Introducción a la Inferencia Bayesiana usando RStan

Ignacio Evangelista

27 de agosto de 2019

```
library(rstan)
library(ggplot2)
library(bayesplot)
```

Ejemplo 1: Caso Simple

```
N      <- c(20,100)
Y      <- c(4,70)
```

Modelo 1

```
model1_stan <- "
data {
  int<lower=1> n;
  int N[n];
  int Y[n];
}
parameters {
  real<lower=0, upper=1> theta;
}
model {
  theta ~ beta(2,2);
  Y ~ binomial(N, theta);
}"
```

```
model1 <- stan_model(model_code = model1_stan)
```

```
data_list <- list(Y=Y, N=N, n=length(Y))
```

```
model1_fit <- sampling(object=model1,
                       data=data_list,
                       chains = 3,
                       iter = 1000,
                       warmup = 100)
```

Ejemplo 2: Regresión Lineal

```
x <- rnorm(100, 10, 1)
y <- 3*x+2
y <- y + rnorm(length(y),0,1)
data <- data.frame(x=x,y=y)
```

Modelo 2

```
model2_stan <- "
data {
  int<lower=0> N;
  vector[N] y;
  vector[N] x;
}
parameters {
  real beta0;
  real beta1;
  real<lower=0> sigma;
}
model {
  beta0 ~ normal(0,10);
  beta1 ~ normal(0,10);
  sigma ~ normal(0,20);
  y ~ normal(beta0 + beta1*x,sigma);
}"
```

```
model2 <- stan_model(model_code = model2_stan)
```

```
data_list <- list(x=x, y=y, N=length(y))
```

```
model2_fit <- sampling(object=model2,
                       data=data_list,
                       chains = 1,
                       iter = 8000,
                       warmup = 1000)
```

Modelo 3

```
model3_stan <- "
data {
  int<lower=0> N;
  vector[N] y;
  vector[N] x;
}
transformed data {
  vector[N] x_std;
  x_std = (x - mean(x)) / sd(x);
}
parameters {
  real beta0;
  real beta1;
  real<lower=0> sigma;
}
model {
  beta0 ~ normal(0,10);
  beta1 ~ normal(0,10);
  sigma ~ normal(0,20);
  y ~ normal(beta0 + beta1*x_std,sigma);
}
generated quantities { // para cada muestra de la cadena!
  real beta1_orig;
  real beta0_orig;
  beta1_orig = beta1 / sd(x);
  beta0_orig = beta0 - beta1 * mean(x) / sd(x);
}"
```

```
model3 <- stan_model(model_code = model3_stan)
```

```
model3_fit <- sampling(object=model3,
                       data=data_list,
                       chains = 1,
                       iter = 8000,
                       warmup = 1000)
```

Modelo 4

```
model4_stan <- "
data {
  int<lower=0> N;
  vector[N] y;
  vector[N] x;
}
transformed data {
  vector[N] x_std;
  x_std = (x - mean(x)) / sd(x);
}
parameters {
  real beta0;
  real beta1;
  real<lower=0> sigma;
}
model {
  beta0 ~ normal(0,10);
  beta1 ~ normal(0,10);
  sigma ~ normal(0,20);
  y ~ normal(beta0 + beta1*x_std,sigma);
}
  generated quantities { // para cada muestra de la cadena!
  real beta1_orig;
  real beta0_orig;
  vector[N] y_rep;
  beta1_orig = beta1 / sd(x);
  beta0_orig = beta0 - beta1 * mean(x) / sd(x);
  for(n in 1:N) {
  y_rep[n] = normal_rng(beta0_orig + beta1_orig * x[n], sigma);
  }
}
"
```

```
model4 <- stan_model(model_code = model4_stan)
```

```
model4_fit <- sampling(object=model4,
                       data=data_list,
                       chains = 1,
                       iter = 4000,
                       warmup = 100)
```

Ejemplo 3: t-test

```
n1 <- 36
n2 <- 30
Ntotal <- n1 + n2
y1 <- rnorm(n1,1.1,0.3)
y2 <- rnorm(n2,0.7,0.6)
y<-c(y1,y2)
group_id<-c(rep(1,n1),rep(2,n2))
```

Modelo 5

```
model5_stan <- "
data {
  int n1;
  int n2;
  int N;
  vector[N] y;
  int group_id[N];
  real mu_prior_location;
  real mu_prior_scale;
}
parameters {
  real mu[2];
  real<lower=0, upper=1000> sigma[2];
  real<lower=0, upper=1000> nu;
}
model {
  mu ~ normal(mu_prior_location, mu_prior_scale);
  sigma ~ normal(0, 20);
  nu ~ exponential(1.0/29.0);
  y ~ student_t(nu, mu[group_id],sigma[group_id]);
}
generated quantities {
  real diff_mu;
  vector[n1] y1_rep;
  vector[n2] y2_rep;

  diff_mu = mu[1] - mu[2];

  for(n in 1:n1) {
  y1_rep[n] = student_t_rng(nu,mu[1],sigma[1]);
  }

  for(n in 1:n2) {
  y2_rep[n] = student_t_rng(nu,mu[2],sigma[2]);
  }
}"
```

```
model5 <- stan_model(model_code = model5_stan)
```

```
data_list <- list(group_id=group_id,
                  y=y,
                  n1=n1,
                  n2=n2,
                  N=Ntotal,
                  mu_prior_location=(mean(y1)+mean(y2))/2,
                  mu_prior_scale=max(sd(y1),sd(y2))*1000)
```

```
model5_fit <- sampling(object=model5,
                       data=data_list,
                       chains = 4,
                       iter = 4000,
                       warmup = 100)
```