臟病與否 (Y) 的二元性數據 (0或1), 如圖2所示, Y可視爲伯努力分配的變數, 其成功比例爲 $Pr(G = 1|X = x)$ 。

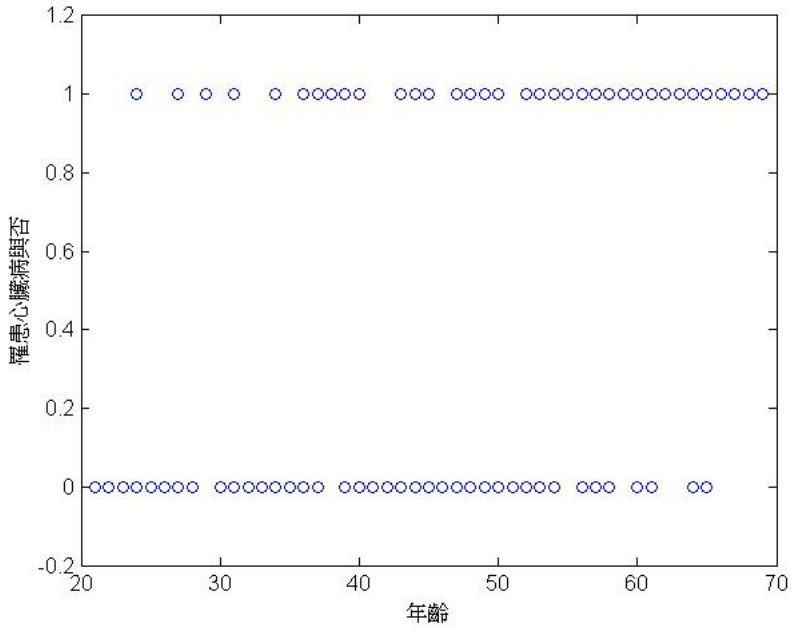

圖 2: 年齡與罹患心臟病的調查結果

當拿 X 與二元性資料 Y 作 Logistic Regression 時, 一般採 maximum likelihood

的觀念來估算參數值。假設在已知自變數 X 及參數β下, 因變數 Y 的條件機率密度函數爲

$$f(y|x,\beta) = Pr(\text{G: 罹患心臟病與否}|x,\beta) \tag{3}$$

爲方便分析起見, 假設式 (2) 的迴歸模式爲線性, 即$\beta = [\beta_0\ \beta_1]^T$。藉由式 (1) 罹患心臟病的機率及因變數觀察值的二元性, 因變數 Y 的條件機率密度函數 (3) 可以進一步寫成

$$\begin{aligned} f(Y=1|x,\beta) = Pr(\text{罹患心臟病}|x,\beta) &= \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}} = p(x,\beta) \\ f(Y=0|x,\beta) = 1 - Pr(\text{罹患心臟病}|x,\beta) &= \frac{1}{1+e^{\beta_0+\beta_1 x}} = 1-p(x,\beta) \end{aligned} \tag{4}$$

1.2 Maximum Likelihood

在 N 個獨立觀察值下, 因變數 Y 的聯合機率密度函數 Joint Probability Density Function 寫成

$$f(\mathbf{y}|\mathbf{x},\beta) = \prod_{k=1}^{N} f_k(y_k|x_k,\beta) \tag{5}$$

上式稱爲概似函數 (Likelihood Function)。套用式 (4), 這個概似函數可以改寫爲

$$L(\beta) = f(\mathbf{y}|\mathbf{x},\beta) = \prod_{k=1}^{N} f_k(y_k|x_k,\beta) = \prod_{k=1}^{N} p(x_k,\beta)^{y_k}(1-p(x_k,\beta))^{1-y_k} \tag{6}$$

