STATS 205P HW4

Chuqi Wang 79167724

2024-05-30

Q1:

```
getwd()
```

```
## [1] "/Users/chuqiwang/Desktop/UCI/STATS205P/hw4"
```

```
pima_data = read.csv("pima.csv", header = TRUE)
head(pima_data)
```

```
##   npreg glu bp skin insulin  bmi   ped age diabetic
## 1     6 148 72   35       0 33.6 0.627  50        1
## 2     1  85 66   29       0 26.6 0.351  31        0
## 3     8 183 64    0       0 23.3 0.672  32        1
## 4     1  89 66   23      94 28.1 0.167  21        0
## 5     0 137 40   35     168 43.1 2.288  33        1
## 6     5 116 74    0       0 25.6 0.201  30        0
```

Given that the outcome variable diabetic is the response and covariates are bp, bmi and age. Suppose y_i is binary outcome variable (0 or 1, indicating whether a person is diabetic). Then y_i follows bernoulli distribution, our generalized linear model will have logit link function. Then the likelihood of our model is given by:

$$y_i \sim Bernoulli(p_i)$$

$$p_i = \frac{exp(\alpha + X_i\beta)}{1 + exp(\alpha + X_i\beta)}$$

The priors are given by:

$$\alpha \sim N(0, 1000)$$

$$\beta_j \sim N(0, 1000) \; for \; j = 1, 2, 3$$

The logistic regression model is given by:

$$g(p_i) = log(\frac{p_i}{1 - p_i}) = \alpha + X_i\beta = \alpha + bp_i \cdot \beta_1 + bmi_i \cdot \beta_2 + age_i \cdot \beta_3$$

```
library(rstan)
```

```
## Loading required package: StanHeaders
```

```
##
## rstan version 2.32.6 (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)
```

```
library(ggplot2)
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
```

```
pima_data <- pima_data %>%
  select(diabetic, bp, bmi, age)

stan_data1 <- list(
  N = nrow(pima_data),
  y = pima_data$diabetic,
  X = as.matrix(pima_data[, c("bp", "bmi", "age")])
)
```

```
# Define the Stan model
stan_model1 <- "
data {
  int<lower=0> N;          // number of observations
  int<lower=0, upper=1> y[N]; // binary outcome variable
  matrix[N, 3] X;          // matrix of predictors
}
parameters {
  vector[3] beta;          // coefficients for predictors
  real alpha;              // intercept
}
model {
  // Priors
  alpha ~ normal(0, 10000);
  beta ~ normal(0, 10000);

  // Likelihood
  y ~ bernoulli_logit(alpha + X * beta);
}
"
```

```
fit1 = stan(model_code = stan_model1, data = stan_data1)
```

```
## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 14.0.3 (clang-1403.0.22.14.1)'
## using SDK: 'MacOSX13.3.sdk'
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
```

```
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Cor
## #include <cmath>
##          ^~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 5.5e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.55 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.645 seconds (Warm-up)
## Chain 1:                0.674 seconds (Sampling)
## Chain 1:                1.319 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.9e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.634 seconds (Warm-up)
```

```
## Chain 2:                0.596 seconds (Sampling)
## Chain 2:                1.23 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.9e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.821 seconds (Warm-up)
## Chain 3:                0.569 seconds (Sampling)
## Chain 3:                1.39 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.9e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.697 seconds (Warm-up)
## Chain 4:                0.568 seconds (Sampling)
## Chain 4:                1.265 seconds (Total)
## Chain 4:
```

```
summary(fit1)
```

```
## $summary
##                  mean      se_mean          sd          2.5%           25%
## beta[1] -9.231391e-03 9.376003e-05 0.004719403   -0.01856588   -0.01229553
## beta[2]  1.048025e-01 3.326832e-04 0.013026828    0.08023847    0.09558495
## beta[3]  4.947429e-02 1.784174e-04 0.007438466    0.03510093    0.04427646
## alpha   -5.108711e+00 1.480871e-02 0.543594829   -6.20788428   -5.47119203
## lp__    -4.379755e+02 3.847528e-02 1.451130876 -441.71850612 -438.66740360
##                  50%           75%         97.5%    n_eff     Rhat
## beta[1] -9.267343e-03 -6.061644e-03  7.498573e-05 2533.604 1.000196
## beta[2]  1.047383e-01  1.136199e-01  1.310212e-01 1533.259 1.001981
## beta[3]  4.932173e-02  5.451040e-02  6.449355e-02 1738.170 1.001069
## alpha   -5.105804e+00 -4.725142e+00 -4.079785e+00 1347.461 1.003693
## lp__    -4.376689e+02 -4.369064e+02 -4.361827e+02 1422.491 1.001661
##
## $c_summary
## , , chains = chain:1
##
##          stats
## parameter          mean          sd          2.5%           25%           50%
##   beta[1] -9.121069e-03 0.004543793   -0.01782187   -0.01208467 -9.391366e-03
##   beta[2]  1.056891e-01 0.013751414    0.07933014    0.09568020  1.053691e-01
##   beta[3]  4.968897e-02 0.007531413    0.03420072    0.04448360  4.924134e-02
##   alpha   -5.153907e+00 0.551356821   -6.27620044   -5.51395458 -5.127181e+00
##   lp__    -4.380206e+02 1.479299256 -442.04773303 -438.68129448 -4.376365e+02
##          stats
## parameter           75%         97.5%
##   beta[1] -6.059138e-03  4.349926e-05
##   beta[2]  1.147956e-01  1.326549e-01
##   beta[3]  5.476377e-02  6.597271e-02
##   alpha   -4.768412e+00 -4.121189e+00
##   lp__    -4.369560e+02 -4.362016e+02
##
## , , chains = chain:2
##
##          stats
## parameter          mean          sd          2.5%           25%           50%
##   beta[1] -9.346486e-03 0.004490009   -0.01761321   -0.01257365 -9.373216e-03
##   beta[2]  1.038189e-01 0.012611153    0.07933706    0.09557818  1.038082e-01
##   beta[3]  4.939534e-02 0.007560977    0.03423261    0.04441496  4.943338e-02
##   alpha   -5.063819e+00 0.526650683   -6.14529432   -5.40513704 -5.061912e+00
##   lp__    -4.379327e+02 1.425967800 -441.46953777 -438.63506176 -4.376879e+02
##          stats
## parameter           75%         97.5%
##   beta[1] -6.304468e-03 -4.630058e-04
##   beta[2]  1.122221e-01  1.277930e-01
##   beta[3]  5.467194e-02  6.381481e-02
##   alpha   -4.703439e+00 -4.071750e+00
##   lp__    -4.368467e+02 -4.361662e+02
##
## , , chains = chain:3
##
##          stats
```

```
## parameter          mean          sd          2.5%          25%          50%
##   beta[1] -9.070813e-03 0.004723280   -0.01891169   -0.01210502   -0.00902542
##   beta[2]  1.050959e-01 0.012531050    0.08111632    0.09619876    0.10470145
##   beta[3]  4.949523e-02 0.007325256    0.03585624    0.04417273    0.04942980
##   alpha   -5.126969e+00 0.541397429   -6.17335080   -5.49240339   -5.13244163
##   lp__    -4.379432e+02 1.432927582 -441.67172671 -438.63567177 -437.66889171
##          stats
## parameter           75%         97.5%
##   beta[1] -5.874954e-03  1.357217e-04
##   beta[2]  1.134071e-01  1.312819e-01
##   beta[3]  5.444088e-02  6.473672e-02
##   alpha   -4.733901e+00 -4.074562e+00
##   lp__    -4.369156e+02 -4.361342e+02
##
## , , chains = chain:4
##
##          stats
## parameter          mean          sd          2.5%          25%          50%
##   beta[1] -9.387196e-03 0.005095760   -0.01964733   -0.01253983   -0.00930891
##   beta[2]  1.046061e-01 0.013124504    0.08026152    0.09511426    0.10502805
##   beta[3]  4.931762e-02 0.007339069    0.03666175    0.04402776    0.04897963
##   alpha   -5.090148e+00 0.551055055   -6.18434686   -5.47257730   -5.11920006
##   lp__    -4.380055e+02 1.465822511 -441.68641037 -438.70612996 -437.66369511
##          stats
## parameter           75%         97.5%
##   beta[1] -6.063762e-03  4.594560e-04
##   beta[2]  1.139250e-01  1.294433e-01
##   beta[3]  5.433490e-02  6.380032e-02
##   alpha   -4.695407e+00 -4.029607e+00
##   lp__    -4.369121e+02 -4.362186e+02
```

```
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 Rhat
## beta[1]   -0.01    0.00 0.00   -0.02   -0.01   -0.01   -0.01    0.00  2534    1
## beta[2]    0.10    0.00 0.01    0.08    0.10    0.10    0.11    0.13  1533    1
## beta[3]    0.05    0.00 0.01    0.04    0.04    0.05    0.05    0.06  1738    1
## alpha     -5.11    0.01 0.54   -6.21   -5.47   -5.11   -4.73   -4.08  1347    1
## lp__    -437.98    0.04 1.45 -441.72 -438.67 -437.67 -436.91 -436.18  1422    1
##
## Samples were drawn using NUTS(diag_e) at Fri May 31 23:28:53 2024.
## 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).
```

```
posterior <- extract(fit1)
alpha_posterior <- posterior$alpha
beta_posterior <- posterior$beta

print(mean(alpha_posterior))
```

```
## [1] -5.108711
```

```
print(mean(beta_posterior[, 1])) # bp
```

```
## [1] -0.009231391
```

```
print(mean(beta_posterior[, 2])) # bmi
```

