Assignment_3

Patrik Molnar, Ditlev Kræn, Bryan Roemelt, Sara Szabo, Manuela Skov Thomasen

2022-11-09

#Assignment 3

##Part 1 - Simulating data

Use meta analysis reported in Parola et al (2020) to create informed simulated data - 100 pairs of schizophrenia and controls, each participant producing 10 repeated measures (10 trials with their speech recorded), for each recording produce 10 acoustic measures (6 from meta analysis and 4 with random noise)

- Do the same for a baseline data set including only 10 noise variables ##Sara ### seeting up variables

```
###Data simulation
n <- 100
trials <- 10

#Effect sizes definition = Informed effect mean and Skeptic effect mean
IEM <- c(-0.5,-1.26,-.74,1.89,0.25,1.3,0,0,0,0)
SEM <- rep(0,10)

#Defining individual vairability from populationd accross trials measurement error
ISD <- 1
TSD <- 0.5
E <- 0.2
```

Simulating the true effect size for each varibale for all pairs of participants

```
for (i in seq(10)){
  temp_informed <- tibble(
    ID=seq(n),
    TrueEffect = rnorm(n, IEM[i], ISD),
    Variable = paste0("V",i))
  temp_skeptic <- tibble(
    ID=seq(n),
    TrueEffect = rnorm(n, SEM[i], ISD),
    Variable = paste0("V",i))

  if(i==1){
    d_informed_true <- temp_informed
    d_skeptic_true <- temp_skeptic
  } else {
    d_informed_true <-  rbind(d_informed_true, temp_informed)
    d_skeptic_true <- rbind(d_skeptic_true, temp_skeptic)
  }
}
```

Creating one row per trial

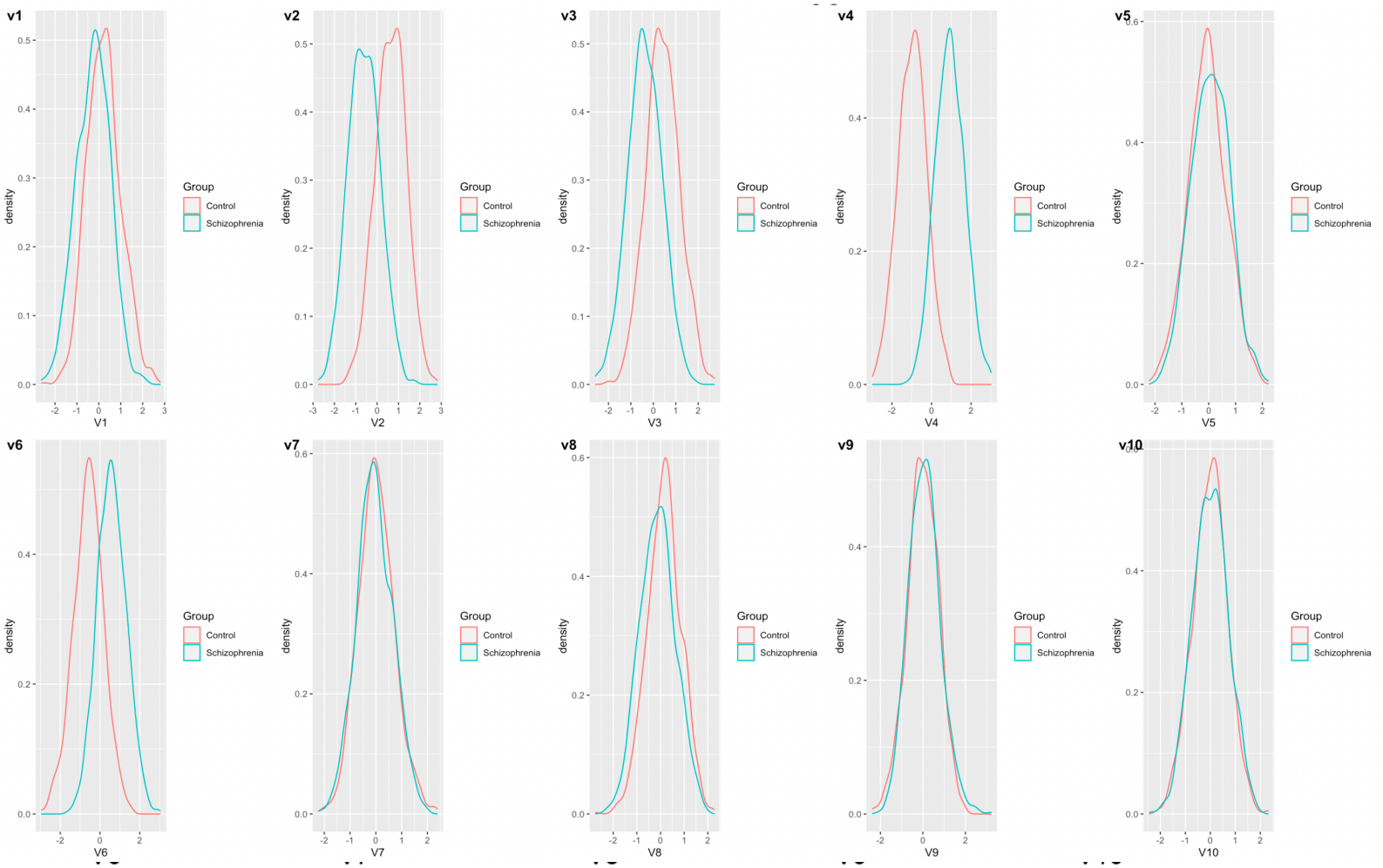
```
d_trial <- tibble(expand_grid(ID=seq(n), Trial = seq(trials), Group = c("Schizophr
enia", "Control")))

d_informed <- merge(d_informed_true, d_trial)
d_skeptic <- merge(d_skeptic_true, d_trial)

for ( i in seq(nrow(d_informed))){
  d_informed$measurement[i] <- ifelse(d_informed$Group[i]=="Schizophrenia",
                                      rnorm(1, rnorm(1, d_informed$TrueEffect[i]/2
, TSD), E),
                                      rnorm(1, rnorm(1, (-d_informed$TrueEffect[i]
)/2, TSD), E))

  d_skeptic$measurement[i] <- ifelse(d_skeptic$Group[i]=="Schizophrenia",
                                      rnorm(1, rnorm(1, d_skeptic$TrueEffect[i]/2,
TSD), E),
                                      rnorm(1, rnorm(1, (-d_skeptic$TrueEffect[i])
/2, TSD), E))
}
```

Transforming the dataframe to a wide format based on the variable

```
d_informed_wide <- d_informed %>%
  mutate(TrueEffect=NULL) %>%
  pivot_wider(names_from = Variable,
              values_from = measurement)
d_skeptic_wide <- d_skeptic %>%
  mutate(TrueEffect=NULL) %>%
  pivot_wider(names_from = Variable,
              values_from = measurement)

Schizo_ID <- d_informed_wide %>%
  filter(Group == 'Schizophrenia')

 control_ID <- d_informed_wide %>%
  filter(Group == 'Control')

control_ID[, 1] <- control_ID[,1] + 100

 d_informed_wide <- rbind(control_ID,Schizo_ID)


 Schizo_ID_skeptic <- d_skeptic_wide %>%
  filter(Group == 'Schizophrenia')

 control_ID_skeptic <- d_skeptic_wide %>%
  filter(Group == 'Control')

 control_ID_skeptic[,1] <- control_ID_skeptic[,1] + 100

 d_skeptic_wide <- rbind(control_ID_skeptic,Schizo_ID_skeptic)
```

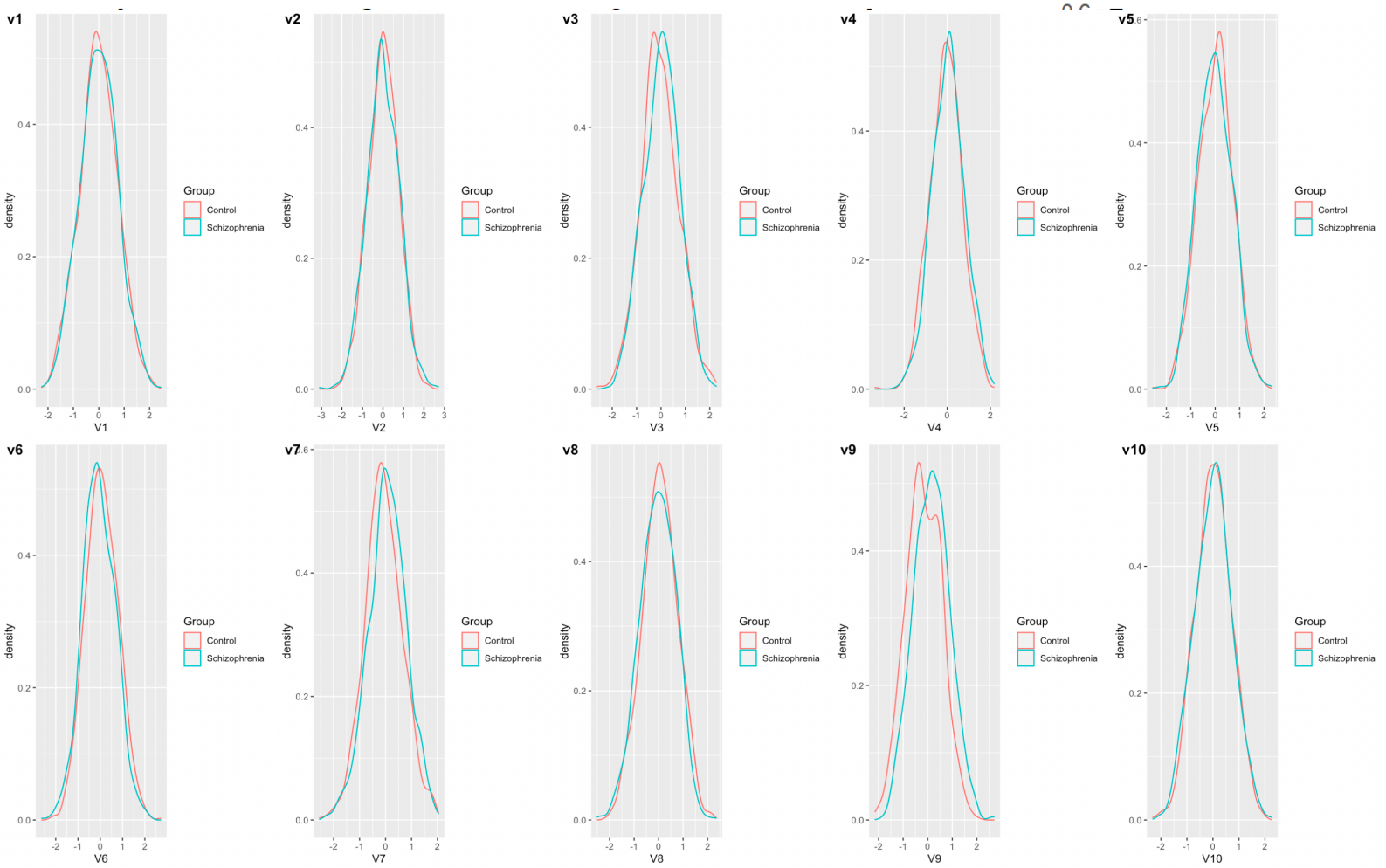
Visualizing the simulated informed data

```
plot1 <- d_informed_wide %>%
  ggplot(aes(x = V1, color = Group))+
  geom_density()

plot2 <- d_informed_wide %>%
  ggplot(aes(x = V2, color = Group))+
  geom_density()

plot3 <- d_informed_wide %>%
  ggplot(aes(x = V3, color = Group))+
  geom_density()

plot4 <- d_informed_wide %>%
  ggplot(aes(x = V4, color = Group))+
  geom_density()

plot5 <- d_informed_wide %>%
  ggplot(aes(x = V5, color = Group))+
  geom_density()

plot6 <- d_informed_wide %>%
  ggplot(aes(x = V6, color = Group))+
  geom_density()

plot7 <- d_informed_wide %>%
  ggplot(aes(x = V7, color = Group))+
  geom_density()

plot8 <- d_informed_wide %>%
  ggplot(aes(x = V8, color = Group))+
  geom_density()

plot9 <- d_informed_wide %>%
  ggplot(aes(x = V9, color = Group))+
  geom_density()

plot10 <- d_informed_wide %>%
  ggplot(aes(x = V10, color = Group))+
  geom_density()

cowplot::plot_grid(plot1, plot2, plot3, plot4, plot5, plot6, plot7, plot8, plot9,
plot10,
          labels = c("v1", "v2", "v3",'v4','v5','v6','v7','v8','v9','v10'),
          ncol = 5, nrow = 2)
```



Visualizing the simulated data for skeptical data frame

```
plot1_s <- d_skeptic_wide %>%
  ggplot(aes(x = V1, color = Group))+
  geom_density()

plot2_s <- d_skeptic_wide %>%
  ggplot(aes(x = V2, color = Group))+
  geom_density()

plot3_s <- d_skeptic_wide %>%
  ggplot(aes(x = V3, color = Group))+
  geom_density()

plot4_s <- d_skeptic_wide %>%
  ggplot(aes(x = V4, color = Group))+
  geom_density()

plot5_s <- d_skeptic_wide %>%
  ggplot(aes(x = V5, color = Group))+
  geom_density()

plot6_s <- d_skeptic_wide %>%
  ggplot(aes(x = V6, color = Group))+
  geom_density()

plot7_s <- d_skeptic_wide %>%
  ggplot(aes(x = V7, color = Group))+
  geom_density()

plot8_s <- d_skeptic_wide %>%
  ggplot(aes(x = V8, color = Group))+
  geom_density()

plot9_s <- d_skeptic_wide %>%
  ggplot(aes(x = V9, color = Group))+
  geom_density()

plot10_s <- d_skeptic_wide %>%
  ggplot(aes(x = V10, color = Group))+
  geom_density()

cowplot::plot_grid(plot1_s, plot2_s, plot3_s, plot4_s, plot5_s, plot6_s, plot7_s,
plot8_s, plot9_s, plot10_s,
          labels = c("v1", "v2", "v3",'v4','v5','v6','v7','v8','v9','v10'),
          ncol = 5, nrow = 2)
```



##Patrik ##Part 2 - Machine larning pipeline on simulated data Build a machine leaning pipeline (separately on the 2 datasets) - create a data budget (e.g., balanced training and test sets) pree-process the data (e.g., scaling the features) - fit and assess a classification algorithm on the training data (e.g., bayesian multilevel logistic regression) - assess performance on the test set - discuss whether performance and feature importance is as expected

Data budget (splitting) and pree-processing (scaling)

