群組分析：羅吉斯迴歸（Logistic Regression）

January 24, 2019

迴歸分析會因爲資料的型態的不同而採取不同的模式。例如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所採取的適當模式。本章要探討當因變數爲類別資料時，邏吉斯迴歸（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最大概似法（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，本章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，邏吉斯迴歸只是簡單的開場白。另提出 MATLAB 在機器學習（Machine Learning）套件的解決方式作爲對比，以利學習邏吉斯迴歸的概念與更有效率的方案。

本章將學到關於程式設計

- MATLAB 提供關於邏吉斯迴歸的解決方式。
- 兩個變數函數的繪圖方式。

〈本章關於 MATLAB 的指令與語法〉
指令：fminsearch, fitnlm, glmfit, predict

1 背景介紹

1.1 邏吉斯迴歸

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，前章提過的 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

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

其中 X 與 G 分別代表自變數及因變數所代表的的群組別。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），上式可以簡化爲一線性函數

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

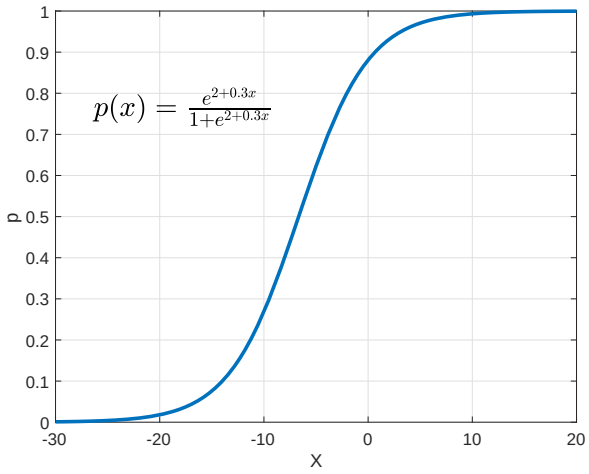

圖 1: sigmoid function

邏吉斯迴歸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，直接採用如圖 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

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

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 $\beta_0, \beta_1, \cdots, \beta_k$。

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。隨著自變數逐漸變大，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 作多項式迴歸稱爲邏吉斯迴歸。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，假設 **x** 與 $Pr(G = 1|X = \mathbf{x})$ 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 $Pr(G = 1|X = \mathbf{x})$ 。

當拿連續型變數 X 與二元性資料 Y 作邏吉斯迴歸時，一般採最大概似法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

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

爲方便分析起見，假設式 (2) 爲簡單線性迴歸模式，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

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

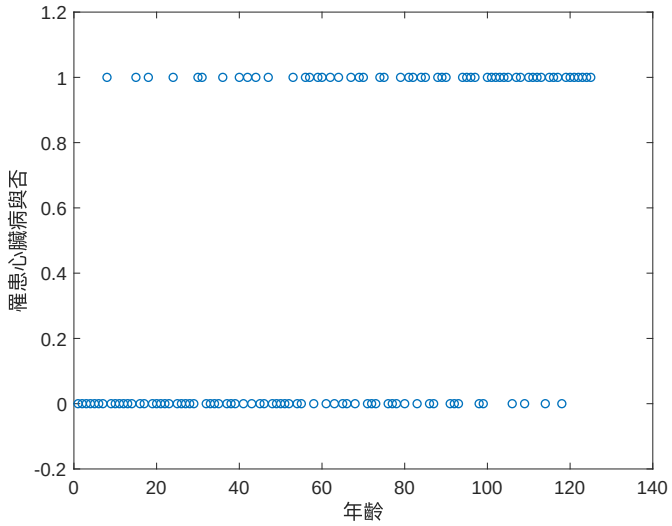

圖 2: 年齡與罹患心臟病的調查結果。0 代表「否」，1 代表「是」。

1.2 最大概似估計

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

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

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

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

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 $\mathbf{x_k} = [\mathbf{1}\ \mathbf{x_k}]^{\mathbf{T}}$。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

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

1.3 MATLAB 的作法

範例 1 利用附件資料 heart_attack.txt 試著畫出對數概似函數 (8)。請注意：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。

