MA678 Homework 5

Yuchen Huang

2023-10-20

15.1 Poisson and negative binomial regression

The folder `RiskyBehavior` contains data from a randomized trial targeting couples at high risk of HIV infection. The intervention provided counseling sessions regarding practices that could reduce their likelihood of contracting HIV. Couples were randomized either to a control group, a group in which just the woman participated, or a group in which both members of the couple participated. One of the outcomes examined after three months was “number of unprotected sex acts.”

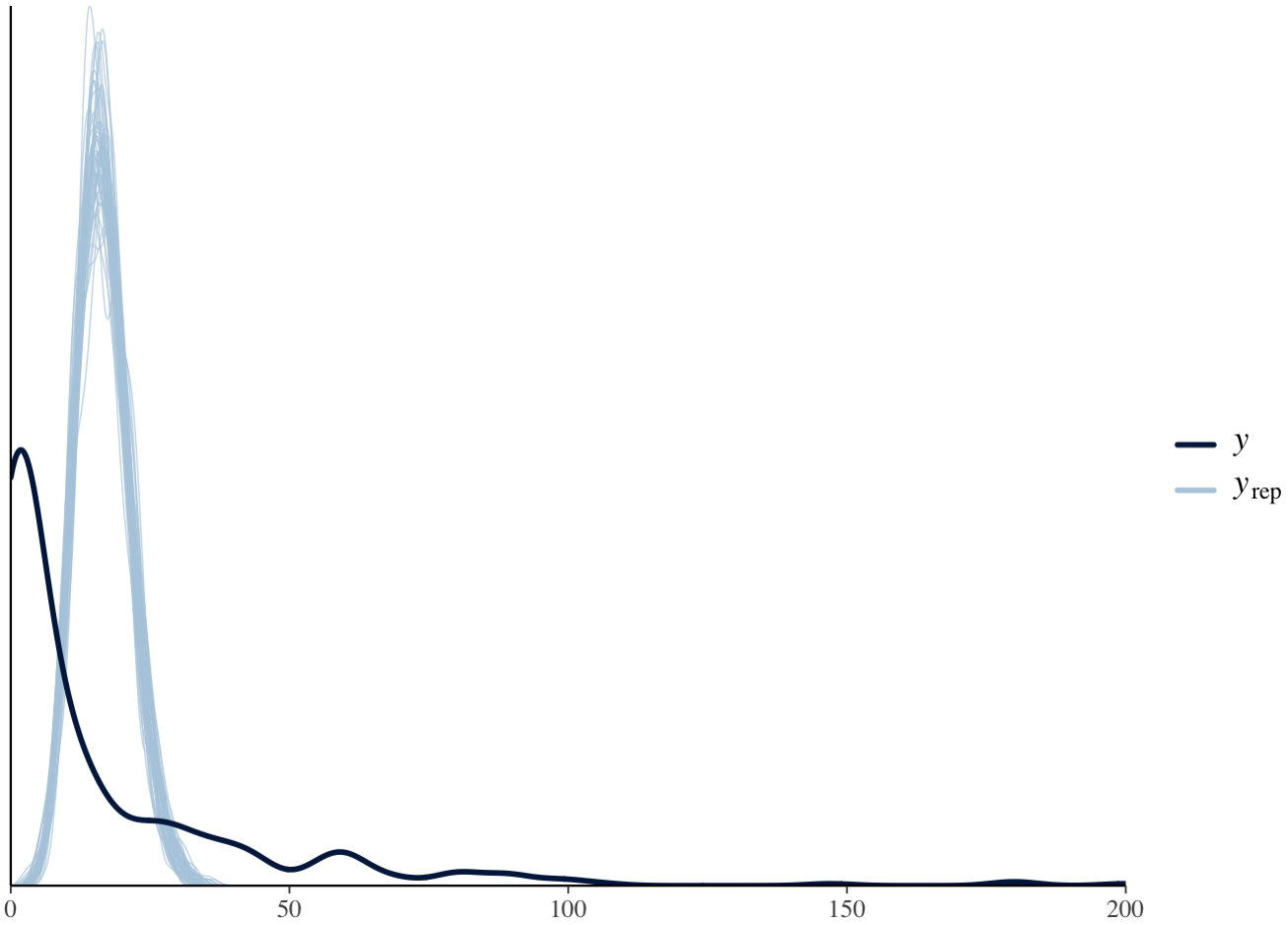
a)

Model this outcome as a function of treatment assignment using a Poisson regression. Does the model fit well? Is there evidence of overdispersion?

```
risky <- read.csv("risky.csv", header = T)
#create a new row called treatment. 3  = couple, 2 = women alone, 1 = education
risky <- risky |>
    mutate(treatment = ifelse(couples == 1, 3, ifelse(women_alone == 1, 2, 1))) |>
    mutate(fupacts = round(fupacts))
m <-  stan_glm(data = risky, fupacts~treatment, family = poisson(link="log"), refresh =
0)
summary(m)
```

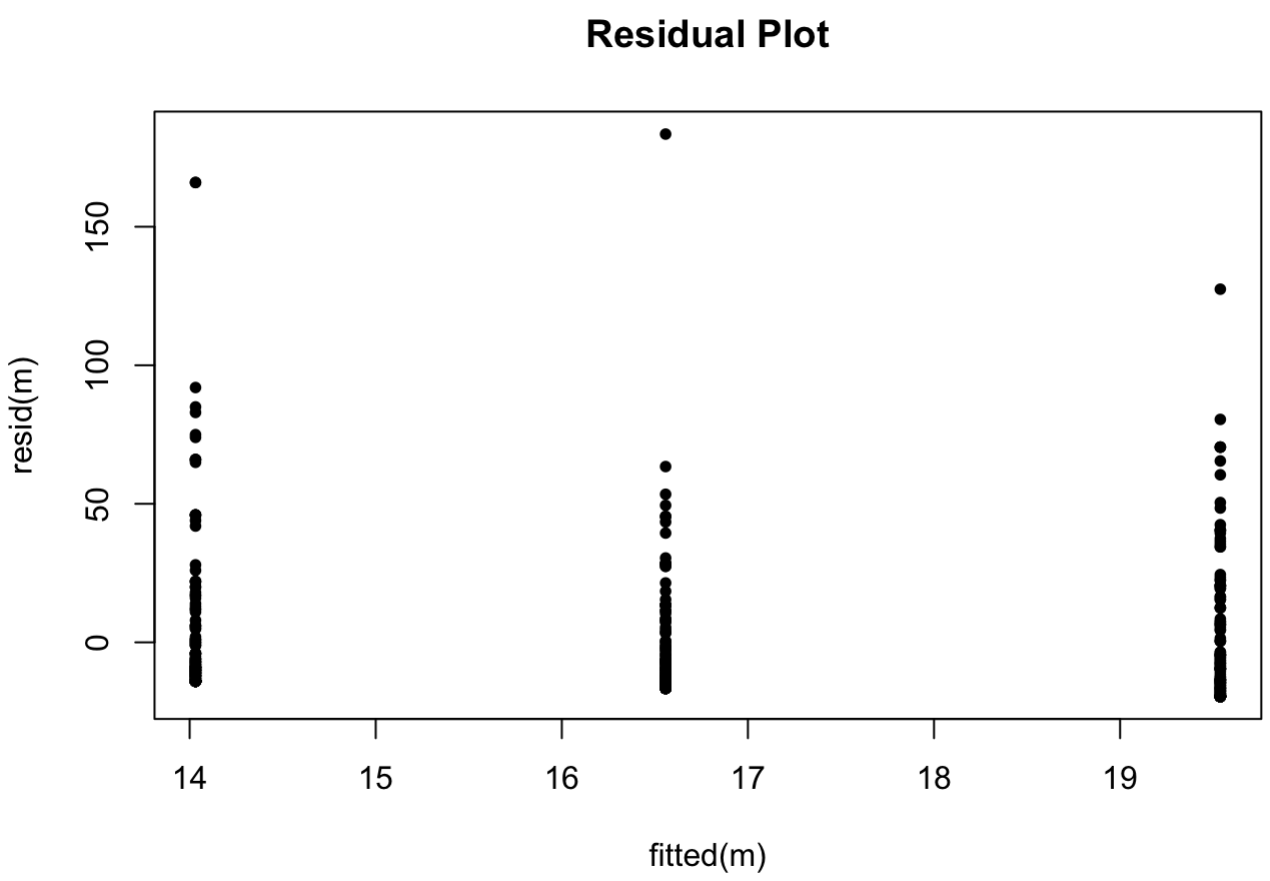
```
##
## Model Info:
##  function:     stan_glm
##  family:       poisson [log]
##  formula:      fupacts ~ treatment
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 434
##  predictors:   2
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept)  3.1    0.0  3.1   3.1   3.2
## treatment   -0.2    0.0 -0.2  -0.2  -0.1
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 16.5    0.3 16.1  16.5  16.9
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome v
ariable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  2696
## treatment     0.0  1.0  2503
## mean_PPD      0.0  1.0  2735
## log-posterior 0.0  1.0  1676
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of e
ffective sample size, and Rhat is the potential scale reduction factor on split chains
(at convergence Rhat=1).
```

```
pp_check(m)
```



From the posterior predictive check plot, we can see our model predicts less 0s than the real data， so there might be 0 inflation.

```
#Use reisidual plot to see the dispersion
plot(fitted(m), resid(m), pch = 20, main = "Residual Plot")
```



From the residual plot, we can see a huge amount of data points are above 0, which indicates that the it is overdispersion. Also, through the dispersion test, the result is 44 which is a large number indicating it's overdispersion.

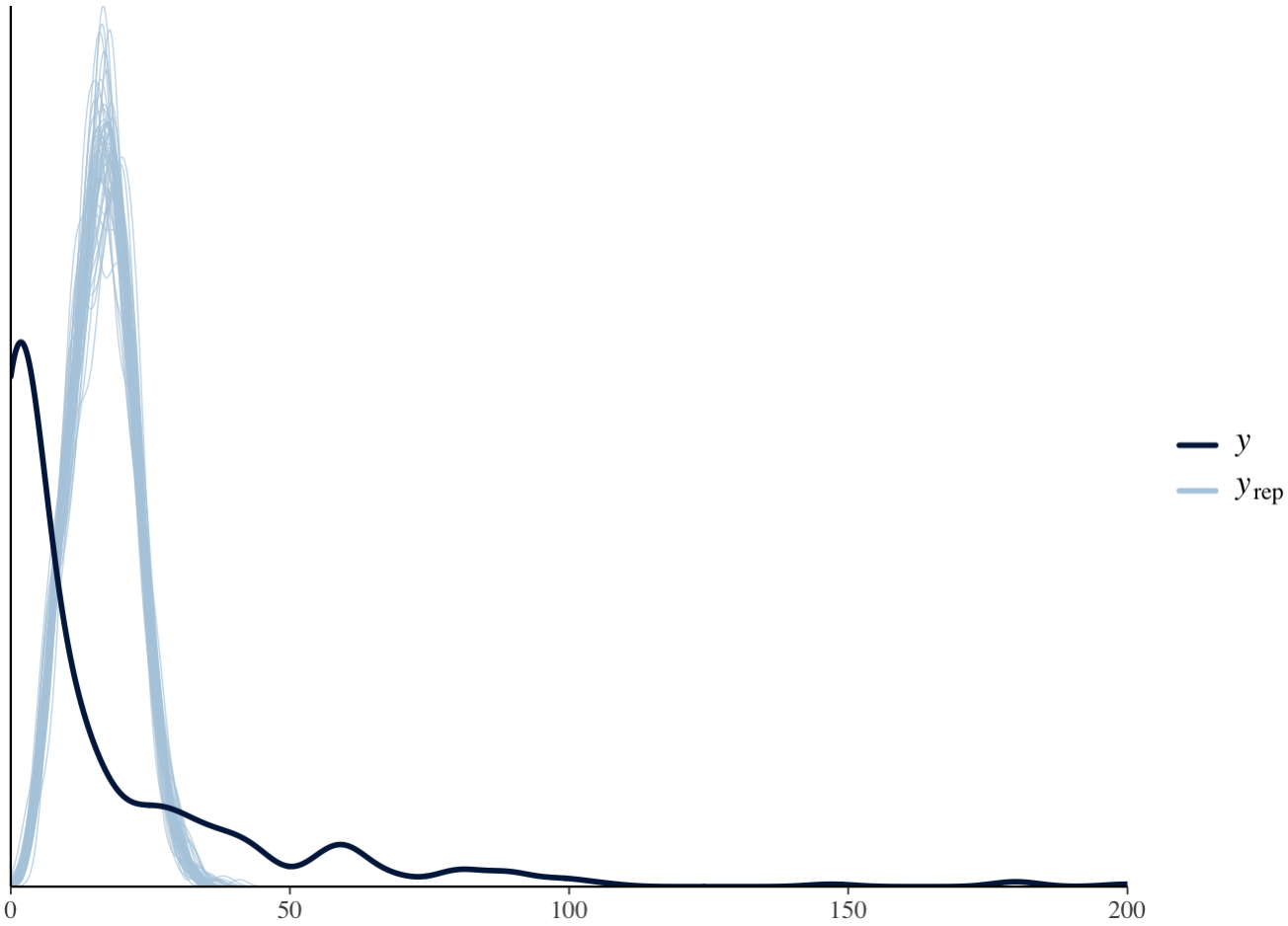
b)