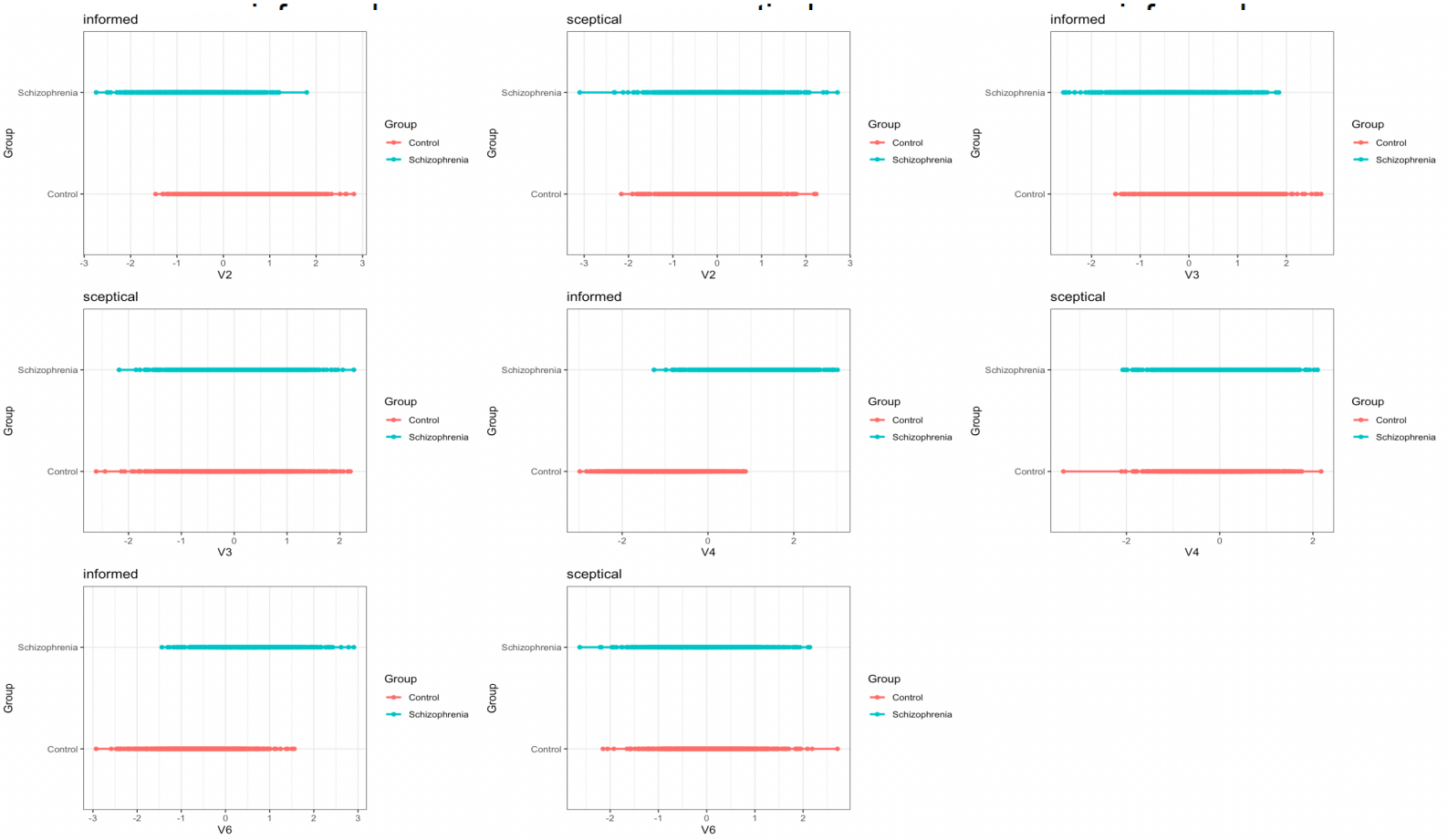
```
set.seed(260)


d_informed_wide <- d_informed_wide %>%
  mutate(pair_ID=ID) %>%
  mutate(pair_ID= ifelse(ID>100, ID-100, ID))

d_skeptic_wide <- d_skeptic_wide %>%
  mutate(pair_ID=ID) %>%
  mutate(pair_ID= ifelse(ID>100, ID-100, ID))

split_inf <- initial_split(d_informed_wide, prop = 4/5)

train_informed <- training(split_inf)
test_informed <- testing(split_inf)
```

```
split_skep <- initial_split(d_skeptic_wide, prop = 4/5)
train_skeptic <- training(split_skep)
test_skeptic <- testing(split_skep)

train_informed$ID <- as.factor(train_informed$ID)
train_skeptic$ID <- as.factor(train_skeptic$ID)

test_informed$ID <- as.factor(test_informed$ID)
test_skeptic$ID <- as.factor(test_skeptic$ID)

train_informed$pair_ID <- as.factor(train_informed$pair_ID)
train_skeptic$pair_ID <- as.factor(train_skeptic$pair_ID)

test_informed$pair_ID <- as.factor(test_informed$pair_ID)
test_skeptic$pair_ID <- as.factor(test_skeptic$pair_ID)

train_informed$Trial <- as.factor(train_informed$Trial)
train_skeptic$Trial <- as.factor(train_skeptic$Trial)

test_informed$Trial <- as.factor(test_informed$Trial)
test_skeptic$Trial <- as.factor(test_skeptic$Trial)


rec_informed <- train_informed %>%
  recipe(Group~.) %>%
  update_role(ID, new_role = 'ID') %>%
  step_scale(all_numeric()) %>%
  step_center(all_numeric()) %>%
  prep(training=train_informed, retain=TRUE)

rec_skeptic <- train_skeptic %>%
  recipe(Group~.) %>%
  update_role(ID, new_role = 'ID') %>%
  step_scale(all_numeric()) %>%
  step_center(all_numeric()) %>%
  prep(training=train_informed, retain=TRUE)


train_informed_s <- juice(rec_informed)
test_informed_s <- bake(rec_informed, new_data = test_informed)

train_skeptic_s <- juice(rec_skeptic)
test_skeptic_s <- bake(rec_skeptic, new_data = test_skeptic)
```

Visual inspection of the data

```
plot_i2 <- ggplot(train_informed, aes(V2, Group, colour=Group))+
  geom_point()+
  geom_smooth(method = "glm", se=FALSE)+
  theme_bw()+
  ggtitle("informed")

plot_s2 <- ggplot(train_skeptic, aes(V2, Group, colour=Group))+
  geom_point()+
  geom_smooth(method = "glm", se=FALSE)+
  theme_bw()+
  ggtitle("sceptical")

plot_i3 <- ggplot(train_informed, aes(V3, Group, colour=Group))+
  geom_point()+
  geom_smooth(method = "glm", se=FALSE)+
  theme_bw()+
  ggtitle("informed")

plot_s3 <- ggplot(train_skeptic, aes(V3, Group, colour=Group))+
  geom_point()+
  geom_smooth(method = "glm", se=FALSE)+
  theme_bw()+
  ggtitle("sceptical")

plot_i4 <- ggplot(train_informed, aes(V4, Group, colour=Group))+
  geom_point()+
  geom_smooth(method = "glm", se=FALSE)+
  theme_bw()+
  ggtitle("informed")

plot_s4 <- ggplot(train_skeptic, aes(V4, Group, colour=Group))+
  geom_point()+
  geom_smooth(method = "glm", se=FALSE)+
  theme_bw()+
  ggtitle("sceptical")

plot_i6 <- ggplot(train_informed, aes(V6, Group, colour=Group))+
  geom_point()+
  geom_smooth(method = "glm", se=FALSE)+
  theme_bw()+
  ggtitle("informed")

plot_s6 <- ggplot(train_skeptic, aes(V6, Group, colour=Group))+
  geom_point()+
  geom_smooth(method = "glm", se=FALSE)+
  theme_bw()+
  ggtitle("sceptical")

plot_grid(plot_i2, plot_s2, plot_i3, plot_s3, plot_i4, plot_s4, plot_i6, plot_s6)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



fit and asses a classification algorithm on training data (Baysian)

```
##Setting up the model
PR_f0 <-bf(Group~1+V1+V2+V3+V4+V5+V6+V7+V8+V9+V10)

PR_f1 <-bf(Group~1+V1+V2+V3+V4+V5+V6+V7+V8+V9+V10+(1|ID))

get_prior(PR_f0, train_informed_s, family = bernoulli)
```

```
##                 prior     class coef group resp dpar nlpar lb ub       source
##                (flat)         b                                       default
##                (flat)         b   V1                             (vectorized)
##                (flat)         b  V10                             (vectorized)
##                (flat)         b   V2                             (vectorized)
##                (flat)         b   V3                             (vectorized)
##                (flat)         b   V4                             (vectorized)
##                (flat)         b   V5                             (vectorized)
##                (flat)         b   V6                             (vectorized)
##                (flat)         b   V7                             (vectorized)
##                (flat)         b   V8                             (vectorized)
##                (flat)         b   V9                             (vectorized)
##  student_t(3, 0, 2.5) Intercept                                        default
```

```
get_prior(PR_f1, train_informed_s, family = bernoulli)
```

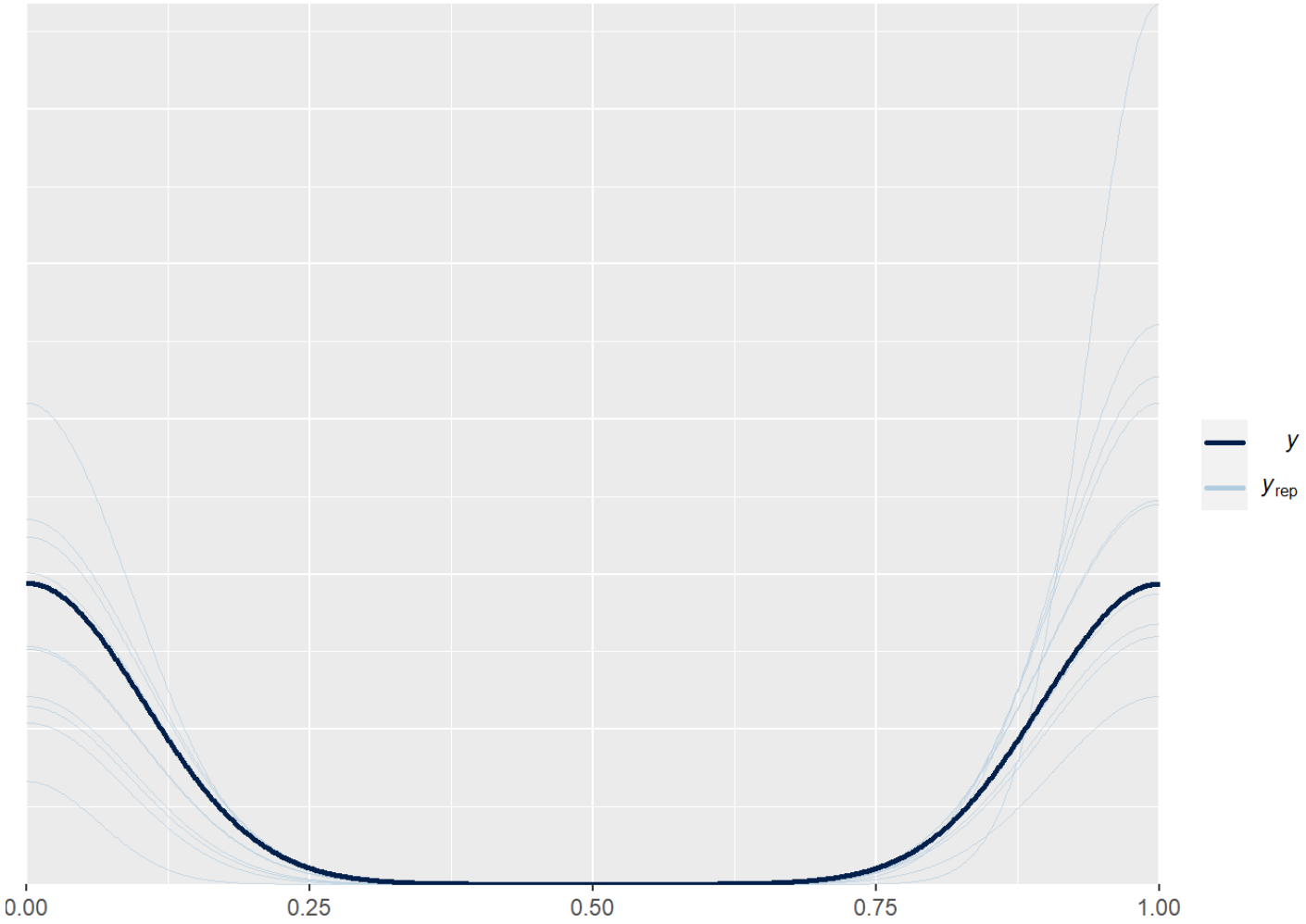
```
##                 prior     class      coef group resp dpar nlpar lb ub
##                (flat)         b
##                (flat)         b        V1
##                (flat)         b       V10
##                (flat)         b        V2
##                (flat)         b        V3
##                (flat)         b        V4
##                (flat)         b        V5
##                (flat)         b        V6
##                (flat)         b        V7
##                (flat)         b        V8
##                (flat)         b        V9
##  student_t(3, 0, 2.5) Intercept
##  student_t(3, 0, 2.5)        sd                                   0
##  student_t(3, 0, 2.5)        sd              ID                   0
##  student_t(3, 0, 2.5)        sd Intercept    ID                   0
##       source
##      default
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##      default
##      default
##  (vectorized)
##  (vectorized)
```

Setting priors

```
PR_p0 <- c(
  prior(normal(0, 1), class=Intercept),
  prior(normal(0, 0.3), class=b)
)

PR_p1 <- c(
  prior(normal(0, 1), class=Intercept),
  prior(normal(0, 0.3), class=sd),
  prior(normal(0, 0.3), class=b)
)
```

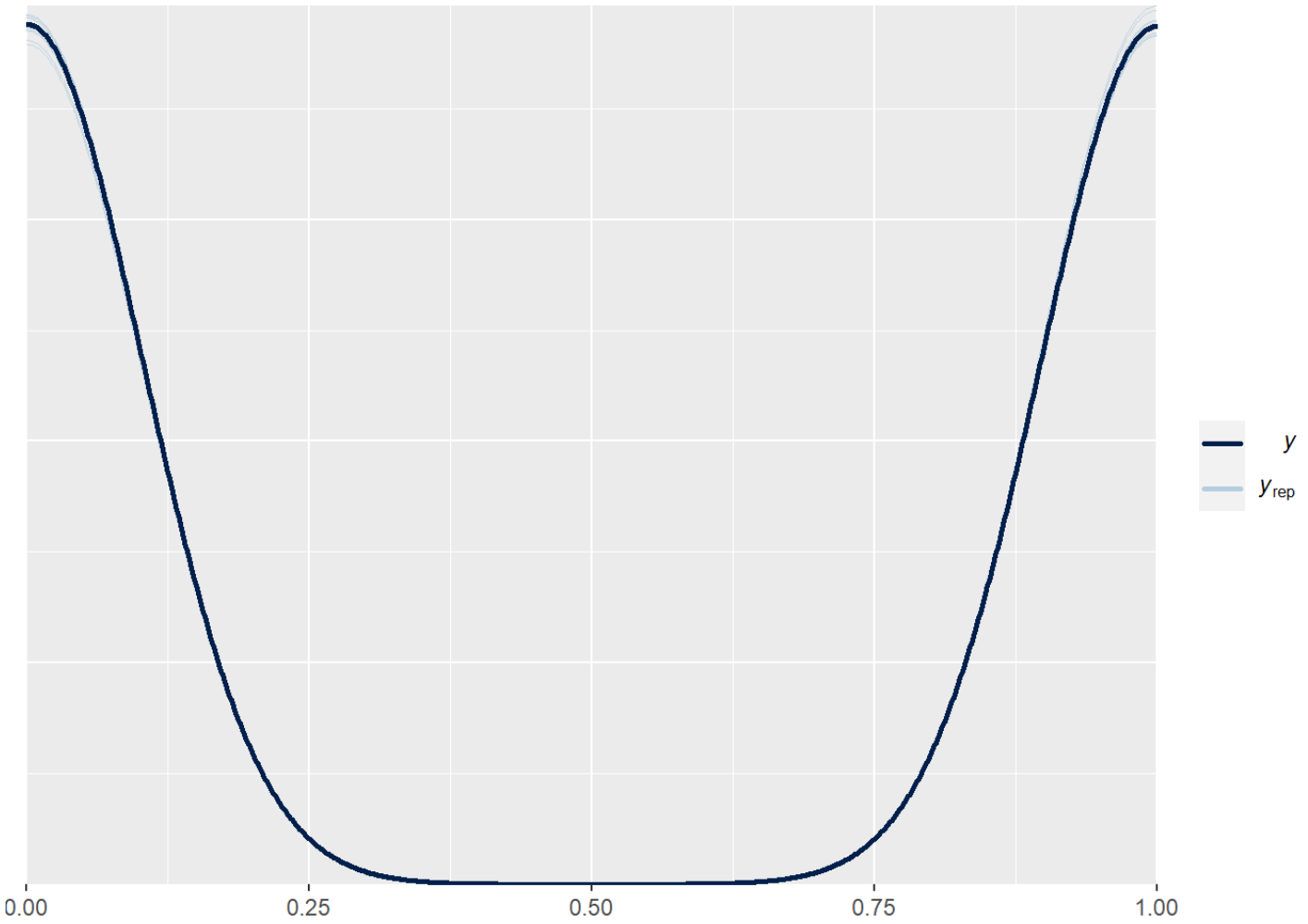
Fitting the first model on both the skeptical and informed data