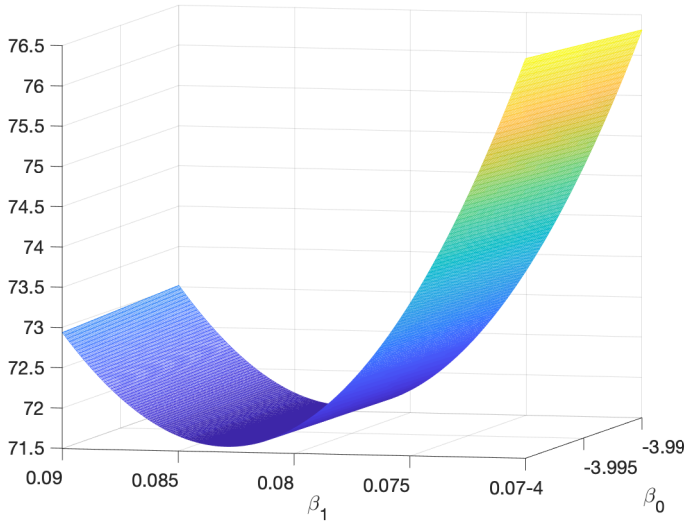

圖 3: 在「谷底」的部分比較平滑的對數概似函數

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 (8) 畫出圖 3。

```
D=load('heart_attack.txt'); % 讀入資料 heart_attack
x=D(:, 1) % 年齡資料
y=D(:, 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)•
```

範例 2 使用 MATLAB 指令 fminsearch 計算式 (10) 的最大概似估計問題。

```
D=load('heart_attack.txt'); % 讀入資料 heart_attack
x=D(:, 1) % 年齡資料
y=D(:, 2); % 罹病與否
Lf=@(b) sum(log(1+exp(b(1)+b(2)*x))-y.*(b(1)+b(2)*x)); % 對數概似函數之負數
ini=[4 1];
[x_opt, fmin]=fminsearch(Lf, ini)
```

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。執行結果當得到 x_opt=[-3.9933 0.0828]，函數之最小值約爲 fmin=71.5678，也就是最大概似函數值爲 -71.5678。

範例 3 MATLAB 也提供指令 fitglm 與 glmfit 兩個指令，直接進行各種迴歸模式的參數估計與相關統計量計算。本範例利用這兩個指令與上述的資料，進行羅吉斯迴歸分析。

MATLAB 指令 glmfit 提供廣義線性模型的迴歸分析（Generalized Linear Model Regression）。所謂「廣義」，指的是對於因變數可能的型態提供適當的轉換模式，[1]最後仍與自變數配適成線性的迴歸關係。

```
D=load('heart_attack.txt'); % 讀入資料 heart_attack
x=D(:, 1) % 年齡資料
y=D(:, 2); % 罹病與否
b = glmfit(x, y, 'binomial', 'link', 'logit'); % 選擇 logit 轉換模式
yfit = glmval(b, x, 'logit');
plot(x, yfit, 'o')
```

這個程式碼執行結果與上一個範例得到一樣的迴歸參數。至於這個估計好或是不好，可以利用指令 glmval 計算擬合值 yfit，並與圖 1 配適看看。結果如圖 4。

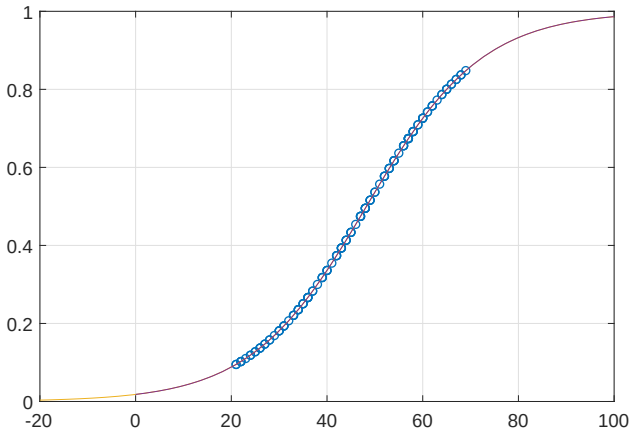

圖 4: 羅吉斯迴歸的模型配適情形

MATLAB 自從推出 Machine Learning 套件之後，將相關的指令重新編排，維持使用的一致性。於是 glmfit 另設爲 fitglm，使用方式如下程式碼所示:

```
mdl = fitglm(x, y,'Distribution', 'binomial'); % 選擇因變數爲二項分配
yfit =predict(mdl, x);
```

輸出結果 mdl 也是標準化的形式，如圖 5 所示。而指令 predict 是機器學習套件的共用指令，任何學習模式的結果都可以用 predict 作爲訓練資料的擬合值計算與新資料的預測。