Next extend the model to include pre-treatment measures of the outcome and the additional pre-treatment variables included in the dataset. Does the model fit well? Is there evidence of overdispersion?

```
#indicators are treatment and bs_hiv
m1 <- stan_glm(data = risky, fupacts~treatment+bs_hiv, family = poisson(link="log"), ref
resh = 0)
summary(m1)
```

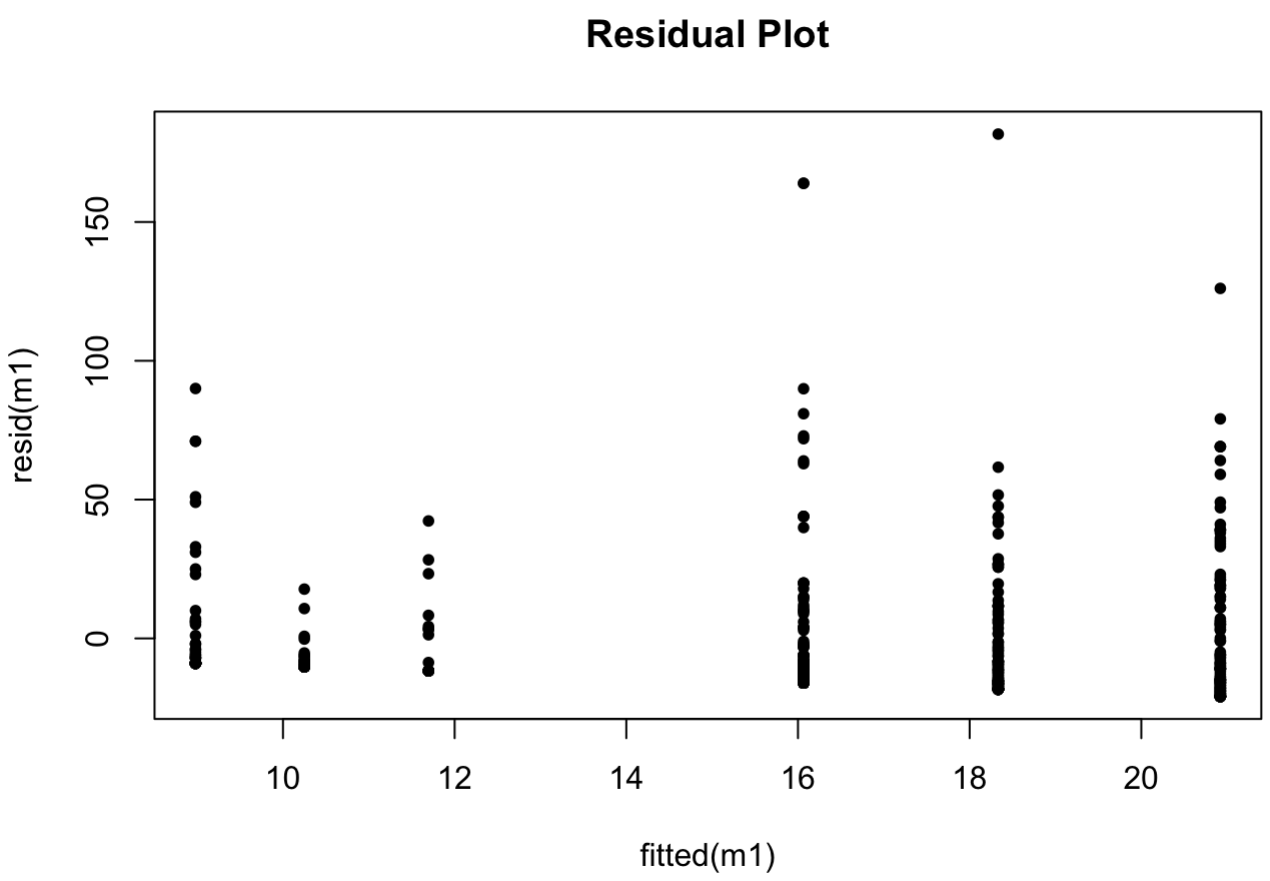
```
##
## Model Info:
##  function:     stan_glm
##  family:       poisson [log]
##  formula:      fupacts ~ treatment + bs_hiv
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 434
##  predictors:   3
##
## Estimates:
##                   mean   sd   10%   50%   90%
## (Intercept)      3.2    0.0  3.1   3.2   3.2
## treatment       -0.1    0.0 -0.2  -0.1  -0.1
## bs_hivpositive -0.6    0.0 -0.6  -0.6  -0.5
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 16.5    0.3 16.1  16.5  16.9
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome v
ariable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##                mcse Rhat n_eff
## (Intercept)    0.0  1.0  3777
## treatment      0.0  1.0  3577
## bs_hivpositive 0.0  1.0  3097
## mean_PPD       0.0  1.0  4003
## log-posterior  0.0  1.0  1512
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of e
ffective sample size, and Rhat is the potential scale reduction factor on split chains
(at convergence Rhat=1).
```

```
pp_check(m1)
```



We can still observe our model create less 0s than the actual data.

```
plot(fitted(m1), resid(m1), pch = 20, main = "Residual Plot")
```



From the dispersion test and residual plot, we can see it's still overdispersion since there are many points far away from 0.

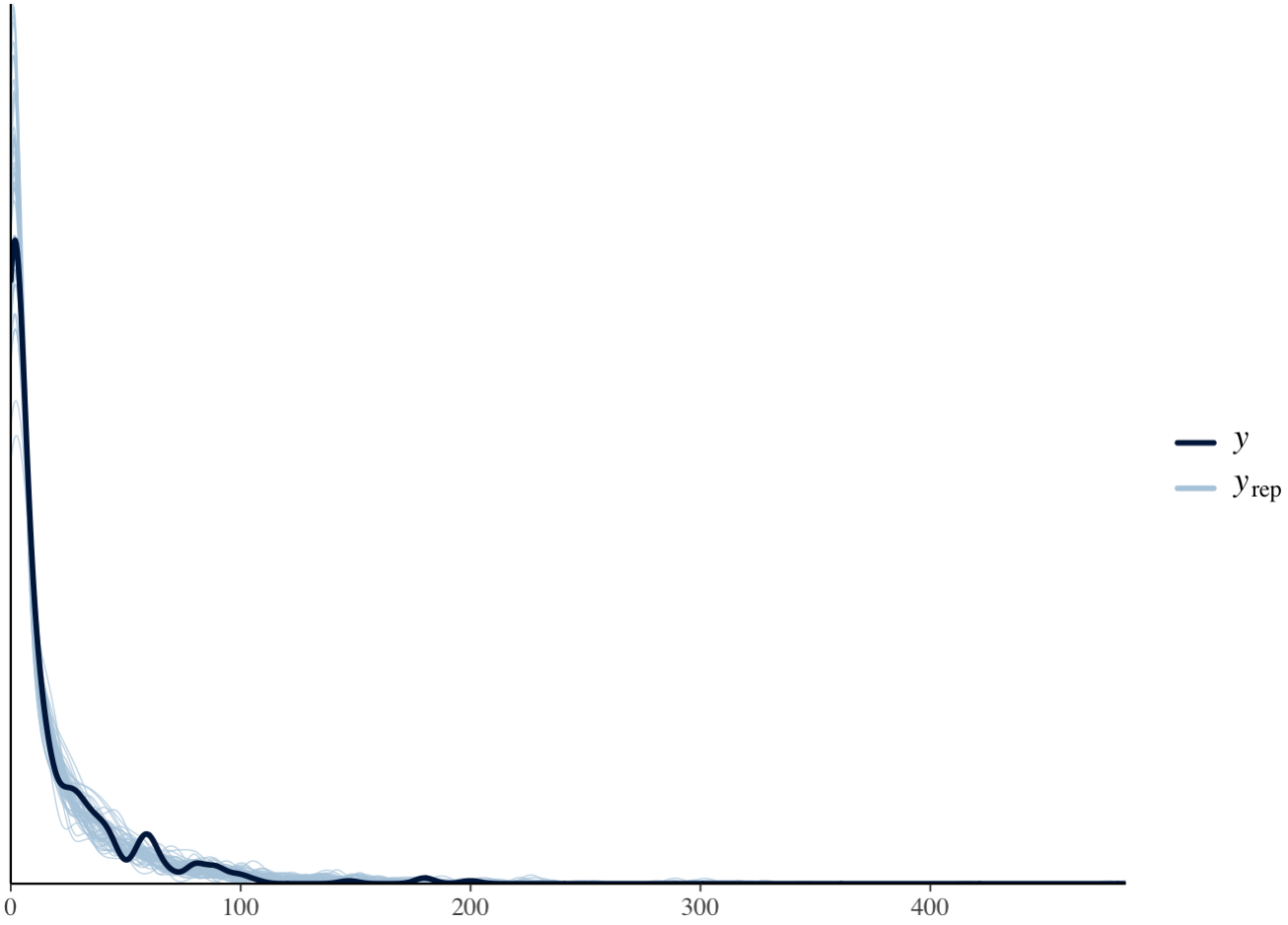
c)

Fit a negative binomial (overdispersed Poisson) model. What do you conclude regarding effectiveness of the intervention?

```
nb <- stan_glm(fupacts~treatment+bs_hiv, data = risky,
               family = neg_binomial_2(link = "log"), refresh = 0)
summary(nb)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       neg_binomial_2 [log]
##  formula:      fupacts ~ treatment + bs_hiv
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 434
##  predictors:   3
##
## Estimates:
##                          mean   sd   10%   50%   90%
## (Intercept)             3.1    0.2  2.8   3.1   3.4
## treatment              -0.1    0.1 -0.2  -0.1   0.0
## bs_hivpositive         -0.6    0.2 -0.8  -0.6  -0.3
## reciprocal_dispersion  0.3    0.0  0.3   0.3   0.4
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 16.7    2.0 14.2  16.6  19.3
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome v
ariable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##                        mcse Rhat n_eff
## (Intercept)            0.0  1.0  4625
## treatment              0.0  1.0  4334
## bs_hivpositive         0.0  1.0  3795
## reciprocal_dispersion 0.0  1.0  4031
## mean_PPD               0.0  1.0  4386
## log-posterior          0.0  1.0  1918
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of e
ffective sample size, and Rhat is the potential scale reduction factor on split chains
(at convergence Rhat=1).
```

```
pp_check(nb)
```




```
exp(coef(nb)[2])
```

```
## treatment
## 0.9213029
```

The model fits better, but the model contains more 0s than the real data.