所謂最大概似值的觀念便是選擇一組參數值 $\beta = \beta^o$ 使得概似函數 $L(\beta)$ 最大, 這表示當 $\beta = \beta^o$ 的情況下, 出現 N 個觀察值 $\mathbf{y}$ 的機會最大。這相當於要解決下列最佳化問題,

$$\max_{\beta} L(\beta)$$

由於概似函數的「長相」, 對概似函數取對數後再求最佳值在計算上比較簡單, 在不影響最佳值的情況下, 問題變爲

$$\max_{\beta} \log L(\beta) \tag{7}$$

其中對數概似函數經化簡後 (作業1), 變成

$$\log L(\beta) = \sum_{k=1}^{N} \left(y_k \beta^T \mathbf{x}_k - \log(1 + e^{\beta^T \mathbf{x}_k}) \right) \tag{8}$$

其中 $\mathbf{x}_k = [1 \; x_k]^T$。從對數概似函數的一次導數 (即梯度向量 gradient vector)

$$\nabla \log L(\beta) = \sum_{k=1}^{N} \left(y_k - \frac{e^{\beta^T \mathbf{x}_k}}{1 + e^{\beta^T \mathbf{x}_k}} \right) \mathbf{x}_k \tag{9}$$

看出當 $\nabla \log L(\beta) = \mathbf{0}$ 時, 對 β 而言是一組非線性的方程式, 沒有 closed-form 的解答。必須採用迭代遞迴 (Iteration) 的方式逐步迭代出最佳解。

由於多數的演算法都以求最小值爲主, 最大概似值的問題可以改寫成

$$\max_{\beta} \log L(\beta) = \min_{\beta} - \log L(\beta) \tag{10}$$

逐步迭代法則:

1. 選擇適當的起始值 β_0
2. $\beta_{k+1} = \beta_k + \alpha \mathbf{d}_k$
3. 直到區域最小值

step size 的選擇, 演算法的細部過程可以參考前面的單元。

應用如上述的迭代法則, 當 $\mathbf{d}_k = -\nabla \log L(\beta)$, 稱爲 Steepest Descent Direction, 是一個歷史悠久且被許多應用證實有效的迭代方向。這個方向的選擇最大的缺點在於面對比較平滑的函數時, 收斂的速度往往不能令人滿意, 如圖3的函數在「谷底」的部分比較平滑, 對最小值的尋找比較慢。雖號稱 steepest, 常常是迂迴蛇行, 令人氣結。優點是計算較爲簡單, 因此這個選擇倒是不差, 本單元將重點放在如何以這個方向寫作迭代程式。

