Project 4

Bayesian Analysis of Question 4 from the Master's Proficiency Exam

Sam Albertson, Suz Angermeier, and Dani Vaithalingam

April 17, 2024

Contents

```r
knitr::opts_chunk$set(echo = TRUE, warning=FALSE, message=FALSE)
library(rstanarm)
```

```
## Warning: package 'rstanarm' was built under R version 4.3.2

## Loading required package: Rcpp

## This is rstanarm version 2.26.1

## - See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!

## - Default priors may change, so it's safest to specify priors, even if equivalent to the defaults.

## - For execution on a local, multicore CPU with excess RAM we recommend calling

##   options(mc.cores = parallel::detectCores())
```

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.2

## Warning: package 'forcats' was built under R version 4.3.2

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
path = here::here("bayesSim.R")
source(path)
ost = read.csv(here::here("Project 4", "osteo.csv")) %>%
  mutate(fracture.b = ifelse(fracture=="Yes", 1, 0),
         priorfrac.b = ifelse(priorfrac == "Yes", 1, 0))
```

1 Background

The enclosed dataset called OSTEO.CSV includes the results of a prospective cohort study that followed 500 women age 55 or older for 1 year after diagnosis with osteoporosis. The primary outcome of the study was bone fracture (yes/no) within the 1st year of osteoporosis diagnosis. The investigators hypothesized that women who had a history of bone fractures prior to their osteoporosis diagnosis would be at higher risk of having an osteoporosis-related fracture than those who had no such prior history. **A potentially important prognostic variable was age at diagnosis of osteoporosis, which was recorded in years**. You have already done a frequentist analysis of this dataset. Now, you will analyze the same dataset using a Bayesian logistic regression model. You will complete the analysis by answering the following questions.

2 Question 1: Write the Bayesian Logistic Regression Model

Write a simple Bayesian logistic regression model for fracture at 1 year post-diagnosis with osteoporosis based only on a history of prior fracture. Follow the example notation used in your textbook in Chapter 13, equation 13.5. In addition to specifying the model in mathematical notation, give a plain English description of what the notation means. Do this separately for the data model and the priors as indicated below.

2.1 Write the data model

$Y_i|\beta_0, \beta_1 \sim Bernoulli(\pi_i)$, where Y_i is a random variable denoting whether participant i has bone fracture and π_i is the probability of bone fracture for participant i.

$\pi_i = \frac{e^{\beta_0+\beta_1 X_{i1}}}{1+e^{\beta_0+\beta_1 X_{i1}}}$

In words, the value of the random variable Y_i, given the values of β_0 and β_1, comes from a Bernoulli distribution with probability π_i, where π_i is a function of the parameters β_1 and β_0 and X_{i1}, which is an indicator variable for whether the participant in question had a prior history of fractures. In order to this relationship we use the model:

$ln[\frac{\pi_i}{1-\pi_i}] = \beta_0 + \beta_1 X_{i1}$

2.2 Write the model for the prior

$\beta_0 \sim N(\mu_{\beta_0}, \sigma^2_{\beta_0})$ $\beta_1 \sim N(\mu_{\beta_1}, \sigma^2_{\beta_1})$

In words, the intercept parameter β_0, which is the log-odds of fracture in women with no prior fracture, has a normally distributed prior centered at some mean with some standard deviation, both of which will be specified in the next section.

The slope parameter β_1 represents the increase in log-odds of fracture in women with prior fracture compared to women without a prior fracture history and this parameter also has a normally distributed prior.

Specify "vague" normal prior distributions for the parameters in your data model as follows.

2.2.1 Prior for Beta0

First, assume your prior information says that 20% of women will have fractures at 1 year when they don't have a prior history of fracture. Your level of uncertainty about this is quite high. A prior that has a 95% credible interval ranging from about 12% to 31% would capture your uncertainty well. What are the parameters of a normally distributed prior for β_0 that express this uncertainty?

We want μ_{β_0} such that $\frac{e^{\mu_{\beta_0}}}{1+e^{\mu_{\beta_0}}} = 0.20$.

If we solve for μ_{β_0}, we get μ_{β_0} = -1.3863

Then, given the 95% credible interval, logit(0.12) to logit(0.31), we get the log odds CI bounds as $log(0.12/0.88)$ = -1.957 and $log(0.31/0.69)$ = -0.787. We can then use these to calculate the standard deviation $\sigma^2_{\mu_{\beta_0}}$.

The width of the interval (from -1.957 to -0.787) is approximately $-0.787 - (-1.957) = 1.17$.

Therefore 1.96 * $\sigma_{\mu_{\beta_0}} = 1.17$ and $\sigma_{\mu_{\beta_0}} = 0.597$.

$\beta_0 \sim N(\mu_{\beta_0} = -1.386, \sigma^2_{\beta_0} = 0.597^2)$

2.2.2 Prior for Beta1

Assume that you have prior information that suggests a history of fractures is expected to increase the probability of fracture by 10% over not having a prior history (i.e., a 30% probability of fracture at 1 year in women who have a prior history of fracture vs. 20% probability in women without a prior history). But your prior information is highly uncertain. In fact, it seems plausible that women with a prior history of fracture may have as little as a 10% probability of fracture at 1 year (the same risk in women without a prior history!) or as high as a 62% risk of fracture at 1 year (a very large increase in risk over women without a prior history). Based on this information, specify a normally distributed prior for β_1 on the log-odds scale and write an interpretation of the mean and 95% credible interval with respect the clinical problem (i.e., whether a prior history promotes or reduces the risk of fracture at 1 year).

State the mean and 95% credible interval for the β_1 prior on the odds ratio scale and write an interpretation.

First, we convert the given probabilities to log-odds. The log-odds of fracture for not having a prior history are $log(0.20/0.80) = -1.386$ and with a prior history are $log(0.30/0.70) = -0.847$.

The mean of the prior is the difference in log odds of fracture between having a prior history and not having a prior history, $\mu_{\beta_1} = -0.847 - (-1.386) = 0.539$.

To get the standard deviation of the prior, we use the 95% credible interval that ranges from a 10% probability (log odds ≈ -2.197) to a 62% probability (log odds ≈ 0.465). The width of this interval is approximately $0.465 - (-2.197) = 2.662$. For a 95% credible interval (± 1.96 standard deviations for a normal distribution), we calculate the standard deviation (σ): 1.96 x σ = 2.662 -> $\sigma = \frac{2.662}{1.96} = 1.359$

$\beta_1 \sim N(\mu_{\beta_1} = 0.539, \sigma^2_{\beta_1} = 1.359^2)$

The prior mean suggests that having a prior history of fracture is associated with a $\beta_1 = 0.539$ (-2.123, 3.201) increase in the log odds of fracture at 1 year compared to not having a prior history of fracture. The 95% credible interval, tells us that there is a 95% probability that the true mean lies within the interval -2.123 and 3.201, given the evidence provided.

3 Question 2: Simulation of the Priors

Simulate the priors for your model. Plot the theoretical priors as well as the MCMC simulated priors and verify the simulations are correct. State the mean, median, and 95% credible intervals for your priors and write interpretations for these.

HINT: Follow the instructions in the stan_glm() help pages and use a model without an intercept. Include a term for no prior history of fracture instead of the intercept:

"If you prefer to specify a prior on the intercept without the predictors being auto-centered, then you have to omit the intercept from the formula and include a column of ones as a predictor, in which case some element of [prior] specifies the prior on it, rather than [prior_intercept]."

More information is here.

$$\beta_0 \sim N(\mu_{\beta_0} = -1.386, \sigma^2_{\beta_0} = 0.597^2)$$

$$\beta_1 \sim N(\mu_{\beta_1} = 0.539, \sigma^2_{\beta_1} = 1.359^2)$$

```
library(rstanarm)
library(bayesplot)
library(tidyverse)
library(tidybayes)
library(broom.mixed)

sd0 = 0.597^2
```

```r
sd1 = 1.359^2

mod = stan_glm(fracture.b ~ priorfrac.b,
               data = ost,
               family = binomial,
               prior_intercept = normal(-1.386, sd0, autoscale = TRUE),
               prior = normal(0.539,sd1, autoscale = TRUE),
               chains = 4,
               iter = 5000*2,
               seed = 1345,
               prior_PD = TRUE)
```

```
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.5e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.25 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.087 seconds (Warm-up)
## Chain 1:                0.141 seconds (Sampling)
## Chain 1:                0.228 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 9e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
```

```
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.088 seconds (Warm-up)
## Chain 2:                0.099 seconds (Sampling)
## Chain 2:                0.187 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 9e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.094 seconds (Warm-up)
## Chain 3:                0.109 seconds (Sampling)
## Chain 3:                0.203 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.2e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 4: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 4: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 4: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 4: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 4: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 4: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 4: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 4: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 4: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 4: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
```