```
##Model fit on priors
pr_m0_inf <- brm(
  PR_f0,
  data = train_informed_s,
  prior = PR_p0,
  family = bernoulli,
  refresh=0,
  sample_prior = 'only',
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)


#pp_check(pr_m0_inf)

pr_m0_skep <- brm(
  PR_f0,
  data = train_skeptic_s,
  prior = PR_p0,
  family = bernoulli,
  refresh=0,
  sample_prior = 'only',
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)


pp_check(pr_m0_skep)
```

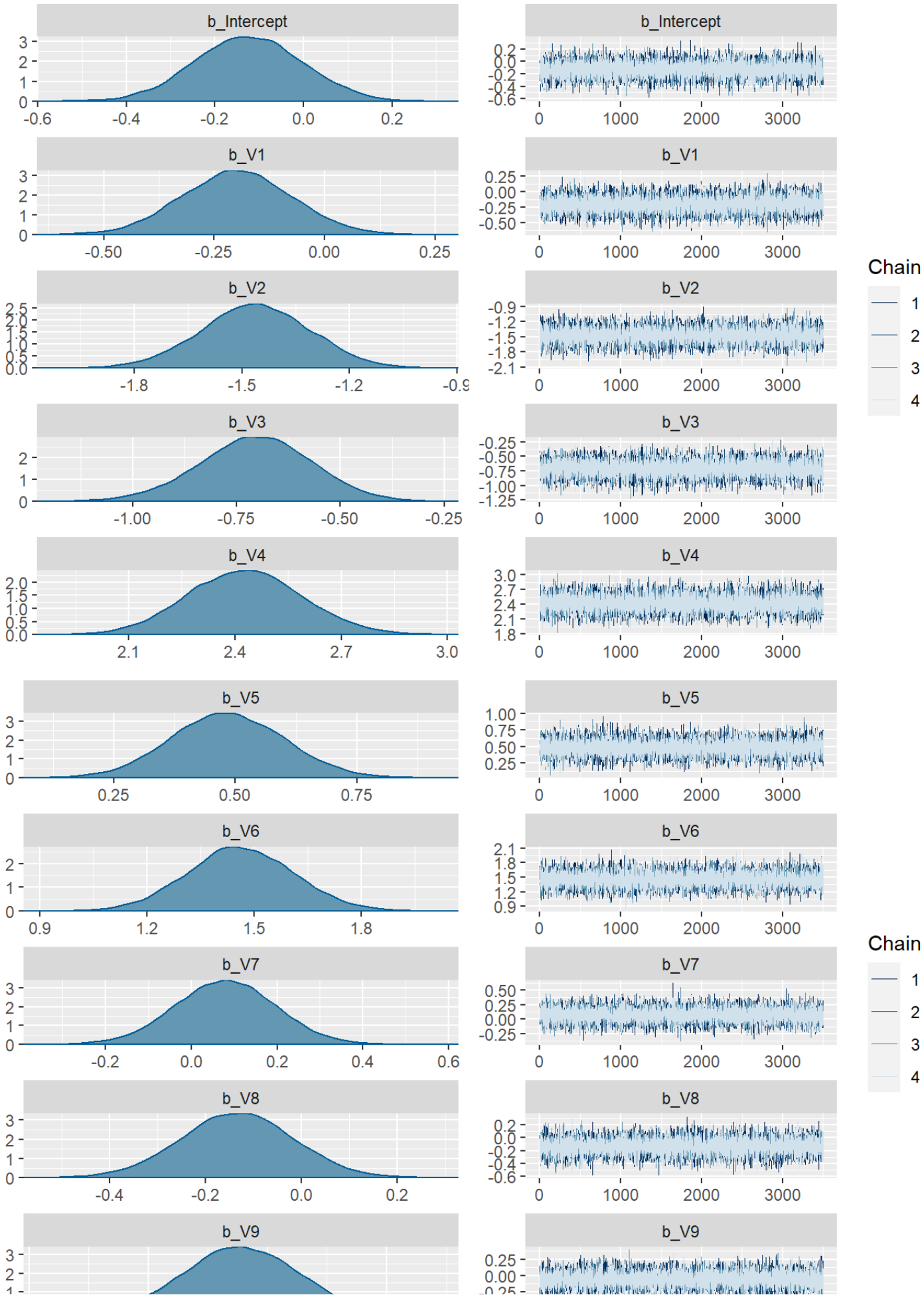


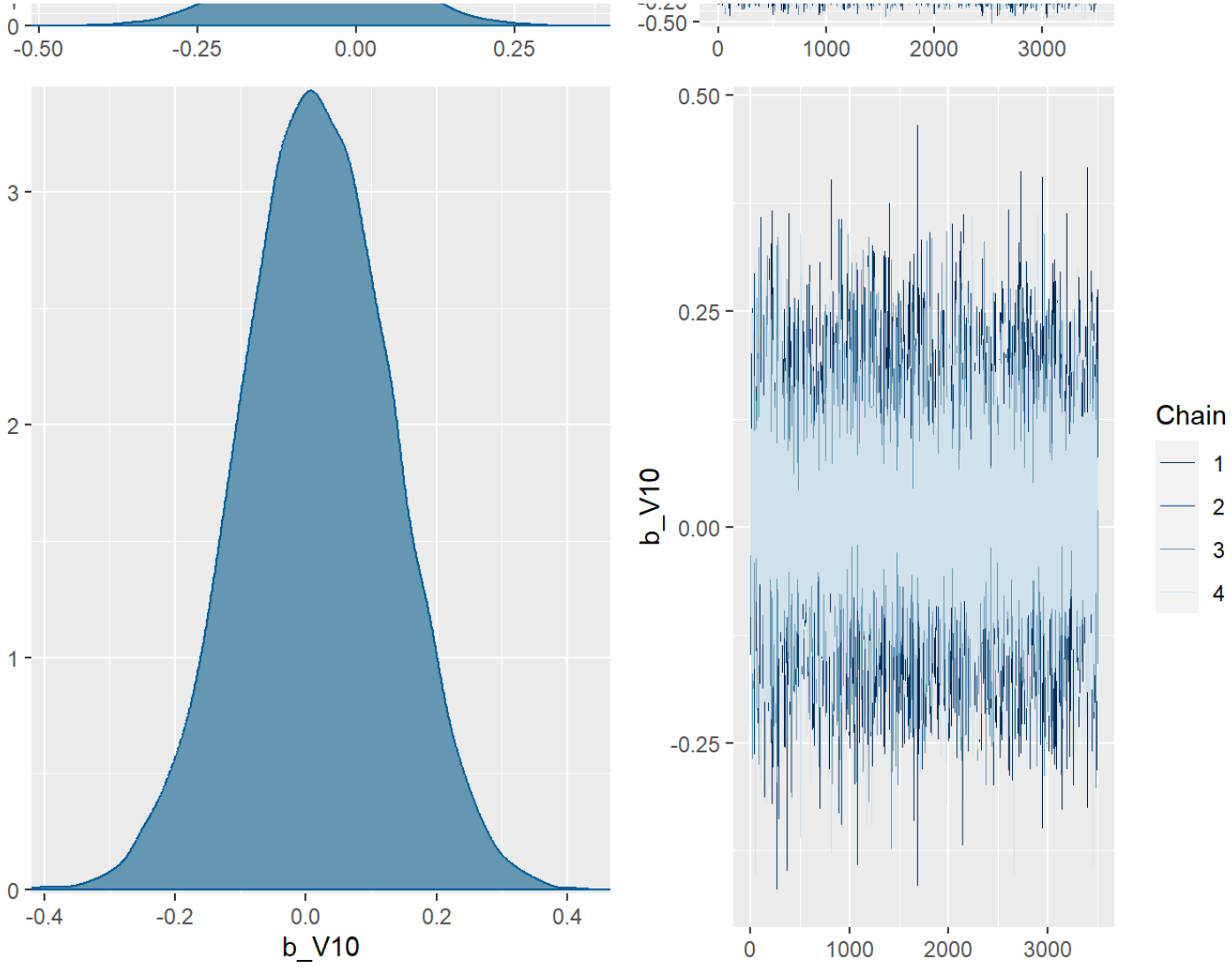
##Bryan

```
##Model fit on informed data
pr_m0_fit_inf <- brm(
  PR_f0,
  data = train_informed_s,
  prior = PR_p0,
  family = bernoulli,
  refresh=0,
  sample_prior = TRUE,
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)

pp_check(pr_m0_fit_inf)
```



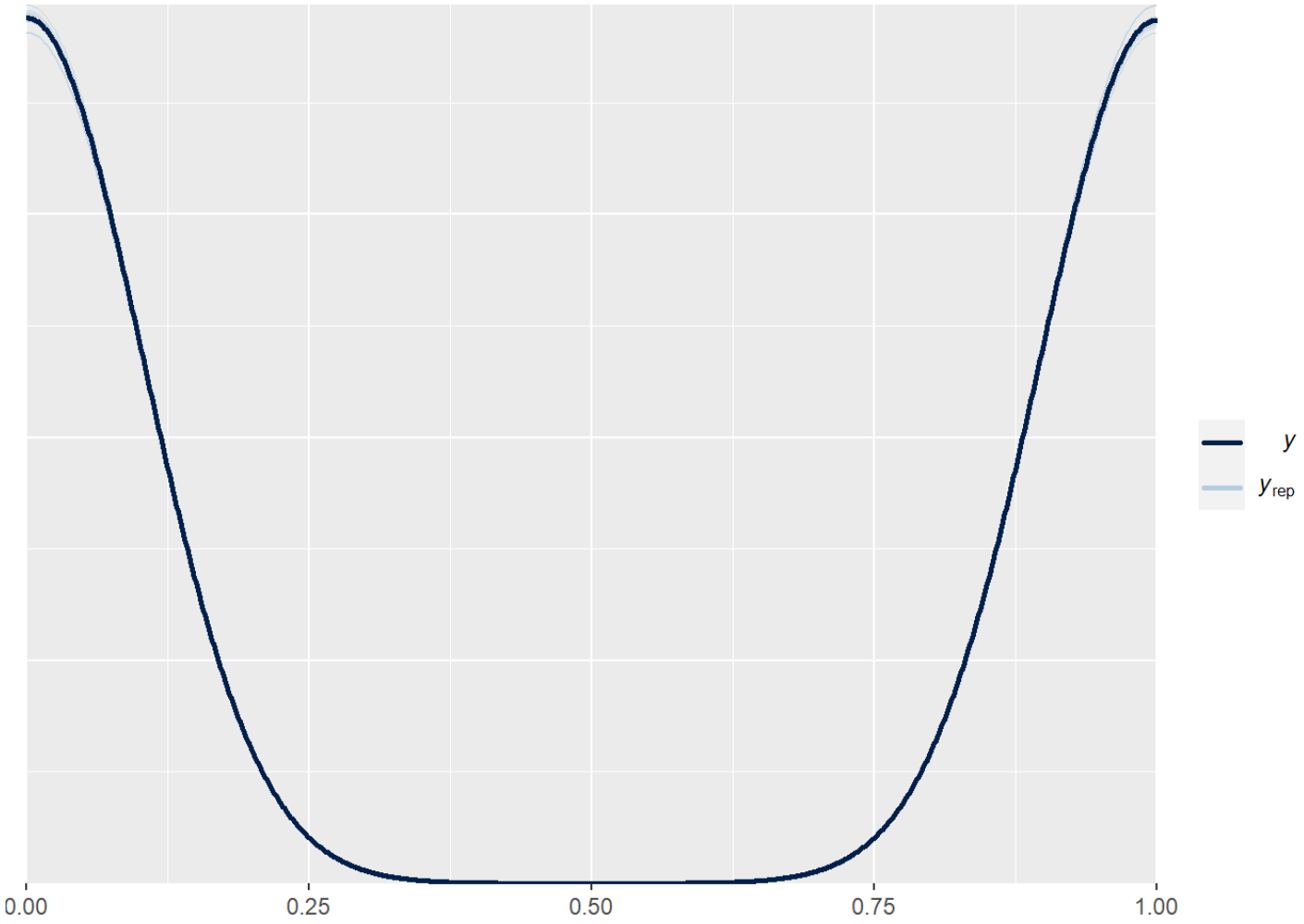

```
plot(pr_m0_fit_inf)
```


b_Intercept
b_V1
b_V2
b_V3
b_V4
b_V5
b_V6
b_V7
b_V8
b_V9
Chain
1
2
3
4

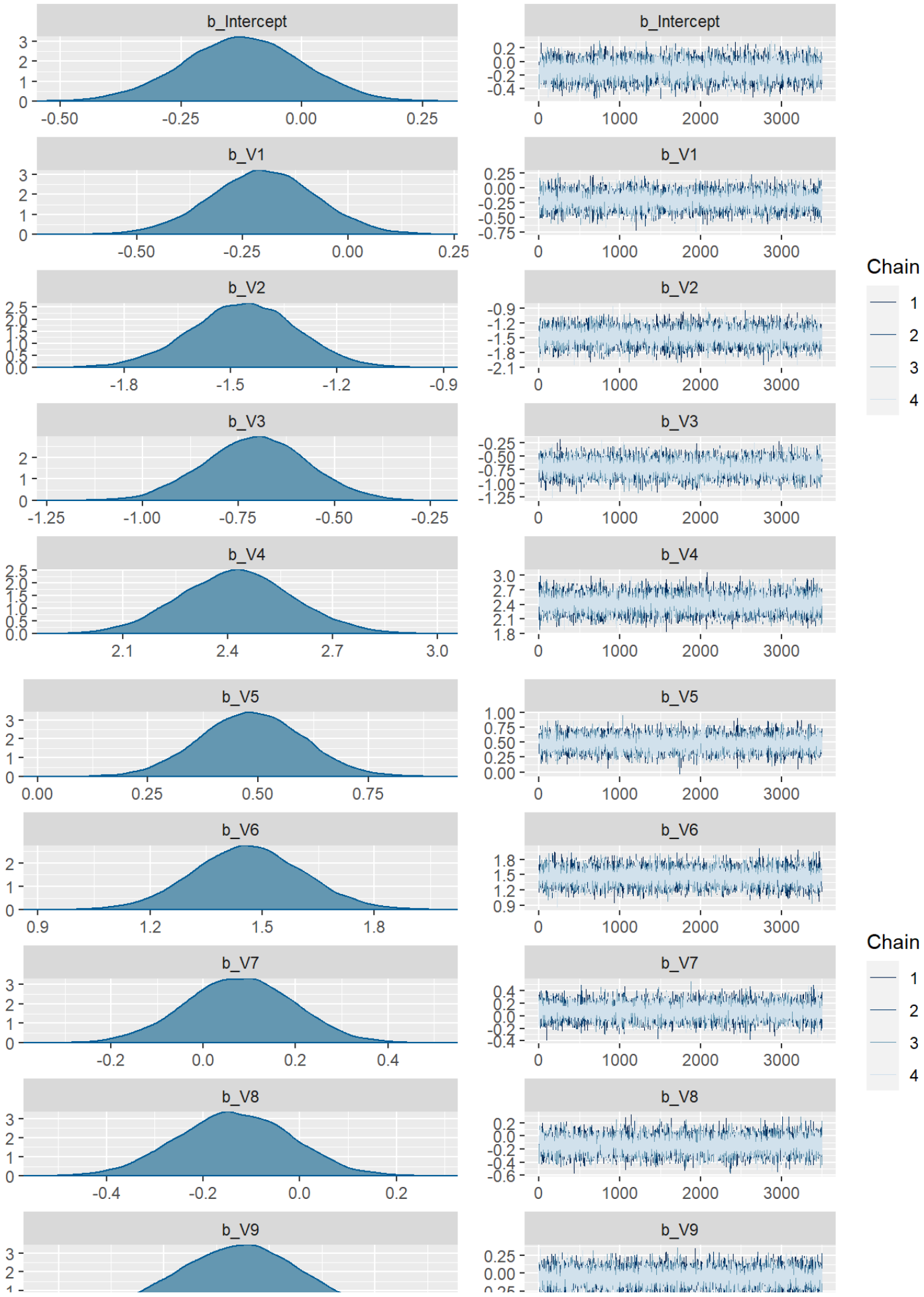


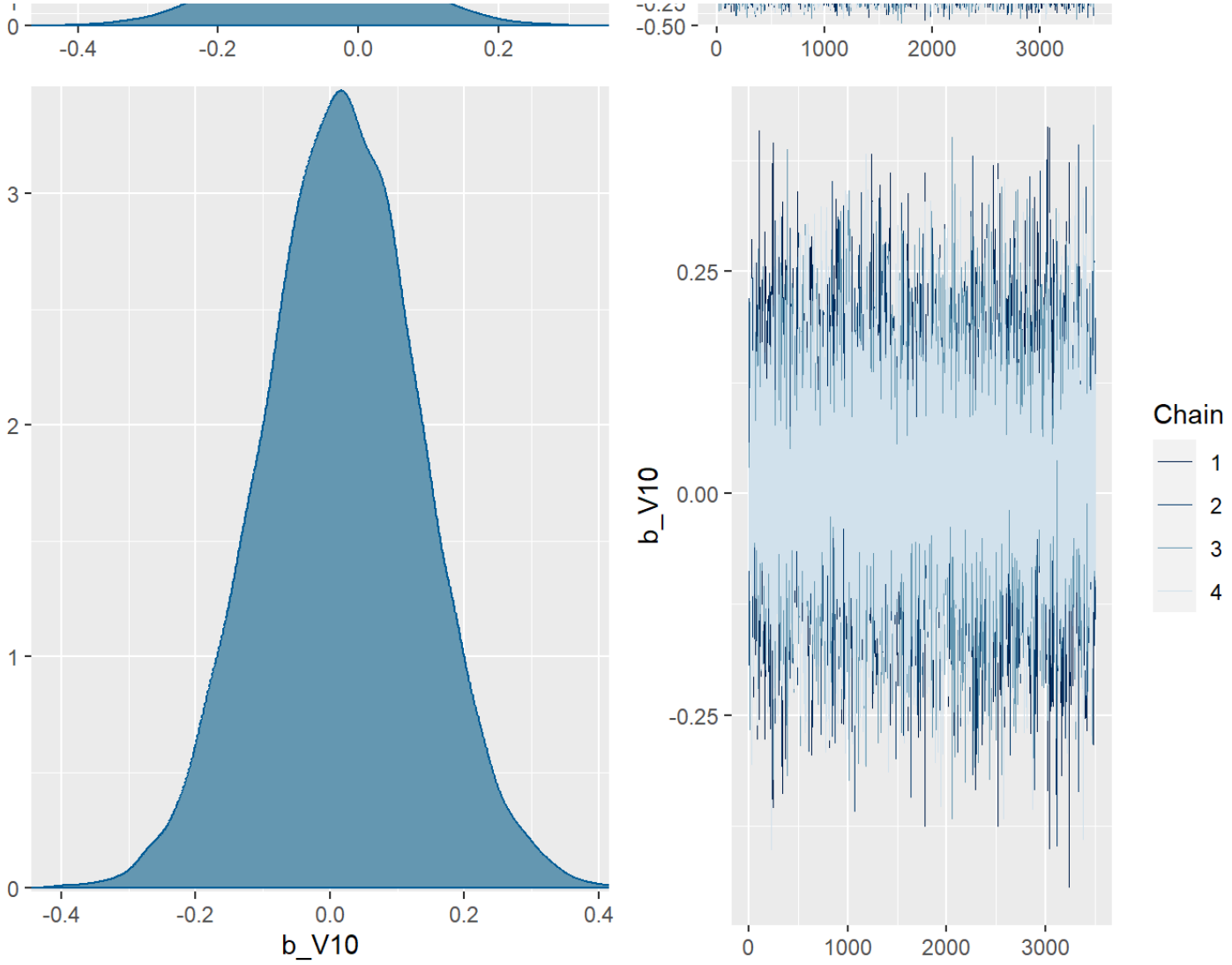