The coefficient shows that if the treatment applied, unprotected sex will decrease by 8.2%.

d)

These data include responses from both men and women from the participating couples. Does this give you any concern with regard to our modeling assumptions?

I think the mode will fit better if the data specify the gender of the one who received education, with adding one more indicator we might be able to decrease the 0s in the predicting model.

15.3 Binomial regression

Redo the basketball shooting example on page 270, making some changes:

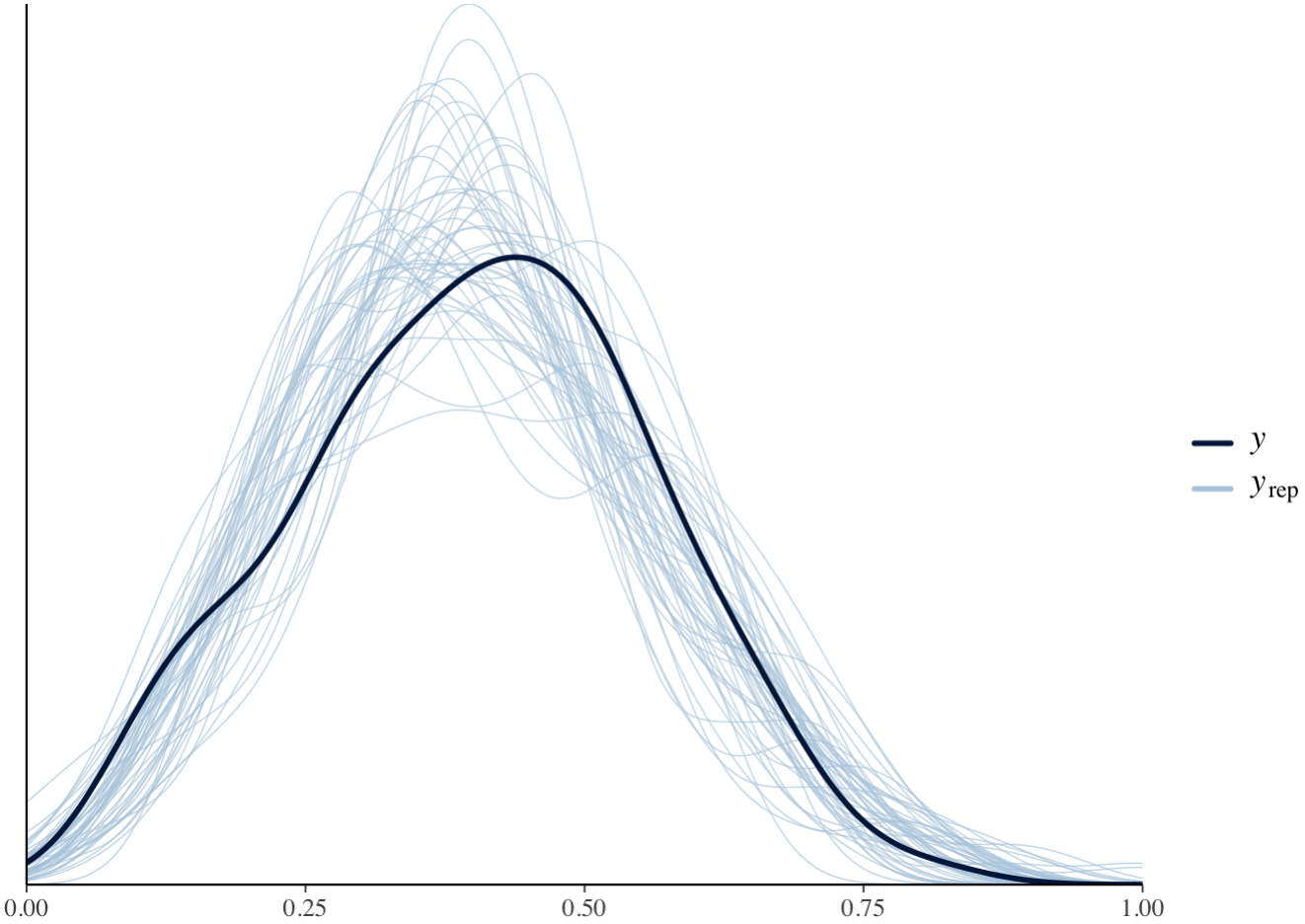
(a)

Instead of having each player shoot 20 times, let the number of shots per player vary, drawn from the uniform distribution between 10 and 30.

```
#
height <- rnorm(100, 72, 3)
p <- 0.4 + 0.1*(height-72)/3
n <- round(runif(100, 10, 30))
y <- rbinom(100, n, p )
bb <- data.frame(n = n, y = y, height = height)
m <- stan_glm(cbind(y,n-y) ~ height, family = binomial(link="logit"), data = bb,refresh
= 0)
summary(m)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       binomial [logit]
##  formula:      cbind(y, n - y) ~ height
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 100
##  predictors:   2
##
## Estimates:
##               mean   sd    10%   50%   90%
## (Intercept) -13.6    1.4 -15.4 -13.5 -11.8
## height        0.2    0.0   0.2   0.2   0.2
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 7.8    0.3  7.5   7.8   8.2
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome v
ariable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  2476
## height        0.0  1.0  2487
## mean_PPD      0.0  1.0  2922
## log-posterior 0.0  1.0  1708
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of e
ffective sample size, and Rhat is the potential scale reduction factor on split chains
(at convergence Rhat=1).
```

```
pp_check(m)
```



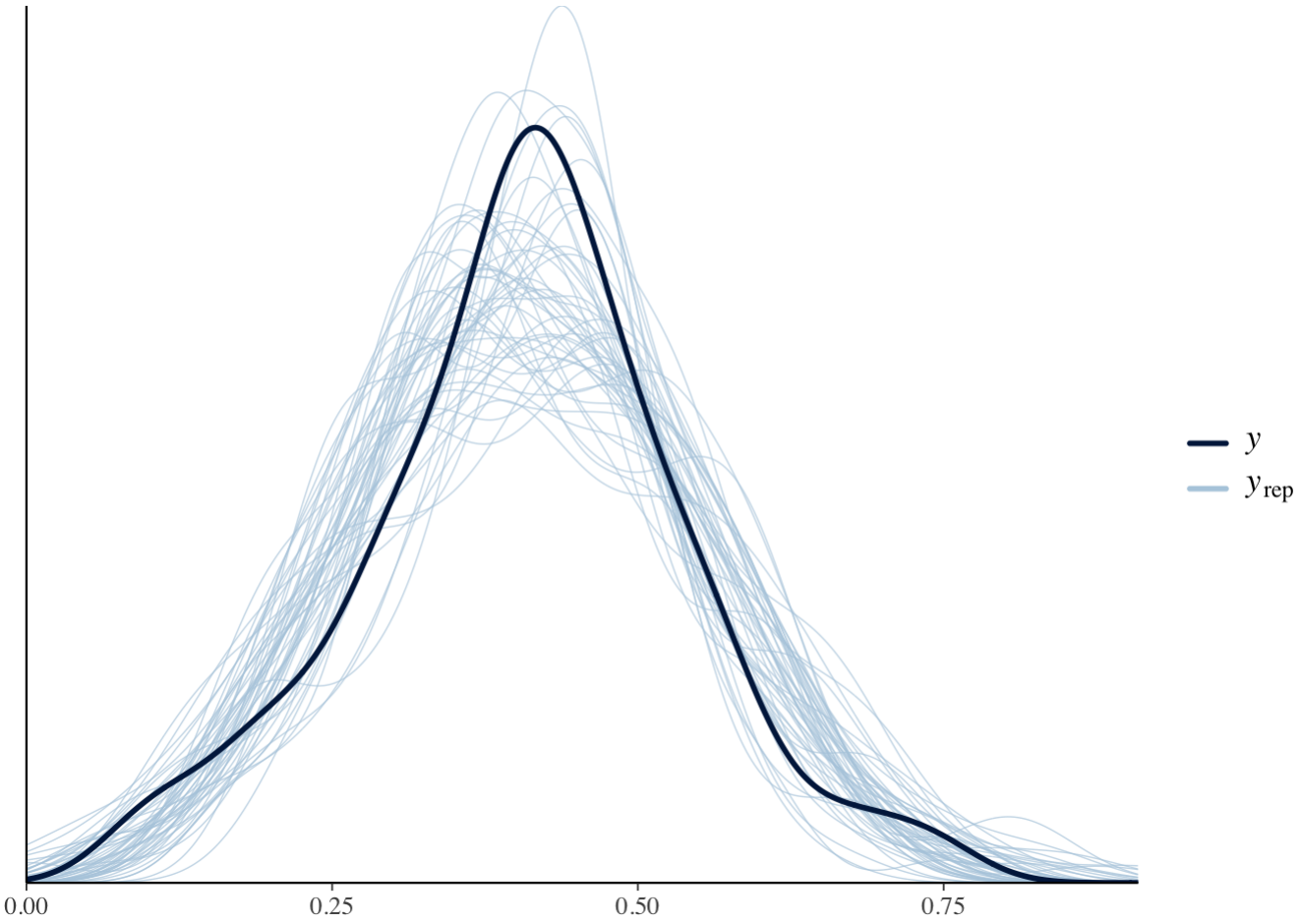
(b)

Instead of having the true probability of success be linear, have the true probability be a logistic function, set so that Pr(success) = 0.3 for a player who is 5’9” and 0.4 for a 6’ tall player.

```
p <- invlogit(rstanarm::logit(0.4) + (rstanarm::logit(0.4) - rstanarm::logit(0.3))/3 *(h
eight-72))
n <- round(runif(100,10,30), 0)
y <- rbinom(100, n, p)
new_bb <- data.frame(n,y,height)
m1 <- stan_glm(cbind(y,n-y) ~ height, family = binomial(link="logit"), data = new_bb, re
fresh = 0)
m1
```

```
## stan_glm
##  family:       binomial [logit]
##  formula:      cbind(y, n - y) ~ height
##  observations: 100
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) -8.6    1.2
## height       0.1    0.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
pp_check(m1)
```



15.7 Tobit model for mixed discrete/continuous data

Experimental data from the National Supported Work example are in the folder `Lalonde` . Use the treatment indicator and pre-treatment variables to predict post-treatment (1978) earnings using a Tobit model. Interpret the model coefficients.

```
lalonde <- read.dta("NSW_dw_obs.dta")
m <- vglm(log(re78+1) ~ treat + re75, tobit(Lower = 0, Upper=10), data = lalonde, refres
h = 0)
```

```
## Warning in eval(slot(family, "initialize")): replacing response values >
## 'Upper' by 'Upper'
```

```
## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 23
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12
```

```
## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12
```

```
## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 13
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12
```

```
## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 1
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12
```

```
## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 5
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12
```

```
## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12

## Warning in checkwz(wz, M = M, trace = trace, wzepsilon = control$wzepsilon): 2
## diagonal elements of the working weights variable 'wz' have been replaced by
## 1.819e-12
```

```
summary(m)
```

```
##
## Call:
## vglm(formula = log(re78 + 1) ~ treat + re75, family = tobit(Lower = 0,
##     Upper = 10), data = lalonde, refresh = 0)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept):1 4.647e+00  7.017e-02  66.223   <2e-16 ***
## (Intercept):2 1.589e+00  7.713e-03 206.072   <2e-16 ***
## treat         8.280e-01  3.795e-01   2.182   0.0291 *
## re75          3.624e-04  4.489e-06  80.726   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: mu, loglink(sd)
##
## Log-likelihood: -34510.66 on 37330 degrees of freedom
##
## Number of Fisher scoring iterations: 21
##
## No Hauck-Donner effect found in any of the estimates
```

As we don't consider the upper and lower bound, when all the predictors are 0, log(re78+1) would be 4.647. If we condier the bounds, when all the predictors are 0, log(re78+1) would be 1.589. Keep other indicator the same, as one unit increase of treat, the log(re78+1) will increase by 0.828. Keep otehr indicator the same, as one unit increase in re75, the log(78+1) will increase by 3.624e-04.