```
## Chain 4:
## Chain 4:  Elapsed Time: 0.089 seconds (Warm-up)
## Chain 4:                0.101 seconds (Sampling)
## Chain 4:                0.19 seconds (Total)
## Chain 4:
```

```
prior_summary(mod)
```

```
## Priors for model 'mod'
## ------
## Intercept (after predictors centered)
##  ~ normal(location = -1.4, scale = 0.36)
##
## Coefficients
##   Specified prior:
##     ~ normal(location = 0.54, scale = 1.8)
##   Adjusted prior:
##     ~ normal(location = 0.54, scale = 4.2)
## ------
## See help('prior_summary.stanreg') for more details
```

```
samples = as.matrix(mod)
summary(samples)
```

```
##   (Intercept)       priorfrac.b
##  Min.   :-5.9575   Min.   :-16.6556
##  1st Qu.:-2.2763   1st Qu.: -2.3929
##  Median :-1.5053   Median :  0.4627
##  Mean   :-1.5107   Mean   :  0.4967
##  3rd Qu.:-0.7414   3rd Qu.:  3.3759
##  Max.   : 3.0231   Max.   : 18.2776
```

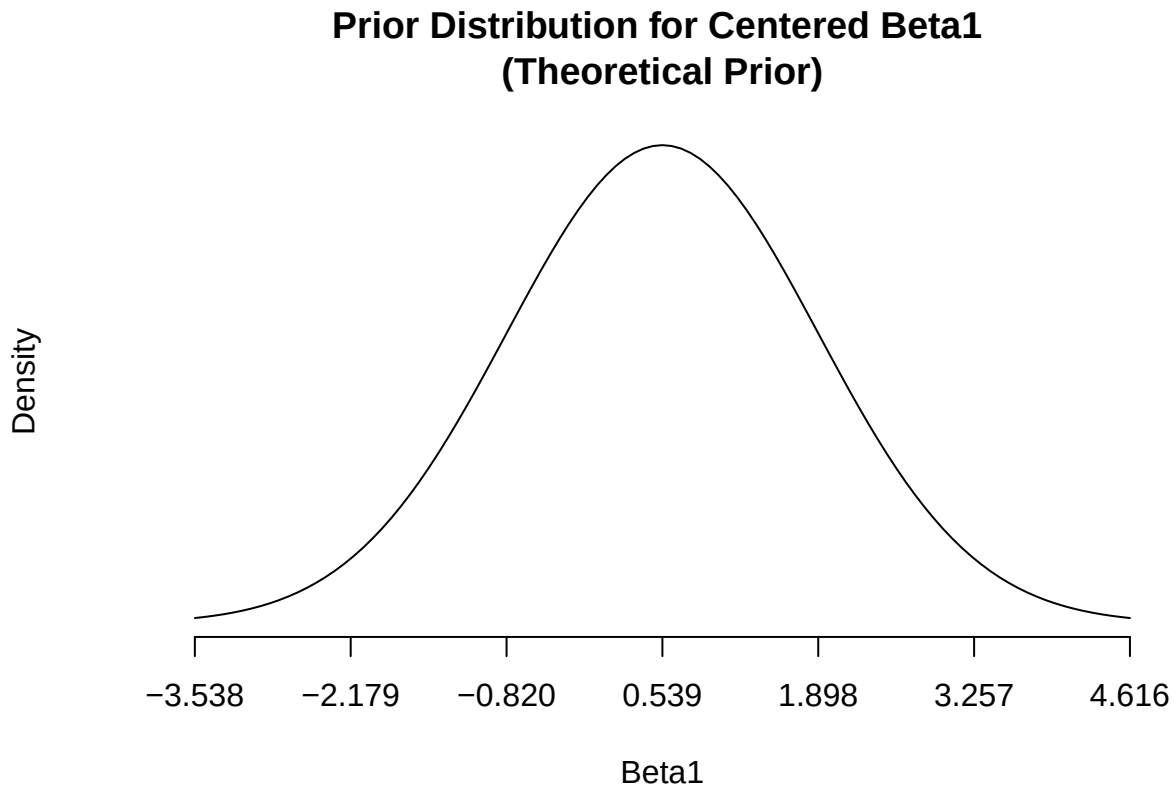
```
mean.b1 = mean(samples[, "priorfrac.b"])
median.b1 = median(samples[, "priorfrac.b"])

cred.int = hdi(samples[, "priorfrac.b"])
cred.int.exp = hdi(exp(samples[, "priorfrac.b"]))
med.b1.exp = median(samples[, "priorfrac.b"])

## theoretical prior -> symmetric so mean = median
mu = 0.539
var = sd1
sigma = sqrt(var)

ciLower = mu - 2 * sqrt(var)
ciUpper = mu + 2 * sqrt(var)

probLower = exp(ciLower)/(1 + exp(ciLower))
probUpper = exp(ciUpper)/(1 + exp(ciUpper))

round(probLower, 2)
```

```
## [1] 0.1
```

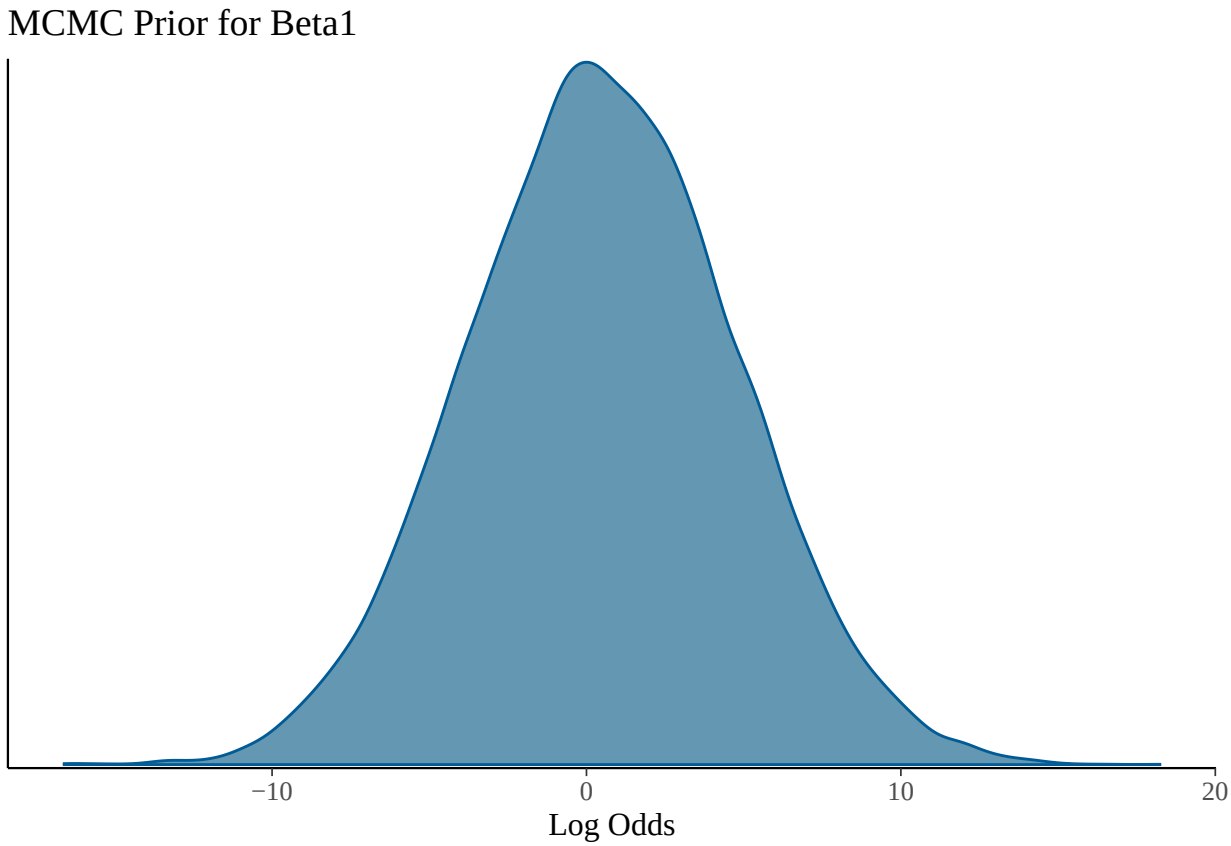
```
round(probUpper, 2)
```

