BDA - Project Work

Jacopo Losi, Nicola Saljoughi

Contents

Introduction

This project is based on a study carried out in 2015 by a group of researchers to estimate the incidence of serious suicide attempts in Shandong, China, and to examine the factors associated with fatality among the attempters.
We have chosen to examine a dataset on suicides because it is a really important but often underconsidered problem in today's society. Not only this problem reflects a larger problem in a country societal system but it can also be a burden for hospital resources. We think that by being able to talk about it more openly and by truly trying to estimate its size and impact we can start to understand where the causes are rooted and what can be done to fight it.
We invite the reader to check the source section to further read about the setting and results of the named paper.

Analysis Problem

The objective of the project is to use the bayesian approach to develop models to evaluate the most influential factors related to serious suicide attempts (SSAs, defined as suicide attempts resulting in either death or hospitalisation) and being able to make predictions for the years following the period where the study was set.

Data

Data from two independent health surveillance systems were linked, constituted by records of suicide deaths and hospitalisations that occured among residents in selected countries during 2009-2011.
The data set is constituted by 2571 observations of 11 variables:

- `Person_ID`: ID number, $1, ..., 2571$
- `Hospitalised`: *yes* or *no*
- `Died`: *yes* or *no*
- `Urban`: *yes*, *no* or *unknown*
- `Year`: 2009, 2010 or 2011
- `Month`: $1, ..., 12$
- `Sex`: *female* or *male*
- `Age`: years
- `Education`: *iliterate*, *primary*, *Secondary*, *Tertiary* or *unknown*
- `Occupation`: one of ten categories
- `method`: one of nine methods

It is important to notice that the population in the study is predominantly rura and that the limitation of the study is that the incidence estimates are likely to be underestimated due to underreporting in both surveillance systems.

Source

Sun J, Guo X, Zhang J, Wang M, Jia C, Xu A (2015) "Incidence and fatality of serious suicide attempts in a predominantly rural population in Shandong, China: a public health surveillance study," BMJ Open 5(2): e006762. https://doi.org/10.1136/bmjopen-2014-006762

Data downloaded via Dryad Digital Repository. https://doi.org/10.5061/dryad.r0v35

Analysis

In this section we will carry out our analysis following the bayesian approach. First we will develop a multiple logistic regression model using all the variables, after that we will do variable selection to determine which are the most influential factors and develop a second multiple logistic regression model using the selected variables. After that, we will assess the convergence and efficiency of the models, do posterior predictive checking and compare the models. To conclude we carry out sensitivity analysis with respect to prior choices and eventually andwer our analysis problem.

Model description

In order to evaluate the factors which influence the probability of SSA the most it is an obvious chioice to develop a multiple logistic regression model. Two different models have been implemented which will then be compared in the following analysis:

- **Full logistic regression model** where all the parameters are included
- **Reduced logistic regression model** that only includes the parameters selected in the variable selection phase.

Full Logistic Regression Model

Reduced Logistic Regression Model

Prior choices

Code

Data loading

First of all we load the data. Notice that some processing was done on the original data removing samples with missing entries (that resulted to constitute less than the 6 % of the dataset) and turning labels from strings into integers.

```
## Create Stan data
dat <- list(N        = nrow(mydata),
            p        = ncol(mydata) - 2,
            died     = as.numeric(mydata$Died),
            urban    = as.numeric(mydata$Urban),
            year     = as.numeric(mydata$Year),
            season   = as.numeric(mydata$Season),
            sex      = as.numeric(mydata$Sex),
            age      = as.numeric(mydata$Age),
            edu      = as.numeric(mydata$Education),
            job      = as.numeric(mydata$Occupation),
            method   = as.numeric(mydata$method))
```

In this phase we are working on testing different models, therefore it is worth to take only some random samples from the data. As a matter of fact, the dataset that we have is big and thus the computation on the whole dataset will take a lot of time.

Therefore, we will proceed as follows: * we will generate a vector of 50 random number taken from our dataset; * we will test the models with this data, that are sufficient for not loosing in generality; * we will run the final model on the whole dataset.

```
random_index <- sample(mydata$Person_ID, size = 50, replace = TRUE)

data_reduced <- mydata[random_index, ]
data_reduced <- na.omit(data_reduced)
```

```r
## Create Stan data
dat_red <- list(N    = nrow(data_reduced),
                p        = ncol(data_reduced) - 2,
                died     = as.numeric(data_reduced$Died),
                urban    = as.numeric(data_reduced$Urban),
                year     = as.numeric(data_reduced$Year),
                season   = as.numeric(data_reduced$Season),
                sex      = as.numeric(data_reduced$Sex),
                age      = as.numeric(data_reduced$Age),
                edu      = as.numeric(data_reduced$Education),
                job      = as.numeric(data_reduced$Occupation),
                method   = as.numeric(data_reduced$method))
```

Full logistic regression model

Here we start by implementing the full logistic regression model.

```r
## FULL LOGISTIC REGRESSION MODEL

## Load Stan Model
fileNameOne <- "./logistic_regression_model.stan"
stan_code_full <- readChar(fileNameOne, file.info(fileNameOne)$size)
cat(stan_code_full)
```

```
## data {
##   // Define variables in data
##   // Number of observations (an integer)
##   int<lower=0> N;
##
##   // Number of parameters
##   int<lower=0> p;
##
##   // Variables
##   int  died[N];
##   int<lower=0>  year[N];
##   int<lower=0>  urban[N];
##   int<lower=0>  season[N];
##   int<lower=0>  sex[N];
##   int<lower=0>  age[N];
##   int<lower=0>  edu[N];
##   int<lower=0>  job[N];
##   int<lower=0>  method[N];
## }
##
## parameters {
##   // Define parameters to estimate
##   real beta[p];
## }
##
## transformed parameters  {
##   // Probability trasformation from linear predictor
##   real<lower=0> odds[N];
##   real<lower=0, upper=1> prob[N];
##   for (i in 1:N) {
```

```
##     odds[i] = exp(beta[1] + beta[2]*year[i]  + beta[3]*urban[i] +
##                           beta[4]*season[i] + beta[5]*sex[i]  +
##                           beta[6]*age[i]    + beta[7]*edu[i]  +
##                           beta[8]*job[i]    + beta[9]*method[i] );
##     prob[i] = odds[i] / (odds[i] + 1);
##   }
## }
##
## model {
##   // Prior part of Bayesian inference (flat if unspecified)
##
##   // Likelihood part of Bayesian inference
##     died ~ bernoulli(prob);
## }
```