```
summary(pr_m0_fit_inf)
```

```
##Model fit on skeptical data
pr_m0_fit_skep <- brm(
  PR_f0,
  data = train_skeptic_s,
  prior = PR_p0,
  family = bernoulli,
  refresh=0,
  sample_prior = TRUE,
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)

pp_check(pr_m0_fit_skep)
```




```
plot(pr_m0_fit_skep)
```

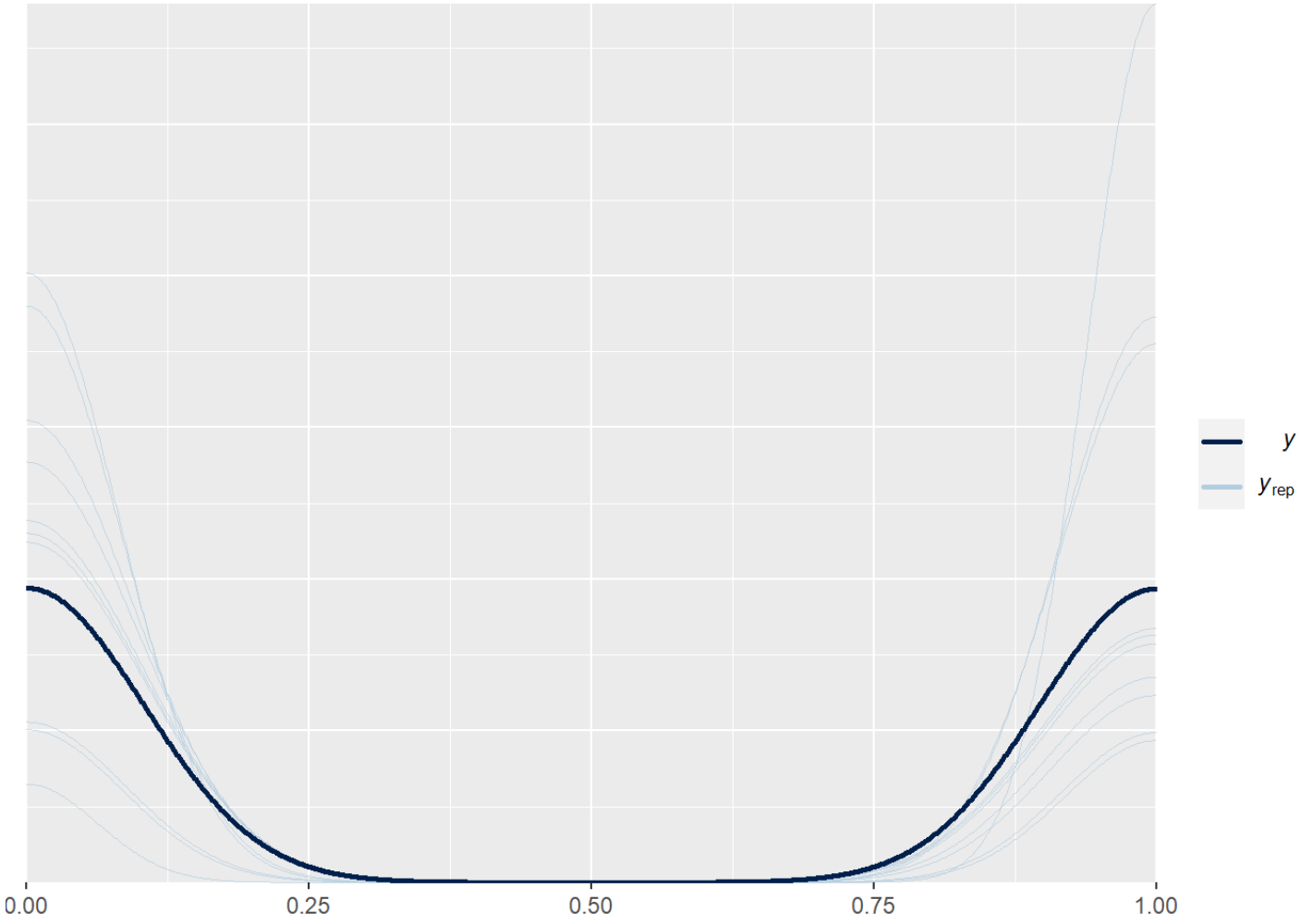

b_Intercept
b_V1
b_V2
b_V3
b_V4
b_V5
b_V6
b_V7
b_V8
b_V9
Chain
1
2
3
4



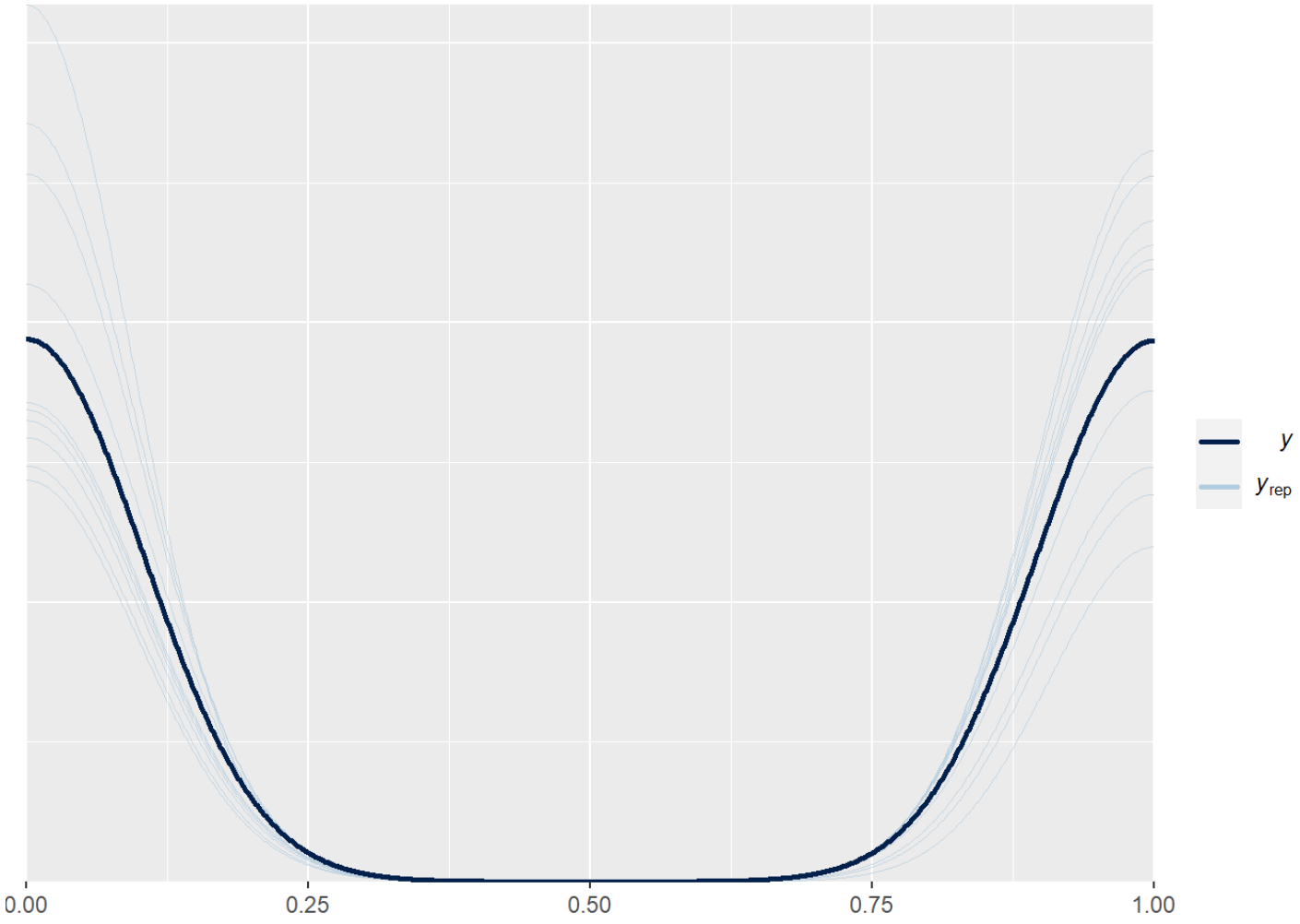

```
summary(pr_m0_fit_skep)
```

Fitting the second model on skeptical and informed data

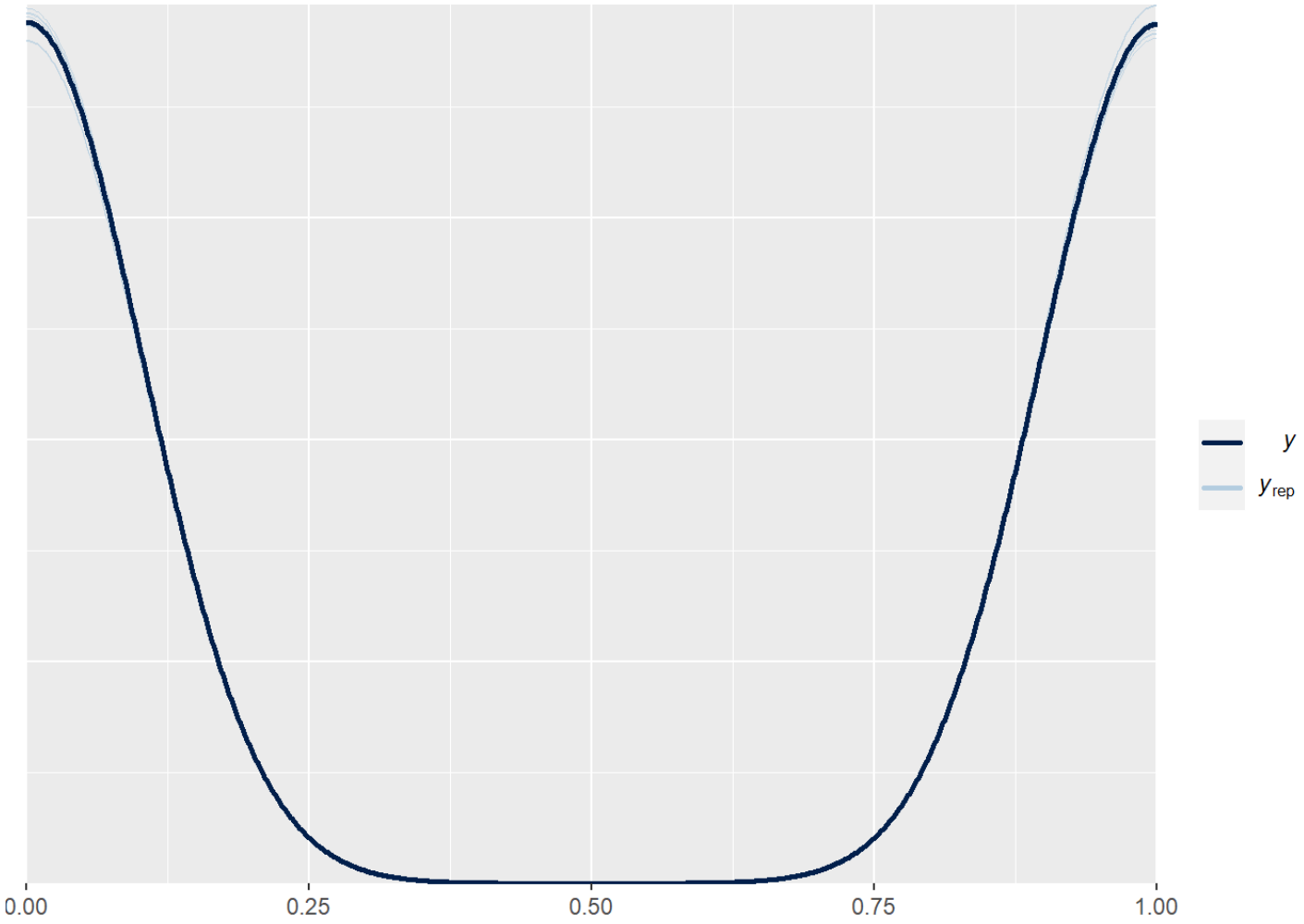
```
##Second model fit on priors
pr_m1_inf <- brm(
  PR_f1,
  data = train_informed_s,
  prior = PR_p1,
  family = bernoulli,
  refresh=0,
  sample_prior = 'only',
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)