```
## [1] 0.96
```

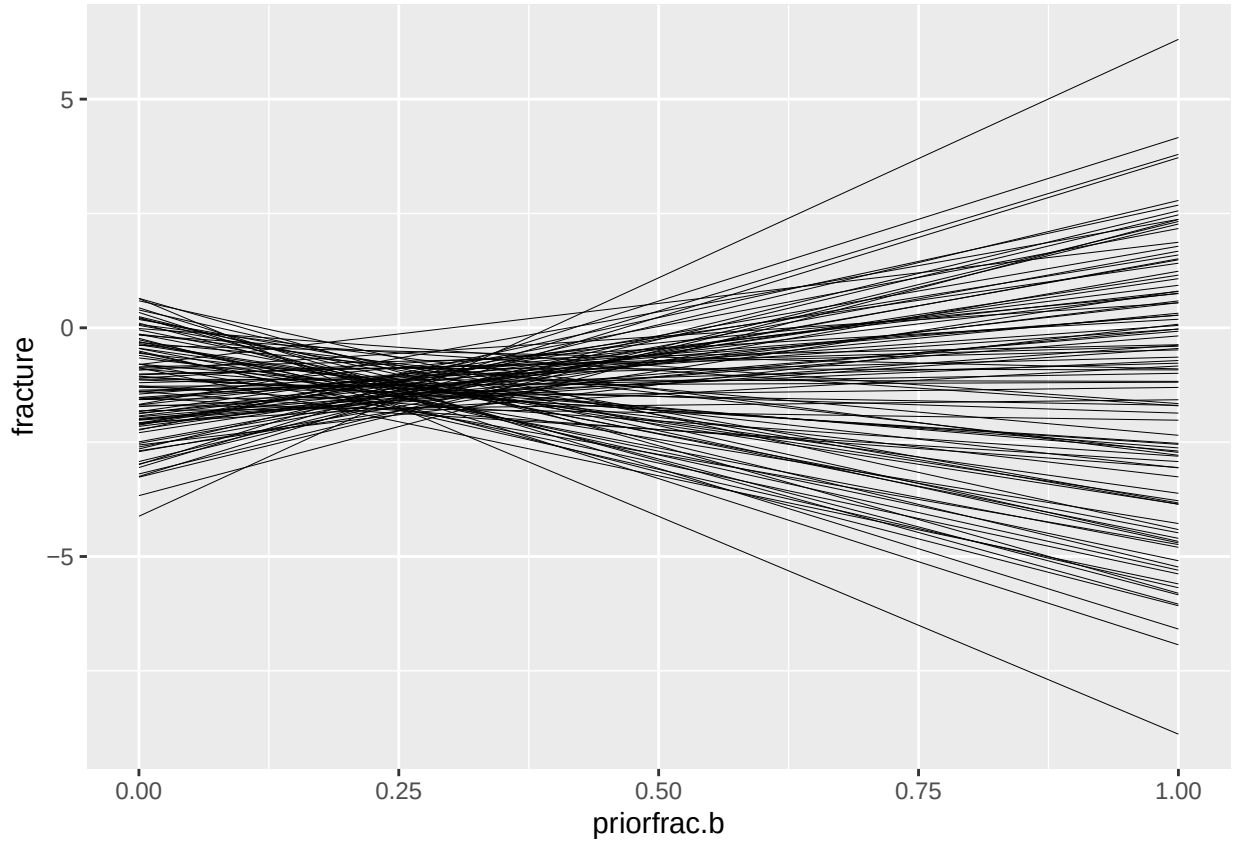
```
curve(dnorm(x, mu, sigma), from = mu-3*sigma, to=mu+3*sigma, axes = FALSE, xlab = "Beta1", ylab = "Dens

axis(1, at=c(mu-3*sigma, mu-2*sigma, mu-sigma, mu,
mu+sigma, mu+2*sigma, mu+3*sigma)
)
```




```
# Plot the prior distribution for the slope coefficient, Beta1
mcmc_dens(mod,pars=c("priorfrac.b")) +
scale_x_continuous(name="Log Odds") +
ggtitle("MCMC Prior for Beta1")
```




```
## To check that the simulation is good, randomly sample 100 possible prior models

ost %>%
  add_linpred_draws(mod, ndraws = 100)%>%
  ggplot(aes(x = priorfrac.b, y = fracture))+
  geom_line(aes(y = .linpred, group = .draw), size = 0.1)
```



The mean of β_1, 0.5, is the average expected increase in log odds of fracture associated with having a prior history of fracture compared to not having prior history. The median, 0.46 is the center value of the distribution of β_1. The 95% credible interval -7.8 to 8.8, so there is a 95% probability of that the true median estimate is between -7.8 and 8.8.

The graph of 100 models that represent the prior belief about the effects of prior history of fracture at 1 year. This helps us visualize the prior belief of our regression coefficients, from 100 possibilities for our set of parameters. There are lines that are horizontal, positive, and negative. This tells us that there is a lot of uncertainty of the effects of prior fractures on current fractures. So we have reasonable tuning of our priors.

4 Question 3: Update your prior belief using the OSTEO data

Now, use the OSTEO data set to update your priors. Plot the posterior distribution for the change in log-odds of fracture at 1 year associated with a prior history of fracture. Interpret the median and 95% highest posterior density credible interval with respect to the research question about whether prior history of fracture is a risk factor for fracture at 1 year after diagnosis with osteoporosis. Repeat these steps on the odds ratio scale.

```
mod.post = update(mod, prior_PD = FALSE)
```

```
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.8e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.28 seconds.
```

```
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.572 seconds (Warm-up)
## Chain 1:                0.76 seconds (Sampling)
## Chain 1:                1.332 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2.7e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.27 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.647 seconds (Warm-up)
## Chain 2:                0.785 seconds (Sampling)
## Chain 2:                1.432 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 3.6e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.36 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
```

```
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.62 seconds (Warm-up)
## Chain 3:                0.781 seconds (Sampling)
## Chain 3:                1.401 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 2.8e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.28 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 4: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 4: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 4: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 4: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 4: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 4: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 4: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 4: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 4: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 4: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.666 seconds (Warm-up)
## Chain 4:                0.801 seconds (Sampling)
## Chain 4:                1.467 seconds (Total)
## Chain 4:
```

```
print(mod.post, digits= 3)
```

```
## stan_glm
##  family:       binomial [logit]
##  formula:      fracture.b ~ priorfrac.b
##  observations: 500
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) -1.442  0.123
## priorfrac.b  1.074  0.224
##
```

```
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
mcmc.chain = as.matrix(mod.post)
ci = hdi(mcmc.chain[, "priorfrac.b"]); ci
```

```
##            [,1]     [,2]
## [1,] 0.6425326 1.504506
```

```
med = median(mcmc.chain[, "priorfrac.b"]); med
```

```
## [1] 1.074491
```

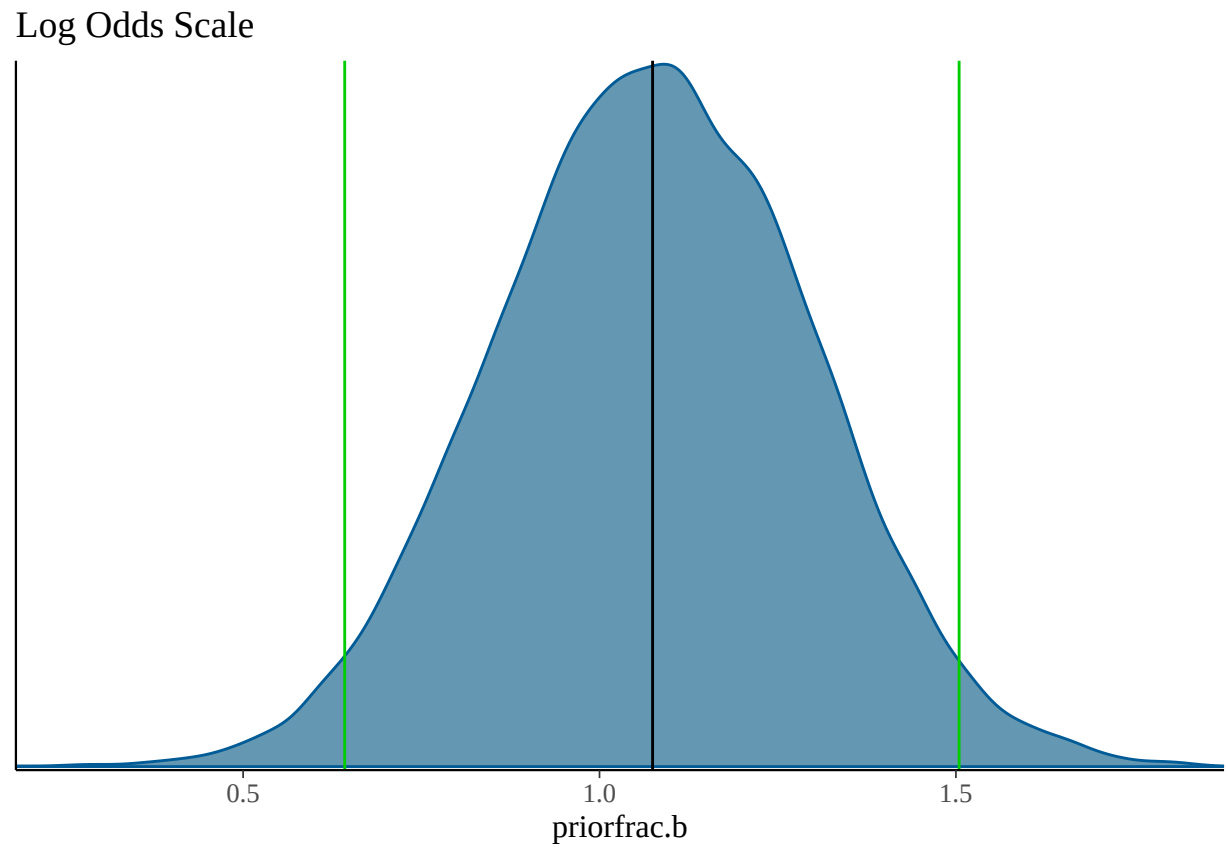
```
ci.exp = hdi(exp(mcmc.chain[, "priorfrac.b"])); ci.exp
```