Stan Code Running

The Stan models are run by using five chains constituted by 2000 iterations, a warmup length of 800 iterations and a thin equal to 10. Thin is a positive integer that specifies the period for saving samples; it is set by default = 1, and it is normally left to defaults. In our case though our posterior distribution takes up a lot of memory even when using a reduced dataset and we require a large numer of iteration to achieve effective sample size and therefore we decide to set it to 10 in this phase.

```
## Warning: There were 600 transitions after warmup that exceeded the maximum treedepth. Increase max_t
## http://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

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

```
## Warning: The largest R-hat is 1.89, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#r-hat
```

```
## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess
```

```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

```
## Inference for Stan model: 44efd1e4898e49d7c3da763fa46eaad0.
## 5 chains, each with iter=2000; warmup=800; thin=10;
## post-warmup draws per chain=120, total post-warmup draws=600.
##
##             mean se_mean      sd     2.5%      25%      50%      75%
## beta[1] -4117.96  992.19 2673.92 -9177.55 -6320.21 -3846.01 -1740.94
## beta[2]     2.86    0.18    1.11     0.83     2.00     2.77     3.61
## beta[3]     5.48    0.42    2.72     0.38     3.55     5.54     7.32
## beta[4]    -1.08    0.07    0.56    -2.12    -1.49    -1.07    -0.71
## beta[5]     2.55    0.20    1.46     0.04     1.56     2.53     3.36
## beta[6]     0.99    0.13    0.88    -0.56     0.34     0.93     1.59
## beta[7]    -4.71    0.31    1.59    -7.75    -5.96    -4.63    -3.54
## beta[8]  4119.42  993.26 2674.77   149.08  1739.38  3846.87  6327.03
## beta[9]    -0.61    0.12    0.53    -1.57    -0.97    -0.60    -0.26
##           97.5% n_eff Rhat
## beta[1] -154.91     7 2.22
## beta[2]    5.08    40 1.10
```

```
## beta[3]   10.94    43 1.08
## beta[4]    0.07    72 1.07
## beta[5]    5.93    53 1.06
## beta[6]    2.85    46 1.09
## beta[7]   -1.93    27 1.18
## beta[8] 9184.22     7 2.22
## beta[9]    0.47    21 1.16
##
## Samples were drawn using NUTS(diag_e) at Fri Dec 06 19:45:15 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).
```

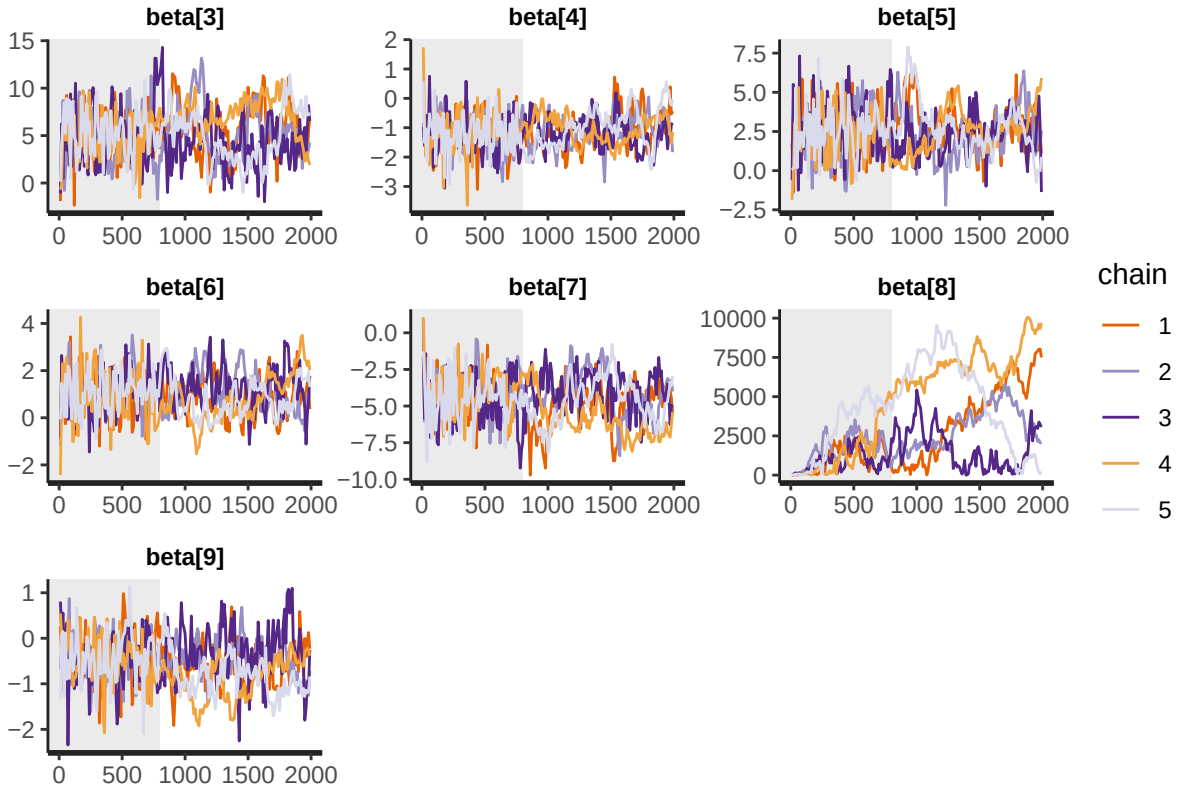
Variable selection

In this section we will evaluate the most influential factors and their correlation in order to select the most descriptive ones that will be used to contruct our second model (the reduced logistic regression model). First of all we process our data:

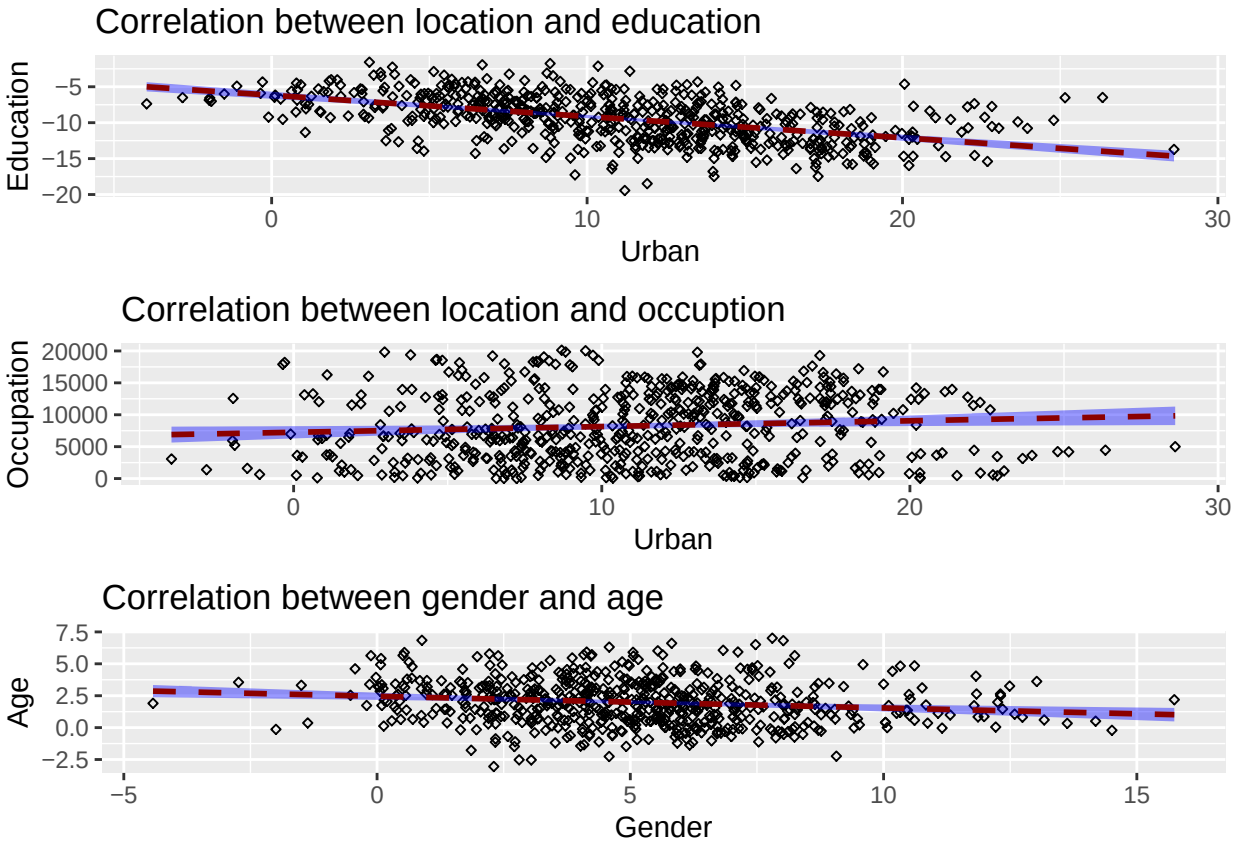
```
# Transform fitting over beta in a dataframe for the plots

beta_matrix <- zeros(length(extract(resStanFull)$beta[,1]), ncol(data_reduced) - 2)

for (i in 1:ncol(data_reduced) - 2)
  beta_matrix[,i] = beta_matrix[,i] + extract(resStanFull)$beta[,i]

beta_df <- as.data.frame(beta_matrix)
```

Now we show traceplots and generate scatter plots in order to evaluate the correlation between the parameters:

```
# Show traceplot
traceplot(resStanFull, pars = c('beta[3]','beta[4]', 'beta[5]',
                                'beta[6]', 'beta[7]', 'beta[8]',
                                'beta[9]'), inc_warmup = TRUE)
```




```
# Generate some scatter plots in order to see the correlations between parameters
scatter_1 <- ggplot(beta_df, aes(x=V3, y=V7)) +
                    ggtitle("Correlation between location and education") +
                    xlab("Urban") + ylab("Education") +
              geom_point(size=1, shape=23) +
              geom_smooth(method=lm, linetype="dashed", color="darkred", fill="blue")

scatter_2 <- ggplot(beta_df, aes(x=V3, y=V8)) +
                    ggtitle("Correlation between location and occuption") +
                    xlab("Urban") + ylab("Occupation") +
              geom_point(size=1, shape=23) +
              geom_smooth(method=lm, linetype="dashed", color="darkred", fill="blue")

scatter_3 <- ggplot(beta_df, aes(x=V5, y=V6)) +
                    ggtitle("Correlation between gender and age") +
                    xlab("Gender") + ylab("Age") +
              geom_point(size=1, shape=23) +
              geom_smooth(method=lm, linetype="dashed", color="darkred", fill="blue")


ggplot2.multiplot(scatter_1,scatter_2,scatter_3, cols=1)
```



Now we overlay histogram, density and mean value of the parameters. The most interesting plots are presented; using the mean value is interesting since we can understand which weakly informative priors can be designed.

```
plot_1 <- qplot(extract(resStanFull)$beta[,3], geom = 'blank',
                xlab = 'Values of weigth', ylab = 'Occurences', main='Urbans') +
  geom_histogram(aes(y = ..density..),col = I('red'), bins = 50) +
  geom_line(aes(y = ..density..), size = 1, col = I('blue'), stat = 'density', ) +
  geom_vline(aes(xintercept=mean(extract(resStanFull)$beta[,3])), col=I('yellow'), linetype="dashed", si

plot_2 <- qplot(extract(resStanFull)$beta[,5], geom = 'blank',
                xlab = 'Values of weigth', ylab = 'Occurences', main='Sex') +
  geom_histogram(aes(y = ..density..),col = I('red'), bins = 50) +
  geom_line(aes(y = ..density..), size = 1, col = I('blue'), stat = 'density', ) +
  geom_vline(aes(xintercept=mean(extract(resStanFull)$beta[,5])), col=I('yellow'), linetype="dashed", si

plot_3 <- qplot(extract(resStanFull)$beta[,6], geom = 'blank',
                xlab = 'Values of weigth', ylab = 'Occurences', main='Age') +
  geom_histogram(aes(y = ..density..),col = I('red'), bins = 50) +
  geom_line(aes(y = ..density..), size = 1, col = I('blue'), stat = 'density', ) +
  geom_vline(aes(xintercept=mean(extract(resStanFull)$beta[,6])), col=I('yellow'), linetype="dashed", si

plot_4 <- qplot(extract(resStanFull)$beta[,7], geom = 'blank',
                xlab = 'Values of weigth', ylab = 'Occurences', main='Education') +
  geom_histogram(aes(y = ..density..),col = I('red'), bins = 50) +
  geom_line(aes(y = ..density..), size = 1, col = I('blue'), stat = 'density', ) +
  geom_vline(aes(xintercept=mean(extract(resStanFull)$beta[,7])), col=I('yellow'), linetype="dashed", si
```

```
plot_5 <- qplot(extract(resStanFull)$beta[,8], geom = 'blank',
                xlab = 'Values of weigth', ylab = 'Occurences', main='Occupation') +
  geom_histogram(aes(y = ..density..),col = I('red'), bins = 50) +
  geom_line(aes(y = ..density..), size = 1, col = I('blue'), stat = 'density', ) +
  geom_vline(aes(xintercept=mean(extract(resStanFull)$beta[,8])), col=I('yellow'), linetype="dashed", si

ggplot2.multiplot(plot_1,plot_2,plot_3,plot_4, plot_5, cols=3)
```

From the analysis done above, and especially looking at the histogram, it is clear that the most important parameters that count in our analysis are: the fact that the people come from urban or rural areas, then their education, occupation and partially if they are man or woman. As a matter of fact, the mean and the maximum values of the coeffcient related to those paramters have the bigger magnitude. This means that those parameters are weighted more in the multi regression function in the model.

