Bayesian Inference

Assignment 3

Sofía Braña Recuero
Roberto Rey Sieiro
Javier Martínez Llamas
Ricardo Hortelano Sánchez
Marta Cortés Ocaña
Santiago César Gómez Fernández

05/03/2020

1 Introduction (A)

We are going to use STAN for logit bayesian regression. Stan is a probabilistic programming language for specifiying statistical models. STAN implements MCMC algorithms for Bayesian inference, gradient-based variational Bayesian methods for approximate Bayesian inference and regression models from simple linear regressions to multilevel generalized linear models.

In our case we have logit regression mode, where we have to predict a binary outcome. Our data is composed by 6 samples where our response variable is the action of customers buying or not buying. We have to classify the customers (Buy/Don't buy) based on the number of bets on a specific website.

2 Model (M)

The hierarchical model that we are assuming is:

$$(Y|X,\theta) \sim Ber(\theta)$$

As we already know logistic regression is a generalized linear model with binary outcomes, the log odds link function is:

$$\text{logit}(x_i) = \log\left(\frac{x_i}{1-x_i}\right) = \alpha + \beta x$$

Where our prior information is

$$\alpha \sim N(0,5) \qquad \beta \sim N(0,3)$$

Assuming Normal distribution is given in terms of Normal(μ,σ). This is relevant since STAN's Normal distribution takes as argument the standard deviation and not the variance.

First of all we create the STAN program where we are going to define the data, the parameters and the model.

```
library(rstan)

rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())

#Data
y <- c(1,0,1,1,0,0)  # for actions (1=Buy)
x <- c(3,5,2,5,10,6) # for the number of bets

#STAN Model

model <- "
  data {
    int<lower=0> N;
    vector[N] x;
    int<lower=0,upper=1> y[N];
  }

  parameters {
    real alpha;
    real beta;
  }

  model {
    alpha ~ normal(0,5);
    beta ~ normal(0,2);
    y ~ bernoulli_logit(alpha + beta * x);
  }
"
```

According to the model part in the code, the block called *data*, reads the external information that you are providing to the model. As STAN is written in C++ it is a compiled language, this means that we have to define the class of our variables before assigning something to them. The parameters block defines our sampling space which is the real numbers.

In the last part of the model we have to define our prior distributions and our likelihood function, we know the distributions for α and β, so we set both of them.

To define our likelihood funcion we have to take the inverse of the logit function:

$$\text{logit}^{-1}(x_i) = \frac{1}{1 + e^{-(\alpha+\beta x)}}$$

This formulation [1] uses the logit-parameterized version of the Bernoulli distribution which can be defined as

$$\text{BernoulliLogit}\ (y|x_i) = \ \text{Bernoulli}\ \left(y|\,\text{logit}^{-1}(x_i)\right) = \begin{cases} \text{logit}^{-1}(x_i) & \text{if } y = 1, \text{ and} \\ 1 - \text{logit}^{-1}(x_i) & \text{if } y = 0 \end{cases}$$

Stan supplies a direct parameterization in terms of a logit-transformed chance-of-success parameter. This function is called as $y \sim bernoullilogit(x_i)$ where in our case is $y \sim bernoullilogit(\alpha + \beta x)$.

2.1 Model Fit

In this section we proceed to fit the model to our data:

```
fit <-  stan(model_code=model, data=list(x=x, y=y, N=length(x)), iter=5000)
fit
```

```
Inference for Stan model: cfcbc26e9b7ae086cf1801b7f8a7f03b.
4 chains, each with iter=5000; warmup=2500; thin=1;
post-warmup draws per chain=2500, total post-warmup draws=10000.

       mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
alpha  5.08    0.08 2.97  0.11  2.96  4.82  6.94 11.60  1509    1
beta  -1.11    0.02 0.62 -2.48 -1.49 -1.05 -0.65 -0.10  1441    1
lp__  -3.49    0.02 1.09 -6.43 -3.89 -3.16 -2.73 -2.45  2306    1

Samples were drawn using NUTS(diag_e) at Sun Mar 08 22:09:50 2020.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

Obtaining an estimated mean α of 5.08 with a standard deviation of 2.97 and a mean β of -1.11 with standard deviation of 0.62.

3 Output Analysis (A^{-1})

In order to analyze our output we are going to plot our *fit* where we can check the posterior distributions for both of the parameters.

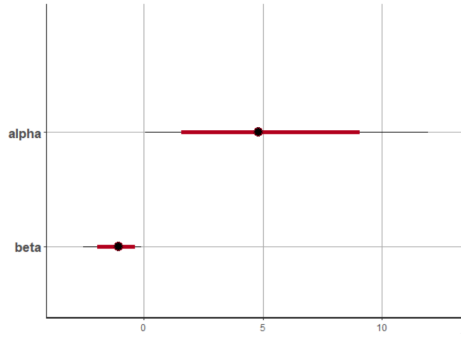

Figure 1: Posterior distribtuions for α and β

The bars represent the 80% intervals (red) and the 95% intervals (outer level). However, the plot provided by default is not very informative so we are going to make use of the library *shinystan*. Shinystan is a R package that provides visual and numerical summaries of model parameters and convergence diagnostics for MCMC simulations.

```
library(shinystan)
launch_shinystan(fit)
```

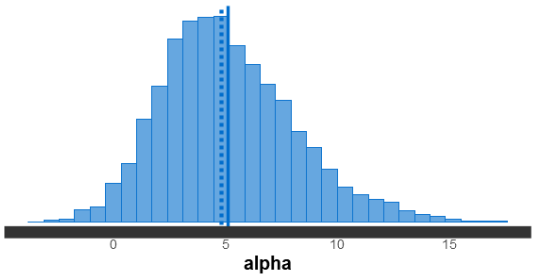

Figure 2: α Posterior Distribution

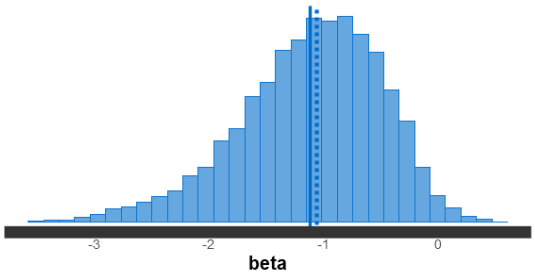

Figure 3: β Posterior Distribution

The following table provides us with the Rhat of the parameters. Since neither for α nor for β is larger than 1 the convergence of the MCMC has been successful.

Parameter	Rhat	n_eff	mean	sd	2.5%	50%	97.5%
alpha	1.0	1623	5.1	3.0	0.1	4.8	11.9
beta	1.0	1630	-1.1	0.6	-2.5	-1.1	-0.1

There is, however, an obvious issue with our posterior distribution: both distributions for α and β are skewed. This causes an increase in the variance of the estimator and reduces the accuracy. Nonetheless, this is due to our really small sample (of size 6) and could be easily improved as this sample size increases.

References

[1] S. R. Bowling, M. T. Khasawneh, S. Kaewkuekool, and B. R. Cho, "A logistic approximation to the cumulative normal distribution," *Journal of Industrial Engineering and Management*, vol. 2, no. 1, 2009.