```
##           [,1]     [,2]
## [1,] 1.793326 4.315921
```

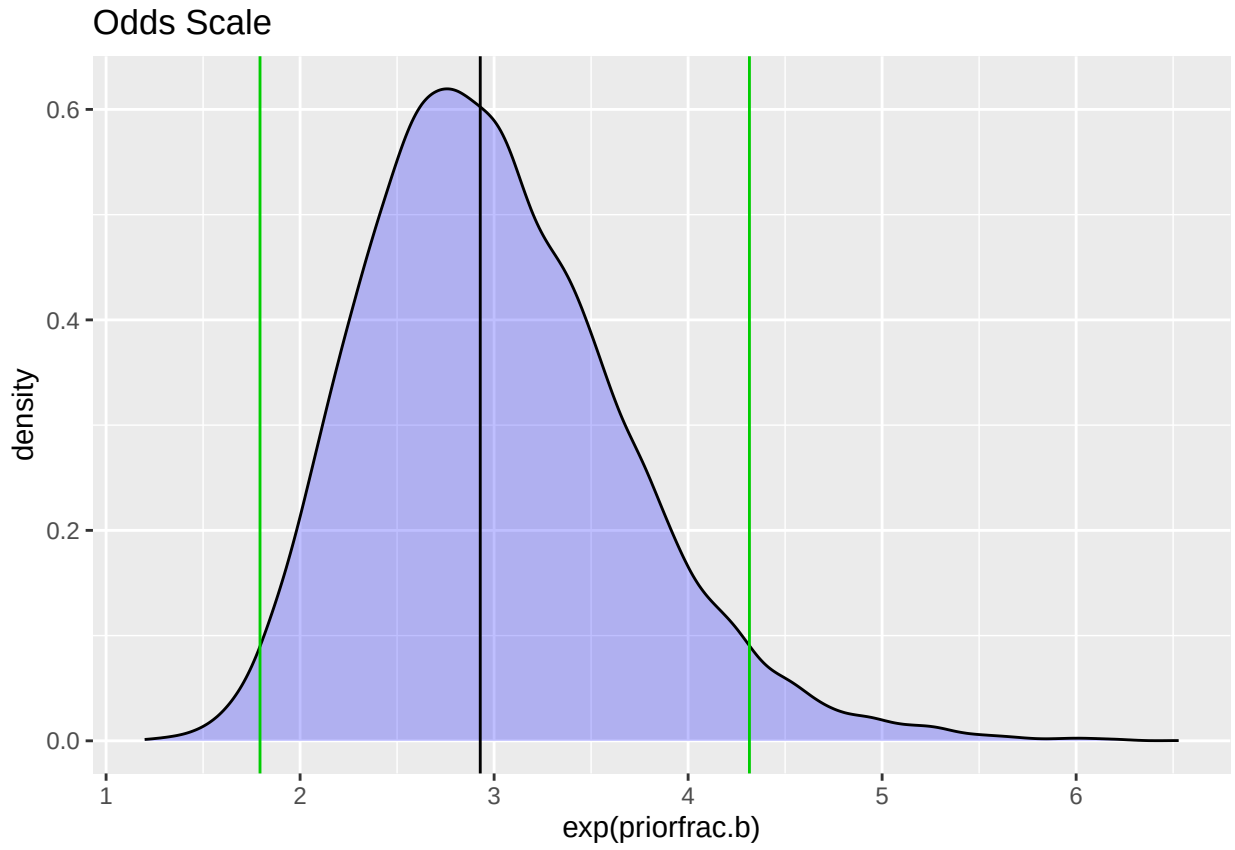
```
med.exp = median(exp(mcmc.chain[,"priorfrac.b"]), na.rm = TRUE); med.exp
```

```
## [1] 2.928501
```

```
mcmc_dens(mod.post, pars = c("priorfrac.b")) + geom_vline(xintercept = med, color = "black") + geom_vli
```



```r
mcd = as.data.frame(mcmc.chain)

ggplot(data = mcd) + geom_density(aes(x = exp(priorfrac.b)), fill = "blue", alpha = 0.25) + geom_vline(
```



On the log odds scale, the median is 1.1, as shown in black on the plot above, with a 95% credible interval of 0.64 to 1.5. In words, the log odds ratio of fracture for participants with prior fracture compared to those without is approximately 1.1 [95% Credible interval: 0.64 to 1.5]. There is a 95% probability that the true median of the prior fracture log OR is between 0.64 and 1.5, given our observed data.

On the odds scale, the median is 2.9, as shown in black on the plot above, with a 95% credible interval of 1.8 to 4.3. In words the odds ratio of fracture for participants with prior fracture compared to those without is approximately 2.9 [95% Credible interval: 1.8 to 4.3]. There is a 95% probability that the true median of the prior fracture OR is between 1.8 and 4.3, given our observed data.

What is the posterior probability that the odds ratio is greater than 1? What does this imply about the risk of fracture at 1 in women with vs. without a prior history of fracture?

```r
sum(ifelse(mcmc.chain[, "priorfrac.b"] > 0, 1, 0)) / length(mcmc.chain[,"priorfrac.b"]) ## odds ratio 0
```

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

The posterior probability that the odds ratio is greater than 1 is: approximately 1.0.

This implies that history of prior fracture is positively associated with the probability of fracture.

What is the posterior probability that the odds ratio is greater than 2?

```r
sum(ifelse(mcmc.chain[, "priorfrac.b"] > log(2), 1, 0)) / length(mcmc.chain[,"priorfrac.b"]) ## odds ra
```

```
## [1] 0.95825
```

The posterior probability that the odds ratio is greater than 2 is 0.96.

What is the value of the odds ratio at which the posterior probability exceeds 80%?

```r
post.prob = 0
i = 0
ORs = seq(10, 2, by = -0.1)
while(post.prob < 0.8){
  OR = ORs[i]
  post.prob = sum(ifelse(mcmc.chain[, "priorfrac.b"] > log(OR), 1, 0)) / length(mcmc.chain[,"priorfrac.
  i = i + 1
}
cat("OR = ", OR, "post.prob =", post.prob)
```

```
## OR =  2.4 post.prob = 0.8138
```

```r
post.prob = 0
i = 0
ORs = seq(3, 2, by = -0.001)
while(post.prob < 0.8){
  OR = ORs[i]
  post.prob = sum(ifelse(mcmc.chain[, "priorfrac.b"] > log(OR), 1, 0)) / length(mcmc.chain[,"priorfrac.
  i = i + 1
}
cat("OR = ", OR, "post.prob =", post.prob)
```

```
## OR =  2.427 post.prob = 0.80055
```

The highest value of the odds ratio for which the posterior probability exceeds 80% is 2.427.

Based on the above posterior probabilities, write a summary sentence that describes the strength of the evidence you know that that a prior history of fracture is a risk factor for fracture 1 year after diagnosis with osteoporosis.

Based on the above posterior probabilities, we have very strong evidence that the prior history of fracture is a risk factor for fracture 1 year after diagnosis with osteoporosis as the posterior probability of the odds ratio being greater than 1 is approximately 100%. Additionally, there is a greater than 80% probability that the odds ratio is greater than 2.427.

5 Question 4: Including Age in the Model

5.1 Writing the Data Model

Write the data model when the age term is included. Use a similar format to what you used to write the data model in question 1. In your notation, use β_0, β_1, and β_2 as the intercept, prior history of fracture, and age respectively and explain what each parameter means. Remember, rstanarm gives model results without

centering (even though the prior for the intercept is based on the other predictors being centered; see the next question).

The data model including the age term is as follows:

$ln[\frac{\pi_i}{1-\pi_i}] = \beta_0 + \beta_1 X_{i1} + \beta_2 Age_{i1}$

Where $ln[\frac{\pi_i}{1-\pi_i}]$ represents the log-odds of bone fracture in a given patient compared to non-osteoperosis patients at birth. β_0 is the log-odds in the reference group, while β_1 is the increase in risk associated with an osteoperosis diagnosis, and $beta_2$ is the increase associated with a one year increase in age.

5.2 Prior Distributions for the Regression Parameters

So far you have used informative but vague priors for the risk of fracture in women with and without a prior history of fracture. The priors are vague in the sense that they suggest an expected difference in the population comparing women with and without a prior history, but the level of uncertainty is very high.

Suppose now that you want to include age in the model since you already know age is a confounder in the population and your descriptive analysis demonstrates age is a confounder in your dataset. Suppose however that you don't have any pre-study information about the risk of fracture in women with and without a prior history according to age. So, instead of using vague, informative priors you will not let rstanarm select weakly informative priors scaled to match your data.

Create an MCMC simulation of weakly informative priors for a model that includes prior history of fracture and age as covariates. Write the priors that rstanarm has identified.