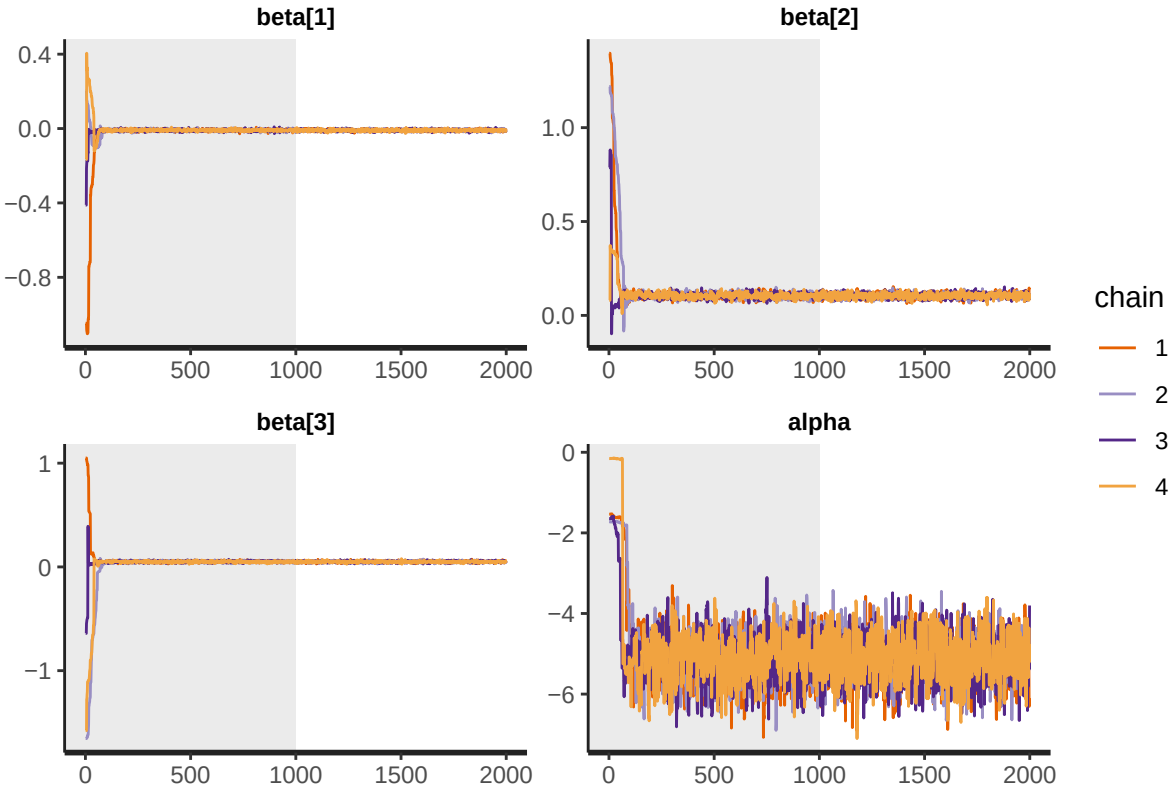
```
## [1] 0.1048025
```

```
print(mean(beta_posterior[, 3])) # age
```

```
## [1] 0.04947429
```