15.8 Robust linear regression using the t model

The folder `Congress` has the votes for the Democratic and Republican candidates in each U.S. congressional district in 1988, along with the parties' vote proportions in 1986 and an indicator for whether the incumbent was running for reelection in 1988. For your analysis, just use the elections that were contested by both parties in both years.

```
congress <- read.csv("congress.csv", header = T)
c1988 <- data.frame(
    vote=congress$v88_adj,
    pastvote=congress$v86_adj,
    inc=congress$inc88)
```

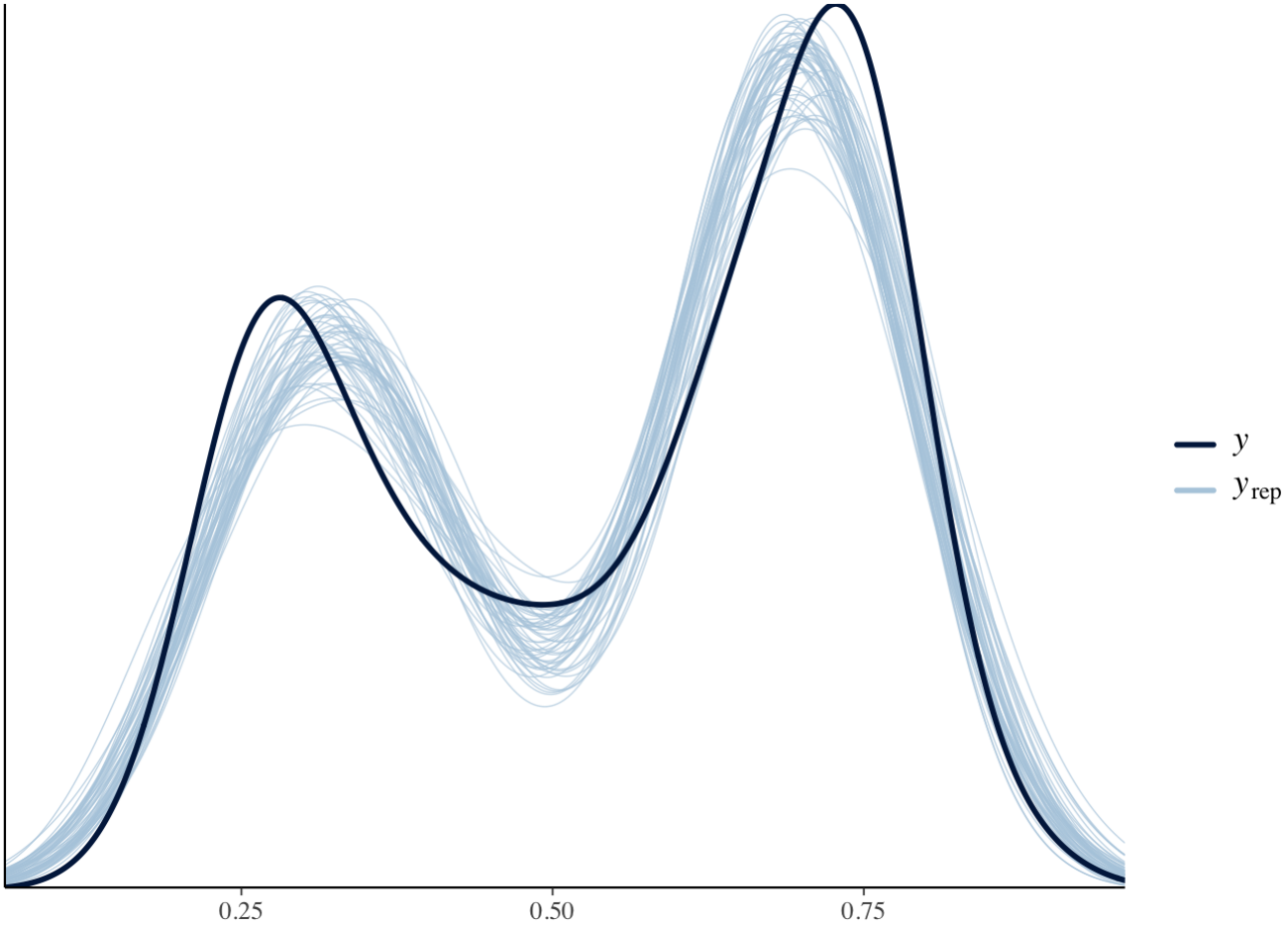
(a)

Fit a linear regression using `stan_glm` with the usual normal-distribution model for the errors predicting 1988 Democratic vote share from the other variables and assess model fit.

```
m8 <- stan_glm(vote~pastvote + inc, data = c1988, refresh = 0)
summary(m8)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      vote ~ pastvote + inc
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 435
##  predictors:   3
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) 0.2    0.0  0.2   0.2   0.3
## pastvote    0.5    0.0  0.5   0.5   0.6
## inc         0.1    0.0  0.1   0.1   0.1
## sigma       0.1    0.0  0.1   0.1   0.1
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 0.5    0.0  0.5   0.5   0.5
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome v
ariable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  1721
## pastvote      0.0  1.0  1664
## inc           0.0  1.0  1717
## sigma         0.0  1.0  2253
## mean_PPD      0.0  1.0  3918
## log-posterior 0.0  1.0  1783
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of e
ffective sample size, and Rhat is the potential scale reduction factor on split chains
(at convergence Rhat=1).
```

```
pp_check(m8)
```



(b)

Fit the same sort of model using the `brms` package with a t distribution, using the `brm` function with the student family. Again assess model fit.

```
mb <- brm(vote~pastvote + inc, data = c1988, family = "student", chains = 2, iter = 200
0, refresh = 0)
```

```
## Compiling Stan program...
```

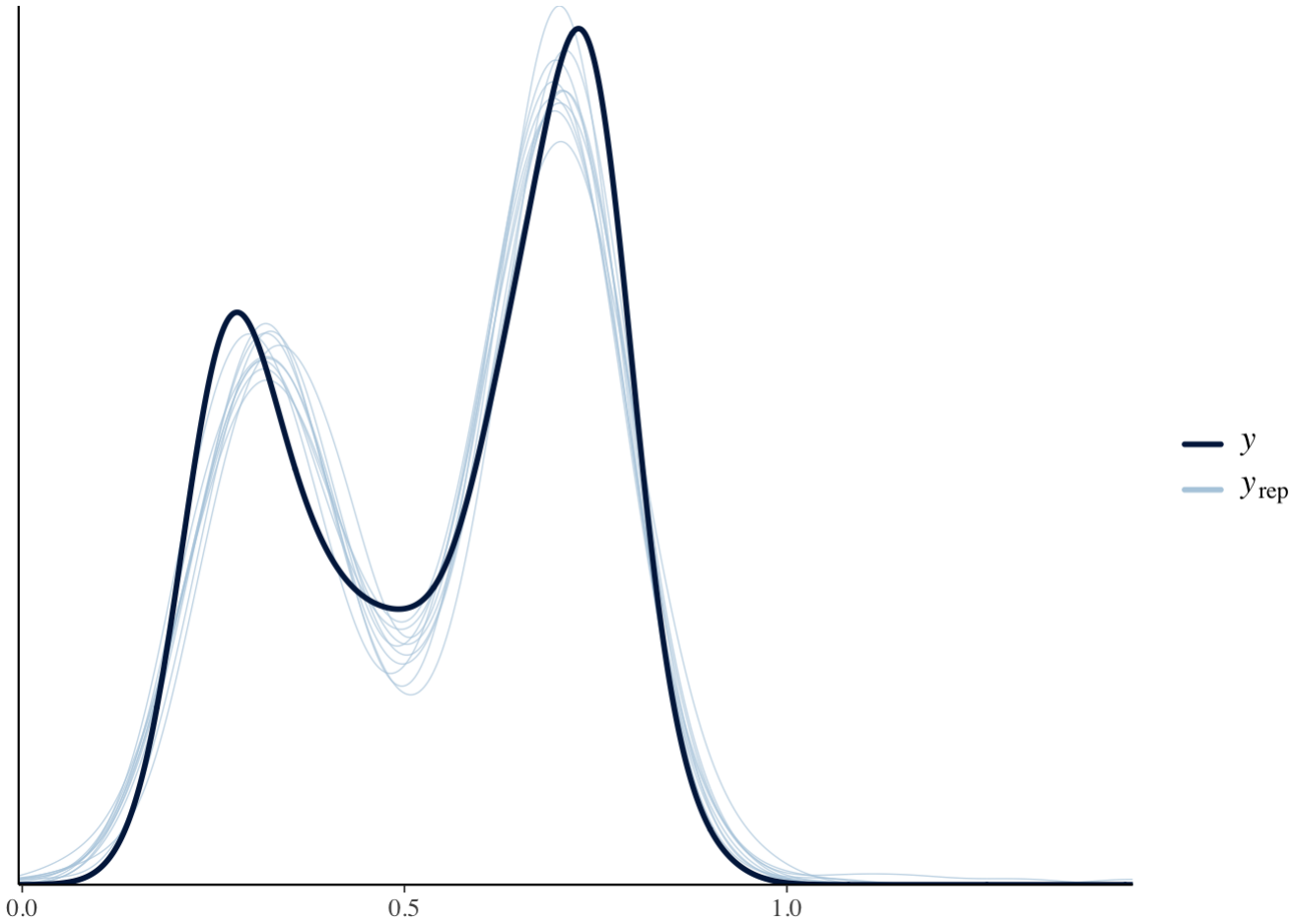
```
## Start sampling
```

```
summary(mb)
```

```
##  Family: student
##   Links: mu = identity; sigma = identity; nu = identity
## Formula: vote ~ pastvote + inc
##    Data: c1988 (Number of observations: 435)
##   Draws: 2 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 2000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.22      0.02     0.19     0.26 1.00      812      784
## pastvote      0.55      0.03     0.49     0.62 1.00      789      865
## inc           0.09      0.01     0.08     0.11 1.00      830      871
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.05      0.00     0.05     0.06 1.00      893     1040
## nu        6.23      2.41     3.36    12.52 1.00      877      869
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
pp_check(mb)
```

```
## Using 10 posterior draws for ppc type 'dens_overlay' by default.
```



(c)

Which model do you prefer?

I would prefer the student-t model since it fits the real data better than the normal distribution model.

15.9 Robust regression for binary data using the robit model

Use the same data as the previous example with the goal instead of predicting for each district whether it was won by the Democratic or Republican candidate.