```
mod = stan_glm(fracture.b ~ priorfrac.b + age,
               data = ost,
               family = binomial,
               prior_intercept = normal(0, 2.5, autoscale = TRUE),
               prior = normal(0, 2.5, autoscale = TRUE),
               chains = 4,
               iter = 5000*2,
               seed = 1345,
               prior_PD = TRUE)
```

```
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 9e-06 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
```

```
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.143 seconds (Warm-up)
## Chain 1:                0.16 seconds (Sampling)
## Chain 1:                0.303 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.3e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.137 seconds (Warm-up)
## Chain 2:                0.162 seconds (Sampling)
## Chain 2:                0.299 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.2e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.143 seconds (Warm-up)
## Chain 3:                0.257 seconds (Sampling)
```

```
## Chain 3:                0.4 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.4e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.14 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 4: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 4: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 4: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 4: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 4: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 4: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 4: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 4: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 4: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 4: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.141 seconds (Warm-up)
## Chain 4:                0.148 seconds (Sampling)
## Chain 4:                0.289 seconds (Total)
## Chain 4:
```

```
p_sum <- prior_summary(mod)
print(p_sum)
```

```
## Priors for model 'mod'
## ------
## Intercept (after predictors centered)
##  ~ normal(location = 0, scale = 2.5)
##
## Coefficients
##   Specified prior:
##     ~ normal(location = [0,0], scale = [2.5,2.5])
##   Adjusted prior:
##     ~ normal(location = [0,0], scale = [5.75,0.28])
## ------
## See help('prior_summary.stanreg') for more details
```

```
sig1 <- 2.5/p_sum$prior$adjusted_scale[1]
sig2 <- 2.5/p_sum$prior$adjusted_scale[2]

samples = as.matrix(mod)
summary(samples)
```

```
##   (Intercept)         priorfrac.b             age
##  Min.   :-75.84029   Min.   :-23.33856   Min.   :-1.114664
##  1st Qu.:-13.08709   1st Qu.: -3.97905   1st Qu.:-0.185819
```

```
##  Median : -0.11556   Median : -0.03228   Median : 0.002191
##  Mean   : -0.06643   Mean   : -0.03968   Mean   : 0.001114
##  3rd Qu.: 12.85214   3rd Qu.:  3.88793   3rd Qu.: 0.189567
##  Max.   : 77.40415   Max.   : 22.06350   Max.   : 1.053818
```

```
mean.b1 = mean(samples[, "priorfrac.b"])
median.b1 = median(samples[, "priorfrac.b"])
cred.int = hdi(samples[, "priorfrac.b"])
```

rstan identified a prior of $\mu_1 = 0$ and $\sigma_1 = 0.435$ for the priorfrac coefficient. rstan identified a prior of $\mu_2 = 0$ and $\sigma_2 = 8.99$ for the age coefficient.

5.3 Posterior

Now update the priors with the data in the OSTEO dataset. Plot the posterior distribution for prior history of fracture and state the median and 95% highest posterior density credible interval on the odds ratio scale. What has happened to the posterior distribution for β_1 now that age is included in the model (as compared to the first model you ran without age)? Is this consistent with what you expected would happen? Why or why not?

```
mod.post = update(mod, prior_PD = FALSE)
```

```
##
## SAMPLING FOR MODEL 'bernoulli' 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 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.104 seconds (Warm-up)
## Chain 1:                1.328 seconds (Sampling)
## Chain 1:                2.432 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 3.3e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.33 seconds.
```

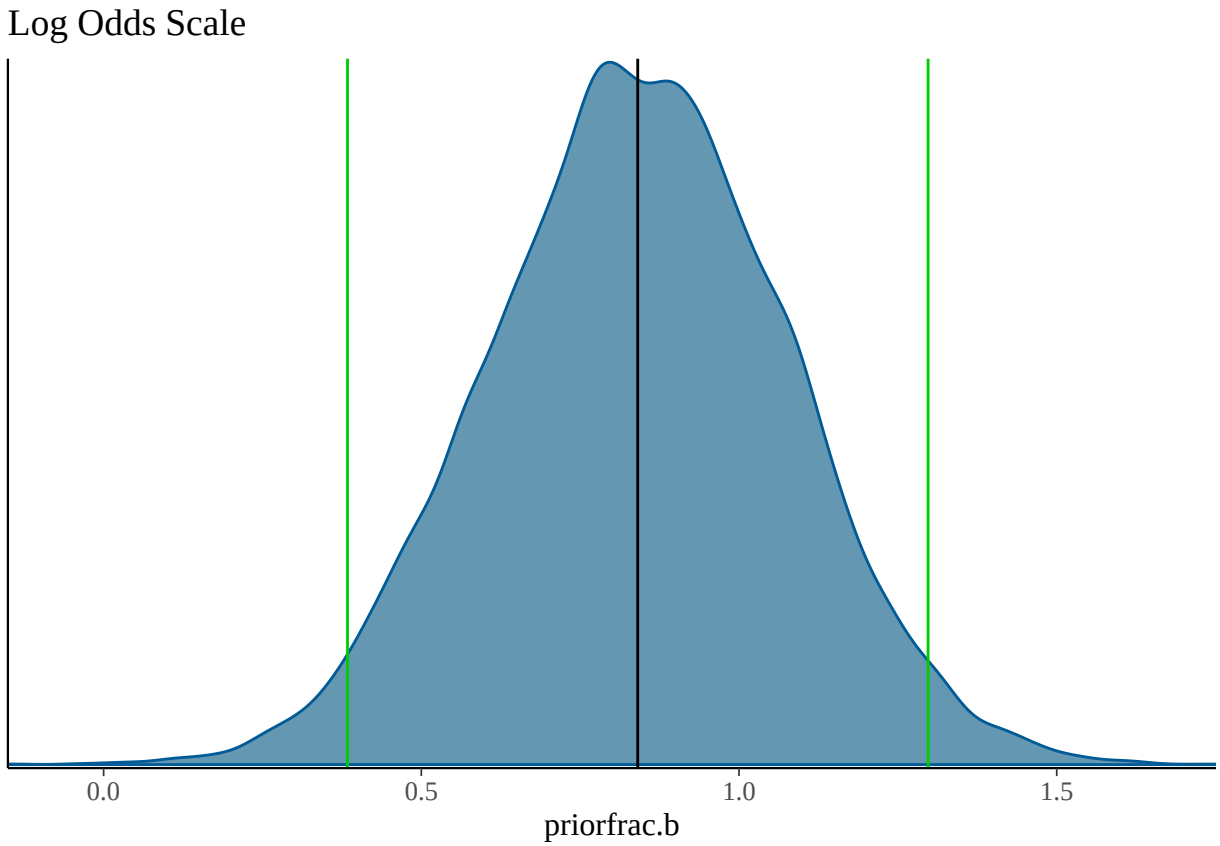
```
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.91 seconds (Warm-up)
## Chain 2:                0.935 seconds (Sampling)
## Chain 2:                1.845 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 3e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.3 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.767 seconds (Warm-up)
## Chain 3:                0.901 seconds (Sampling)
## Chain 3:                1.668 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 4e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.4 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
```