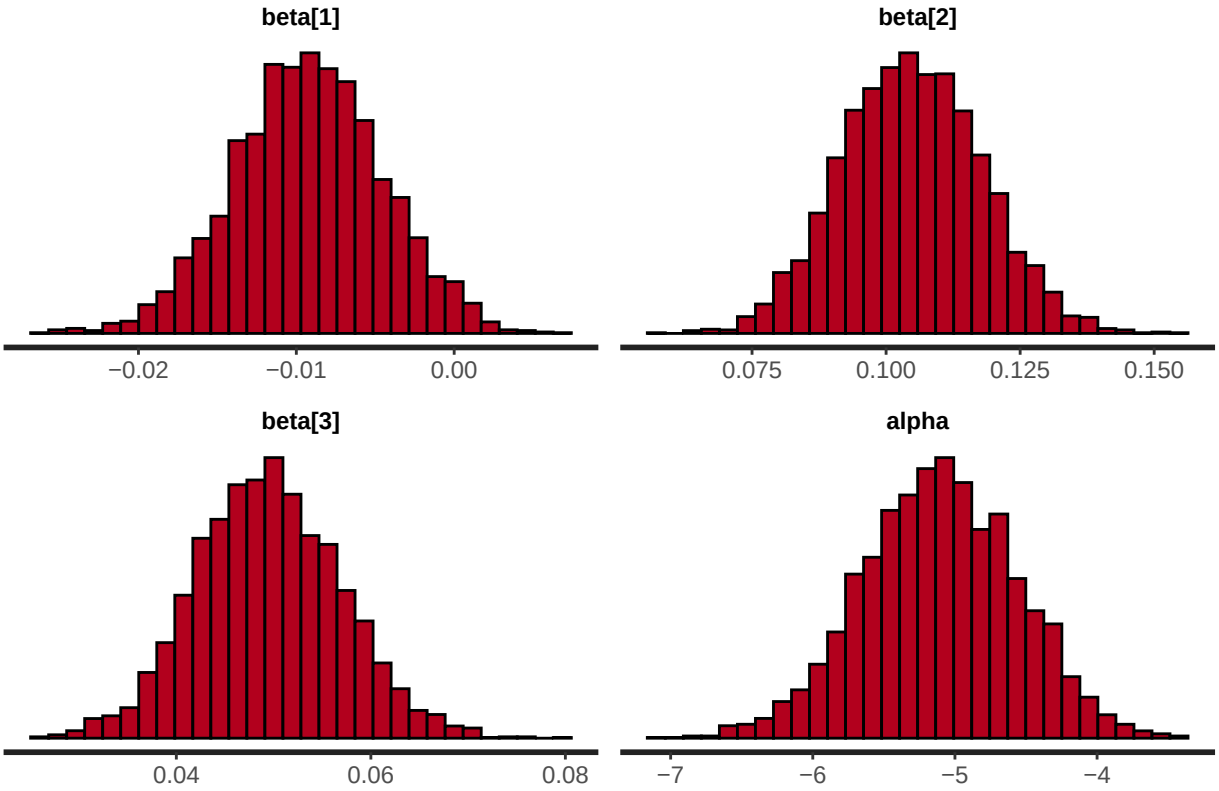
The posterior distributions are shown below.

```
traceplot(fit1, inc_warmup = TRUE)
```




```
plot(fit1, plotfun = "hist")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



After MCMC, the posterior distributions of α, β_1, β_2 and β_3 are given by:

$$\alpha \sim N(-5.10, 0.54^2)$$

$$\beta_1 \sim N(-0.01, 0)$$

$$\beta_2 \sim N(0.1, 0.01^2)$$

$$\beta_3 \sim N(0.05, 0.01^2)$$

we found that our logistic regression model is given by:

$$\hat{p_i} = \frac{exp(\hat{\eta_i})}{1 + exp(\hat{\eta_i})}$$

where

$$\hat{\eta_i} = -5.10 - bp_i \cdot 0.01 + bmi_i \cdot 0.1 + age_i \cdot 0.05$$

Credible intervals for model parameters:

```
# Compute 95% credible intervals
alpha_ci <- quantile(posterior$alpha, probs = c(0.025, 0.975))
beta_ci <- apply(posterior$beta, 2, function(x) quantile(x, probs = c(0.025, 0.975)))

# Print credible intervals
print(alpha_ci)
```

```
##      2.5%     97.5%
## -6.207884 -4.079785
```

```
print(beta_ci)
```

```
##
##                   [,1]       [,2]       [,3]
##   2.5%  -1.856588e-02 0.08023847 0.03510093
##   97.5%  7.498573e-05 0.13102125 0.06449355
```

The 95% credible interval for α is (-6.142455, -4.067584). This suggests that the baseline log odds of being diabetic, when all predictors are zero, is significantly less than zero. For β_1 is (-0.01885, 0.00016) which includes 0 which means that diastolic blood pressure(bp) may not have a significant effect on the likelihood of being diabetic in this dataset. For β_2 is (0.08064, 0.13043), this suggests a significant positive effect of body mass index on the likelihood of being diabetic. For β_3 is (0.03462, 0.06343), which does not include zero. This indicates a significant positive effect of age on the likelihood of being diabetic.

Q2:

```r
absent_data = read.table("absent.txt", header = TRUE)
head(absent_data)
```

```
##   male      math langarts daysabs
## 1    1 56.988830 42.45086       4
## 2    1 37.094160 46.82059       4
## 3    0 32.275460 43.56657       2
## 4    0 29.056720 43.56657       3
## 5    0  6.748048 27.24847       3
## 6    0 61.654280 48.41482      13
```

In a Poisson regression model, the expected value of the count variable y_i is related to a set of predictor variables X_i via a log link function. The model can be expressed as:

$$y_i \sim Poisson(\lambda_i)$$

where λ_i is the rate parameter of the Poisson distribution, and it is modeled as:

$$log(\lambda_i) = \alpha + \beta_1 \cdot male_i + \beta_2 \cdot math_i + \beta_3 \cdot langarts_i$$

The priors are given by:

$$\alpha \sim N(0, 1000)$$

$$\beta_j \sim N(0, 1000) \; for \; j = 1, 2, 3$$

```r
# Define the Stan model as a string
stan_model2 <- "
data {
  int<lower=0> N;              // number of observations
  int<lower=0> y[N];           // outcome variable (daysabs)
  matrix[N, 3] X;              // predictor matrix (male, math, langarts)
}
parameters {
  real alpha;                  // intercept
  vector[3] beta;              // coefficients for predictors
}
model {
  alpha ~ normal(0, 1000);
  beta ~ normal(0, 1000);
  y ~ poisson_log(alpha + X * beta);
}
"
```

```r
stan_data2 <- list(
  N = nrow(absent_data),
  y = absent_data$daysabs,
  X = as.matrix(absent_data[, c('male', 'math', 'langarts')])
)
```

```r
fit2 = stan(model_code = stan_model2, data = stan_data2)
```

```
## Trying to compile a simple C file
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: ‘Apple clang version 14.0.3 (clang-1403.0.22.14.1)’
## using SDK: ‘MacOSX13.3.sdk’
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Framew
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## In file included from /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/
## /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/RcppEigen/include/Eigen/src/Core
## #include <cmath>
##          ^~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 3.3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.33 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.955 seconds (Warm-up)
## Chain 1:                0.33 seconds (Sampling)
## Chain 1:                1.285 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 5e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
```

```
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.116 seconds (Warm-up)
## Chain 2:                0.089 seconds (Sampling)
## Chain 2:                0.205 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 4e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.04 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.139 seconds (Warm-up)
## Chain 3:                0.107 seconds (Sampling)
## Chain 3:                0.246 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 4e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.04 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
```

```
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.113 seconds (Warm-up)
## Chain 4:                0.096 seconds (Sampling)
## Chain 4:                0.209 seconds (Total)
## Chain 4:
```

```
summary(fit2)
```

```
## $summary
##                  mean      se_mean          sd          2.5%           25%
## alpha    2.569584e+00 1.928282e-03 0.083315096  2.399135e+00  2.514509e+00
## beta[1] -5.265655e-01 1.399967e-03 0.057221752 -6.399609e-01 -5.651206e-01
## beta[2] -3.722151e-03 3.886154e-05 0.002179386 -7.913934e-03 -5.243670e-03
## beta[3] -1.654843e-03 4.399400e-05 0.002344127 -6.113027e-03 -3.190549e-03
## lp__     1.463307e+03 3.910305e-02 1.441083694  1.459638e+03  1.462616e+03
##                  50%           75%         97.5%    n_eff      Rhat
## alpha    2.570838e+00  2.626212e+00  2.730034e+00 1866.837 1.0006763
## beta[1] -5.253895e-01 -4.893719e-01 -4.131542e-01 1670.654 1.0006743
## beta[2] -3.721054e-03 -2.295831e-03  5.974876e-04 3145.055 0.9999811
## beta[3] -1.697264e-03 -1.256934e-04  3.034875e-03 2839.066 1.0002537
## lp__     1.463638e+03  1.464382e+03  1.465069e+03 1358.180 1.0030313
##
## $c_summary
## , , chains = chain:1
##
##          stats
## parameter          mean          sd          2.5%           25%           50%
##   alpha    2.572664e+00 0.079455606  2.423940e+00  2.517283e+00  2.573976e+00
##   beta[1] -5.280142e-01 0.054402980 -6.298007e-01 -5.674709e-01 -5.255991e-01
##   beta[2] -3.676923e-03 0.002089028 -7.470507e-03 -5.177273e-03 -3.645484e-03
##   beta[3] -1.752246e-03 0.002292236 -6.024057e-03 -3.323143e-03 -1.717808e-03
##   lp__     1.463465e+03 1.300402186  1.460183e+03  1.462901e+03  1.463760e+03
##          stats
## parameter           75%         97.5%
##   alpha    2.629306e+00  2.721390e+00
##   beta[1] -4.948123e-01 -4.144987e-01
##   beta[2] -2.261829e-03  3.677696e-04
##   beta[3] -1.785203e-04  2.633941e-03
##   lp__     1.464383e+03  1.465040e+03
##
## , , chains = chain:2
##
##          stats
## parameter          mean          sd          2.5%           25%           50%
##   alpha    2.563971e+00 0.085255934  2.396679e+00  2.509234e+00  2.564715e+00
##   beta[1] -5.244182e-01 0.060223006 -6.491174e-01 -5.625144e-01 -5.246021e-01
```