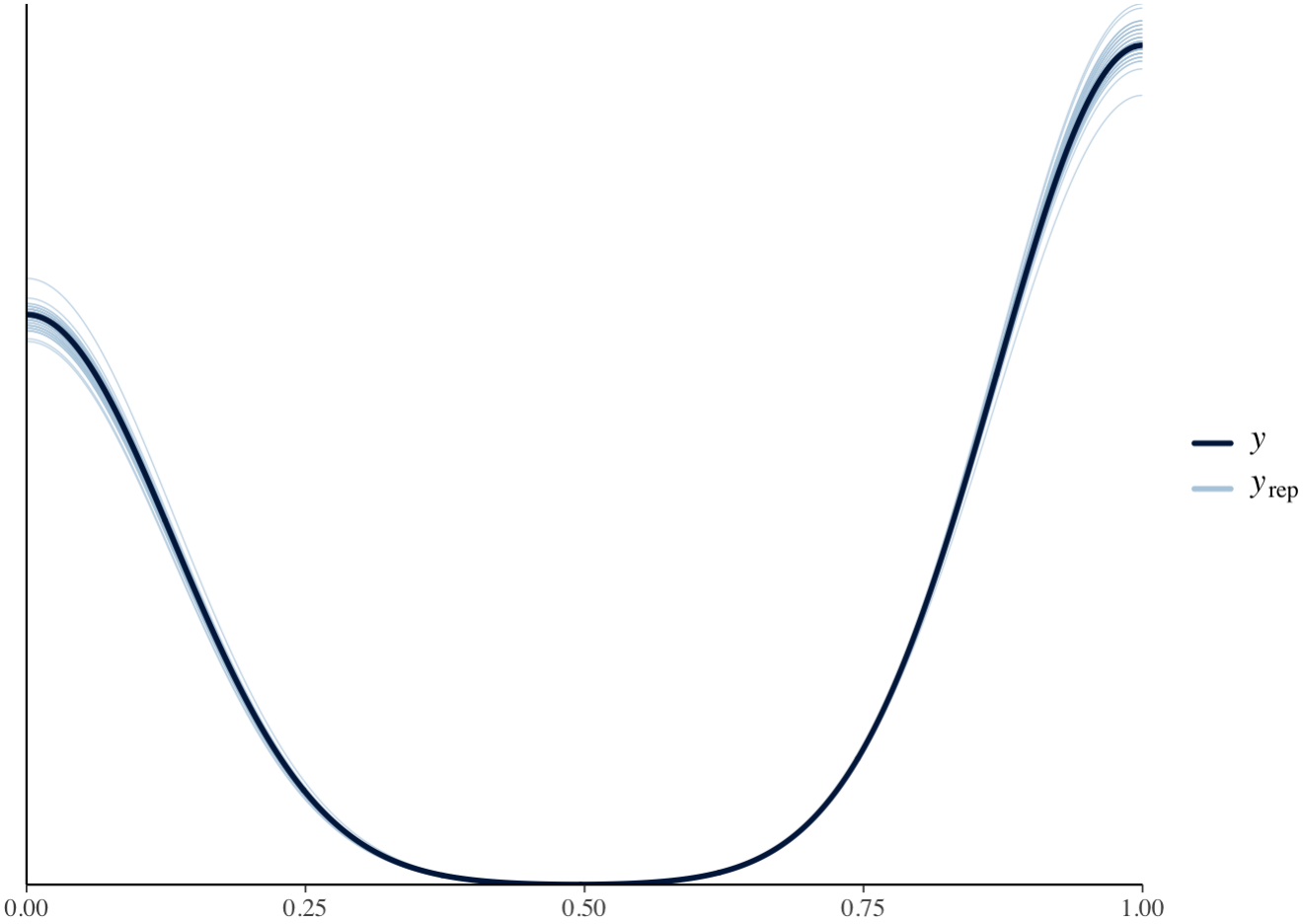
(a)

Fit a standard logistic or probit regression and assess model fit.

```
c1988 <- c1988|>
    mutate(p = ifelse(vote > 0.5, 1, 0))
mlog <- stan_glm(p~pastvote + inc, data = c1988, family = binomial(link = "logit"), refr
esh = 0)
summary(mlog)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       binomial [logit]
##  formula:      p ~ pastvote + inc
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 435
##  predictors:   3
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) -5.7    1.3 -7.4  -5.6  -4.0
## pastvote    11.5    2.6  8.3  11.4  14.8
## inc          2.7    0.5  2.2   2.7   3.4
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 0.6    0.0  0.6   0.6   0.6
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome v
ariable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  2558
## pastvote      0.1  1.0  2498
## inc           0.0  1.0  2233
## mean_PPD      0.0  1.0  3705
## log-posterior 0.0  1.0  1595
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of e
ffective sample size, and Rhat is the potential scale reduction factor on split chains
(at convergence Rhat=1).
```

```
pp_check(mlog)
```



The logit model fits well, and it's better than the previous two models.

(b)

Fit a robit regression and assess model fit.

```
mr <- brm(p~pastvote+inc, data=c1988, family=student(link="logit"))
```

```
## Compiling Stan program...
```

```
## Start sampling
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000147 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.47 seco
nds.
## 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: 33.352 seconds (Warm-up)
## Chain 1:                89.945 seconds (Sampling)
## Chain 1:                123.297 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 5.7e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.57 seco
nds.
## 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: 44.67 seconds (Warm-up)
## Chain 2:                30.704 seconds (Sampling)
## Chain 2:                75.374 seconds (Total)
## Chain 2:
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 5.7e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.57 seco
nds.
## 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: 0.893 seconds (Warm-up)
## Chain 3:                0.573 seconds (Sampling)
## Chain 3:                1.466 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0.000346 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 3.46 seco
nds.
## 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: 0.939 seconds (Warm-up)
## Chain 4:                1.163 seconds (Sampling)
## Chain 4:                2.102 seconds (Total)
## Chain 4:
```

```
## Warning: There were 2836 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: There were 1009 transitions after warmup that exceeded the maximum treedept
h. Increase max_treedepth above 10. See
## https://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

```
## Warning: There were 1 chains where the estimated Bayesian Fraction of Missing Informa
tion was low. See
## https://mc-stan.org/misc/warnings.html#bfmi-low
```

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

```
## Warning: The largest R-hat is 4.05, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#r-hat
```

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

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

```
summary(mr)
```

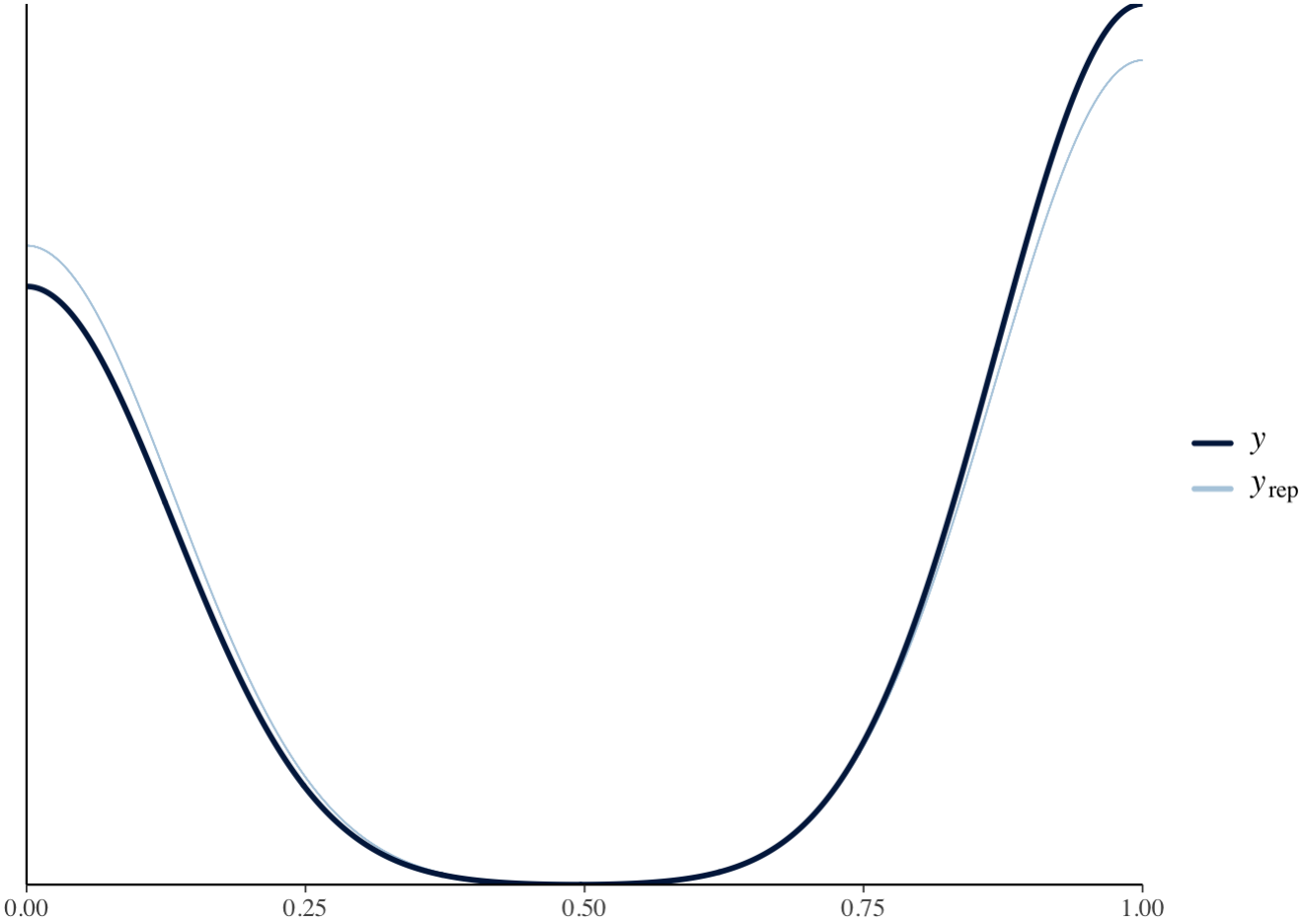
```
## Warning: Parts of the model have not converged (some Rhats are > 1.05). Be
## careful when analysing the results! We recommend running more iterations and/or
## setting stronger priors.
```

```
## Warning: There were 2836 divergent transitions after warmup. Increasing
## adapt_delta above 0.8 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
##  Family: student
##   Links: mu = logit; sigma = identity; nu = identity
## Formula: p ~ pastvote + inc
##    Data: c1988 (Number of observations: 435)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept  -246.56     97.92  -451.52  -160.58 3.11        5       11
## pastvote     23.46     10.36    12.61    34.43 3.64        4       15
## inc         373.58    196.23   197.93   839.48 3.60        4       11
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.00      0.00     0.00     0.00 1.15       22       NA
## nu        1.00      0.00     1.00     1.00 4.06        4       11
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
pp_check(mr)
```