Therefore, for further analysis, it will be good to develop specific analysis using only these parameters, in order to have a more precise evalution considering only the most relevant parameters.

Frequentist approach

```
outcomeModel <- glm(as.numeric(Died) ~ as.numeric(Urban) +
                                       as.numeric(Year) +
                                       as.numeric(Season) +
                                       as.numeric(Sex) +
                                       as.numeric(Age) +
                                       as.numeric(Education) +
                                       as.numeric(Occupation) +
```

```
                                        as.numeric(method), data = mydata,
                    family = binomial(link = "logit"))
summary(outcomeModel)
```

```
##
## Call:
## glm(formula = as.numeric(Died) ~ as.numeric(Urban) + as.numeric(Year) +
##     as.numeric(Season) + as.numeric(Sex) + as.numeric(Age) +
##     as.numeric(Education) + as.numeric(Occupation) + as.numeric(method),
##     family = binomial(link = "logit"), data = mydata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3828  -0.8351   0.3501   0.8233   2.5409
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -0.239151   0.325168  -0.735   0.4621
## as.numeric(Urban)       0.296072   0.161150   1.837   0.0662 .
## as.numeric(Year)        0.291617   0.061971   4.706 2.53e-06 ***
## as.numeric(Season)      0.008641   0.044876   0.193   0.8473
## as.numeric(Sex)         0.424288   0.099000   4.286 1.82e-05 ***
## as.numeric(Age)         0.331736   0.052953   6.265 3.73e-10 ***
## as.numeric(Education)  -1.248306   0.080832 -15.443  < 2e-16 ***
## as.numeric(Occupation)  0.518808   0.131602   3.942 8.07e-05 ***
## as.numeric(method)     -0.051068   0.045090  -1.133   0.2574
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3344.4  on 2413  degrees of freedom
## Residual deviance: 2560.3  on 2405  degrees of freedom
## AIC: 2578.3
##
## Number of Fisher Scoring iterations: 4
```

Comparison

```
## Bayesian
print(resStanFull, pars = c("beta"))
```

```
## Inference for Stan model: 44efd1e4898e49d7c3da763fa46eaad0.
## 5 chains, each with iter=2000; warmup=800; thin=10;
## post-warmup draws per chain=120, total post-warmup draws=600.
##
##             mean se_mean      sd     2.5%      25%      50%      75%
## beta[1] -4117.96  992.19 2673.92 -9177.55 -6320.21 -3846.01 -1740.94
## beta[2]     2.86    0.18    1.11     0.83     2.00     2.77     3.61
## beta[3]     5.48    0.42    2.72     0.38     3.55     5.54     7.32
## beta[4]    -1.08    0.07    0.56    -2.12    -1.49    -1.07    -0.71
## beta[5]     2.55    0.20    1.46     0.04     1.56     2.53     3.36
## beta[6]     0.99    0.13    0.88    -0.56     0.34     0.93     1.59
## beta[7]    -4.71    0.31    1.59    -7.75    -5.96    -4.63    -3.54
```

```
## beta[8]  4119.42  993.26 2674.77   149.08  1739.38  3846.87  6327.03
## beta[9]    -0.61    0.12    0.53    -1.57    -0.97    -0.60    -0.26
##           97.5% n_eff Rhat
## beta[1] -154.91     7 2.22
## beta[2]    5.08    40 1.10
## beta[3]   10.94    43 1.08
## beta[4]    0.07    72 1.07
## beta[5]    5.93    53 1.06
## beta[6]    2.85    46 1.09
## beta[7]   -1.93    27 1.18
## beta[8] 9184.22     7 2.22
## beta[9]    0.47    21 1.16
##
## Samples were drawn using NUTS(diag_e) at Fri Dec 06 19:45:15 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).
```

```
## Frequentist
tableone::ShowRegTable(outcomeModel, exp = FALSE)
```

```
##                        coef [confint]       p
## (Intercept)            -0.24 [-0.88, 0.40]   0.462
## as.numeric(Urban)       0.30 [-0.02, 0.61]   0.066
## as.numeric(Year)        0.29 [0.17, 0.41]   <0.001
## as.numeric(Season)      0.01 [-0.08, 0.10]   0.847
## as.numeric(Sex)         0.42 [0.23, 0.62]   <0.001
## as.numeric(Age)         0.33 [0.23, 0.44]   <0.001
## as.numeric(Education)  -1.25 [-1.41, -1.09] <0.001
## as.numeric(Occupation)  0.52 [0.26, 0.78]   <0.001
## as.numeric(method)     -0.05 [-0.14, 0.04]   0.257
```

Some clustering on the data

Let us try to cluster the data using the specific year in order to do a prediction on the following year

```
indexYear2009 <- which(mydata$Year == 2009)
data_year_2009 <- mydata[indexYear2009,]

indexYear2010 <- which(mydata$Year == 2010)
data_year_2010 <- mydata[indexYear2010,]

indexYear2011 <- which(mydata$Year == 2011)
data_year_2011 <- mydata[indexYear2011,]
```

Full logistic regression model

For completeness we report here the full model once again.

```
## FULL LOGISTIC REGRESSION MODEL

## Load Stan Model
fileNameOne <- "./logistic_regression_model.stan"
stan_code_full <- readChar(fileNameOne, file.info(fileNameOne)$size)
cat(stan_code_full)
```

```
## data {
##   // Define variables in data
##   // Number of observations (an integer)
##   int<lower=0> N;
##
##   // Number of parameters
##   int<lower=0> p;
##
##   // Variables
##   int  died[N];
##   int<lower=0>  year[N];
##   int<lower=0>  urban[N];
##   int<lower=0>  season[N];
##   int<lower=0>  sex[N];
##   int<lower=0>  age[N];
##   int<lower=0>  edu[N];
##   int<lower=0>  job[N];
##   int<lower=0>  method[N];
## }
##
## parameters {
##   // Define parameters to estimate
##   real beta[p];
## }
##
## transformed parameters  {
##   // Probability trasformation from linear predictor
##   real<lower=0> odds[N];
##   real<lower=0, upper=1> prob[N];
##   for (i in 1:N) {
##     odds[i] = exp(beta[1] + beta[2]*year[i]   + beta[3]*urban[i] +
##                             beta[4]*season[i] + beta[5]*sex[i]   +
##                             beta[6]*age[i]    + beta[7]*edu[i]   +
##                             beta[8]*job[i]    + beta[9]*method[i] );
##     prob[i] = odds[i] / (odds[i] + 1);
##   }
## }
##
## model {
##   // Prior part of Bayesian inference (flat if unspecified)
##
##   // Likelihood part of Bayesian inference
##     died ~ bernoulli(prob);
## }
```