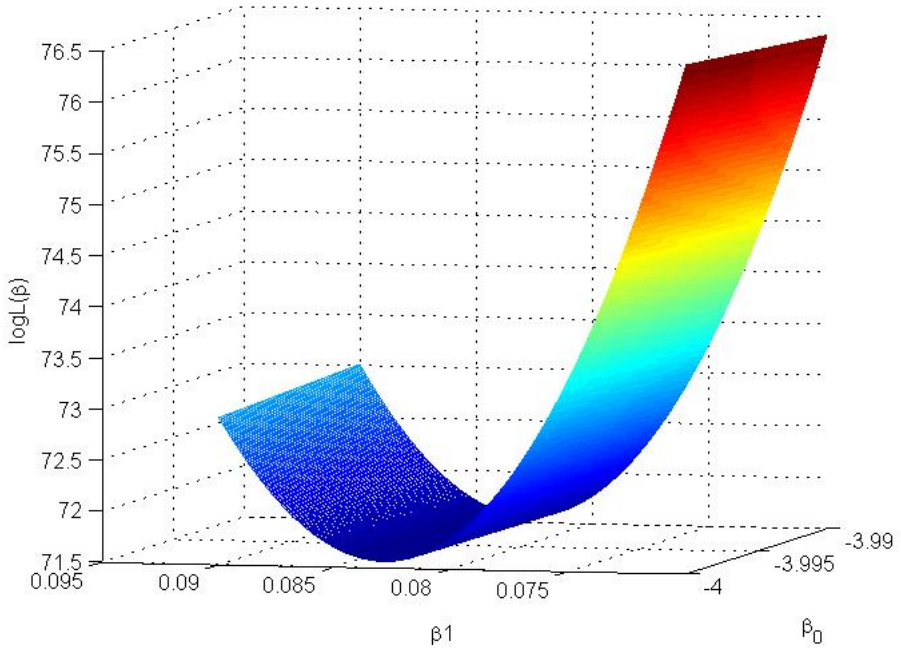

圖 3: 在「谷底」的部分比較平滑的對數概似函數

1.3 MATLAB的技巧

式 (9) 的表示法並不利程式的撰寫。傳統的方法會利用迴圈來解決大量的加法, 不過 Matlab 卻擅長利用矩陣作最精簡迅速的計算。將式 (9) 改寫成矩陣形式是 Matlab 寫作中最重要的養成技術。在此只將結果列出, 過程請自行推導 (作業2);

$$\nabla \log L(\beta) = X(\mathbf{z} - \mathbf{y}) \tag{11}$$

其中

$$X = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N \end{bmatrix} \quad \mathbf{z} = \begin{bmatrix} \frac{e^{\beta^T \mathbf{x}_1}}{1+e^{\beta^T \mathbf{x}_1}} \\ \frac{e^{\beta^T \mathbf{x}_2}}{1+e^{\beta^T \mathbf{x}_2}} \\ \vdots \\ \frac{e^{\beta^T \mathbf{x}_N}}{1+e^{\beta^T \mathbf{x}_N}} \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

利用 Matlab 計算 (11) 時, 資料矩陣 X 與 $\mathbf{y}$ 比較容易建構起來, 但是 $\mathbf{z}$ 的形成便有點棘手, 不過卻是Matlab 程式設計上很典型的轉換, 推演過程如下:

1. 先建立

$$\begin{bmatrix} \beta^T\mathbf{x}_1 \\ \beta^T\mathbf{x}_2 \\ \vdots \\ \beta^T\mathbf{x}_N \end{bmatrix} \Longrightarrow X^T\beta$$

2. 套入 exp 函式

$$\begin{bmatrix} e^{\beta^T\mathbf{x}_1} \\ e^{\beta^T\mathbf{x}_2} \\ \vdots \\ e^{\beta^T\mathbf{x}_N} \end{bmatrix} \Longrightarrow exp(X^T\beta)$$

3. 完成

$$\mathbf{z} = \begin{bmatrix} \frac{e^{\beta^T\mathbf{x}_1}}{1+e^{\beta^T\mathbf{x}_1}} \\ \frac{e^{\beta^T\mathbf{x}_2}}{1+e^{\beta^T\mathbf{x}_2}} \\ \vdots \\ \frac{e^{\beta^T\mathbf{x}_N}}{1+e^{\beta^T\mathbf{x}_N}} \end{bmatrix} \Longrightarrow exp(X^T\beta)./(1+exp(X^T\beta))$$

程式中只要將資料矩陣 X與 $\mathbf{y}$ 準備好, $\mathbf{z}$ 的計算只要一條指令便可解決, 非常簡潔, 速度也快。

另外, 迭代估計的過程中, 對數概似函數的函數值 $\log L(\beta)$ 也必須計算, 同樣的採矩陣的方式計算, Matlab的指令可以寫成:(若不熟悉, 請按前項的方式推演)

$$\log L(\beta) \doteq sum(log(1+exp(\beta^T X))) - \beta^T X\mathbf{y} \tag{12}$$

同樣是一條指令解決, 乾淨清爽。

1.4 鑑別函數與群組判別

Logistic Regression 的鑑別函數爲

$$f(\mathbf{x}) = \hat{\beta}^T\mathbf{x}$$

其中$\hat{\beta} = [\hat{\beta}_0\ \hat{\beta}_1\ \cdots\ \hat{\beta}_k\ ]$,$\mathbf{x} = [1\ x_1\ x_2\ \cdots\ x_k]$