```
## Using 10 posterior draws for ppc type 'dens_overlay' by default.
```



(c)

Which model do you prefer?

By observing the pp graph, I prefer the logit model.

15.14 Model checking for count data

The folder `RiskyBehavior` contains data from a study of behavior of couples at risk for HIV; see Exercise 15.1.

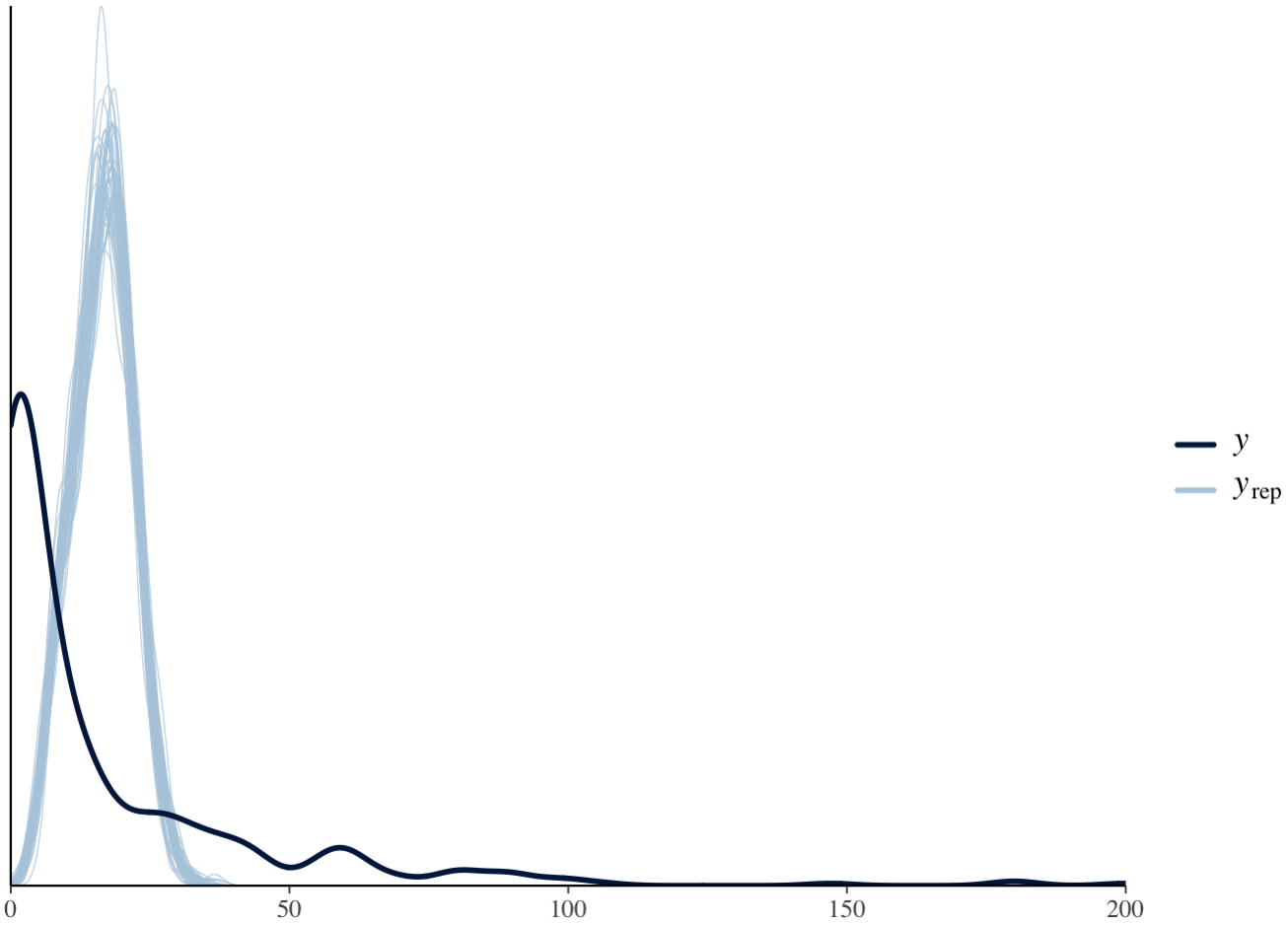
(a)

Fit a Poisson regression predicting number of unprotected sex acts from baseline HIV status. Perform predictive simulation to generate 1000 datasets and record the percentage of observations that are equal to 0 and the percentage that are greater than 10 (the third quartile in the observed data) for each. Compare these to the observed value in the original data.

```
mp <- stan_glm(fupacts~bs_hiv, data = risky, family = poisson(link = "log"), refresh =
0)
summary(mp)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       poisson [log]
##  formula:      fupacts ~ bs_hiv
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 434
##  predictors:   2
##
## Estimates:
##                  mean   sd   10%   50%   90%
## (Intercept)     2.9    0.0  2.9   2.9   2.9
## bs_hivpositive -0.6    0.0 -0.7  -0.6  -0.6
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 16.5    0.3 16.1  16.5  16.8
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome v
ariable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##                mcse Rhat n_eff
## (Intercept)    0.0  1.0  2381
## bs_hivpositive 0.0  1.0  2323
## mean_PPD       0.0  1.0  2720
## log-posterior  0.0  1.0  1669
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of e
ffective sample size, and Rhat is the potential scale reduction factor on split chains
(at convergence Rhat=1).
```

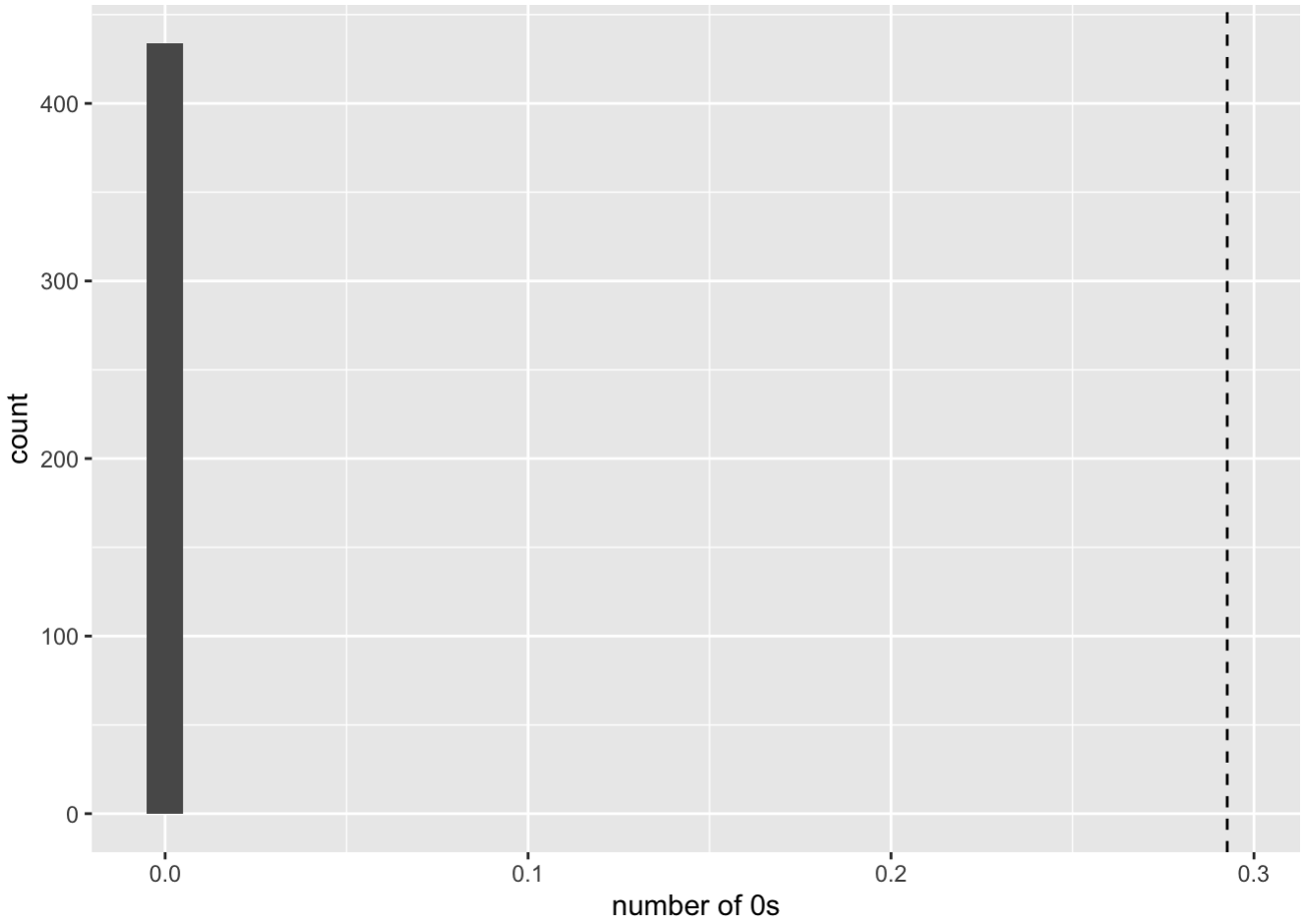
```
pp_check(mp)
```




```
pp <- posterior_predict(mp, draw = 1000)
obs_p <- data.frame(
    num0 = apply(pp, 2, function(x) mean(x==0)),
    num10 = apply(pp, 2, function(x) mean(x>=10)))

ggplot(data = obs_p, aes(x = num0))+
    geom_histogram(aes(x = num0))+
    geom_vline(aes(xintercept=mean(risky$fupacts == 0)), linetype = "dashed")+
    labs(x = "number of 0s")
```

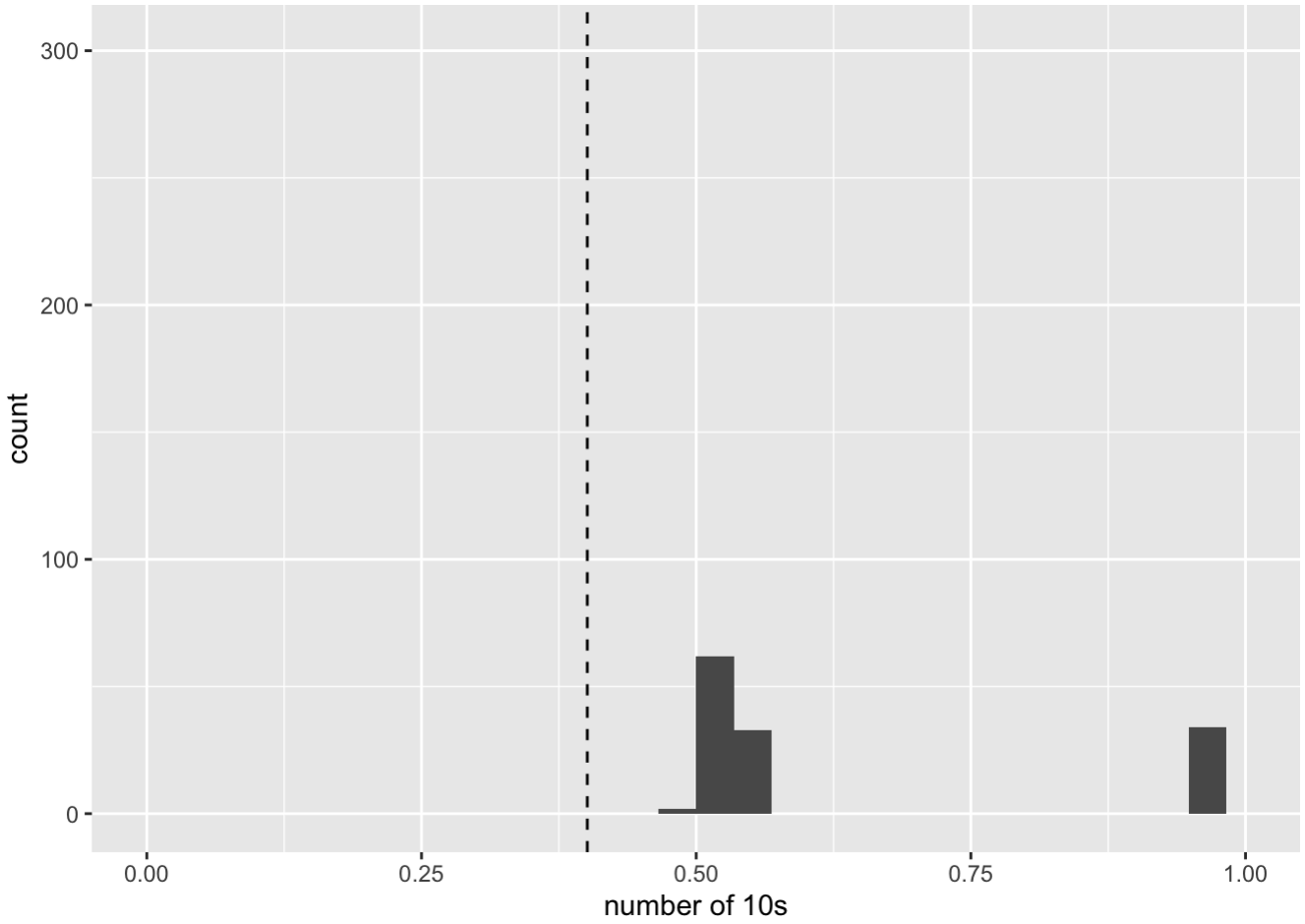
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(data = obs_p, aes(x = num10))+
    geom_histogram(aes(x = num10))+
    geom_vline(aes(xintercept=mean(risky$fupacts >= 10)), linetype = "dashed")+
    labs(x = "number of 10s")+
    xlim(c(0,1))
```