Reduced logistic regression model

We can now implement the reduced logistic regression model using the selected parameters.

```
## REDUCED LOGISTIC REGRESSION MODEL

## Load Stan Model
fileNameOneDef <- "./logistic_regression_model_def.stan"
stan_code_simple_def <- readChar(fileNameOneDef, file.info(fileNameOneDef)$size)
cat(stan_code_simple_def)
```

```
## data {
##   // Define variables in data
##   // Number of observations (an integer)
##   int<lower=0> N;
##
##   // Number of parameters
##   int<lower=0> p;
##
##   // Variables
##   int  died[N];
##   int<lower=0>  sex[N];
##   int<lower=0>  age[N];
##   int<lower=0>  edu[N];
## }
##
## parameters {
##   // Define parameters to estimate
##   real beta[p];
## }
##
## transformed parameters  {
##   // Probability trasformation from linear predictor
##   real<lower=0> odds[N];
##   real<lower=0, upper=1> prob[N];
##   for (i in 1:N) {
##     odds[i] = exp(beta[1] + beta[2]*sex[i]  + beta[3]*age[i] +
##                               beta[4]*edu[i] );
##     prob[i] = odds[i] / (odds[i] + 1);
##   }
## }
##
## model {
##   // Prior part of Bayesian inference (flat if unspecified)
##
##   // Likelihood part of Bayesian inference
##     died ~ bernoulli(prob);
## }
```

Stan Code Running

Now we are going to run the model on the full dataset.
First we define the Stan data to run the second (reduced) model.

```
## Create Stan data
dat_def <- list(N        = nrow(mydata),
                p        = 4,
                died     = as.numeric(mydata$Died),
                sex      = as.numeric(mydata$Sex),
                age      = as.numeric(mydata$Age),
                edu      = as.numeric(mydata$Education))
```

We first run the full model. The settings are the same as before except that now we are using the full dataset and default value for thin.

```
## Inference for Stan model: 44efd1e4898e49d7c3da763fa46eaad0.
## 5 chains, each with iter=2000; warmup=800; thin=1;
## post-warmup draws per chain=1200, total post-warmup draws=6000.
##
##          mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
## beta[1] -0.24    0.01 0.32 -0.89 -0.45 -0.23 -0.02  0.38  3099    1
## beta[2]  0.29    0.00 0.06  0.17  0.25  0.29  0.33  0.41  5156    1
## beta[3]  0.30    0.00 0.16 -0.02  0.19  0.30  0.41  0.61  5810    1
## beta[4]  0.01    0.00 0.05 -0.08 -0.02  0.01  0.04  0.10  5706    1
## beta[5]  0.43    0.00 0.10  0.22  0.36  0.43  0.49  0.62  5494    1
## beta[6]  0.33    0.00 0.05  0.23  0.30  0.33  0.37  0.44  3438    1
## beta[7] -1.26    0.00 0.08 -1.41 -1.31 -1.26 -1.20 -1.10  4174    1
## beta[8]  0.52    0.00 0.13  0.27  0.44  0.52  0.61  0.77  5065    1
## beta[9] -0.05    0.00 0.05 -0.14 -0.08 -0.05 -0.02  0.04  4599    1
##
## Samples were drawn using NUTS(diag_e) at Fri Dec 06 19:51:56 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).
```

Now we run the reduced order model..

```
## Inference for Stan model: 42f8f42f99ebe8f9576f152a7e8a0f9d.
## 5 chains, each with iter=2000; warmup=800; thin=1;
## post-warmup draws per chain=1200, total post-warmup draws=6000.
##
##          mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
## beta[1]  0.59       0 0.22  0.19  0.44  0.59  0.74  1.03  2161    1
## beta[2]  0.48       0 0.10  0.30  0.42  0.48  0.55  0.67  3360    1
## beta[3]  0.34       0 0.05  0.24  0.31  0.34  0.38  0.44  2480    1
## beta[4] -1.24       0 0.08 -1.40 -1.29 -1.24 -1.18 -1.09  2519    1
##
## Samples were drawn using NUTS(diag_e) at Fri Dec 06 19:56:17 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).
```

Convergence Analysis

In this section we are going to analyse the implemented models, both in terms of convergence (assessed using R-hat and HMC specific convergence diagnostic) and efficiency (by computing the Effective Sample Size).

R-hat

R-hat convergence diagnostic compares between- and within-chain estimates for model parameters and other univariate quantities of interest. If chains have not mixed well R-hat is larger than 1. In practical terms, it is good practice to use at least four chains and using the sample if R-hat is less than 1.05.
We can see from the result of `print(fit)` we have just displayed that all the Rhat values are equal to one for both the models and therefore we have convergence.

HMC

Here we compute convergence diagnostic specific to Hamiltonian Monte Carlo, and in particular divergences and tree depth.
The following code computes the diagnostic for the full model:

```
## Full model HMC diagnostic

check_hmc_diagnostics(resStanFull)
```

```
##
## Divergences:
## 0 of 6000 iterations ended with a divergence.
##
## Tree depth:
## 0 of 6000 iterations saturated the maximum tree depth of 10.
##
## Energy:
## E-BFMI indicated no pathological behavior.
```

As we can see none of the interations ended with a divergence nor saturated the maximum tree depth.
Now we compute the diagnostic for the reduced model:

```
## Reduced model HMC diagnostic

check_hmc_diagnostics(resStanRed)
```

```
##
## Divergences:
## 0 of 6000 iterations ended with a divergence.
##
## Tree depth:
## 0 of 6000 iterations saturated the maximum tree depth of 10.
##
## Energy:
## E-BFMI indicated no pathological behavior.
```

Also for the reduced model none of the iterations ended with a divergence nor saturated the maximum tree depth.

ESS

Effective sample size (ESS) measures the amount by which autocorrelation within the chains increases uncertainty in estimates.
As for the Rhat values we can directly observe the effective sample size values of the chains using the command `print(fit)`, already used. We can see that the sample size values are all sufficiently high for both model.

