STA 601 - Homework 14

Kedar Prabhudesai

November 9, 2013

Probit Model:

Model Specification:

X is $(p \times n)$ matrix of predictor variables and β is a $(p \times 1)$ vector of regression co-efficients. Y is $(n \times 1)$ vector of binary responses. We use an additional latent variable Z to define our probit model.

General form of Probit Model:

$$\begin{aligned} z_i &\sim \mathcal{N}(x_i'\beta, 1) \\ y_i &= \mathbb{1}(z_i > 0) \end{aligned}$$

Which is the same as,

$$\begin{aligned} P(y_i = 1 \mid x_i, \beta) &= \Phi(x_i'\beta) \\ \Phi(z) &= \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} e^{-s^2/2} ds \end{aligned}$$

Complete Data Likelihood:

$$L(x^n, y^n \mid z^n, \beta) = \prod_{i=1}^{n} \mathcal{N}(z_i; x_i'\beta, 1) \times \prod_{i=1}^{n} \mathbb{1}(z_i > 0) y_i + \mathbb{1}(z_i < 0)(1 - y_i)$$

Prior Specification:

$$\beta \sim \mathcal{N}_p(\beta_0, \Sigma_\beta)$$

Posterior:

The posterior is given as,

$$p(z^n, \beta \mid x^n, y^n) \propto \mathcal{N}_p(\beta; \beta_0, \Sigma_\beta) \times \prod_{i=1}^{n} \mathcal{N}(z_i; x_i'\beta, 1) \times \prod_{i=1}^{n} [\mathbb{1}(z_i > 0) y_i + \mathbb{1}(z_i < 0)(1 - y_i)]$$

Full Conditionals:

To compute this posterior we can use Gibbs Sampling, for which we need to compute full conditionals.

$$\begin{aligned} p(\beta \mid x^n, y^n, z^n) &\propto \mathcal{N}_p(\beta; \beta_0, \Sigma_\beta) \times \prod_{i=1}^{n} \mathcal{N}(z_i; x_i'\beta, 1) \\ &\propto \mathcal{N}_p(\beta^*, \Sigma_\beta^*) \end{aligned}$$

Where, (Referring to class notes,)

$$\Sigma_\beta^* = (\Sigma_\beta^{-1} + X'X)^{-1}$$

Let $\Sigma_\beta^{-1} = 0$. (Improper Prior)

$$\begin{aligned} \beta^* &= (X'X)^{-1}X'z \\ \Sigma_\beta^* &= (X'X)^{-1} \end{aligned}$$

$$\begin{aligned} p(z_i \mid z_{\sim i}, \beta, x^n, y^n) &\propto \mathcal{N}(z_i; x_i'\beta, 1) \times [\mathbb{1}(z_i > 0)y_i + \mathbb{1}(z_i < 0)(1 - y_i)] \\ \therefore p(z_i \mid y_i = 1, z_{\sim i}, \beta, x^n) &\propto \mathcal{N}_+(x_i'\beta, 1) \\ \therefore p(z_i \mid y_i = 0, z_{\sim i}, \beta, x^n) &\propto \mathcal{N}_-(x_i'\beta, 1) \end{aligned}$$

Where, $\mathcal{N}_+$ refers to the positive support of the Normal Distribution $[0, \infty]$, and $\mathcal{N}_-$ is the negative support $[-\infty, 0]$.

Simulation:
I used one predictor, with true beta values $[2, 5]$. Using the above full conditionals we can do Gibbs Sampling as follows,

(1) Start with $\{\mathrm{z}_i^{(0)}\}$

(2) Update $\beta^{(s)} \sim \mathcal{N}_p(\beta^*, \Sigma_\beta^*)$.

(3) Update $(z_i^{(s)} \mid y_i = 1) \sim \mathcal{N}_+(x_i'\beta, 1)$ OR $(z_i^{(s)} \mid y_i = 0) \sim \mathcal{N}_-(x_i'\beta, 1)$.

Sampling Results:
I used 5000 samples with 1000 Burn-In. Following are estimates from Gibbs Sampler.
$\beta_0 = 1.9550$ $[0.7548, 3.5509]$.
$\beta_1 = 4.5234$ $[2.6571, 6.7894]$.

Appendix:

```
%% STA 601 - Homework 14
% Author: Kedar Prabhudesai
% Created on: 11/10/2013

close all;
clear all;

% Simulate Data
TrueBeta = [2 5];
% Predictors from Normal distribution with mean 1 and std dev. 2
X = 1 + 2.*randn(100,1);
% We append ones to X
X = cat(2,ones(100,1),X);
% Generate Z - Latent data
Z = X*TrueBeta' + randn(100,1);
% Set Y based on Z
Y = Z > 0;

XXInv = pinv(X'*X);

% Initialize prior values for bivariate beta prior
b0 = [0 0];
Sb = [0 0;0 0];

nTrials = 5000;
nBurnIn = 1000;
betaSamples = zeros(nTrials,2);
zSamples = zeros(nTrials,size(X,1));
% Initialize Latent data
zSamples(1,:) = rand(100,1);
zDistObj = makedist('Normal');

for iTrial = 2:nTrials
    home;disp(iTrial)
    % Update Beta
    bStar = XXInv*X'*zSamples(iTrial-1,:)';
    bHere = mvnrnd(bStar,XXInv);
    betaSamples(iTrial,:) = bHere;

    % Update z
    for iData = 1:size(X,1)
        xHere = X(iData,:);
        XB = xHere*bHere';

        zDistObj.mu = XB;
        zRand = zDistObj.random();

        if Y(iData) == 1
            while zRand < 0
                zRand = zDistObj.random();
            end
        else
            while zRand > 0
                zRand = zDistObj.random();
            end
        end
        zSamples(iTrial,iData) = zRand;
    end
end

```

```
% Burn-In
betaSamples(1:nBurnIn,:) = [];
zSamples(1:nBurnIn,:) = [];
```