bayesAssgnment 5

2025-05-04

# Data

data<- readxl::read\_xlsx("mydata.xlsx")

# 1.

## plot

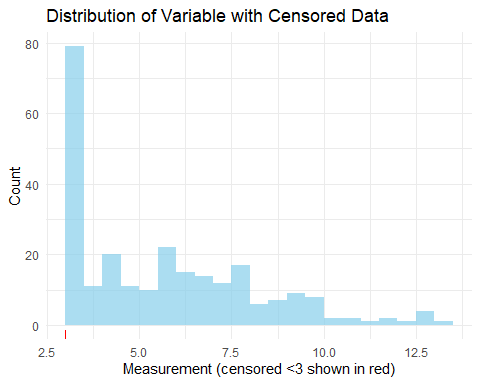
# Suppose your variable is in a column called "var"  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

g<-data.frame(data$Glucose)  
df <-g %>%  
 mutate(  
 censored = grepl("^<", data.Glucose), # TRUE if value starts with <  
 var\_numeric = as.numeric(gsub("^<", "", data.Glucose)) # remove < and convert to numeric  
 )  
library(ggplot2)  
  
ggplot(df, aes(x = var\_numeric)) +  
 geom\_histogram(binwidth = 0.5, fill = "skyblue", alpha = 0.7, boundary = 0, closed = "left") +  
 geom\_rug(data = df %>% filter(censored), aes(x = var\_numeric), color = "red") +  
 labs(title = "Distribution of Variable with Censored Data",  
 x = "Measurement (censored <3 shown in red)",  
 y = "Count") +  
 theme\_minimal()



The data has column for Glucose which is made of non-fasting blood glucose measurements that is the amount of glucose in your blood after eating. The Glucose column has continuous data points but also includes values less than 3 as <3 and looking at the histogram the data is skewed to the right.

The data also has explanatory variables one of them as previous glucose level taken. which has same measure as scale as dependent variable. School and urban columns are categorical with missing values, and Age and BMI are continuous with missing values also.

# 2. Imputation.

library(mice)

##   
## Attaching package: 'mice'

## The following object is masked from 'package:stats':  
##   
## filter

## The following objects are masked from 'package:base':  
##   
## cbind, rbind

data\_impute<- data[,c(-2,-3)]  
data\_impute$Urban <- as.factor(data\_impute$Urban)

imputed\_data <- mice(data\_impute, m=10, method="pmm", maxit=50, seed=500)

## Warning: Number of logged events: 1

completed\_data\_1 <- complete(imputed\_data, 1)

# 3. Between imputation standard deviation.

mean\_vec <- numeric(10)  
  
for (i in 1:10) {  
 completed\_data <- complete(imputed\_data, i)  
 mean\_vec[i] <- mean(completed\_data$BMI, na.rm = TRUE)  
}  
mu<- mean(mean\_vec)  
btw\_std<- sqrt(var(mean\_vec))

# 4.

# data

completed\_data\_1 <- complete(imputed\_data, 1)   
completed\_data\_1$Glucose <- data$Glucose  
as.numeric(completed\_data\_1$Glucose[completed\_data\_1$Glucose == "<3"] <- 2.9)

## [1] 2.9

N<- length(completed\_data\_1$Glucose)

# FIT

stan\_code <- "  
data {  
 int<lower=0> N;  
 int<lower=1> K;  
 matrix[N, K] X;  
 vector[N] y;  
}  
parameters {  
 vector[K] beta;  
 real<lower=0> sigma;  
}  
model {  
 target += -log(sigma);  
 for (n in 1:N) {  
 target += -log(2 \* sigma) - fabs(y[n] - X[n] \* beta) / sigma;  
 }  
}  
"

library(rstan)

## Loading required package: StanHeaders

##   
## rstan version 2.32.7 (Stan version 2.32.2)

## For execution on a local, multicore CPU with excess RAM we recommend calling  
## options(mc.cores = parallel::detectCores()).  
## To avoid recompilation of unchanged Stan programs, we recommend calling  
## rstan\_options(auto\_write = TRUE)  
## For within-chain threading using `reduce\_sum()` or `map\_rect()` Stan functions,  
## change `threads\_per\_chain` option:  
## rstan\_options(threads\_per\_chain = 1)

## Do not specify '-march=native' in 'LOCAL\_CPPFLAGS' or a Makevars file

X <- model.matrix(Glucose~ BMI + Age + Urban, data = completed\_data\_1)  
y <-as.numeric(completed\_data\_1$Glucose)  
data\_list <- list(N = nrow(X), K = ncol(X), X = X, y = y)

print(fit)

