Completely Pooled

Yash Amonkar

February 3, 2019

Loading Required Libraries and Pre-requisites

```
library(dplyr)
library(rstan)
options(mc.cores = parallel::detectCores()) #Local Multicore CPU excess
```

Data Input

Reading the cleaned data

```
input_data <- read.table("Dataset for Regression.txt", sep =" ", header = TRUE)
head(input_data)
```

```
##   county_FIPS   week_end dum_1stwk_viol dum_2ndwk_viol defl_medsales_pc
## 1        1001 2013-03-02              0              0      0.005202615
## 2        1001 2015-11-28              0              0      0.004205149
## 3        1001 2015-10-24              0              0      0.004734095
## 4        1001 2013-02-23              0              0      0.004074375
## 5        1001 2011-05-07              0              0      0.004465746
## 6        1001 2016-07-30              0              0      0.003311779
##   defl_bottlesales_pc
## 1          0.05323357
## 2          0.07018714
## 3          0.06845623
## 4          0.05924734
## 5          0.08861730
## 6          0.07787990
```

```
input_data <- subset(input_data, county_FIPS < 1010) #This line is to be included in t
he demonstration only. REMOVE WHEN RUNNING ACTUAL PROGRAM
```

Stan Model Used

```
writeLines(readLines("Completely_Pooled_Multiparameter.stan"))
```

```
## data {
## int<lower=0> N;                      //Number of observations
## int<lower=0> K;           //Number of Variables including the intercept
## matrix[N,K] X;        //Covariate Matrix.Note:- First column is intercept
## int<lower=0, upper=1> z[N];      //Violations Status
## real<lower=0> sd_variables;         //Prios for the coefficients
## }
##
## parameters {
## vector[K] beta;                       //Coefficient
## real<lower=0, upper=1> nu;            //Reporting Rate
## }
##
## model {
## //priors
## beta ~ normal(0,sd_variables);
## nu ~ beta(1,1);
##
## //model
## z ~ bernoulli(nu*inv_logit(X*beta)); //Pr(z=1) = nu*inverse_logit(X*beta)
##
## }
```

Stan Parameters

```
sd_beta <- 25
chains <- 6
iter <- 1000
warmup <- 500
X <- as.matrix(data.frame(1, input_data$defl_medsales_pc,input_data$defl_bottlesales_p
c))
K <- dim(X)[2]
data_list <- list(N=dim(input_data)[1], K = K, X = X, z=input_data$dum_2ndwk_viol, sd_
variables = sd_beta)
```

Stan Model

```
model_stan <- stan(file = "Completely_Pooled_Multiparameter.stan", data=data_list, cha
ins = chains, iter = iter, warmup = warmup,control = list(adapt_delta = 0.95))
```

```
## In file included from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/BH/include/
boost/config.hpp:39:0,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/BH/include/
boost/math/tools/config.hpp:13,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/stan/math/rev/core/var.hpp:7,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/stan/math/rev/core/gevv_vvv_vari.hpp:5,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/stan/math/rev/core.hpp:12,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/stan/math/rev/mat.hpp:4,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/stan/math.hpp:4,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/src/stan/model/model_header.hpp:4,
##                  from file39649bb1e2.cpp:8:
## C:/Users/Yash Amonkar/Documents/R/win-library/3.3/BH/include/boost/config/compiler/
gcc.hpp:186:0: warning: "BOOST_NO_CXX11_RVALUE_REFERENCES" redefined
##  #  define BOOST_NO_CXX11_RVALUE_REFERENCES
##  ^
## <command-line>:0:0: note: this is the location of the previous definition
## In file included from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/stan/math/rev/core.hpp:44:0,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/stan/math/rev/mat.hpp:4,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/stan/math.hpp:4,
##                  from C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeader
s/include/src/stan/model/model_header.hpp:4,
##                  from file39649bb1e2.cpp:8:
## C:/Users/Yash Amonkar/Documents/R/win-library/3.3/StanHeaders/include/stan/math/re
v/core/set_zero_all_adjoints.hpp:14:17: warning: 'void stan::math::set_zero_all_adjoin
ts()' defined but not used [-Wunused-function]
##      static void set_zero_all_adjoints() {
##                  ^
```

```
## Warning: There were 1 divergent transitions after warmup. Increasing adapt_delta ab
ove 0.95 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
print("Stan Program has run and is completed")
```

```
## [1] "Stan Program has run and is completed"
```

```
print(model_stan)
```

```
## Inference for Stan model: Completely_Pooled_Multiparameter.
## 6 chains, each with iter=1000; warmup=500; thin=1; 
## post-warmup draws per chain=500, total post-warmup draws=3000.
## 
##           mean se_mean    sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## beta[1]  -3.52    0.80  6.68  -6.73  -5.74  -5.08  -4.22  18.96    70 1.08
## beta[2]  -1.81    0.69 23.11 -48.62 -16.71  -2.11  14.07  42.27  1131 1.00
## beta[3]  10.94    0.36  8.20  -5.35   7.51  11.11  14.74  26.33   523 1.01
## nu        0.47    0.02  0.30   0.01   0.20   0.47   0.73   0.97   304 1.01
## lp__    -82.73    0.20  1.98 -88.14 -83.44 -82.14 -81.33 -80.50    97 1.05
## 
## Samples were drawn using NUTS(diag_e) at Sun Feb 03 13:45:01 2019.
## 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).
```

Saving the Results

```
data_summary <- summary(model_stan)$summary[1:(K+1),] #Here we analyze check the global parameters.
data_draws <- as.data.frame(extract(model_stan))[,1:(K+1)]

write.table(data_summary, "Stan Model Summary - Pooled.txt", sep = " ")
write.table(data_draws, "Draws - Pooled.txt", sep = " ")
print("Saving data complete")
```

```
## [1] "Saving data complete"
```

```
pdf(file="Histrogram Plots.pdf")
for(i in 1:ncol(data_draws)){
  hist(data_draws[,i], freq = FALSE, main = colnames(data_draws)[i], xlab = colnames(data_draws)[i])
  abline(v = mean(data_draws[,i]), col = 'red', lwd = 2)
}
dev.off()
```

```
## png 
##   2
```

```
print("Plotting is complete")
```

```
## [1] "Plotting is complete"
```