```
##   beta[2] -3.784608e-03 0.002222225 -8.049553e-03 -5.335893e-03 -3.783103e-03
##   beta[3] -1.505241e-03 0.002426040 -6.223284e-03 -3.110242e-03 -1.533997e-03
##   lp__     1.463197e+03 1.597559857  1.458956e+03  1.462431e+03  1.463540e+03
##          stats
## parameter           75%         97.5%
##   alpha    2.618196e+00  2.735282e+00
##   beta[1] -4.875704e-01 -4.002977e-01
##   beta[2] -2.349767e-03  6.927072e-04
##   beta[3]  8.981119e-05  3.200946e-03
##   lp__     1.464400e+03  1.465081e+03
##
## , , chains = chain:3
##
##          stats
## parameter          mean          sd          2.5%           25%           50%
##   alpha    2.572827e+00 0.081649994  2.397299e+00  2.522965e+00  2.577154e+00
##   beta[1] -5.292250e-01 0.058330591 -6.407156e-01 -5.700024e-01 -5.275656e-01
##   beta[2] -3.678635e-03 0.002159340 -7.858523e-03 -5.183874e-03 -3.724984e-03
##   beta[3] -1.720308e-03 0.002290367 -6.028182e-03 -3.167687e-03 -1.773605e-03
##   lp__     1.463304e+03 1.421119769  1.459678e+03  1.462586e+03  1.463598e+03
##          stats
## parameter           75%         97.5%
##   alpha    2.626429e+00  2.718909e+00
##   beta[1] -4.901657e-01 -4.197058e-01
##   beta[2] -2.132161e-03  7.580110e-04
##   beta[3] -2.295983e-04  2.865626e-03
##   lp__     1.464382e+03  1.465090e+03
##
## , , chains = chain:4
##
##          stats
## parameter          mean          sd          2.5%           25%           50%
##   alpha    2.568872e+00 0.086521117  2.394088e+00  2.507979e+00  2.567943e+00
##   beta[1] -5.246046e-01 0.055682587 -6.408453e-01 -5.608286e-01 -5.243181e-01
##   beta[2] -3.748437e-03 0.002244894 -8.077598e-03 -5.272080e-03 -3.717813e-03
##   beta[3] -1.641577e-03 0.002360946 -6.111659e-03 -3.290758e-03 -1.652160e-03
##   lp__     1.463262e+03 1.418230093  1.459753e+03  1.462542e+03  1.463618e+03
##          stats
## parameter           75%         97.5%
##   alpha    2.627915e+00  2.738768e+00
##   beta[1] -4.885843e-01 -4.190411e-01
##   beta[2] -2.338123e-03  5.861730e-04
##   beta[3] -1.219102e-04  3.142936e-03
##   lp__     1.464352e+03  1.465029e+03
```

```
print(fit2)
```

```
## 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
## alpha     2.57    0.00 0.08    2.40    2.51    2.57    2.63    2.73  1867    1
## beta[1]  -0.53    0.00 0.06   -0.64   -0.57   -0.53   -0.49   -0.41  1671    1
## beta[2]   0.00    0.00 0.00   -0.01   -0.01    0.00    0.00    0.00  3145    1
```

