Partially Pooled Model

Yash Amonkar

February 3, 2019

Loading Required Libraries and Pre-requisites

```
library(rstan)
library(dplyr)
options(mc.cores = parallel::detectCores()) #Local Multicore CPU excess
print("Library Loading Complete")
```

```
## [1] "Library Loading Complete"
```

Data Input

Reading the cleaned data

```
input_data <- read.table("Dataset for Regression.txt", sep =" ", header = TRUE)
print("Data Input successful. We now run the stan program")
```

```
## [1] "Data Input successful. We now run the stan program"
```

Subsetting_Dataset_to_state_level

This file is for running a partailly pooled model for each state. Texas is 48 oklahoma is 40

```
state_FIPS_code <- 48
state_data <- input_data %>%
              filter(county_FIPS > state_FIPS_code*1000 & county_FIPS < (state_FIPS_co
de+1)*1000 )
```

```
## Warning: package 'bindrcpp' was built under R version 3.3.3
```

```
## Warning: `as_dictionary()` is soft-deprecated as of rlang 0.3.0.
## Please use `as_data_pronoun()` instead
## This warning is displayed once per session.
## Warning: `new_overscope()` is soft-deprecated as of rlang 0.2.0.
## Please use `new_data_mask()` instead
## This warning is displayed once per session.
```

```
## Warning: The `parent` argument of `new_data_mask()` is deprecated.
## The parent of the data mask is determined from either:
##
##   * The `env` argument of `eval_tidy()`
##   * Quosure environments when applicable
## This warning is displayed once per session.
```

```
## Warning: `overscope_clean()` is soft-deprecated as of rlang 0.2.0.
## This warning is displayed once per session.
```

```
state_data <- state_data %>% group_by(county_FIPS) %>% mutate(defl_medsales_pc = scale
(defl_medsales_pc), defl_bottlesales_pc = scale(defl_bottlesales_pc))

state_data <- transform(state_data, id=match(county_FIPS, unique(county_FIPS)))
head(state_data)
```

```
##   county_FIPS   week_end dum_1stwk_viol dum_2ndwk_viol defl_medsales_pc
## 1       48001 2015-07-04              1              1      -0.99775249
## 2       48001 2015-07-11              0              1       0.18810499
## 3       48001 2008-04-19              0              0      -0.07863564
## 4       48001 2010-08-07              0              0       0.87653457
## 5       48001 2008-07-19              0              0       0.54973691
## 6       48001 2015-04-11              0              0      -0.26062889
##   defl_bottlesales_pc id
## 1          1.59264905  1
## 2          2.03546264  1
## 3         -0.90782945  1
## 4          0.79347718  1
## 5          0.04681694  1
## 6          1.20106078  1
```

```
state_data <- subset(state_data, county_FIPS < 48006) #This line is to be included in
the demonstration only. REMOVE WHEN RUNNING ACTUAL PROGRAM
print("Data Subsetting successful. We now run the stan program")
```

```
## [1] "Data Subsetting successful. We now run the stan program"
```

Stan Code

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

```
## data {
## int<lower=1> N;  //The number of data points
## int<lower=1> K;  //The number of covarites plus intercept.
## int<lower=1> J;  // The number of groups or states of hierarhicy
## matrix[N,K] X;   // The Data Matrix
## int<lower=1> id[N];  //vector of department indices.
## int<lower=0, upper=1> z[N];  //The response variable
## }
##
## parameters {
## real beta[K];  //The national level coefficients
## vector[K] beta_group[J]; //matrix of group-level regression coefficients
## real<lower=0, upper=1> nu;
## real<lower=0,upper=1> nu_group[J];
## }
##
##
## model {
## //priors
## for(j in 1:J){beta_group[j] ~ normal(beta,100);}
## for(j in 1:J){nu_group[j] ~ beta(((1-nu)*nu-(nu*(1-nu)/J))*nu/(nu*(1-nu)/J),((1-nu)
*nu-(nu*(1-nu)/J))*nu*(1-nu)/(nu*nu*(1-nu)/J)); }
## beta ~ normal(0,10);
## nu ~ beta(1,1);
##
## //model
## for(n in 1:N){
## z[n] ~ bernoulli((nu_group[id[n]]*exp(X[n] * beta_group[id[n]]))/(exp(X[n] * beta_g
roup[id[n]])+1));
## }
## }
```

Stan Parameters

```
N <- dim(state_data)[1]
J <- max(state_data$id)
groups <- state_data$id
chains <- 3
iter <- 1000
warmup <- 500
X <- as.matrix(data.frame(1, state_data$defl_medsales_pc,state_data$defl_bottlesales_p
c))
K <- dim(X)[2]
data_list <- list(N=N, K = K, J=J, X = X, z=state_data$dum_2ndwk_viol, id = groups)
```

Running_the_Stan_Model

```
model_stan <- stan(file = "Partial_Pooling_Multiparameter.stan", data=data_list, chain
s = 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 file3bd8211193e.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 file3bd8211193e.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 1500 transitions after warmup that exceeded the maximum treedep
th. Increase max_treedepth above 10. See
## http://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

```
## 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, pars = c("nu", "nu_group"))
```

```
## Inference for Stan model: Partial_Pooling_Multiparameter.
## 3 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=1500.
##
##             mean se_mean   sd 2.5%  25%  50%  75% 97.5% n_eff Rhat
## nu          0.39    0.14 0.19 0.10 0.24 0.34 0.62  0.65     2 3.31
## nu_group[1] 0.05    0.00 0.01 0.04 0.05 0.05 0.06  0.07     6 1.83
## nu_group[2] 0.03    0.00 0.01 0.02 0.03 0.03 0.03  0.04     2 2.24
##
## Samples were drawn using NUTS(diag_e) at Sun Feb 03 17:12:10 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 Model Ouput and Simulation Draws

```
data_summary <- summary(model_stan)$summary #Here we analyze check the global paramete
rs.
data_draws <- as.data.frame(extract(model_stan))
print("Saving data complete")
```

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

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

The simulation draws are saved for computation of different quantities if needed.