群組間的分界線 (Hyperplane) 爲集合

$$\{\mathbf{x}|\hat{\beta}^T\mathbf{x} = 0\}$$

$$G = \begin{cases} CLASS1 \text{ if } \hat{\beta}^T\mathbf{x} \leq 0 \\ CLASS2 \text{ if } \hat{\beta}^T\mathbf{x} > 0 \end{cases}$$

2 練習

範例1: 群組發生的後驗機率密度函數

$$Pr(Y = k|X = x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}$$

是個漸增函數 (monotinic function), 其中的兩個因數 β_0, β_1 對函數的影響值得觀察。試著繪製下列兩張圖:

- 固定 $\beta_1 = 0.3$, 令 $\beta_0 = -2, 2, 4$ 分別畫出三條函數線在同一張圖上。
- 固定 $\beta_0 = 2$, 令 $\beta_1 = 0.3, 0.6, 1.2$, 分別畫出三條函數線在另一張圖上。

從圖4可以清楚的看出係數 β_0, β_1 對函數的影響。當 β_1 愈大時, 函數愈接近所謂 step function, 代表機率從 0 到 1 的進程非常快速, 當 x 大於某個值後, 機率馬上跳到 1, 這也說明在 x 的變數軸上, 群組的分別性很高。
以下的練習以圖2的「年齡與罹患心臟病的調查結果」爲資料來源。

範例2: 不論採用Steepest Descent Method 或 Newton-Raphson Method, 都需要計算對數概似函數的梯度向量 (9), MATLAB利用矩陣的特色, 可以精簡的式 (11) 來計算。首先建構資料矩陣 X 與 $\mathbf{y}$, 並按上述的步驟逐步建構 $\mathbf{z}$, 最後再合成爲一個

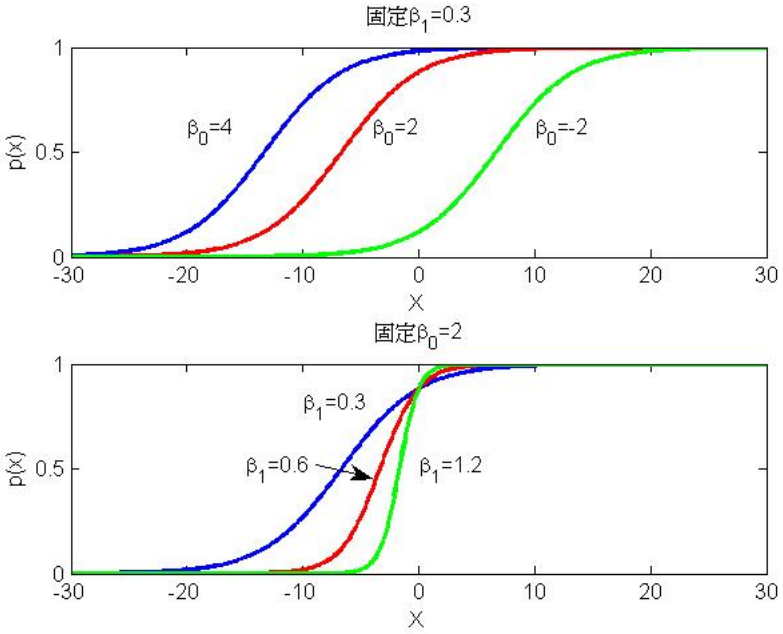

圖 4: 群組發生的後驗機率密度函數與其係數的關係

指令。

資料矩陣 X 與 **y**完全來自原始資料, 建構方式在前面的單元都已經練習過。值得注意的是建構向量變數 **z** 時, 未知變數 β 必須先給一組起始值才能計算。在程式建立的初期, 爲確定程式的正確性, 起始值的選擇可以盡量靠近函數的最小值, 當然這只在函數圖形可以呈現出來的情況下, 譬如圖3。