```
## beta[3]    0.00    0.00 0.00   -0.01    0.00    0.00    0.00    0.00  2839    1
## lp__    1463.31    0.04 1.44 1459.64 1462.62 1463.64 1464.38 1465.07  1358    1
##
## Samples were drawn using NUTS(diag_e) at Fri May 31 23:29:18 2024.
## 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).
```

```
posterior <- extract(fit2)
alpha_posterior <- posterior$alpha
beta_posterior <- posterior$beta

print(mean(alpha_posterior))
```

```
## [1] 2.569584
```

```
print(mean(beta_posterior[, 1])) # male
```

```
## [1] -0.5265655
```

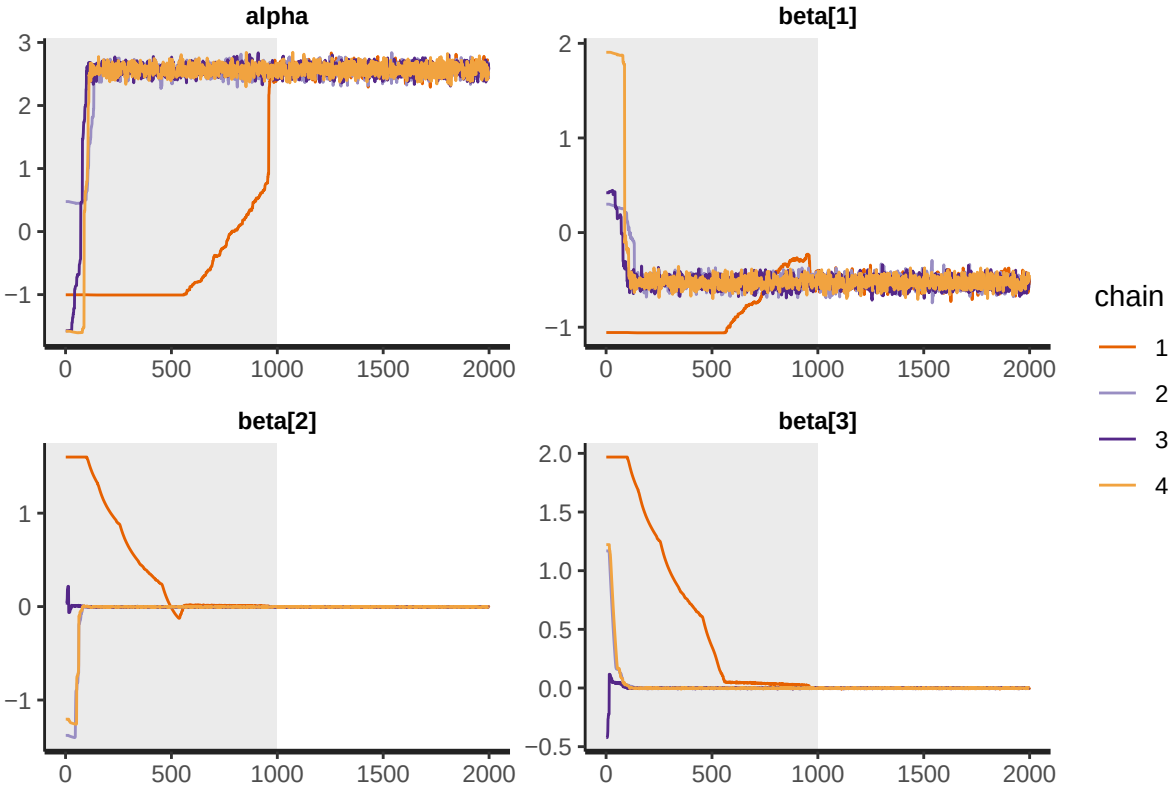
```
print(mean(beta_posterior[, 2])) # math
```