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

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



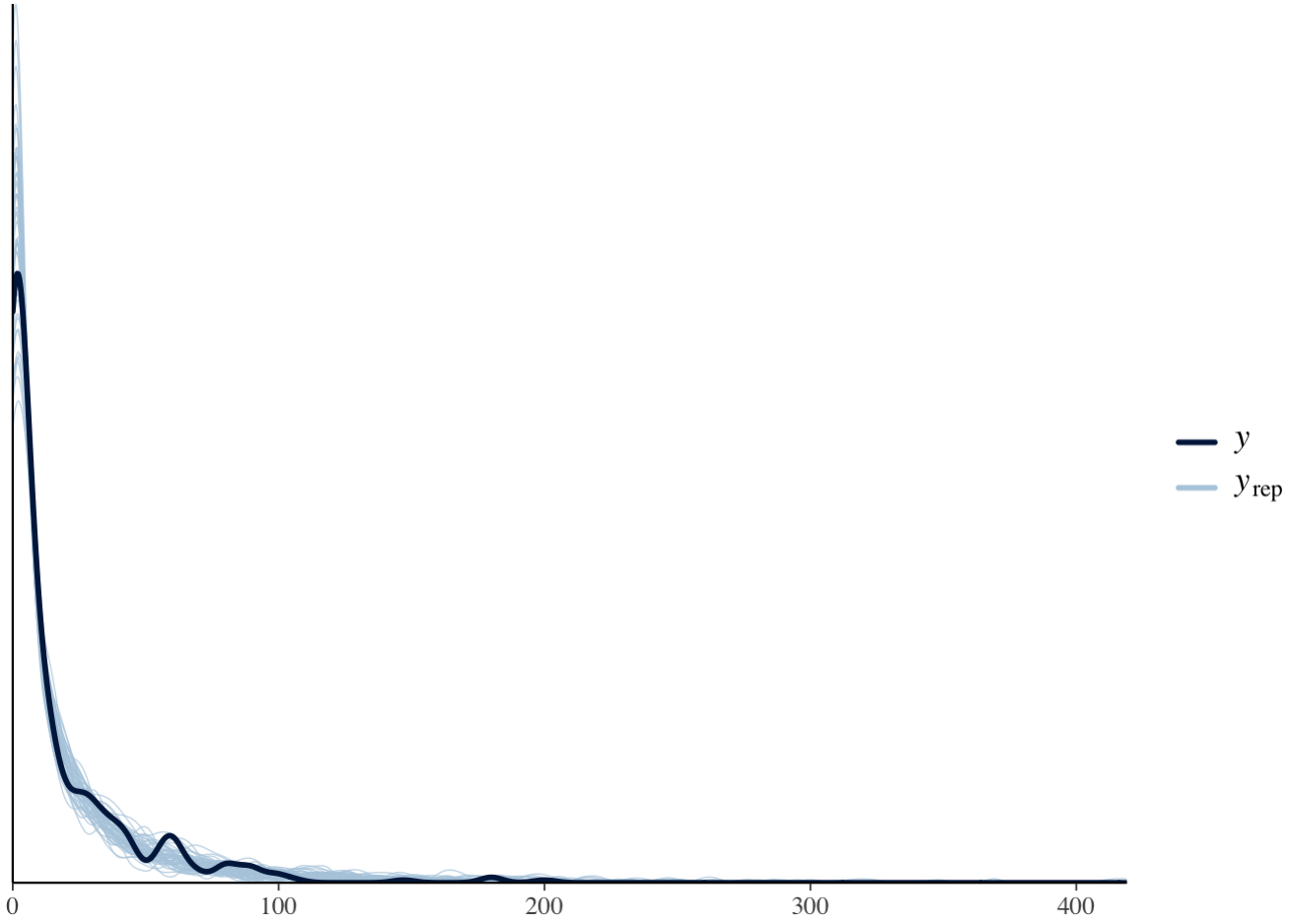
(b)

Repeat (a) using a negative binomial (overdispersed Poisson) regression.

```
mnb <- stan_glm(fupacts~bs_hiv, data = risky, family = neg_binomial_2(link = "log"), ref
resh = 0)
summary(mnb)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       neg_binomial_2 [log]
##  formula:      fupacts ~ bs_hiv
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 434
##  predictors:   2
##
## Estimates:
##                          mean   sd   10%   50%   90%
## (Intercept)             2.9    0.1  2.8   2.9   3.0
## bs_hivpositive         -0.6    0.2 -0.9  -0.6  -0.3
## reciprocal_dispersion  0.3    0.0  0.3   0.3   0.4
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 16.6    2.1 14.1  16.5  19.3
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome v
ariable (for details see help('summary.stanreg')).
##
## MCMC diagnostics
##                       mcse Rhat n_eff
## (Intercept)           0.0  1.0  3720
## bs_hivpositive        0.0  1.0  3685
## reciprocal_dispersion 0.0  1.0  3717
## mean_PPD              0.0  1.0  3931
## log-posterior         0.0  1.0  1798
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of e
ffective sample size, and Rhat is the potential scale reduction factor on split chains
(at convergence Rhat=1).
```

```
pp_check(mnb)
```



(c)

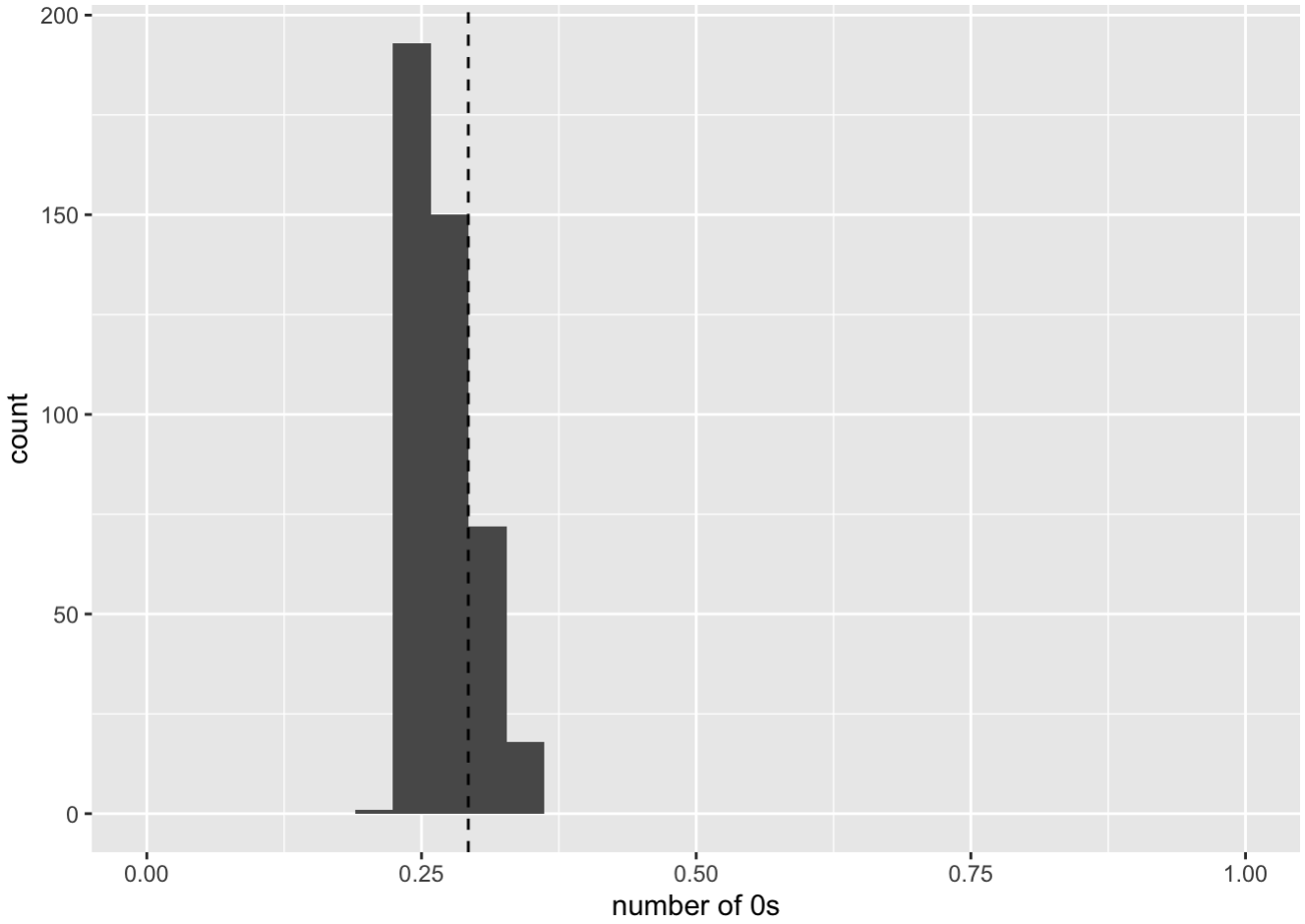
Repeat (b), also including ethnicity and baseline number of unprotected sex acts as inputs.

```
ppc <- posterior_predict(mnb, draw = 1000)
obs_nb <- data.frame(
    num0 = apply(ppc, 2, function(x) mean(x==0)),
    num10 = apply(ppc, 2, function(x) mean(x>=10)))

ggplot(data = obs_nb, aes(x = num0))+
    geom_histogram(aes(x = num0))+
    geom_vline(aes(xintercept=mean(risky$fupacts == 0)), linetype = "dashed")+
    labs(x = "number of 0s")+
    xlim(c(0,1))
```