Posterior Predictive Checking

Model Comparison

```
datFile <- "suicide attempt data_2.csv"
datCsv <- read.csv(datFile, stringsAsFactors=FALSE)
datSet <- as.data.frame(datCsv)

datSet$Season <- datSet$Month
datSet$Month = NULL
```

```r
## Remove unknown labels

indexUnkn_1 <- which(datSet$Education == 'unknown')
indexUnkn_2 <- which(datSet$Urban == 'unknown')
indexUnkn_3 <- which(datSet$Occupation == 'others/unknown')
datSet <- datSet[-c(indexUnkn_1, indexUnkn_2,indexUnkn_3),]

datSet$Month = NULL

# Hospitalised
indexHosp   <- which(datSet$Hospitalised == 'yes')
indexNoHosp <- which(datSet$Hospitalised == 'no')
datSet$Hospitalised[indexHosp]   <- 1    # 1 --> yes
datSet$Hospitalised[indexNoHosp] <- 0    # 0 --> no


# Died
indexDied   <- which(datSet$Died == 'yes')
indexNoDied <- which(datSet$Died == 'no')
datSet$Died[indexDied]   <- 1    # 1 --> yes
datSet$Died[indexNoDied] <- 0    # 0 --> no


# Urban
indexUrban   <- which(datSet$Urban == 'yes')
indexNoUrban <- which(datSet$Urban == 'no')
datSet$Urban[indexUrban]   <- 1    # 1 --> yes
datSet$Urban[indexNoUrban] <- 0    # 0 --> no

#Year
indexYear2009 <- which(datSet$Year == 2009)
indexYear2010 <- which(datSet$Year == 2010)
indexYear2011 <- which(datSet$Year == 2011)
datSet$Year[indexYear2009] <- 1    # 1 --> 2009
datSet$Year[indexYear2010] <- 2    # 2 --> 2010
datSet$Year[indexYear2011] <- 3    # 3 --> 2011

# Sex
indexMale   <- which(datSet$Sex == 'male')
indexFemale <- which(datSet$Sex == 'female')
datSet$Sex[indexMale]   <- 1    # 1 --> male
datSet$Sex[indexFemale] <- 0    # 0 --> female

# Education
indexEduZero  <- which(datSet$Education == 'iliterate')
indexEduOne   <- which(datSet$Education == 'primary')
indexEduTwo   <- which(datSet$Education == 'Secondary')
indexEduThree <- which(datSet$Education == 'Tertiary')

datSet$Education[indexEduZero]   <- 0   # 0 --> iliterate
datSet$Education[indexEduOne]    <- 1   # 1 --> primary
datSet$Education[indexEduTwo]    <- 2   # 2 --> Secondary
datSet$Education[indexEduThree]  <- 3   # 3 --> Tertiary
```

```r
# Occupation
indexUnEmpl <- which(datSet$Occupation == 'unemployed')
indexFarm   <- which(datSet$Occupation == 'farming')
indexProf   <- which(datSet$Occupation == 'business/service' | datSet$Occupation == 'professional')

datSet$Occupation[indexUnEmpl]   <- 0    # 1 --> farming
datSet$Occupation[indexFarm]     <- 1    # 0 --> non farming
datSet$Occupation[indexProf]     <- 2
datSet$Occupation[-c(indexUnEmpl, indexFarm, indexProf)]    <- 3

# Method
indexPesticide <- which(datSet$method == 'Pesticide')
indexPoison    <- which(datSet$method == 'Other poison')
indexHanging   <- which(datSet$method == 'hanging')
indexOthers    <- which(datSet$method != 'Pesticide' &
                        datSet$method != 'Other poison' &
                        datSet$method != 'hanging')

datSet$method[indexPesticide] <- 1 # 1 --> Pesticide
datSet$method[indexPoison]    <- 2 # 2 --> Other poison
datSet$method[indexHanging]   <- 3 # 3 --> hanging
datSet$method[indexOthers]    <- 4 # 4 --> All others

# Season
indexSpring <- which(datSet$Season >= 3 & datSet$Season <= 5)
indexSummer <- which(datSet$Season >= 6 & datSet$Season <= 8)
indexAutumn <- which(datSet$Season >= 9 & datSet$Season <= 11)
indexWinter <- which(datSet$Season == 12 | datSet$Season <= 2)

datSet$Season[indexSpring] <- 1  # 1 --> Spring
datSet$Season[indexSummer] <- 2  # 2 --> Summer
datSet$Season[indexAutumn] <- 3  # 3 --> Autumn
datSet$Season[indexWinter] <- 4  # 4 --> Winter


# qplot(datSet$Age, geom = 'blank', xlab = 'Values of weigth', ylab = 'Occurences', main='Urbans') +
#   geom_histogram(aes(y = ..density..),col = I('red'), bins = 50) +
#   geom_line(aes(y = ..density..), size = 1, col = I('blue'), stat = 'density', ) +
#   geom_vline(aes(xintercept=mean(datSet$Age)), col=I('yellow'), linetype="dashed", size=1)
```

```r
data_4_fit <- list(N = nrow(datSet),
                   p = 6,
                   age = as.numeric(datSet$Age),
                   died = as.numeric(datSet$Died),
                   sex = as.numeric(datSet$Sex),
                   job = as.numeric(datSet$Occupation),
                   urban = as.numeric(datSet$Urban),
                   edu = as.numeric(datSet$Education))


fileName <- "./stan_model_with_prior.stan"
stan_code <- readChar(fileName, file.info(fileName)$size)
```

```
cat(stan_code)
```

```
## //
## // This Stan program defines a simple model, with a
## // vector of values 'y' modeled as normally distributed
## // with mean 'mu' and standard deviation 'sigma'.
## //
## // Learn more about model development with Stan at:
## //
## //    http://mc-stan.org/users/interfaces/rstan.html
## //    https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started
## //
##
##
## data {
##   // Define variables in data
##   // Number of observations (an integer)
##   int<lower=0> N;
##
##   // Number of parameters
##   int<lower=0> p;
##
##   // Variables
##   real<lower=0> age[N];
##   int<lower=0>  died[N];
##   int<lower=0>  sex[N];
##   int<lower=0>  job[N];
##   int<lower=0>  urban[N];
##   int<lower=0>  edu[N];
## }
##
## parameters {
##   // A-priori mean for prediction
##   real mu_p;
##   // Define parameters to estimate
##   real beta[p];
##
##   // standard deviation (a positve real number)
##   real<lower=0> sigma;
## }
##
## transformed parameters  {
##   // Mean
##   real mu[N];
##   for (i in 1:N) {
##     mu[i] = beta[1] + beta[2]*died[i] + beta[3]*sex[i] + beta[4]*job[i] +
##             beta[5]*urban[i] + beta[6]*edu[i];
##   }
## }
##
## model {
##   // Weakly informative prior
##   mu_p ~ normal(60, 10);
##   sigma ~ normal(0,10);
```

```
##
##   // Likelihood part of the Bayesian inference
##   age ~ normal(mu, sigma);
##
## }
##
## generated quantities{
##   real predct_age;
##   vector[N] log_lik;
##
##   predct_age = normal_rng(mu_p, sigma);
##   for (i in 1:N)
##     log_lik[i] = normal_lpdf(age[i] | mu[i], sigma);
##
## }
```

```
# Run Stan
fitStan <- stan(model_code = stan_code,
                data = data_4_fit,
                chains = 5,
                iter = 2000,
                warmup = 800,
                thin = 10,
                refresh = 0,
                seed = 12345,
                control = list(adapt_delta = 0.95))
```

```
## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess
```

```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

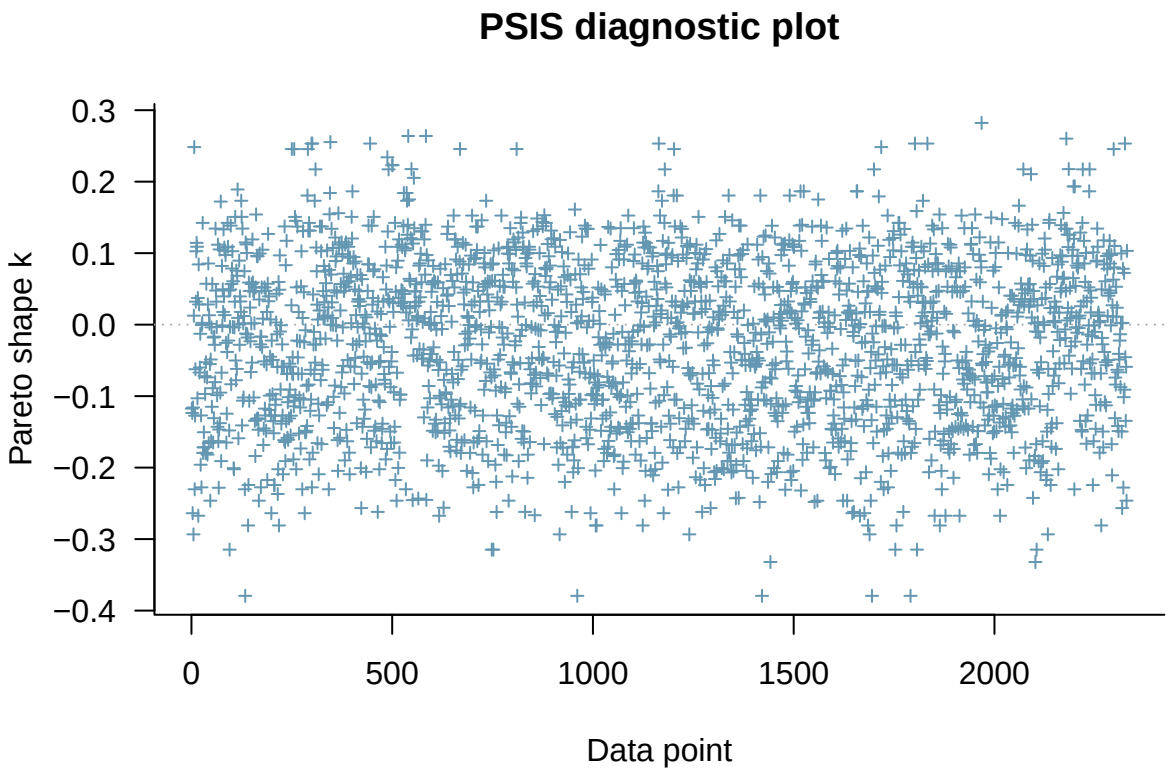
```
print(fitStan, pars = c('beta', 'sigma'))
```