pp_check(pr_m1_inf)
```



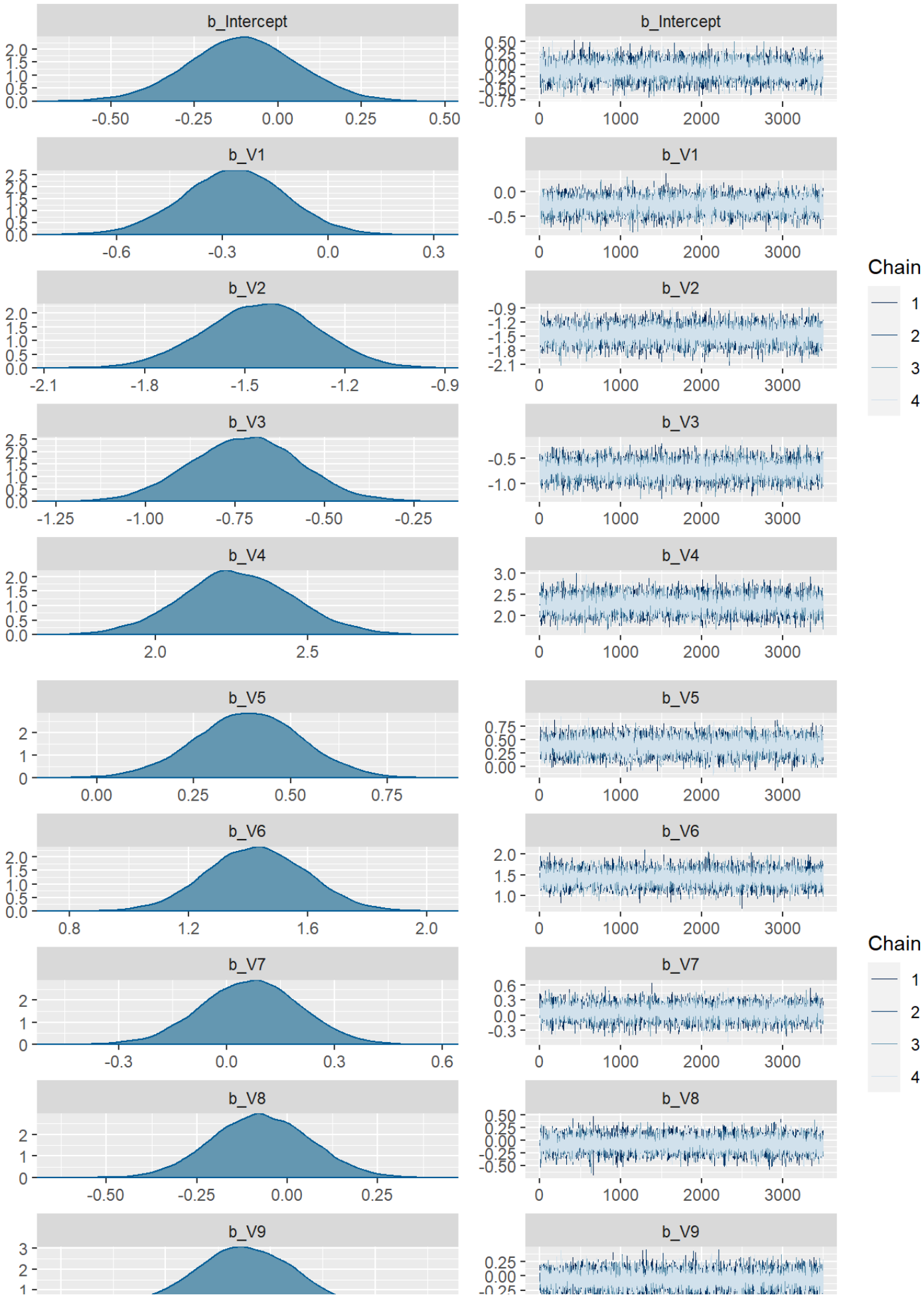
```
pr_m1_skep <- brm(
  PR_f1,
  data = train_skeptic_s,
  prior = PR_p1,
  family = bernoulli,
  refresh=0,
  sample_prior = 'only',
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)


pp_check(pr_m1_skep)
```

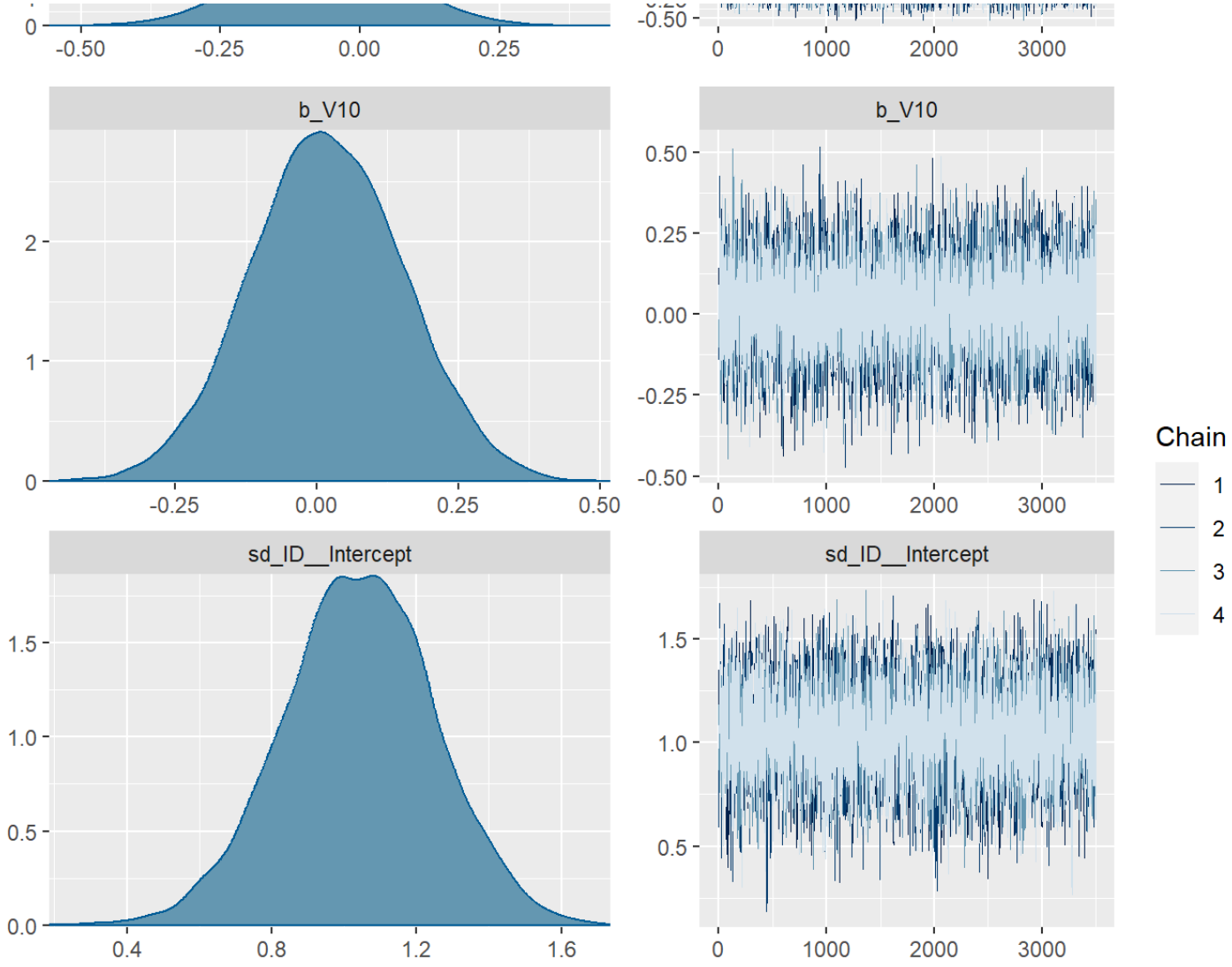


```
##second model fit on informed data
pr_m1_fit_inf <- brm(
  PR_f1,
  data = train_informed_s,
  prior = PR_p1,
  family = bernoulli,
  refresh=0,
  sample_prior = TRUE,
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)

pp_check(pr_m1_fit_inf)
```



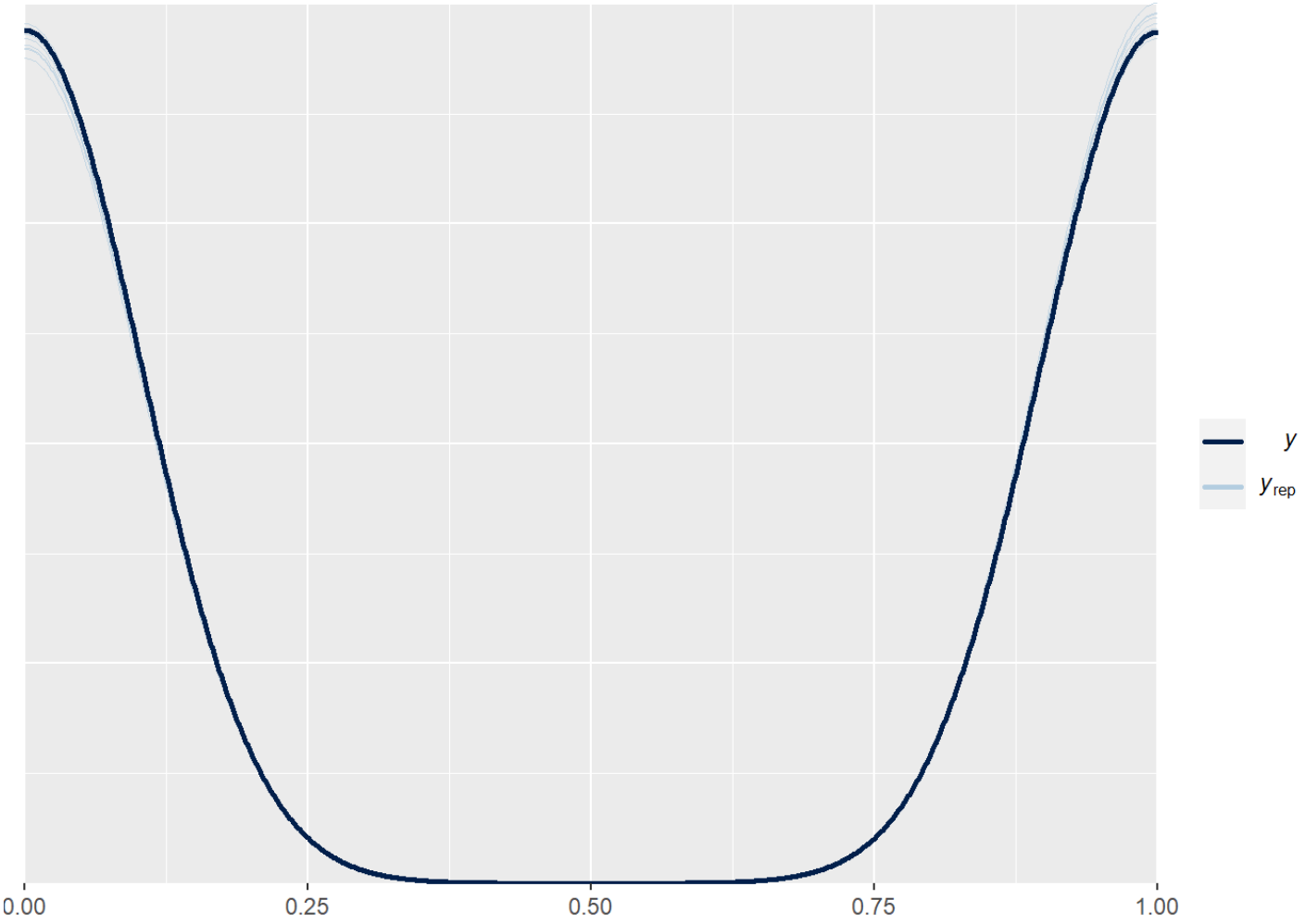

```
plot(pr_m1_fit_inf)
```


b_Intercept
b_V1
b_V2
b_V3
b_V4
b_V5
b_V6
b_V7
b_V8
b_V9
Chain
1
2
3
4

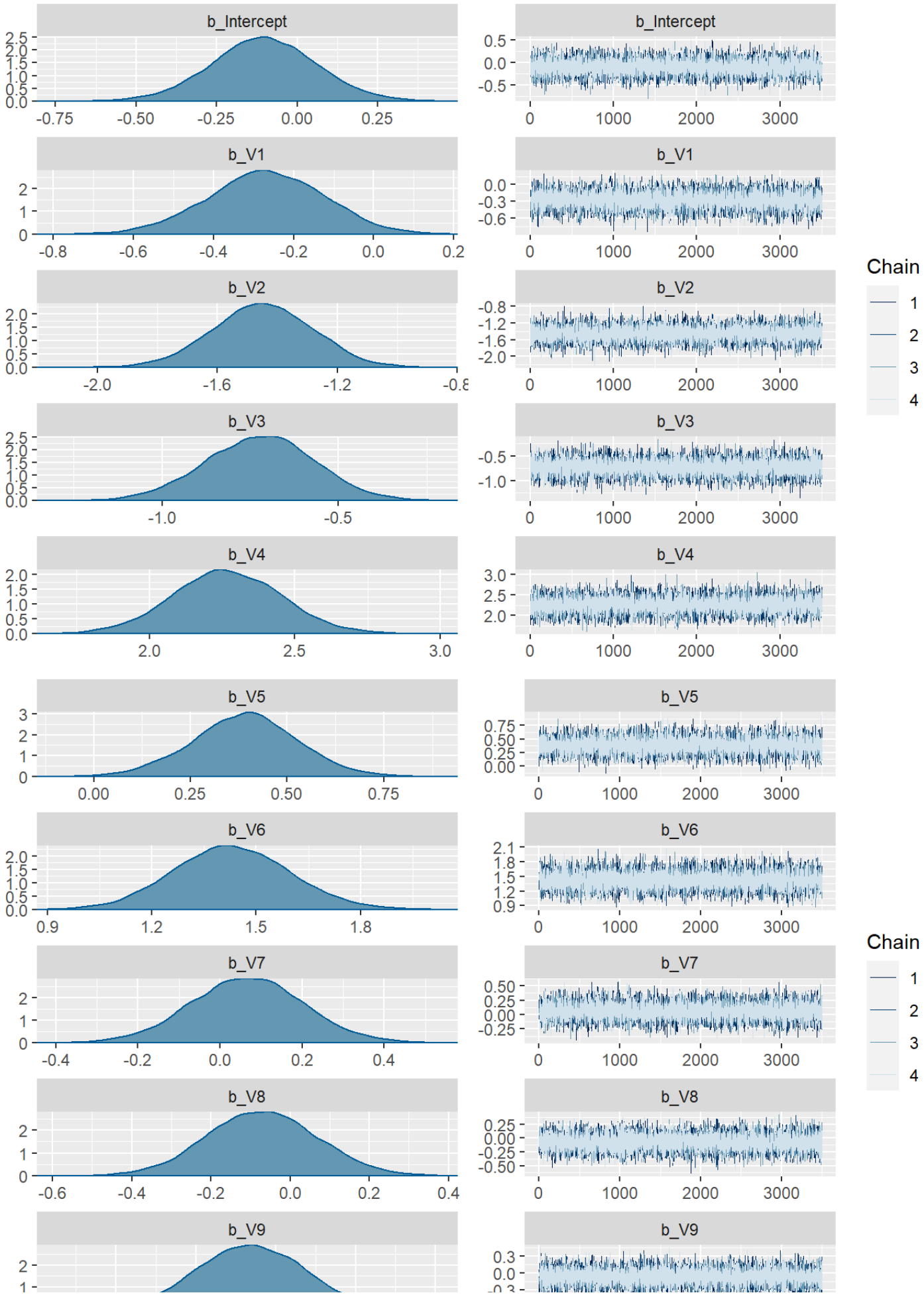


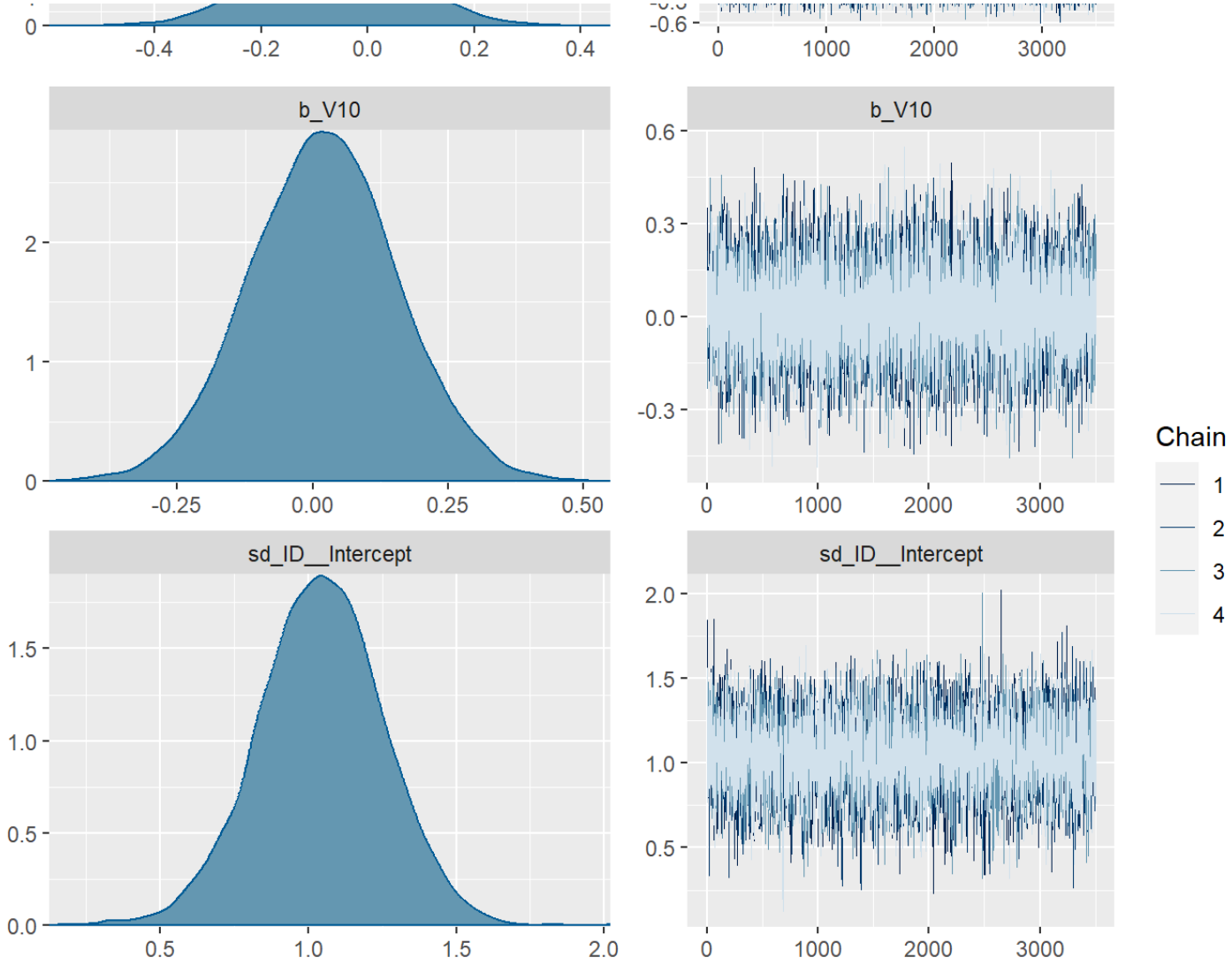