```
## Chain 4: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 4: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 4: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 4: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 4: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 4: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 4: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 4: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 4: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 4: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.799 seconds (Warm-up)
## Chain 4:                0.909 seconds (Sampling)
## Chain 4:                1.708 seconds (Total)
## Chain 4:
```

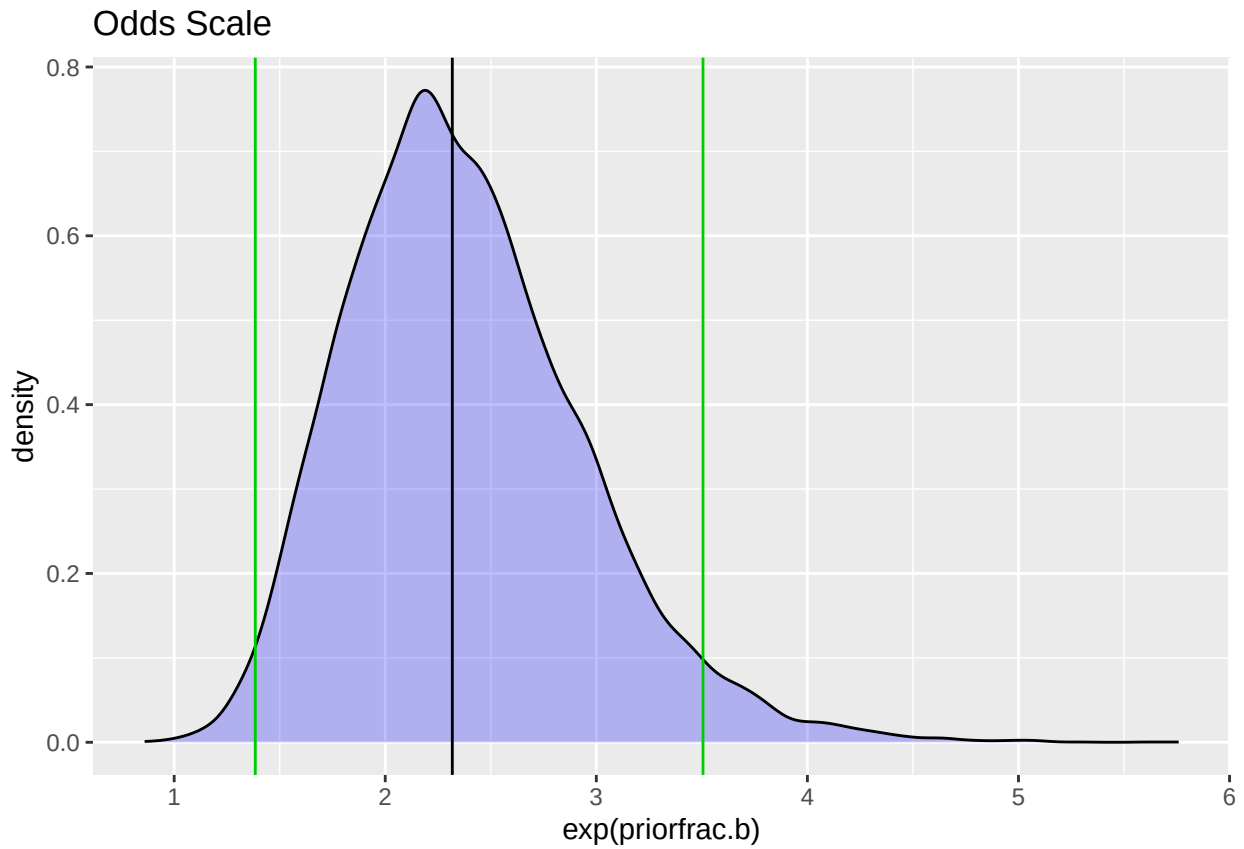
```
print(mod.post, digits= 3)
```

```
## stan_glm
##  family:       binomial [logit]
##  formula:      fracture.b ~ priorfrac.b + age
##  observations: 500
##  predictors:   3
## ------
##             Median MAD_SD
## (Intercept) -4.229  0.851
## priorfrac.b  0.841  0.234
## age          0.041  0.012
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
mcmc.chain = as.matrix(mod.post)
ci = hdi(mcmc.chain[, "priorfrac.b"])
med = median(mcmc.chain[, "priorfrac.b"])

ci.exp = hdi(exp(mcmc.chain[, "priorfrac.b"]))
med.exp = median(exp(mcmc.chain[,"priorfrac.b"]), na.rm = TRUE)
mcmc_dens(mod.post, pars = c("priorfrac.b")) + geom_vline(xintercept = med, color = "black") + geom_vli
```




```
mcd = as.data.frame(mcmc.chain)

ggplot(data = mcd) + geom_density(aes(x = exp(priorfrac.b)), fill = "blue", alpha = 0.25) + geom_vline(
```



The median of the age OR is 2.32 [95% CI: 1.38 - 3.51]. In words the odds ratio of fracture for participants as age increases is approximately 2.32. There is a 95% probability that the true median of the age OR is between 1.38 and 3.51, given our observed data. With age included in the model, the distribution of the coefficient associated with previous fracture history has shifted much lower. This makes sense, because if age is a big risk factor for fracture, then it would create a causal connection between both previous and 1-year post-diagnosis fracture rates. Accounting for this would be expected to reduce the apparent impact of previous fracture as an effect independent of age.

6 Question 5: Evaluating Interaction Between Age and a Prior History of Fracture

6.1 Write the Data Model

Write the data model. Use a similar format to what you used to write the data model in question 1. In your notation, use β_0, β_1, and β_2 as the intercept, prior history of fracture, and age respectively and explain what each parameter means. Remember, rstanarm gives model results without centering (even though the prior for the intercept is based on the other predictors being centered; see the next question).

The data model including the interaction term is as follows:

$ln[\frac{\pi_i}{1-\pi_i}] = \beta_0 + \beta_1 X_{i1} + \beta_2 Age_{i1} + \beta_3 Age_{i1} X_{i1}$

Where $ln[\frac{\pi_i}{1-\pi_i}]$ represents the log-odds of bone fracture in a given patient compared to non-osteoporosis patients at birth. β_0 is the log-odds in the reference group, while β_1 is the increase in risk associated with an osteoporosis diagnosis, and $beta_2$ is the increase associated with a one year increase in age (each holding

the other variable steady). The $beta_3$ coefficient is the increase in $beta_2$ given a patient has a prior history of fracture.

6.2 Prior Distributions for the Regression Parameters

Create an MCMC simulation of weakly informative priors for a model that includes prior history of fracture and age as covariates, as well as the interaction between these two factors. Write the priors that rstanarm has identified.

```
mod = stan_glm(fracture.b ~ priorfrac.b * age,
               data = ost,
               family = binomial,
               prior_intercept = normal(0, 2.5, autoscale = TRUE),
               prior = normal(0, 2.5, autoscale = TRUE),
               chains = 4,
               iter = 5000*2,
               seed = 1345,
               prior_PD = TRUE)
```

```
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.1 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.134 seconds (Warm-up)
## Chain 1:                0.144 seconds (Sampling)
## Chain 1:                0.278 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.2e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
```

```
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.128 seconds (Warm-up)
## Chain 2:                0.145 seconds (Sampling)
## Chain 2:                0.273 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.2e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.135 seconds (Warm-up)
## Chain 3:                0.173 seconds (Sampling)
## Chain 3:                0.308 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.1 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 4: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 4: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 4: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 4: Iteration: 4000 / 10000 [ 40%]  (Warmup)
```

```
## Chain 4: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 4: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 4: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 4: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 4: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 4: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.132 seconds (Warm-up)
## Chain 4:                0.147 seconds (Sampling)
## Chain 4:                0.279 seconds (Total)
## Chain 4:
```

```
p_sum <- prior_summary(mod)
print(p_sum)
```

```
## Priors for model 'mod'
## ------
## Intercept (after predictors centered)
##  ~ normal(location = 0, scale = 2.5)
##
## Coefficients
##   Specified prior:
##     ~ normal(location = [0,0,0], scale = [2.5,2.5,2.5])
##   Adjusted prior:
##     ~ normal(location = [0,0,0], scale = [5.752,0.278,0.078])
## ------
## See help('prior_summary.stanreg') for more details
```

```
sig1 <- 2.5/p_sum$prior$adjusted_scale[1]
sig2 <- 2.5/p_sum$prior$adjusted_scale[2]
sig3 <- 2.5/p_sum$prior$adjusted_scale[3]

samples = as.matrix(mod)
summary(samples)
```

```
##   (Intercept)         priorfrac.b             age
##  Min.   :-80.25936   Min.   :-22.907168   Min.   :-1.029662
##  1st Qu.:-13.31731   1st Qu.: -3.817677   1st Qu.:-0.189552
##  Median : -0.21462   Median : -0.001070   Median : 0.002507
##  Mean   : -0.06366   Mean   :  0.001141   Mean   : 0.001102
##  3rd Qu.: 13.15774   3rd Qu.:  3.787412   3rd Qu.: 0.193540
##  Max.   : 71.73810   Max.   : 24.488221   Max.   : 1.141835
##  priorfrac.b:age
##  Min.   :-0.3125335
##  1st Qu.:-0.0524056
##  Median : 0.0002804
##  Mean   : 0.0003981
##  3rd Qu.: 0.0526989
##  Max.   : 0.3092918
```

```
mean.b1 = mean(samples[, "priorfrac.b"])
median.b1 = median(samples[, "priorfrac.b"])
cred.int = hdi(samples[, "priorfrac.b"])
```

rstan identified a prior of $\mu_1 = 0$ and $\sigma_1 = 0.435$ for the priorfrac coefficient. rstan identified a prior of $\mu_2 = 0$ and $\sigma_2 = 8.99$ for the age coefficient. rstan also identified a prior of $\mu_2 = 0$ and $\sigma_2 = 32.1$ for the age coefficient.

6.3 Posterior

Now update the priors with the data in the OSTEO dataset. Plot the posterior distribution for the interaction term and interpret the median and 95% highest posterior density credible interval. What impact does older age have on the relationship between fracture risk in those with and without prior history?

Write a single sentence that summarizes the strength of evidence you have that age interacts with prior history of fracture to increase the risk of fracture 1 year after diagnosis with osteoporosis?