範例3: 寫作最大概似函數的程式, 需要計算過程中的對數概似函數, 藉以檢查程式或概似函數本身是否正確。請逐步以式(12) 的方式建構對數概似函數 (8) $\log L(\beta)$, 最後以一個指令完成。

寫作程式時, 指令的精簡會讓整個環境看起來清爽, 程式整體的步驟與邏輯容易解讀。但寫作之初卻不容易做到, 實際上也不需要一次「到位,」逐步建構並檢查反而比較容易成功。俟一切無誤後再行合併, 效率往往比較高, 值得初學者參考。

範例4: 試著畫出對數概似函數 (8)(注意:β_0 與 β_1 範圍的訂定非常關鍵, 否則即使畫正確了, 也看不出所以然)。

圖3的概似函數圖是經過適當旋轉後的樣子。當 MATLAB 自行決定的角度不利觀察時, 可以利用繪圖區的旋轉功能, 適度的旋轉, 找到一個最佳的觀察視野。從圖上可以看出概似函數在 β_0 的方向一片平坦, 說明 β_0 比較不具鑑別能力, 反觀在 β_1 的方向則是清楚的凹線。以下程式碼利用式 (12) 畫出圖3。

```
data = load('heart_attack.txt'); % 讀入資料 heart_attack
x = data(:,1); %年齡資料
y = data(:,2); %罹病與否
[B0,B1] = meshgrid(-4:0.0001:-3.99,0.07:0.0001:0.09);
Z1 = sum(y)*B0 + y'*x*B1;
Z2 = zeros(size(Z1));
for i = 1:N
        Z2 = Z2 + log(1+exp(B0+B1*x(i)));
end
Z = -Z1 + Z2;
mesh(B0,B1,Z)
```

也可以直接以式 (6) 直接寫程式畫圖, 配合匿名函數的使用如下:

```
%資料讀入同上
Lf = @(b)sum(log(1+exp(b(1)+b(2)*x))-y.*(b(1)+b(2)*x));
b0 = -4:0.0001:-3.99;
b1 = 0.07:0.0001:0.09;
Z = zeros(length(b0),length(b1));
```

```
for i = 1 : length(b0)
    for j = 1 : length(b1)
        Z(i, j) = Lf([b0(i)  b1(j)]);
    end
end
mesh(b1, b0, Z);
```

當然, 第二個程式耗費的時間稍多一些, 因爲迴圈的使用與匿名函數都會比較費時。

範例5:MATLAB 提供指令 fminsearch 計算多變量函數的極小值, 這個方式並沒有使用到梯度向量, 卻常能有效且迅速的找到極小值。其使用方式同前面單元介紹過的 quad、fzero、fminbnd, 在此利用上述的資料與 fminsearch 指令, 計算式 (10) 的最大概似估計。

```
data = load('heart_attack.txt'); % 讀入資料 heart_attack
x = data(:, 1); %年齡資料
y = data(:, 2); %罹病與否
Lf = @(b)sum(log(1 + exp(b(1) + b(2) * x)) - y. * (b(1) + b(2) * x));
fminsearch(Lf, [4  1]) %初始值爲 [4  1]
```

這裡採用匿名函數的方式, 方便程式運作。其中變數爲 1×2 的向量 b, $b(1)$ 代表 β_0 ,$b(2)$ 代表 β_1。

3 觀察

1. 將上述模式擴展到多元的自變數, 式 (2) 可以改寫爲

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x_2 + \cdots + \beta_r x_r = \beta^T \mathbf{x}$$