```
## Inference for Stan model: 27c2703215c82db517f6e77613d1aace.
## 5 chains, each with iter=2000; warmup=800; thin=10;
## post-warmup draws per chain=120, total post-warmup draws=600.
##
##           mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## beta[1]  67.77    0.05 1.11  65.66  67.05  67.78  68.59  69.79   580 1.01
## beta[2]   5.78    0.03 0.73   4.30   5.31   5.80   6.22   7.25   672 1.00
## beta[3]   1.86    0.03 0.63   0.71   1.43   1.85   2.29   3.11   562 1.00
## beta[4]  -0.38    0.02 0.47  -1.29  -0.72  -0.37  -0.08   0.53   682 1.00
## beta[5]  -1.01    0.04 1.04  -2.99  -1.72  -1.04  -0.30   1.19   694 0.99
## beta[6] -13.82    0.02 0.46 -14.72 -14.12 -13.83 -13.50 -12.87   602 1.00
## sigma    14.82    0.01 0.21  14.43  14.69  14.81  14.97  15.24   489 1.00
##
## Samples were drawn using NUTS(diag_e) at Fri Dec 06 19:59:50 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).
```

```
log_like_model <- extract_log_lik(fitStan, merge_chains = FALSE)
r_eff <- relative_eff(exp(log_like_model))
loo_mod <- loo(log_like_model, r_eff = r_eff)
print(loo_mod)
```

```
##
## Computed from 600 by 2330 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -9591.2 34.3
## p_loo         7.3  0.3
## looic     19182.3 68.6
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
plot(loo_mod)
```




```
elpd_loo_mod <- loo_mod$estimates[1]
elpd_loo_mod
```

```
## [1] -9591.165
```

```
data_4_fit_complete <- list(N = nrow(datSet),
                  p = 10,
```

```
                    age = as.numeric(datSet$Age),
                    died = as.numeric(datSet$Died),
                    hosp = as.numeric(datSet$Hospitalised),
                    year = as.numeric(datSet$Year),
                    sex = as.numeric(datSet$Sex),
                    job = as.numeric(datSet$Occupation),
                    urban = as.numeric(datSet$Urban),
                    edu = as.numeric(datSet$Education),
                    method = as.numeric(datSet$method),
                    season = as.numeric(datSet$Season))


fileName <- "./stan_model_prior_all_params.stan"
stan_code_complete <- readChar(fileName, file.info(fileName)$size)
cat(stan_code_complete)
```

```
## // This Stan program defines a simple model, with a
## // vector of values 'y' modeled as normally distributed
## // with mean 'mu' and standard deviation 'sigma'.
## //
## // Learn more about model development with Stan at:
## //
## //    http://mc-stan.org/users/interfaces/rstan.html
## //    https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started
## //
##
##
## data {
##   // Define variables in data
##   // Number of observations (an integer)
##   int<lower=0> N;
##
##   // Number of parameters
##   int<lower=0> p;
##
##   // Variables
##   real<lower=0> age[N];
##   int<lower=0>  died[N];
##   int<lower=0>  hosp[N];
##   int<lower=0>  year[N];
##   int<lower=0>  sex[N];
##   int<lower=0>  job[N];
##   int<lower=0>  urban[N];
##   int<lower=0>  edu[N];
##   int<lower=0>  method[N];
##   int<lower=0>  season[N];
## }
##
## parameters {
##   // Define parameters to estimate
##   real beta[p];
##
##   // standard deviation (a positve real number)
##   real<lower=0> sigma;
```