```
summary(pr_m1_fit_inf)
```

```
## second model fit on skeptical data
pr_m1_fit_skep <- brm(
  PR_f1,
  data = train_skeptic_s,
  prior = PR_p1,
  family = bernoulli,
  refresh=0,
  sample_prior = TRUE,
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)

pp_check(pr_m1_fit_skep)
```




```
plot(pr_m1_fit_skep)
```


b_Intercept
b_V1
b_V2
b_V3
b_V4
b_V5
b_V6
b_V7
b_V8
b_V9
Chain
1
2
3
4




```
summary(pr_m1_fit_skep)
```

##Ditlev ### Asses performance on test data

```
train_informed_s$PredictionsPerc0 <- predict(pr_m0_fit_inf)[, 1]
train_informed_s$Prediction0[train_informed_s$PredictionsPerc0 > 0.5] <- "Schizoph
renia"
```

```
## Warning: Unknown or uninitialised column: `Prediction0`.
```

```
train_informed_s$Prediction0[train_informed_s$PredictionsPerc0 <= 0.5] <- "Control
"


train_informed_s$PredictionsPerc1 <- predict(pr_m1_fit_inf)[, 1]
train_informed_s$Prediction1[train_informed_s$PredictionsPerc1 > 0.5] <- "Schizoph
renia"
```

```
## Warning: Unknown or uninitialised column: `Prediction1`.
```

```
train_informed_s$Prediction1[train_informed_s$PredictionsPerc1 <= 0.5] <- "Control
"


train_skeptic_s$PredictionsPerc0 <- predict(pr_m0_fit_skep)[, 1]
train_skeptic_s$Prediction0[train_skeptic_s$PredictionsPerc0 > 0.5] <- "Schizophre
nia"
```

```
## Warning: Unknown or uninitialised column: `Prediction0`.
```

```
train_skeptic_s$Prediction0[train_skeptic_s$PredictionsPerc0 <= 0.5] <- "Control"

train_skeptic_s$PredictionsPerc1 <- predict(pr_m1_fit_skep)[, 1]
train_skeptic_s$Prediction1[train_skeptic_s$PredictionsPerc1 > 0.5] <- "Schizophre
nia"
```

```
## Warning: Unknown or uninitialised column: `Prediction1`.
```

```
train_skeptic_s$Prediction1[train_skeptic_s$PredictionsPerc1 <= 0.5] <- "Control"

train_informed_s <- train_informed_s %>%
  mutate(
    Group = as.factor(Group),
    Prediction0 = as.factor(Prediction0),
    Prediction1 = as.factor(Prediction1)
  )

train_skeptic_s <- train_skeptic_s %>%
  mutate(
    Group = as.factor(Group),
    Prediction0 = as.factor(Prediction0),
    Prediction1 = as.factor(Prediction1)
  )


test_informed_s$PredictionsPerc0 <- predict(pr_m0_fit_inf, newdata = test_informed
_s, allow_new_levels = T)[, 1]
test_informed_s$Prediction0[test_informed_s$PredictionsPerc0 > 0.5] <- "Schizophre
nia"
```

```
## Warning: Unknown or uninitialised column: `Prediction0`.
```

```
test_informed_s$Prediction0[test_informed_s$PredictionsPerc0 <= 0.5] <- "Control"


test_informed_s$PredictionsPerc1 <- predict(pr_m1_fit_inf, newdata = test_informed
_s, allow_new_levels = T)[, 1]
test_informed_s$Prediction1[test_informed_s$PredictionsPerc1 > 0.5] <- "Schizophre
nia"
```

```
## Warning: Unknown or uninitialised column: `Prediction1`.
```

```
test_informed_s$Prediction1[test_informed_s$PredictionsPerc1 <= 0.5] <- "Control"


test_skeptic_s$PredictionsPerc0 <- predict(pr_m0_fit_skep, newdata = test_informed
_s, allow_new_levels = T)[, 1]
test_skeptic_s$Prediction0[test_skeptic_s$PredictionsPerc0 > 0.5] <- "Schizophreni
a"
```

```
## Warning: Unknown or uninitialised column: `Prediction0`.
```

```
test_skeptic_s$Prediction0[test_skeptic_s$PredictionsPerc0 <= 0.5] <- "Control"

test_skeptic_s$PredictionsPerc1 <- predict(pr_m1_fit_skep, newdata = test_informed
_s, allow_new_levels = T)[, 1]
test_skeptic_s$Prediction1[test_skeptic_s$PredictionsPerc1 > 0.5] <- "Schizophreni
a"
```

```
## Warning: Unknown or uninitialised column: `Prediction1`.
```

```
test_skeptic_s$Prediction1[test_skeptic_s$PredictionsPerc1 <= 0.5] <- "Control"

test_informed_s <- test_informed_s %>%
  mutate(
    Group = as.factor(Group),
    Prediction0 = as.factor(Prediction0),
    Prediction1 = as.factor(Prediction1)
  )

test_skeptic_s <- test_skeptic_s %>%
  mutate(
    Group = as.factor(Group),
    Prediction0 = as.factor(Prediction0),
    Prediction1 = as.factor(Prediction1)
  )
```

```
confusionMatrix(train_skeptic_s$Group, train_skeptic_s$Prediction0)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Control Schizophrenia
##   Control           778            23
##   Schizophrenia      15           784
##
##                Accuracy : 0.9762
##                  95% CI : (0.9675, 0.9831)
##     No Information Rate : 0.5044
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9525
##
##  Mcnemar's Test P-Value : 0.2561
##
##             Sensitivity : 0.9811
##             Specificity : 0.9715
##          Pos Pred Value : 0.9713
##          Neg Pred Value : 0.9812
##              Prevalence : 0.4956
##          Detection Rate : 0.4863
##    Detection Prevalence : 0.5006
##       Balanced Accuracy : 0.9763
##
##        'Positive' Class : Control
##
```

```
confusionMatrix(test_skeptic_s$Group, test_skeptic_s$Prediction0)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Control Schizophrenia
##   Control           198              9
##   Schizophrenia       5            188
##
##                Accuracy : 0.965
##                  95% CI : (0.942, 0.9807)
##     No Information Rate : 0.5075
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.93
##
##  Mcnemar's Test P-Value : 0.4227
##
##             Sensitivity : 0.9754
##             Specificity : 0.9543
##          Pos Pred Value : 0.9565
##          Neg Pred Value : 0.9741
##              Prevalence : 0.5075
##          Detection Rate : 0.4950
##    Detection Prevalence : 0.5175
##       Balanced Accuracy : 0.9648
##
##        'Positive' Class : Control
##
```

```
confusionMatrix(train_skeptic_s$Group, train_skeptic_s$Prediction1)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Control Schizophrenia
##   Control           797              4
##   Schizophrenia       2            797
##
##                Accuracy : 0.9962
##                  95% CI : (0.9919, 0.9986)
##     No Information Rate : 0.5006
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9925
##
##  Mcnemar's Test P-Value : 0.6831
##
##             Sensitivity : 0.9975
##             Specificity : 0.9950
##          Pos Pred Value : 0.9950
##          Neg Pred Value : 0.9975
##              Prevalence : 0.4994
##          Detection Rate : 0.4981
##    Detection Prevalence : 0.5006
##       Balanced Accuracy : 0.9963
##
##        'Positive' Class : Control
##
```

```
confusionMatrix(test_skeptic_s$Group, test_skeptic_s$Prediction1)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Control Schizophrenia
##   Control           199              8
##   Schizophrenia       2            191
##
##                Accuracy : 0.975
##                  95% CI : (0.9545, 0.9879)
##     No Information Rate : 0.5025
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.95
##
##  Mcnemar's Test P-Value : 0.1138
##
##             Sensitivity : 0.9900
##             Specificity : 0.9598
##          Pos Pred Value : 0.9614
##          Neg Pred Value : 0.9896
##              Prevalence : 0.5025
##          Detection Rate : 0.4975
##    Detection Prevalence : 0.5175
##       Balanced Accuracy : 0.9749
##
##        'Positive' Class : Control
##
```

```
confusionMatrix(train_informed_s$Group, train_informed_s$Prediction0)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Control Schizophrenia
##   Control           777            24
##   Schizophrenia      15           784
##
##                Accuracy : 0.9756
##                  95% CI : (0.9668, 0.9826)
##     No Information Rate : 0.505
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9513
##
##  Mcnemar's Test P-Value : 0.2002
##
##             Sensitivity : 0.9811
##             Specificity : 0.9703
##          Pos Pred Value : 0.9700
##          Neg Pred Value : 0.9812
##              Prevalence : 0.4950
##          Detection Rate : 0.4856
##    Detection Prevalence : 0.5006
##       Balanced Accuracy : 0.9757
##
##        'Positive' Class : Control
##
```

```
confusionMatrix(test_informed_s$Group, test_informed_s$Prediction0)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Control Schizophrenia
##   Control           198              1
##   Schizophrenia       5            196
##
##                Accuracy : 0.985
##                  95% CI : (0.9676, 0.9945)
##     No Information Rate : 0.5075
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.97
##
##  Mcnemar's Test P-Value : 0.2207
##
##             Sensitivity : 0.9754
##             Specificity : 0.9949
##          Pos Pred Value : 0.9950
##          Neg Pred Value : 0.9751
##              Prevalence : 0.5075
##          Detection Rate : 0.4950
##    Detection Prevalence : 0.4975
##       Balanced Accuracy : 0.9851
##
##        'Positive' Class : Control
##
```

```
confusionMatrix(train_informed_s$Group, train_informed_s$Prediction1)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Control Schizophrenia
##   Control           797              4
##   Schizophrenia       2            797
##
##                Accuracy : 0.9962
##                  95% CI : (0.9919, 0.9986)
##     No Information Rate : 0.5006
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9925
##
##  Mcnemar's Test P-Value : 0.6831
##
##             Sensitivity : 0.9975
##             Specificity : 0.9950
##          Pos Pred Value : 0.9950
##          Neg Pred Value : 0.9975
##              Prevalence : 0.4994
##          Detection Rate : 0.4981
##    Detection Prevalence : 0.5006
##       Balanced Accuracy : 0.9963
##
##        'Positive' Class : Control
##
```

```
confusionMatrix(test_informed_s$Group, test_informed_s$Prediction1)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      Control Schizophrenia
##   Control           199              0
##   Schizophrenia       2            199
##
##                Accuracy : 0.995
##                  95% CI : (0.9821, 0.9994)
##     No Information Rate : 0.5025
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.99
##
##  Mcnemar's Test P-Value : 0.4795
##
##             Sensitivity : 0.9900
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9900
##              Prevalence : 0.5025
##          Detection Rate : 0.4975
##    Detection Prevalence : 0.4975
##       Balanced Accuracy : 0.9950
##
##        'Positive' Class : Control
##
```

Discuss whether performance and feature importance is as expected

##Part 3 - Applying the machine learning pipeline to empirical data

```
real_data <- read_csv("assignment_3_data.csv")
```

```
## Rows: 1889 Columns: 398
## ── Column specification ────────────────────────────────────────────────────
─
## Delimiter: ","
## chr   (5): NewID, Diagnosis, Language, Gender, Trial
## dbl (393): PatID, Corpus, Duration_Praat, F0_Mean_Praat, F0_SD_Praat, Intens...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this messag
e.
```