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

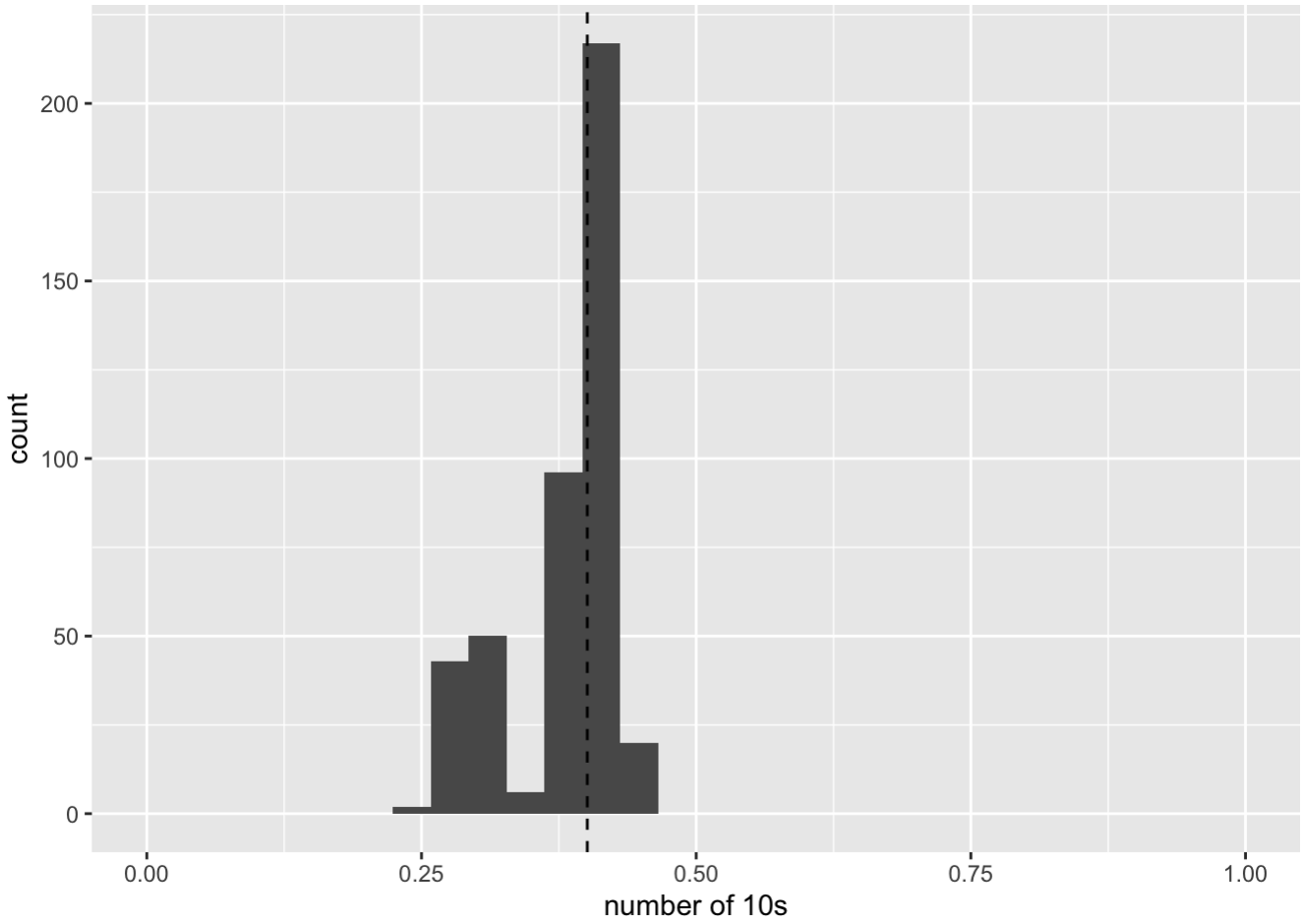
```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```




```
ggplot(data = obs_nb, aes(x = num10))+
    geom_histogram(aes(x = num10))+
    geom_vline(aes(xintercept=mean(risky$fupacts >= 10)), linetype = "dashed")+
    labs(x = "number of 10s")+
    xlim(c(0,1))
```

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

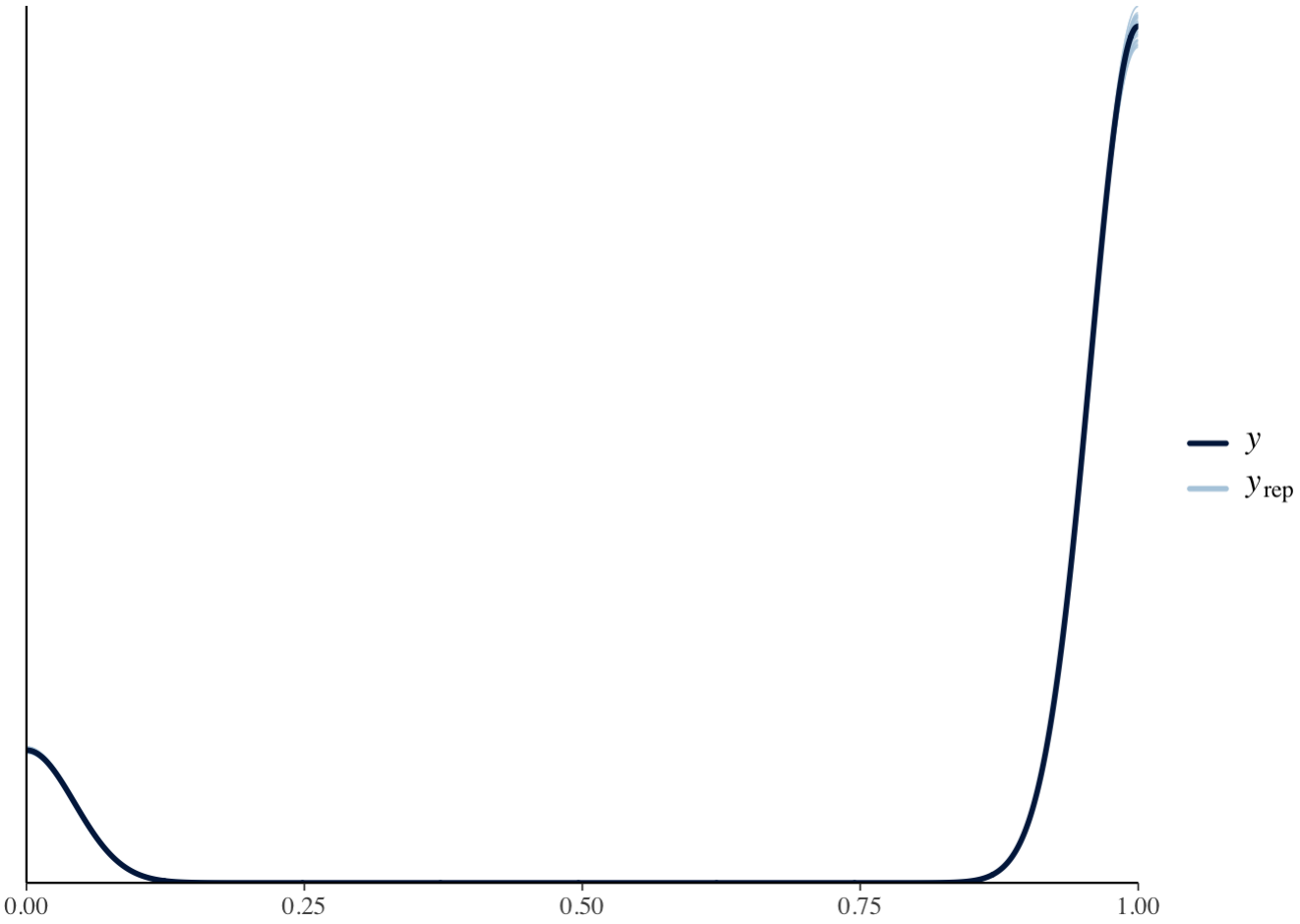
```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



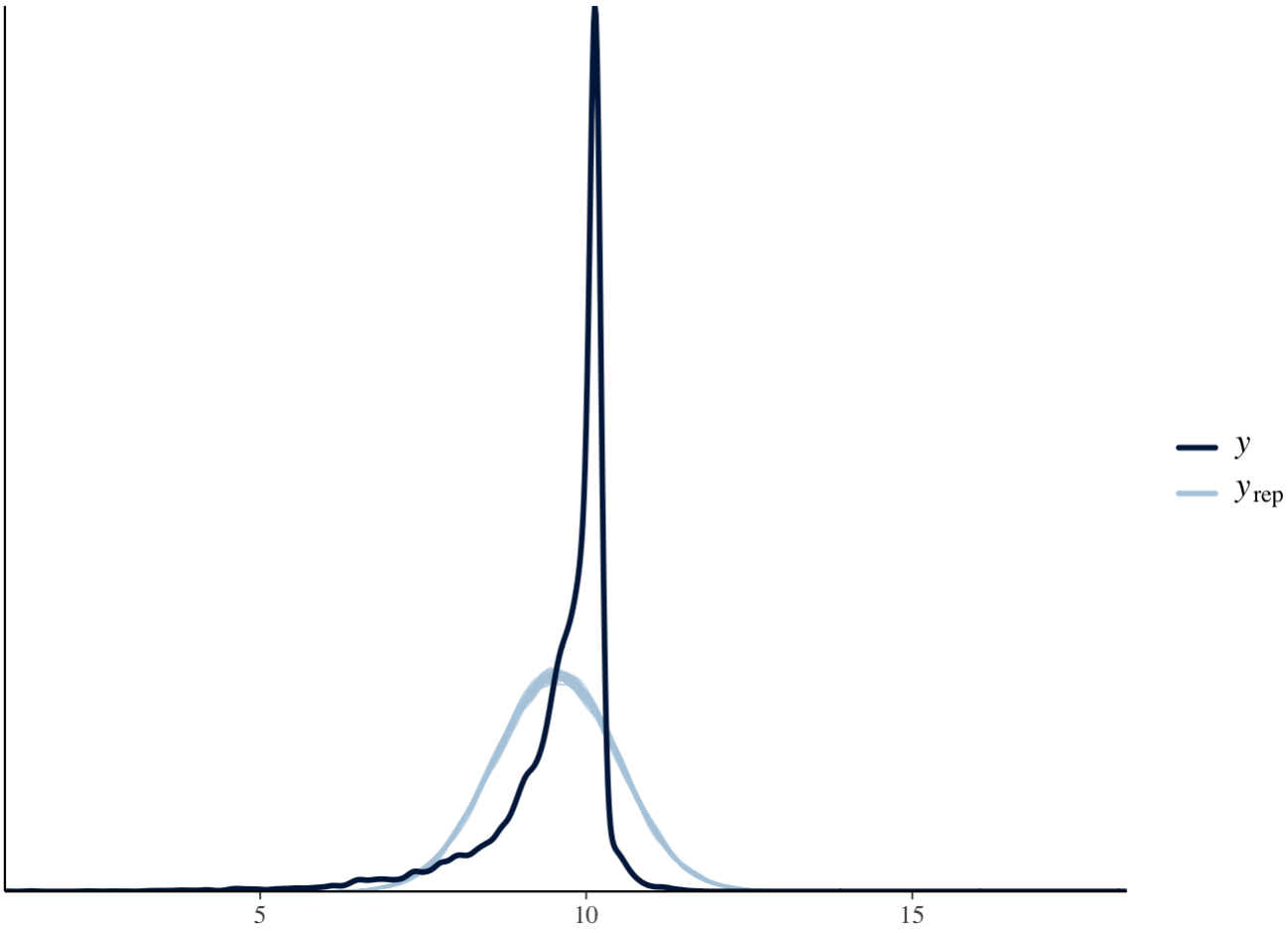
15.15 Summarizing inferences and predictions using simulation

Exercise 15.7 used a Tobit model to fit a regression with an outcome that had mixed discrete and continuous data. In this exercise you will revisit these data and build a two-step model: (1) logistic regression for zero earnings versus positive earnings, and (2) linear regression for level of earnings given earnings are positive. Compare predictions that result from each of these models with each other.

```
lalonde <- read.dta("NSW_dw_obs.dta")
lalonde$bin78 <- ifelse(lalonde$re78 > 0, 1, 0)
m1 = stan_glm(bin78 ~ treat + re75, data = lalonde, family = binomial(link="logit"), ref
resh = 0)
pp_check(m1)
```




```
m2 = stan_glm(log(re78) ~ treat + re75, data=lalonde[lalonde$bin78==1,], refresh=0)
pp_check(m2)
```



a