```
## }
##
## transformed parameters  {
##   // Mean
##   real mu[N];
##   for (i in 1:N) {
##     mu[i] = beta[1] + beta[2]*died[i] + beta[3]*hosp[i] + beta[4]*year[i] +
##             beta[5]*sex[i] + beta[6]*job[i] + beta[7]*urban[i] + beta[8]*edu[i] +
##             beta[9]*method[i] + beta[10]*season[i];
##   }
## }
##
## model {
##   // Weakly informative prior
##   //mu ~ normal(0.5, 10);
##   sigma ~ normal(0,10);
##
##   // Likelihood part of the Bayesian inference
##   age ~ normal(mu, sigma);
##
## }
##
## generated quantities{
##   vector[N] log_lik;
##
##   for (i in 1:N)
##     log_lik[i] = normal_lpdf(age[i] | mu[i], sigma);
##
## }
```

```
# Run Stan
fitStan_complete <- stan(model_code = stan_code_complete,
                data = data_4_fit_complete,
                chains = 5,
                iter = 2000,
                warmup = 800,
                thin = 10,
                refresh = 0,
                seed = 12345,
                control = list(adapt_delta = 0.95))
```

```
## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess
```

```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quanti
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

```
print(fitStan_complete, pars = c('beta', 'sigma'))
```

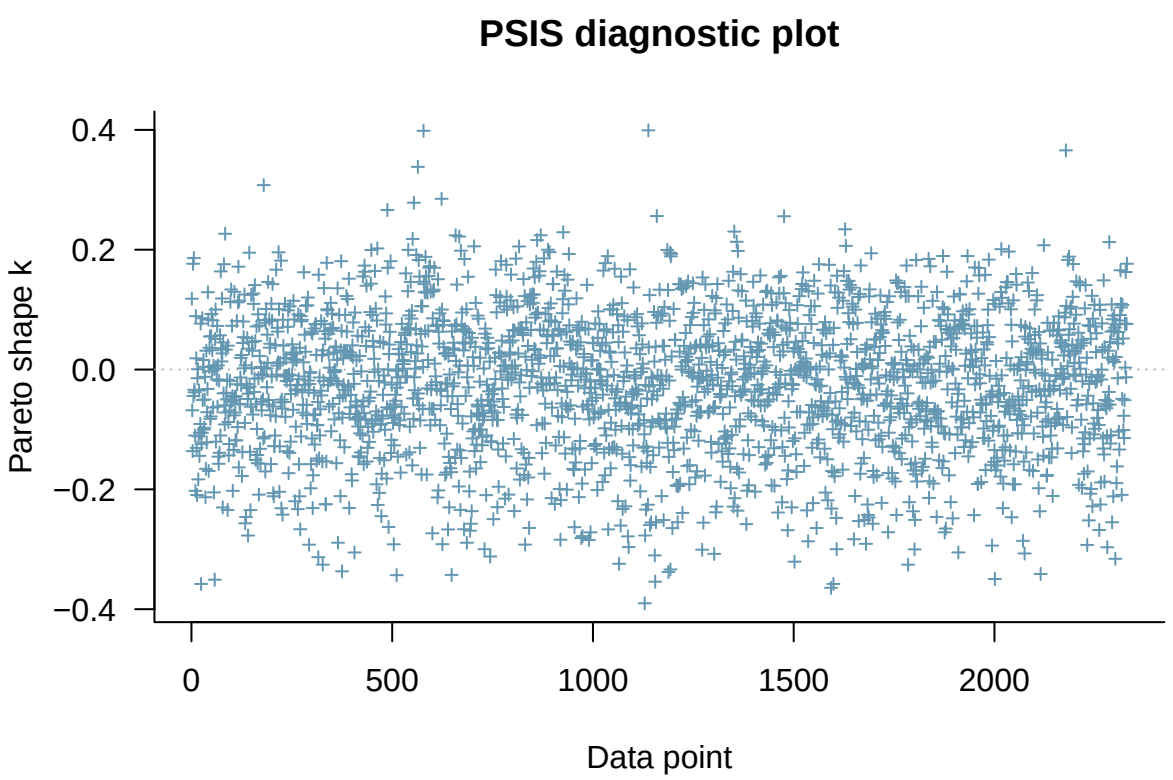
```
## Inference for Stan model: 09e3182dff98db3733ab71c0903a7bef.
## 5 chains, each with iter=2000; warmup=800; thin=10;
## post-warmup draws per chain=120, total post-warmup draws=600.
##
##            mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
```

```
## beta[1]   78.68    0.08 2.07  74.63  77.24  78.77  80.10  82.72   645 0.99
## beta[2]    3.28    0.05 1.13   1.06   2.50   3.31   4.09   5.48   619 0.99
## beta[3]   -2.92    0.04 1.07  -5.03  -3.66  -2.87  -2.17  -0.90   613 1.00
## beta[4]    0.18    0.02 0.38  -0.52  -0.07   0.20   0.41   0.98   508 1.00
## beta[5]    1.34    0.03 0.68  -0.01   0.86   1.32   1.79   2.60   556 1.00
## beta[6]   -0.92    0.02 0.46  -1.81  -1.22  -0.91  -0.62  -0.03   743 1.00
## beta[7]   -0.62    0.05 1.01  -2.70  -1.28  -0.60   0.09   1.24   482 0.99
## beta[8]  -13.36    0.02 0.42 -14.23 -13.62 -13.35 -13.09 -12.54   614 1.00
## beta[9]   -2.07    0.01 0.28  -2.64  -2.24  -2.07  -1.90  -1.49   628 1.00
## beta[10]  -0.52    0.01 0.29  -1.11  -0.71  -0.50  -0.34   0.06   624 1.00
## sigma     14.64    0.01 0.22  14.25  14.49  14.64  14.78  15.07   604 1.00
##
## Samples were drawn using NUTS(diag_e) at Fri Dec 06 20:05:09 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).
```

```
log_like_model_compl <- extract_log_lik(fitStan_complete, merge_chains = FALSE)
r_eff <- relative_eff(exp(log_like_model_compl))
loo_mod_comp <- loo(log_like_model_compl, r_eff = r_eff)
print(loo_mod_comp)
```

```
##
## Computed from 600 by 2330 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -9564.4 34.6
## p_loo        11.5  0.4
## looic     19128.7 69.1
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
plot(loo_mod_comp)
```




```
elpd_loo_mod_comp <- loo_mod_comp$estimates[1]
elpd_loo_mod_comp
```

```
## [1] -9564.371
```

Sensitivity Analysis

Conclusions

Problems encountered

Potential improvements