```
mod.post = update(mod, prior_PD = FALSE)
```

```
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.3 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 3.834 seconds (Warm-up)
## Chain 1:                4.668 seconds (Sampling)
## Chain 1:                8.502 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2.7e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.27 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
```

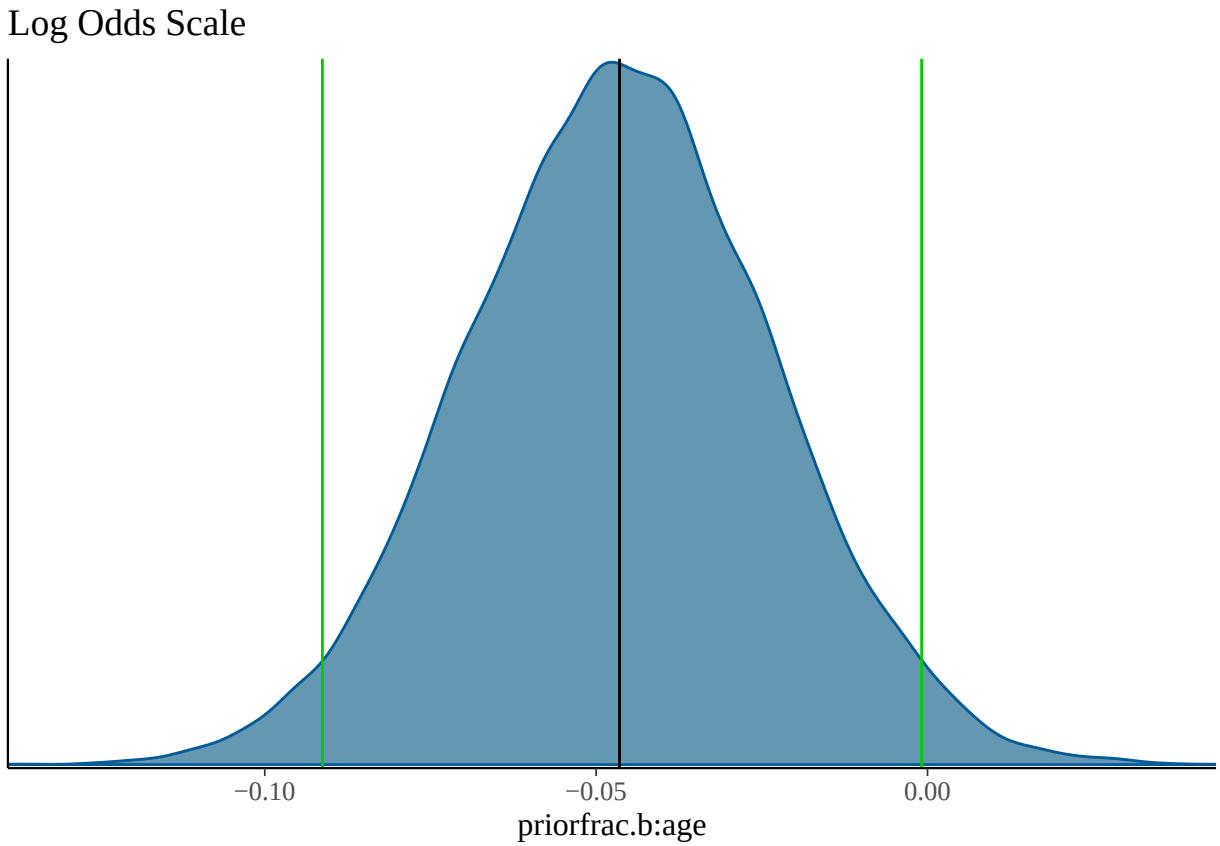
```
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 3.668 seconds (Warm-up)
## Chain 2:                3.938 seconds (Sampling)
## Chain 2:                7.606 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.7e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.27 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 3.567 seconds (Warm-up)
## Chain 3:                4.639 seconds (Sampling)
## Chain 3:                8.206 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'bernoulli' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 3.5e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.35 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 4: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 4: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 4: Iteration: 3000 / 10000 [ 30%]  (Warmup)
```

```
## Chain 4: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 4: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 4: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 4: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 4: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 4: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 4: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 4.435 seconds (Warm-up)
## Chain 4:                4.232 seconds (Sampling)
## Chain 4:                8.667 seconds (Total)
## Chain 4:
```

```
print(mod.post, digits= 3)
```

```
## stan_glm
##  family:       binomial [logit]
##  formula:      fracture.b ~ priorfrac.b * age
##  observations: 500
##  predictors:   4
## ------
##                 Median MAD_SD
## (Intercept)     -5.416  1.038
## priorfrac.b      4.176  1.655
## age              0.059  0.015
## priorfrac.b:age -0.046  0.023
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
mcmc.chain = as.matrix(mod.post)
ci = hdi(mcmc.chain[, "priorfrac.b:age"])
med = median(mcmc.chain[, "priorfrac.b:age"])

ci.exp = hdi(exp(mcmc.chain[, "priorfrac.b:age"]))
med.exp = median(exp(mcmc.chain[,"priorfrac.b:age"]), na.rm = TRUE)
mcmc_dens(mod.post, pars = c("priorfrac.b:age")) + geom_vline(xintercept = med, color = "black") + geom
```




```
## WE DON'T NEED ODDS SCALE PLOT

mcd = as.data.frame(mcmc.chain)
pprb <- length(which(mcd$`priorfrac.b:age`<0)) / nrow(mcd)