## Inference for Stan model: anon\_model.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 2.5% 25% 50% 75% 97.5% n\_eff Rhat  
## beta[1] -3.56 0.01 0.50 -4.58 -3.88 -3.55 -3.23 -2.61 1870 1  
## beta[2] 0.26 0.00 0.02 0.23 0.25 0.26 0.27 0.30 2106 1  
## beta[3] 0.03 0.00 0.01 0.02 0.03 0.03 0.04 0.04 3014 1  
## beta[4] 0.67 0.00 0.24 0.22 0.50 0.66 0.83 1.15 2734 1  
## sigma 1.49 0.00 0.09 1.32 1.42 1.48 1.55 1.68 2757 1  
## lp\_\_ -530.18 0.04 1.65 -534.12 -531.07 -529.87 -528.96 -527.94 1404 1  
##   
## Samples were drawn using NUTS(diag\_e) at Mon May 12 11:46:29 2025.  
## 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).

# 5.

stan<-"  
 data {  
 int<lower=0> N;   
 int<lower=1> K;   
 matrix[N, K] X;   
 vector[N] y;   
 int<lower=0,upper=1> is\_censored[N]; # 1 = censored, 0 = observed  
}  
parameters  
{  
 vector[K] beta;  
 real<lower=0> sigma;  
}  
  
model  
{  
 target += -log(sigma); # // Prior on sigma: log π(σ) ∝ -log(σ)  
  
 for (n in 1:N) {  
 real mu = X[n] \* beta;  
 if (is\_censored[n] == 1) {  
 // Fully observed  
 target += -log(2 \* sigma) - fabs(y[n] - mu) / sigma;  
 } else {  
 // Left-censored: y[n]   
 target += log1m\_exp(-fabs(y[n] - mu) / sigma);  
 }  
 }  
}  
"

censoring\_point <- 3  
is\_censored <- ifelse(completed\_data\_1$Glucose < censoring\_point, 1, 0)  
y\_observed <- pmin(completed\_data\_1$Glucose, censoring\_point)  
  
data\_list1 <- list(  
 N = length(y\_observed),  
 K = ncol(X),  
 X = X,  
 y = as.numeric(y\_observed),  
 is\_censored = is\_censored  
)

fit1 <- stan(model\_code = stan, data = data\_list1,   
 iter = 2000, chains = 4, seed = 123)

## Warning: There were 128 divergent transitions after warmup. See  
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup  
## to find out why this is a problem and how to eliminate them.

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

## Warning: The largest R-hat is 2.25, indicating chains have not mixed.  
## Running the chains for more iterations may help. See  
## https://mc-stan.org/misc/warnings.html#r-hat

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

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

print(fit1)

## Inference for Stan model: anon\_model.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 2.5% 25% 50% 75% 97.5% n\_eff  
## beta[1] -2.91 3.68 5.26 -7.37 -6.31 -5.60 -1.10 7.03 2  
## beta[2] 0.27 0.17 0.25 -0.20 0.19 0.38 0.43 0.50 2  
## beta[3] 0.01 0.02 0.02 -0.03 -0.01 0.02 0.03 0.05 2  
## beta[4] 0.76 0.43 0.70 -0.59 0.34 0.84 1.28 1.95 3  
## sigma 1.36 0.11 0.20 1.05 1.21 1.33 1.47 1.83 4  
## lp\_\_ -230.66 23.12 32.76 -289.47 -237.88 -212.39 -210.47 -208.18 2  
## Rhat  
## beta[1] 7.39  
## beta[2] 8.65  
## beta[3] 3.37  
## beta[4] 2.39  
## sigma 1.50  
## lp\_\_ 19.11  
##   
## Samples were drawn using NUTS(diag\_e) at Mon May 12 11:51:02 2025.  
## 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).

## probability that each coefficient has changed by more than 2%.

percent\_change <- (as.matrix(fit1)- as.matrix(fit) / as.matrix(fit)) \* 100  
prob\_gt\_2 <- apply(abs(percent\_change) > 2, 2, mean)  
print(prob\_gt\_2)

## beta[1] beta[2] beta[3] beta[4] sigma lp\_\_   
## 1.00000 1.00000 1.00000 0.97000 0.99075 1.00000

# 6.

completed\_datasets <- lapply(1:10, function(i) complete(imputed\_data, i))  
  
completed\_datasets <- lapply(completed\_datasets, function(df) {  
 df$Glucose <- completed\_data\_1$Glucose  
 return(df)  
})  
  
results\_list <- list()  
  
for (i in 1:10) {  
 df <- completed\_datasets[[i]]  
  