```
real_data <- real_data %>%
  select(-Language) %>%
  select(-Corpus) %>%
  select(-NewID)
```

Apply your machine learning pipeline to the empirical data

Warning: in simulated data you only have 10 features, now you have many more - Consider the impact a higher number of features will have on your ML inference and decide if you want to cut down the number of features before running the pipeline (alternatibvely expand the pipeline to add feature selection)

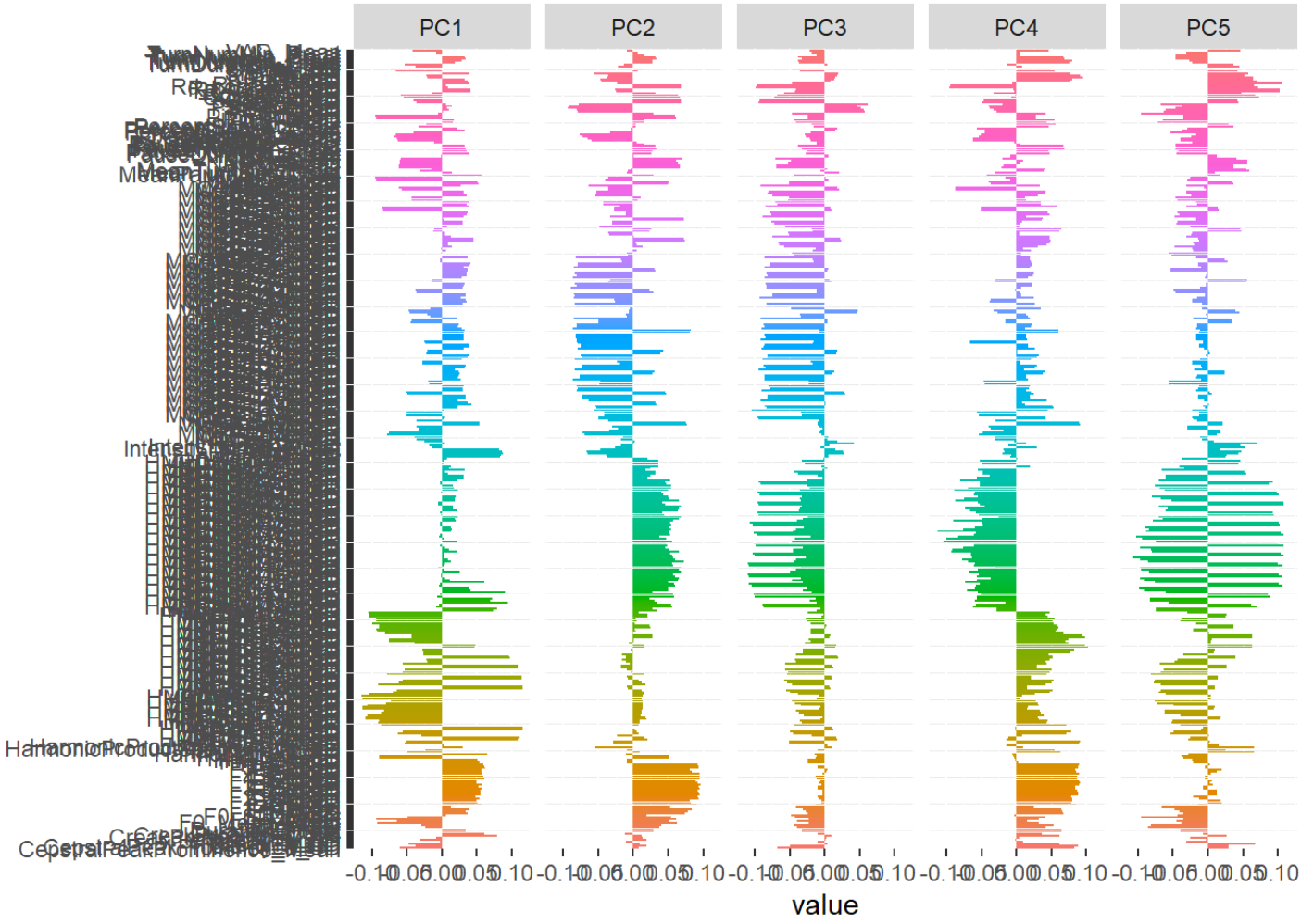
data budgeting and pree-processing of the data

```
set.seed(304)

split_real_data <- initial_split(real_data, prop = 4/5, strata = Gender)

train_data <- training(split_real_data)
test_data <- testing(split_real_data)

train_data <- train_data %>%
  select(-Gender) %>%
  select(-Trial)

test_data <- test_data %>%
  select(-Gender) %>%
  select(-Trial)

train_data$PatID <- as.factor(train_data$PatID)

test_data$PatID <- as.factor(test_data$PatID)
```

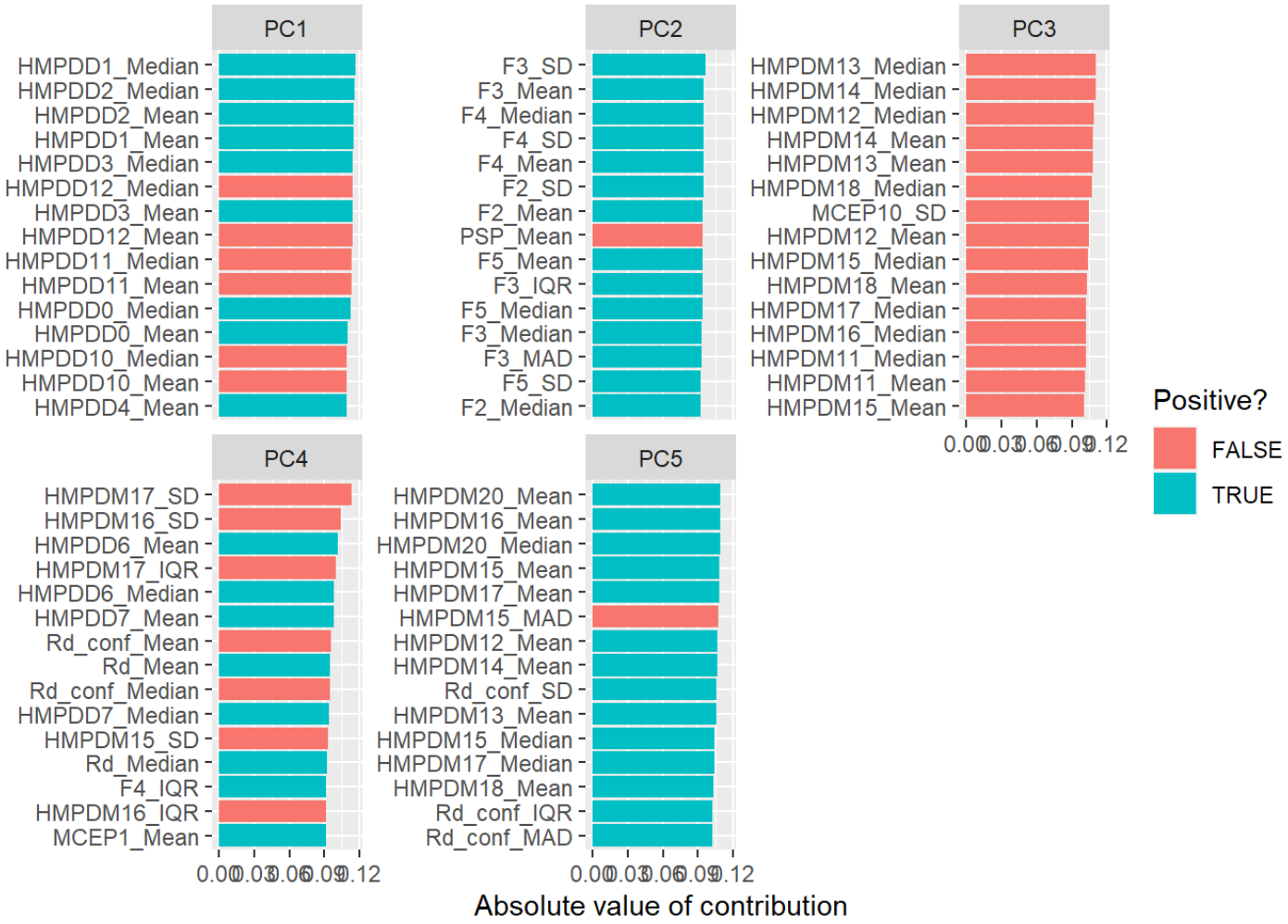
```
recipe_real <- train_data %>%
  recipe(Diagnosis~.) %>%
  update_role(PatID, new_role = 'PatID') %>%
  step_scale(all_numeric()) %>%
  step_center(all_numeric()) %>%
  prep(training=train_data, retain=TRUE)

train_data_s <- juice(recipe_real)
test_data_s <- bake(recipe_real, new_data = test_data)
```

Principle component analysis

```
pca_recipe <- recipe(Diagnosis~., data = train_data_s) %>%
  update_role(PatID, new_role = "id") %>%
  step_pca(all_numeric(), id = "pca")%>%
  prep()

tidy_pca <- tidy(pca_recipe, 1)

bake_pca <- bake(pca_recipe, train_data_s)

pca_b_test <- bake(pca_recipe, test_data_s)

tidy_pca %>%
  filter(component %in% paste0("PC", 1:5)) %>%
  mutate(component = fct_inorder(component)) %>%
  ggplot(aes(value, terms, fill = terms)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~component, nrow = 1) +
  labs(y = NULL)
```



```
tidy_pca %>%
  filter(component %in% paste0("PC", 1:5)) %>%
  group_by(component) %>%
  top_n(15, abs(value)) %>%
  ungroup() %>%
  mutate(terms = reorder_within(terms, abs(value), component)) %>%
  ggplot(aes(abs(value), terms, fill = value > 0)) +
  geom_col() +
  facet_wrap(~component, scales = "free_y") +
  scale_y_reordered() +
  labs(
    x = "Absolute value of contribution",
    y = NULL, fill = "Positive?"
  )
```



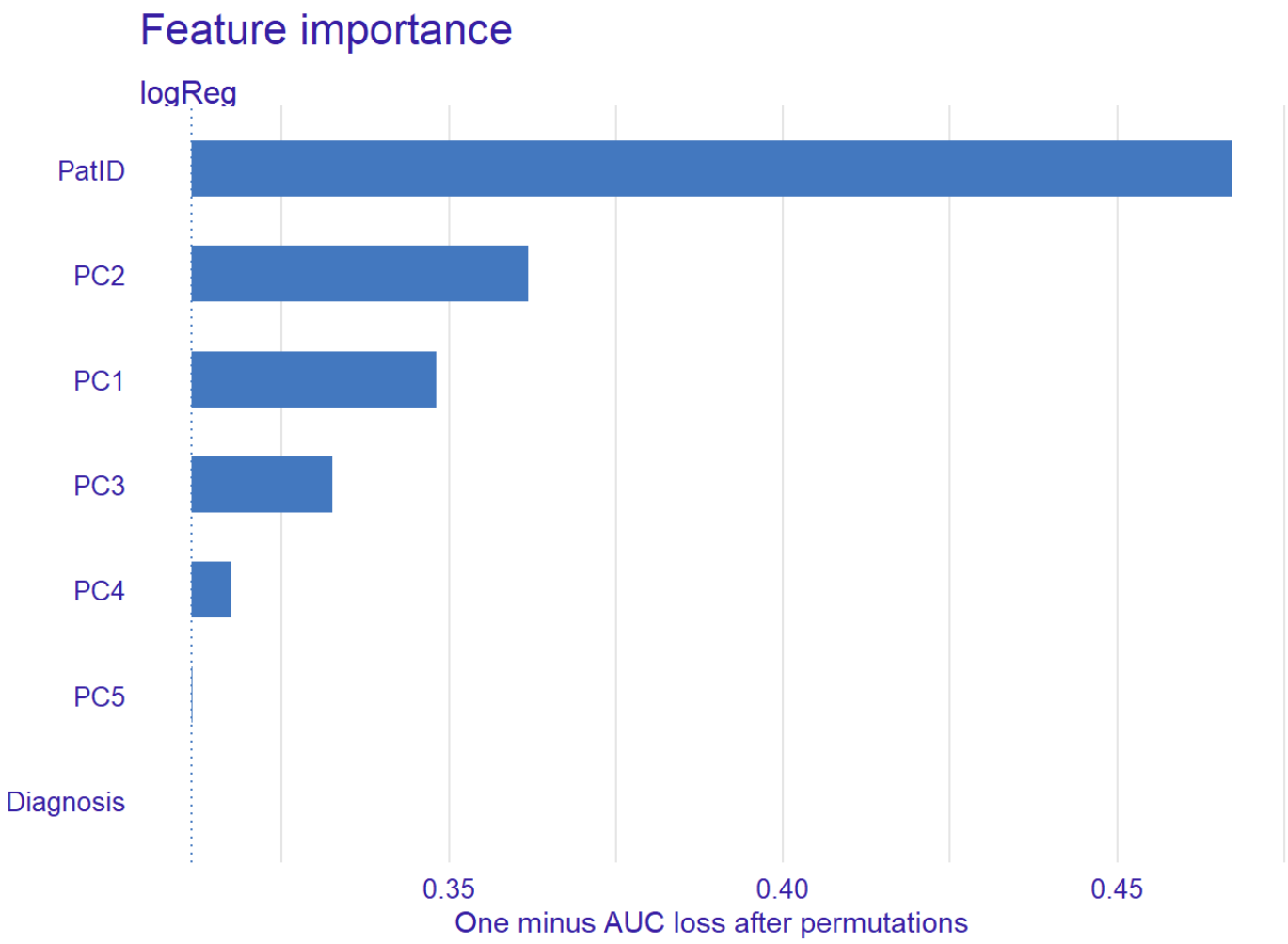

```
??reorder_within
```

```
## starting httpd help server ... done
```

##Manuela ### feature selection

```
d_data <- bake_pca %>%
  mutate(PatID = NULL, Trail = NULL, NewID = NULL, Gender = NULL, Language = NULL,
Corpus = NULL)

LogisticRegression_train<- logistic_reg() %>%
  set_mode('classification') %>%
  set_engine('glm') %>%
  fit(Diagnosis ~ . , data = bake_pca)
```

```
explainer_lm <-
  explain_tidymodels(
    LogisticRegression_train,
    data = bake_pca,
    y = as.numeric(d_data$Diagnosis) -1,
    label = 'logReg',
    verbose = FALSE
  )

explainer_lm %>% model_parts() %>%  plot(show_boxplots = FALSE) + ggtitle('Feature
importance', '')
```



fit and assess a classification algorithm on the training data

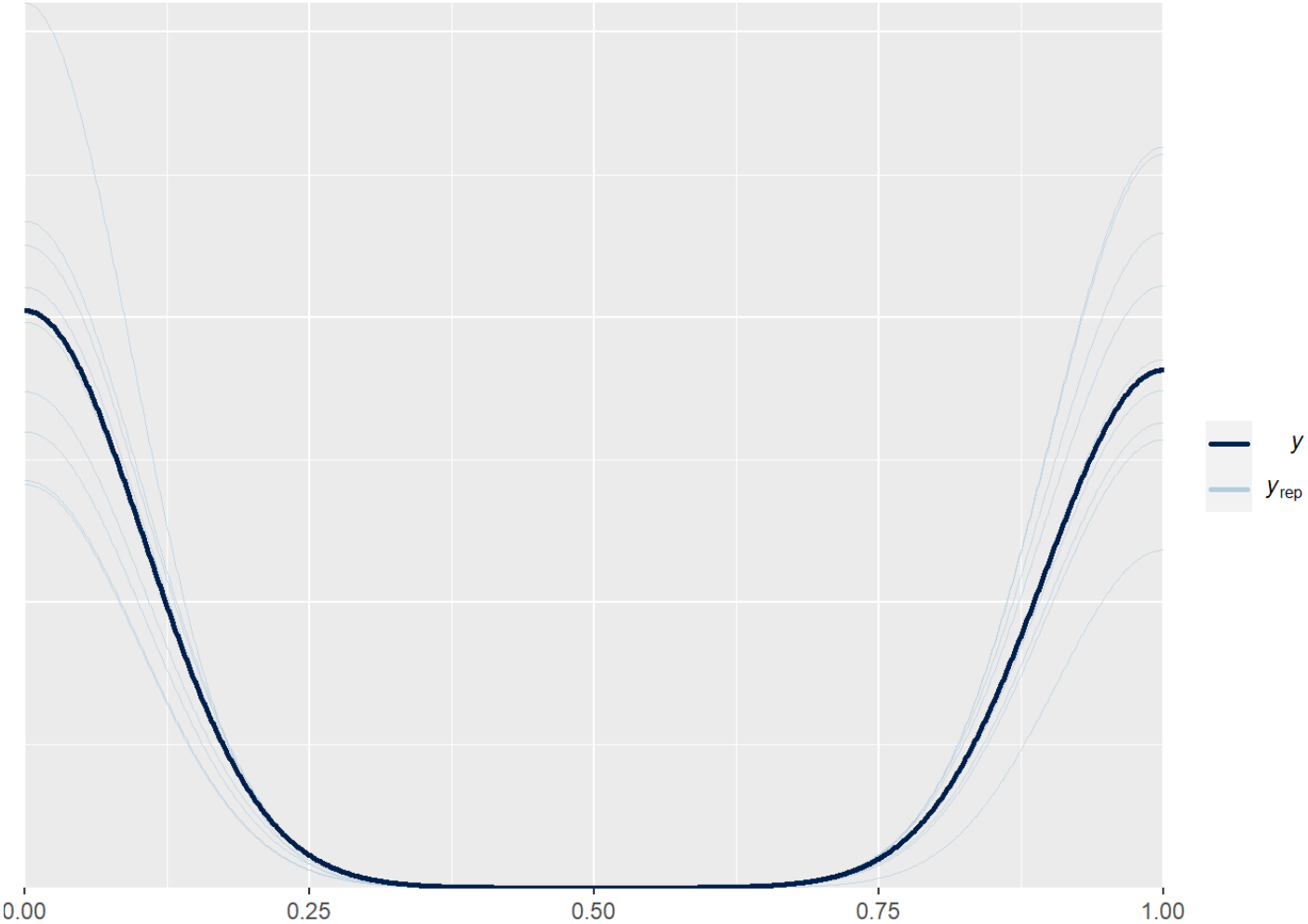
```
form1 <- bf(Diagnosis~1+PC2+PC1+PC3+PC4+(1|PatID))
get_prior(form1, bake_pca, family = bernoulli)
```