Plotting the results and uncertainity

```r
Reporting_Rate <- as.data.frame(summary(model_stan, pars = c("nu", "nu_group")))
Reporting_Rate$index <- 1:dim(Reporting_Rate)[1]
J <- dim(Reporting_Rate)[1]
ticks<- 0:(dim(Reporting_Rate)[1])



Coefficient_Estimates <- as.data.frame(summary(model_stan, pars = c("beta", "beta_grou
p")))
Coefficient_Medicine <- Coefficient_Bottles <- Intercept <- as.data.frame(matrix(0,nco
l=ncol(Coefficient_Estimates), nrow=nrow(Coefficient_Estimates)/3))
colnames(Coefficient_Medicine) <- colnames(Coefficient_Bottles) <- colnames(Intercep
t) <- colnames(Coefficient_Estimates)

for(i in 1:(max(state_data$id)+1)){
  index <- 3*(i-1) + 1
  Intercept[i,] <- Coefficient_Estimates[index,]
  Coefficient_Medicine[i,] <- Coefficient_Estimates[index+1,]
  Coefficient_Bottles[i,] <- Coefficient_Estimates[index+2,]
}
Coefficient_Medicine$index <- Coefficient_Bottles$index <- Intercept$index <- 1:dim(In
tercept)[1]



pdf("Uncertainty_Estimates.pdf")

#Plot for the Medicine Sales
plot (Coefficient_Medicine$index, Coefficient_Medicine$summary.mean, bty="l", pch=20,
xaxt="n", xlab = "Pooled and County Medicine Sales Effect", ylab = "Med Sales Coeffici
ent", main = "Uncertainty across effect of medicine sales")
abline(h=0, lty = 2)
axis(1,at=ticks,labels=ticks)
lines (rep(Coefficient_Medicine$index[1],2), c(Coefficient_Medicine$summary.97.5.[1],
Coefficient_Medicine$summary.2.5.[1]), lwd=.5, lty = 2, col = 'red')
lines (rep(Coefficient_Medicine$index[1],2), c(Coefficient_Medicine$summary.25.[1], Co
efficient_Medicine$summary.75.[1]), lwd=.5, col = 'red')
for (j in 2:J)
  {
  lines (rep(Coefficient_Medicine$index[j],2), c(Coefficient_Medicine$summary.97.5.
[j], Coefficient_Medicine$summary.2.5.[j]), lwd=.5, lty = 2)
  lines (rep(Coefficient_Medicine$index[j],2), c(Coefficient_Medicine$summary.25.[j],
Coefficient_Medicine$summary.75.[j]), lwd=.5)

}


#Plot for Bottle Sales
```

```
plot (Coefficient_Bottles$index, Coefficient_Bottles$summary.mean, bty="l", pch=20, xaxt="n", xlab = "Pooled and County Bottle Sales Effect", ylab = "Bottle Sales Coefficient", main = "Uncertainty across effect of Bottle sales")
abline(h=0, lty = 2)
axis(1,at=ticks,labels=ticks)
lines (rep(Coefficient_Bottles$index[1],2), c(Coefficient_Bottles$summary.97.5.[1], Coefficient_Bottles$summary.2.5.[1]), lwd=.5, lty = 2, col = 'red')
lines (rep(Coefficient_Bottles$index[1],2), c(Coefficient_Bottles$summary.25.[1], Coefficient_Bottles$summary.75.[1]), lwd=.5, col = 'red')
for (j in 2:J)
  {
  lines (rep(Coefficient_Bottles$index[j],2), c(Coefficient_Bottles$summary.97.5.[j], Coefficient_Bottles$summary.2.5.[j]), lwd=.5, lty = 2)
  lines (rep(Coefficient_Bottles$index[j],2), c(Coefficient_Bottles$summary.25.[j], Coefficient_Bottles$summary.75.[j]), lwd=.5)

}




plot (Reporting_Rate$index, Reporting_Rate$summary.mean, bty="l", pch=20, ylim = c(-0.1,1.1), xaxt="n", xlab = "Pooled and County Index", ylab = "Reporting Rate", main = "Uncertainty across reporting rates")
abline(h=0, lty = 2)
axis(1,at=ticks,labels=ticks)
lines (rep(Reporting_Rate$index[1],2), c(Reporting_Rate$summary.97.5.[1], Reporting_Rate$summary.2.5.[1]), lwd=.5, lty = 2, col = 'red')
lines (rep(Reporting_Rate$index[1],2), c(Reporting_Rate$summary.25.[1], Reporting_Rate$summary.75.[1]), lwd=.5, col = 'red')
for (j in 2:J){
  lines (rep(Reporting_Rate$index[j],2), c(Reporting_Rate$summary.97.5.[j], Reporting_Rate$summary.2.5.[j]), lwd=.5, lty = 2)
  lines (rep(Reporting_Rate$index[j],2), c(Reporting_Rate$summary.25.[j], Reporting_Rate$summary.75.[j]), lwd=.5)

}

dev.off()
```

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