  
 X <- model.matrix(Glucose~ BMI + Age + Urban, data = df)   
  
  
 c\_point <- 3  
 is\_censored <- ifelse(df$Glucose < c\_point, 1, 0)  
 y\_obs <- ifelse(is\_censored == 1, c\_point, df$Glucose)  
  
 data\_list <- list(  
 N = nrow(X),  
 K = ncol(X),  
 X = X,  
 y = y\_obs,  
 is\_censored = is\_censored  
 )  
  
  
 fit <- stan(model\_code = stan , data = list(  
 N = length(y\_observed),  
 K = ncol(X),  
 X = X,  
 y = as.numeric(y\_observed),  
 is\_censored = is\_censored  
),  
 iter = 1000, chains = 2, seed = 123 + i,  
 refresh = 0)  
  
 # Store parameter simulations  
 results\_list[[i]] <- as.data.frame(rstan::extract(fit, pars = c("beta", "sigma")))  
}

# 7.

all\_draws <- do.call(rbind, results\_list)  
library(tidyr)

##   
## Attaching package: 'tidyr'

## The following object is masked from 'package:rstan':  
##   
## extract

library(dplyr)  
  
summary\_table <- all\_draws %>%  
 summarise(across(everything(), list(  
 mean = ~mean(.),  
 lower\_95 = ~quantile(., 0.025),  
 upper\_95 = ~quantile(., 0.975)  
 ))) %>%  
 pivot\_longer(cols = everything(),  
 names\_to = c("parameter", "stat"),  
 names\_sep = "\_") %>%  
 pivot\_wider(names\_from = stat, values\_from = value)  
print(summary\_table)

## # A tibble: 5 × 4  
## parameter mean lower upper  
## <chr> <dbl> <dbl> <dbl>  
## 1 beta.1 -3.72 -8.20 5.67   
## 2 beta.2 0.201 -0.245 0.496  
## 3 beta.3 0.0656 -0.0444 0.182  
## 4 beta.4 1.20 -0.320 2.87   
## 5 sigma 1.36 0.980 1.87

# 8.

The estimate of beta 1 is the intercept and it is the median Glucose level when we have other explanatory variables equal to zero, meaning if we do not know the person BMI, Age and we just know is from the rural then that person will have Glucose level of 2.92. The person will observe 0.0014394765 increase in the blood glucose level if their BMI increases by a unit and from the rural, and also age has not changed, if a persons age increases by a unit then we expect the median Glucose to increase by 0.0003382594 and the BMI has not changed and the person is from the rural. if you are from the Urban the median Glucose level is 0.015133721 higher given your same age and have same BMI as the person from the rural.

# 8.

data\_2p<- data[c(13,14),]

Subject 1 is a 32 years old and his previous Glucose is the same as the current one this might mean the subject does not do follow ups this might be because of the missing value in urban column he might have done this tests at a place he does not stay at. subject 2 is 61 with BMI missing it could be due to that they do not know it or it was not recorded by the data handler.

we going to increase both their ages with 3 years and also put subject 1 in rural and make subject 1 BMI 20.

data\_2p[1,"Urban"]<- "rural"  
data\_2p[c(1,2), "Age"]<- c(35,64)  
data\_2p[2,"BMI"]<- 20

## posterior predictive dist

X\_new <- model.matrix(~ BMI + Age + Urban, data = data\_2p)  
  
posterior <- rstan::extract(fit)  
beta\_draws <- posterior$beta  
sigma\_draws <- posterior$sigma  
n\_draws <- length(sigma\_draws)  
  
  
y\_pred\_sub1 <- numeric(n\_draws)  
y\_pred\_sub2 <- numeric(n\_draws)  
  
r\_laplace <- function(mu, b) {  
 u <- runif(length(mu), -0.5, 0.5)  
 mu - b \* sign(u) \* log(1 - 2 \* abs(u))  
}  
# Linear predictors for all draws (matrix multiplication)  
mu\_pred <- X\_new %\*% t(beta\_draws) # result: 2 x n\_draws  
  
# Generate predictions  
sub1\_pred <- r\_laplace(mu = mu\_pred[1, ], b = sigma\_draws)  
sub2\_pred <- r\_laplace(mu = mu\_pred[2, ], b = sigma\_draws)  
  
pred\_sub1=quantile(sub1\_pred, c(0.025, 0.5, 0.975))  
sub1<- pred\_sub1["50%"]  
pred\_sub2=quantile(sub2\_pred, c(0.025, 0.5, 0.975))  
sub2<- pred\_sub2["50%"]  
print(sub1)

## 50%   
## 8.113088

print(sub2)

## 50%   
## 4.905217

The predictions given by the median seem to be good predictions especial1y for the subject 2 4.4 is not that for from observed Glucose and also not that far from previous observed glucose but given that this subject has been keeping in shape.