# ggplot(data = mcd) + geom_density(aes(x = exp(priorfrac.b)), fill = "blue", alpha = 0.25) + geom_vlin
```

The median of the interaction OR is 0.955 [95% CI: 0.912 - 0.998]. In words the odds ratio of fracture for participants of those with prior fractures compared to without prior fractures is approximately 0.955. There is a 95% probability that the true median of the interaction OR is between 0.912 and 0.998, given our observed data. With the interaction included in the model, the distribution of the coefficient associated with previous fracture history has not shifted significantly compared to the linear model with age involved. This makes sense, because if age does not have significant interaction with prior fracture history, then including the interaction term would not explain much of the residual variation and the priorfrac coefficient would not have to shift much.

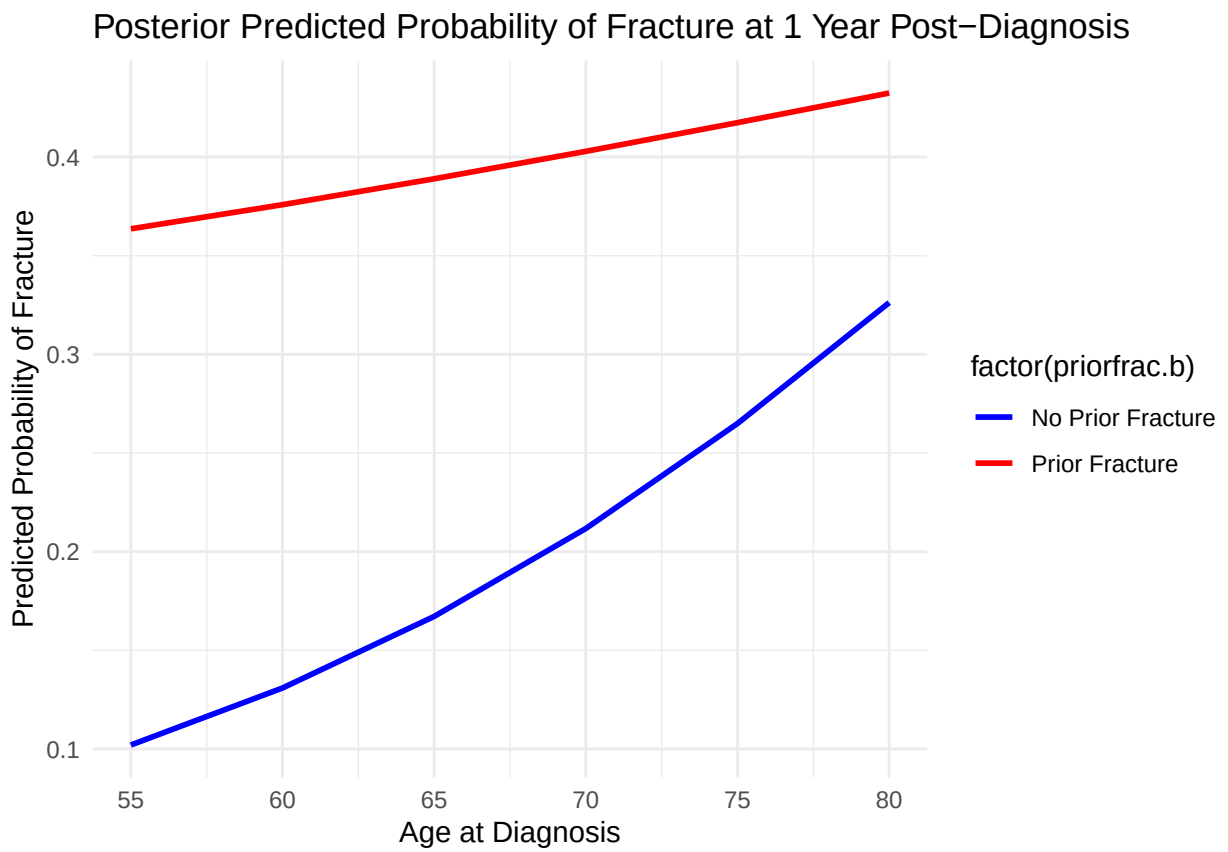
97.8% of the confidence interval is on one side of zero, so we can say that there is strong evidence of an interaction between age and prior history of fracture, even if the effect is small.

7 Question 6: Predicting Fracture Risk Based on Prior History and Age

Make a plot that shows the posterior predicted probability of fracture at 1 year post-diagnosis with osteoporosis by age at diagnosis (55 to 80 years, in increments of 5 years) and prior history of fracture. HINT:

Your plot should show the age at diagnosis on the X-axis and the posterior predicted probability on the Y axis. There should be two lines on the plot: one for women without a prior history of fracture and another for women who have a prior history of fracture.

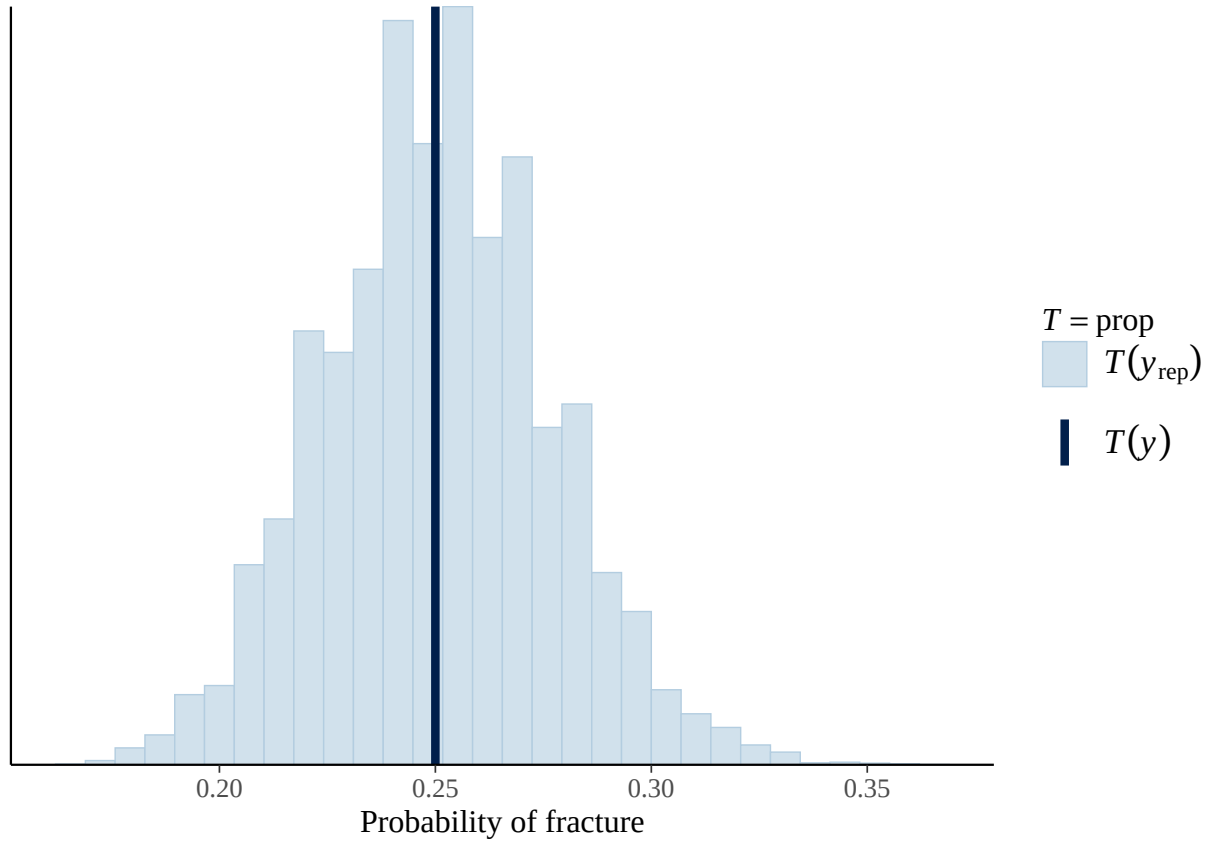
```
library(sjPlot)
age_grid = seq(55, 80, by = 5)

# Create data frame for predictions
pred_data <- expand.grid(age = age_grid, priorfrac.b = c(0, 1))

# Predict posterior probabilities of fracture for each age and prior history
pred_probs <- predict(mod.post, newdata = pred_data, type = "response",
                      allow_new_levels = TRUE, re.form = NA)

# Add predicted probabilities to pred_data
pred_data$predicted_probability <- pred_probs

# Plot posterior predicted probabilities
ggplot(pred_data, aes(x = age, y = predicted_probability, color = factor(priorfrac.b))) +
  geom_line(size = 1) +
  labs(title = "Posterior Predicted Probability of Fracture at 1 Year Post-Diagnosis",
       x = "Age at Diagnosis", y = "Predicted Probability of Fracture") +
  scale_color_manual(values = c("blue", "red"), labels = c("No Prior Fracture", "Prior Fracture")) +
  theme_minimal()
```



8 Question 7: Posterior Predictive Check

Perform a posterior predictive check examining how consistent the model is with the original data. Use 100 datasets. Explain how this posterior predictive check works and what you conclude from the results.

```
set.seed(1234)
prop = function(x){mean(x==1)}
pp_check(mod.post, nreps = 100, plotfun = "stat", "stat" ="prop" ) + xlab("Probability of fracture")
```



The posterior predictive check attempts to use our Bayesian model of the outcome to predict the outcome rate in our dataset via simulation. Because the observed fracture rate is roughly the median of the distribution of simulated fracture rates, we can conclude that our model is well-specified.

9 Question 8: Identifying Women at High Risk of Fracture

Suppose that in clinical practice an intervention would be warranted in women who have a posterior predicted probability of fracture at 1 year that exceeds 40%. First, use this cutoff for predicting a woman is at high risk to evaluate the model based on the data you have. State the sensitivity, specificity, overall accuracy, and positive predictive value and interpret each of these in light of the clinical problem of identifying women eligible for an intervention to prevent fracture within 1 year of diagnosis with osteoporosis. HINT: rstanarm doesn't give you the positive predictive value by default but you can use Bayes' theorem to find positive predictive value based on sensitivity, specificity and prevalence.

```
library(bayesrules)
cs = classification_summary(model = mod.post, data = ost, cutoff = 0.4)

## to get the positive predictive value...

sensitivity = cs$accuracy_rates[1,1]
specificity = cs$accuracy_rates[2,1]
overall_accuracy = cs$accuracy_rates[3,1]

prevalence = mean(pred_data$priorfrac.b == 1)

## calculate positive predictive value (PPV) using Bayes' theorem
ppv = sensitivity * prevalence / (sensitivity * prevalence + (1 - specificity) * (1 - prevalence))
```

The PPV, the probability that a woman with an over 40% chance of fracture will actually have a fracture, is 67.3%. The sensitivity, the probability that a woman who ends up getting a fracture will have over a 40% assessed risk of fracture, is 28%. The specificity, the probability that a woman who does not end up having a fracture has an assessed risk of under 40%, is 86.4%. The overall accuracy, our chance of correctly classifying a woman as being at risk of fracture or not, is 71.8%.

Then, use cross validation to evaluate the future performance of the model. Again, state the sensitivity, specificity, overall accuracy, and positive predictive value and interpret each of these.

```
cscv = classification_summary_cv(model = mod.post, data = ost, cutoff = 0.4, k = 5)

sensitivity = cscv$cv$sensitivity
specificity = cscv$cv$specificity
overall_accuracy = cscv$cv$overall_accuracy

ppv = sensitivity * prevalence / (sensitivity * prevalence + (1 - specificity) * (1 - prevalence))
```

Under cross-validation, the PPV, the probability that a woman with an over 40% chance of fracture will actually have a fracture, is 67.3%. The sensitivity, the probability that a woman who ends up getting a fracture will have over a 40% assessed risk of fracture, is 28.7%. The specificity, the probability that a woman who does not end up having a fracture has an assessed risk of under 40%, is 86.1%. The overall accuracy, our chance of correctly classifying a woman as being at risk of fracture or not, is 71.8%.