   對於參數 β 的估計, 依然可以採用本單元所敍述的方法。

2. 從一個數學式 (9) 到以矩陣方式表示 (11) 再以 Matlab 的指令寫出程式, 這樣的模式必須非常熟練, 方能對許多看似複雜, 但實際也可以很輕鬆以矩陣代表數學式完成程式建置。轉換成矩陣後顯得乾淨俐落許多, 有利於往後進一步的推導。

3. 爲觀察 steepest descent 這個「下坡」方向的收斂速度, 你可以在執行過程中逐步印出結果, 或是每隔一定的迴圈數才列印。

4. 迭代方向最常見的是「Newton-Raphson method,」不過它牽涉到二次導數, 雖然是比較有效的方向, 卻也比較複雜, 往往因此捨棄不用, 改採較簡單但收斂速度較差的其他方法。這些方法通常介於本單元討論的 steepest descent 與 Newton-Raphson 之間。

5. 其實本單元採 Newton-Raphson 的方式還算簡潔, 如下

$$\mathbf{d}_k = -\left(\nabla^2 \log L(\beta)\right)^{-1} \nabla \log L(\beta)$$

其中 $\nabla^2 \log L(\beta)$ 稱爲 Hessian matrix, 在此並假設其反矩陣存在, 其定義及矩陣的表示式如下:

$$\nabla^2 \log L(\beta) = \begin{bmatrix} \frac{\partial^2 \log L(\beta)}{\partial\beta_0\partial\beta_0} & \frac{\partial^2 \log L(\beta)}{\partial\beta_0\partial\beta_1} \\ \frac{\partial^2 \log L(\beta)}{\partial\beta_1\partial\beta_0} & \frac{\partial^2 \log L(\beta)}{\partial\beta_1\partial\beta_1} \end{bmatrix} = \nabla \left(\nabla \log L(\beta)\right)^T = XWX^T$$

W 爲 NxN 對角矩陣, 其中$W(k,k) = p(\mathbf{x}_k, \beta)(1 - p(\mathbf{x}_k, \beta))$。當然將 Hessian matrix 寫入程式時, W 還要另做安排 (如式 (11))。不妨試著推敲看看, 並實際比較 steepest descent 與 Newton-Raphson 速度上的表現。

4 作業

1. 推導式 (8)、(9)。

2. 推導式 MATLAB 的計算式 (11)、(12)。

3. 寫下 Steepest Descent Algorithm 的完整步驟 (含 step size control)。

4. 試著在適當的範圍畫出對數概似函數 $\log L(\beta)$。

5. 完成 β 的估計。注意: 本單元的對數概似函數 $\log L(\beta)$ 在底部接近最小值的附近非常的平滑, $\log L(\beta)$ 的小小改變可能意味在自變數方面不小的異動, 因此當設定程式停止迴圈的條件時, 必須仔細觀察, 必要時也可以利用自變數的微量改變停止迴圈。

6. 利用估計出來的 β 值, 畫出式 (1) 的機率圖。這是 Logistic Regression 主要的模型, 說明年齡與罹患心臟病的關係。

7. 利用 Newton-Raphson 的方法, 完成β 的估計。

8. 將本單元的 Logistic Regression 方法用在之前單元所使用的資料, 如 la_1.txt,la_2.txt, 試著畫出那條分界線, 並與之前做過的簡單迴歸與 LDA 比較。

9. 比較 Logistic Regression 與 LDA 之異同。

10. 當自變數超過1個時,Logistic Regression 的估計結果必須作適當的評估, 依所估計參數的大小檢定其顯著性 (Significance), 請試著以實際資料[1](請從網站下載) 估計所有的參數, 並評估哪一個變數該被去除。

參考文獻

[1] T. Hastie, R. Tibshirani, J. Friedman, "The Elements of Statistical Learning:Data Mining, Inference, and Prediction," Springer.

[2] 陳順宇, 迴歸分析-三版, 華泰書局。

[3] J.E. Dennis, R.B. Schnabel, Numerical Methods for Unconstrained Optimization, Prentice Hall