```
## [1] -0.003722151
```

```
print(mean(beta_posterior[, 3])) # langarts
```

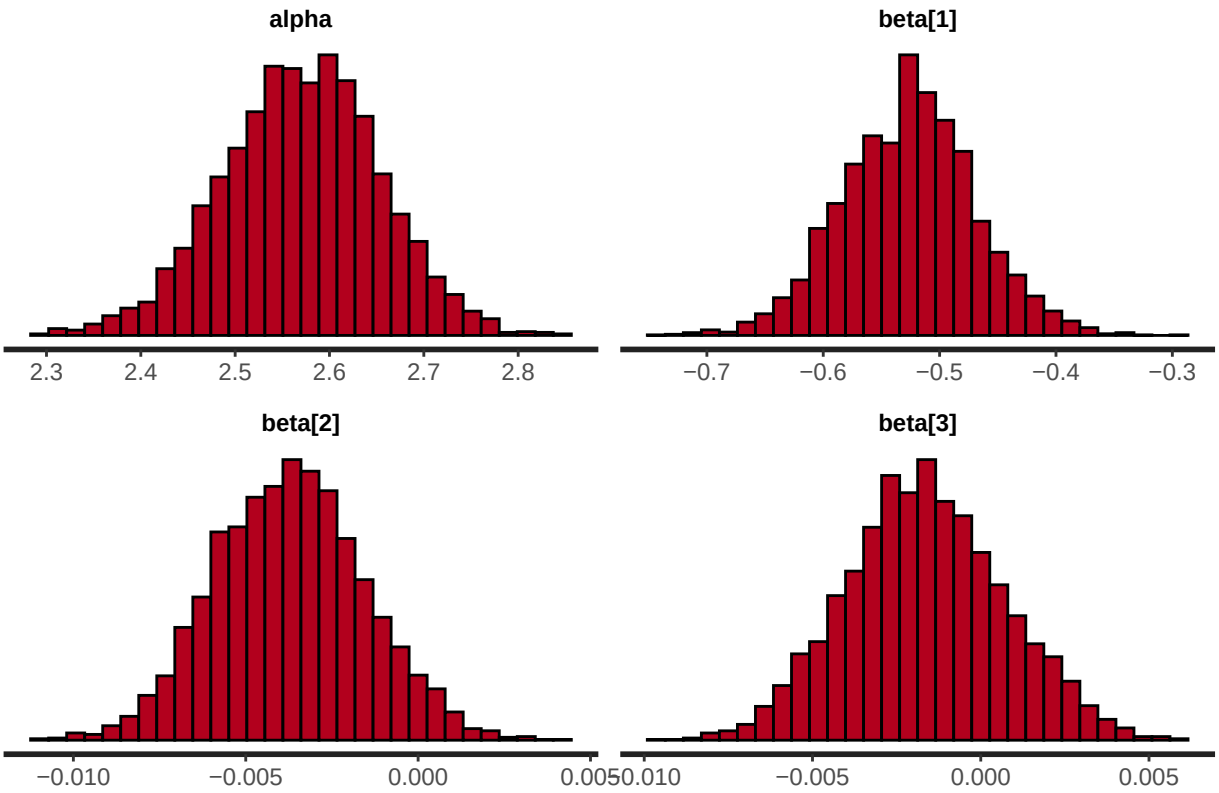
```
## [1] -0.001654843
```

```
traceplot(fit2, inc_warmup = TRUE)
```




```
plot(fit2, plotfun = "hist")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



After MCMC, the posterior distributions of α, β_1, β_2 and β_3 are given by:

$$\alpha \sim N(2.57, 0.08^2)$$

$$\beta_1 \sim N(-0.53, 0.06^2)$$

$$\beta_2 \sim N(0, 0)$$

$$\beta_3 \sim N(0, 0)$$

And our poisson regression model is given by:

$$\hat{\lambda_i} = exp(\hat{\eta_i}) = exp(2.57 - 0.53 \cdot male_i)$$

Credible intervals for model parameters:

```
# Compute 95% credible intervals
alpha_ci <- quantile(posterior$alpha, probs = c(0.025, 0.975))
beta_ci <- apply(posterior$beta, 2, function(x) quantile(x, probs = c(0.025, 0.975)))

# Print credible intervals
print(alpha_ci)
```

```
##     2.5%    97.5%
## 2.399135 2.730034
```

```
print(beta_ci)
```

```
##
##                [,1]          [,2]         [,3]
##   2.5%  -0.6399609 -0.0079139337 -0.006113027
##   97.5% -0.4131542  0.0005974876  0.003034875
```

The 95% credible interval for α is (2.406966, 2.732167). This suggests that the baseline log rate of days absent, when all predictors are zero, is significantly greater than zero. For β_1 is (-0.6388425, -0.6388425) indicating that being male may have a positive effect on the number of days absent, though the effect is small. For β_2 is (-0.007995, 0.000492) which includes 0, suggesting that math test scores may not have a significant effect on the number of days absent. For β_3 is (-0.00617, 0.00296) which also includes 0 indicating that language arts test scores may not have a significant effect on the number of days absent.