[1]因變數的型態，或說分配，可能是二項（Binomial）、常態（Normal）、卜瓦松（Poisson）等，細節請參考使用手册所列的 link 選項。本章主要針對二項形式的因變數，譬如成功或失敗。

```
mdl =


Generalized linear regression model:
    logit(y) ~ 1 + x1
    Distribution = Binomial

Estimated Coefficients:
                   Estimate       SE        tStat       pValue
                   ________    ________    _______    __________

    (Intercept)     -3.9933     0.83558    -4.7791    1.7609e-06
    x1             0.082793    0.017323     4.7793     1.759e-06


125 observations, 123 error degrees of freedom
Dispersion: 1
Chi^2-statistic vs. constant model: 29.2, p-value = 6.59e-08
```

圖 5: 指令 fitglm 執行結果

範例 4 上述範例的資料來自眞實案例，所以估計結果的「品質」好壞無法判斷。一般來說，會以模擬資料來評估模型與演算法估計的優劣。本範例示範因變數爲二項分配資料的模擬，並代入 MATLAB 指令 fitglm 與 predict 做估計與產生擬合值。模擬參數設定爲:

$$\beta_0 = -2, \beta_1 = 1, N = 100, x \sim Unif(0, 5)$$

資料模擬是很重要的訓練，幫助習者更理解統計原理與應用的模型。本範例的自變數訂爲來自均勻分配，這是沒有根據任何實務面的考量，純粹以亂數當作自變數資料。因變數的資料產生便是依據本章對於邏輯斯迴歸模型的描述產生的。讀者最好先自己想一想，再參考如下的程式碼。

```
RandStream.setGlobalStream(RandStream('mt19937ar','seed',sum(100*clock)));
b0=-2; b1=1; N=100;
x=unifrnd(0,5,N,1); % 先產生自變數資料
A=exp(b0+b1*x);
p=A./(1+A);
y=binornd(1, p); % 生成二項分配的因變數資料
mdl = fitglm(x, y,'Distribution', 'binomial'); % 進行估計與計算相關統計量
yfit =predict(mdl, x); % 計算擬合值
f=@(x) exp(b0+b1*x)./(1+exp(b0+b1*x)); % 繪製機率圖
fplot(f), [-5 10])
hold on, plot(x, yfit, 'o'), hold off % 貼上擬合值
```

圖 6 展示上述程式碼的某次執行結果。之所以說是「某次」執行的結果，來自自變數從亂數生成來，而且亂數的「種子」（seed）並不固定。程式碼第一行定義了亂數取用的「種子」來源爲 sum(100*clock)，也就是根據電腦當時的時間做些處理後的數字。如果希望產生固定的模擬資料，必須將種子設爲固定數字。不同的模擬資料當然估計出不同的參數值，因此不能從一次的模擬資料判斷估計的品質。本章在習題中提出一種在學術研究上常見的模擬情境，讀者不妨試著寫程式做做看。

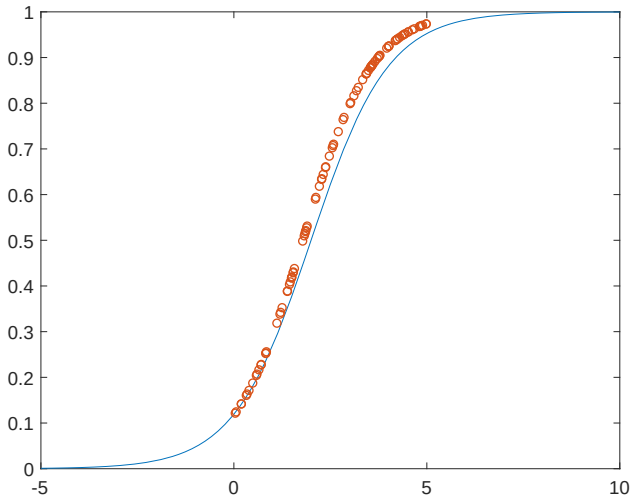

圖 6: 模擬資料的羅吉斯迴歸配適情形

2 觀察與延伸

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。試著模仿範例 4 的做法，模擬二元自變數的資料，即 $r = 2$，並估計參數值 β_0, β_1 與 β_2。

3 習題

1. 推導式 (8)、(9)。
2. 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。
3. 比較 Logistic Regression 與 LDA 之異同。

4. 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。

5. 範例 4 展示模擬資料的邏吉斯迴歸模型參數估計。請針對樣本數 N=50, 100, 500, 1000，各模擬 1 萬筆資料並計算相對應的參數估計值共 1 萬個，計算每個參數估計值的平均數、標準差（SEE: Standard Error of Estimators ）及 95% 的覆蓋機率（CP: Coverage Probability）。

References

[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