```
##                 prior     class      coef group resp dpar nlpar lb ub
##                (flat)         b
##                (flat)         b       PC1
##                (flat)         b       PC2
##                (flat)         b       PC3
##                (flat)         b       PC4
##  student_t(3, 0, 2.5) Intercept
##  student_t(3, 0, 2.5)        sd                                   0
##  student_t(3, 0, 2.5)        sd           PatID                   0
##  student_t(3, 0, 2.5)        sd Intercept PatID                   0
##        source
##       default
##  (vectorized)
##  (vectorized)
##  (vectorized)
##  (vectorized)
##       default
##       default
##  (vectorized)
##  (vectorized)
```

```
prior_f1 <- c(
  prior(normal(0, 1), class=Intercept),
  prior(normal(0, 1), class=sd),
  prior(normal(0, 0.3), class=b)
)
```

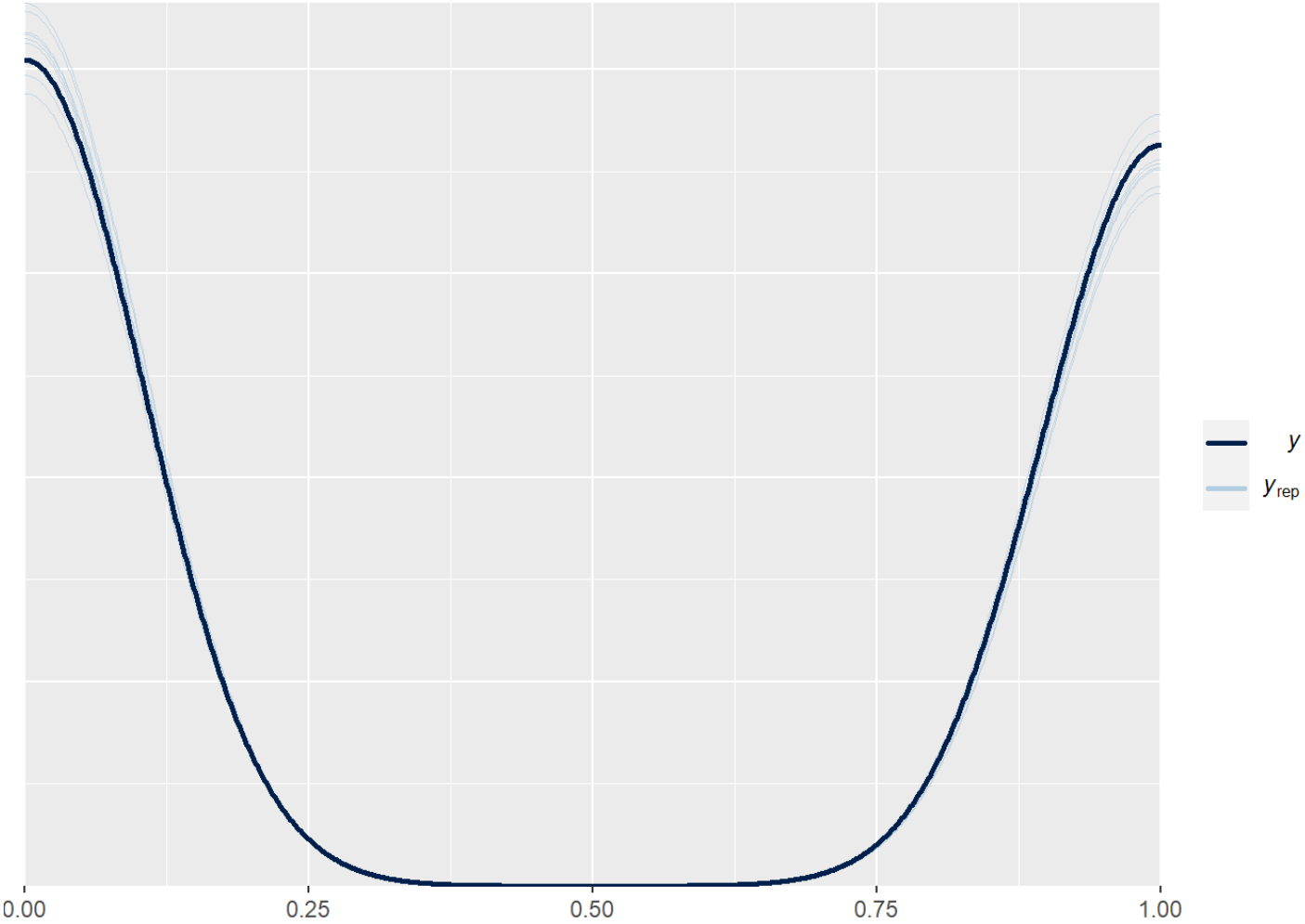
Fitting the model with only the priors

```
pr_m1 <- brm(
  form1,
  data = bake_pca,
  prior = prior_f1,
  family = bernoulli,
  refresh=0,
  sample_prior = 'only',
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)

pp_check(pr_m1)
```

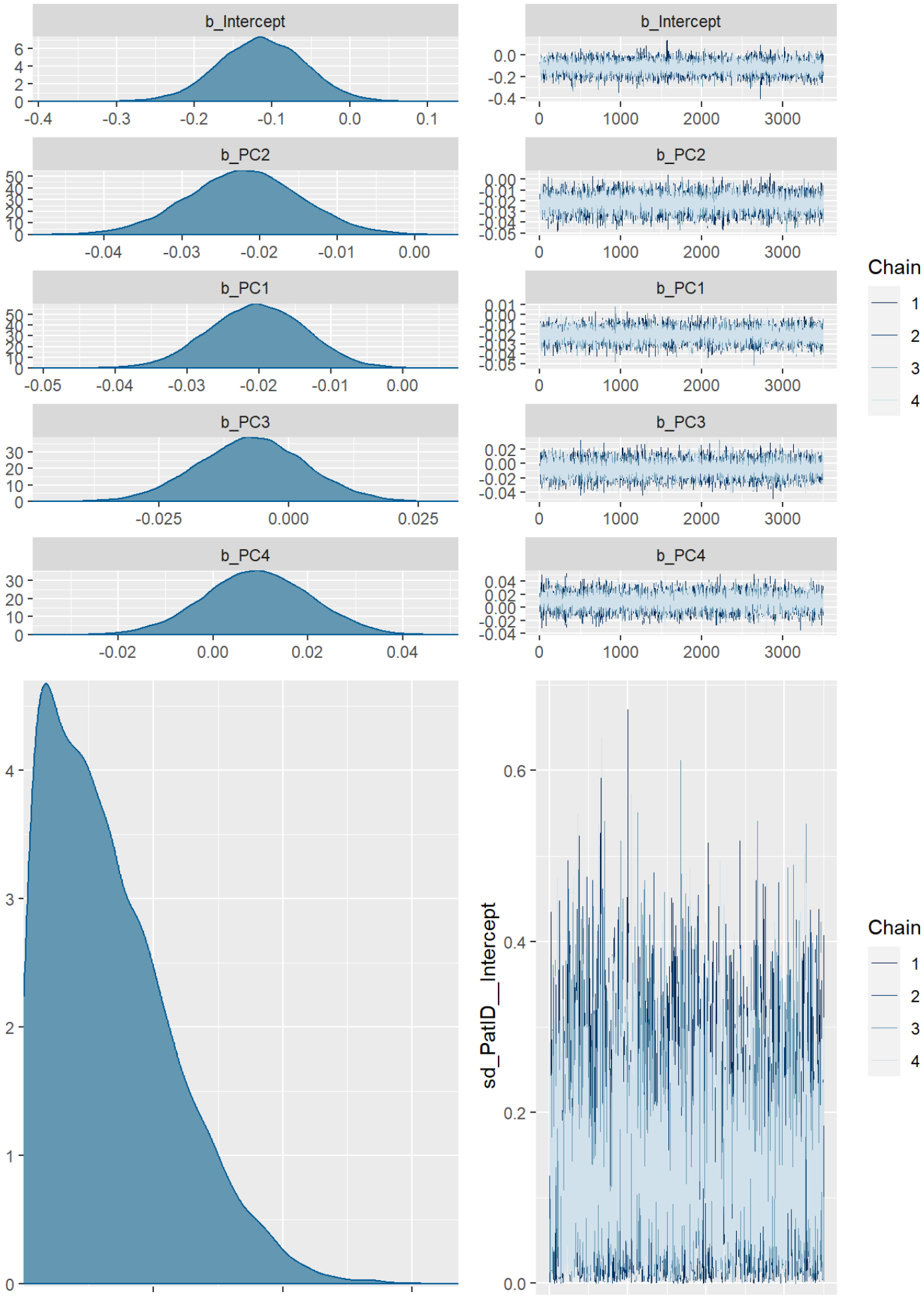


Fitting model on the data

```
pr_m1_fit <- brm(
  form1,
  data = bake_pca,
  prior = prior_f1,
  family = bernoulli,
  refresh=0,
  sample_prior = TRUE,
  iter=6000,
  warmup = 2500,
  backend = "cmdstanr",
  threads = threading(2),
  chains = 4,
  cores = 4,
  control = list(
    adapt_delta = 0.9,
    max_treedepth = 20)
)

pp_check(pr_m1_fit)
```



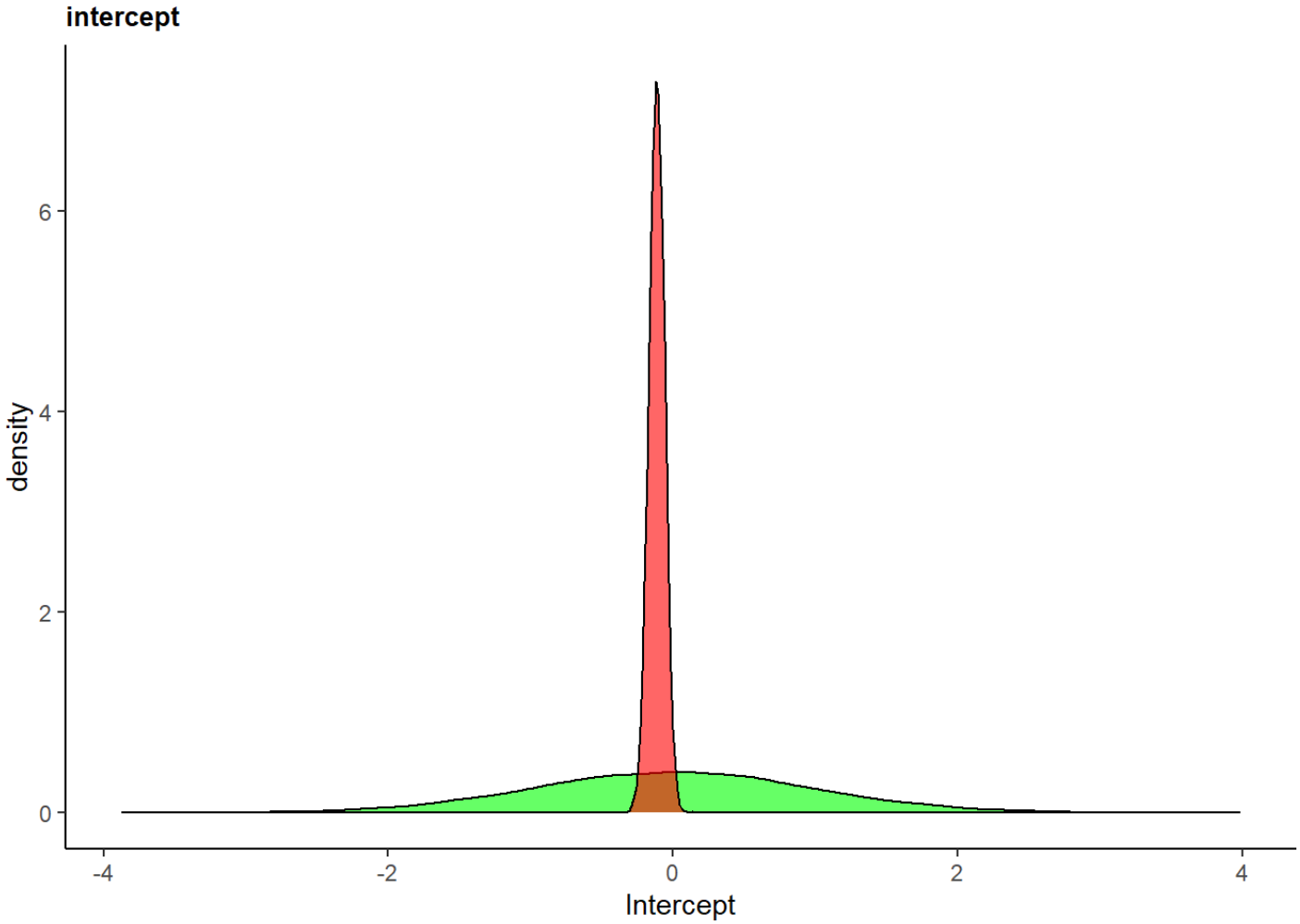

```
plot(pr_m1_fit)
```


b_Intercept
b_PC2
b_PC1
b_PC3
b_PC4
Chain
1
2
3
4
sd_PatID__Intercept


```
summary(pr_m1_fit)
```

Prior posterior update check

```
Posterior_f1 <- as_draws_df(pr_m1_fit)

ggplot(Posterior_f1) +
  geom_density(aes(prior_Intercept), fill="green", color="black",alpha=0.6) +
  geom_density(aes(b_Intercept), fill="red", color="black",alpha=0.6) +
  xlab('Intercept') +
  theme_classic()+
  ggtitle("intercept")+
      theme(plot.title = element_text(size = 10, face = "bold"))
```



assess performance on the test set

```
bake_pca$PredictionsPerc1 <- predict(pr_m1_fit)[, 1]

bake_pca$Prediction1[bake_pca$PredictionsPerc1 > 0.5] <- "SCZ"
```

```
## Warning: Unknown or uninitialised column: `Prediction1`.
```

```
bake_pca$Prediction1[bake_pca$PredictionsPerc1 <= 0.5] <- "CT"

bake_pca <- bake_pca %>%
  mutate(
    Diagnosis = as.factor(Diagnosis),
    Prediction1 = as.factor(Prediction1))

pca_b_test$PredictionsPerc1 <- predict(pr_m1_fit, newdata = pca_b_test, allow_new_
levels = T)[, 1]



pca_b_test$Prediction1[pca_b_test$PredictionsPerc1 > 0.5] <- "SCZ"
```

```
## Warning: Unknown or uninitialised column: `Prediction1`.
```

```
pca_b_test$Prediction1[pca_b_test$PredictionsPerc1 <= 0.5] <- "CT"

pca_b_test <- pca_b_test %>%
  mutate(
    Diagnosis = as.factor(Diagnosis),
    Prediction1 = as.factor(Prediction1)
  )
```

Accuracy

```
confusionMatrix(bake_pca$Diagnosis, bake_pca$Prediction1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  CT SCZ
##        CT  545 251
##        SCZ 444 270
##
##                Accuracy : 0.5397
##                  95% CI : (0.5142, 0.5651)
##     No Information Rate : 0.655
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0637
##
##  Mcnemar's Test P-Value : 3.265e-13
##
##             Sensitivity : 0.5511
##             Specificity : 0.5182
##          Pos Pred Value : 0.6847
##          Neg Pred Value : 0.3782
##              Prevalence : 0.6550
##          Detection Rate : 0.3609
##    Detection Prevalence : 0.5272
##       Balanced Accuracy : 0.5346
##
##        'Positive' Class : CT
##
```

```
confusionMatrix(pca_b_test$Diagnosis, pca_b_test$Prediction1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  CT SCZ
##        CT  115  78
##        SCZ 104  82
##
##                Accuracy : 0.5198
##                  95% CI : (0.4682, 0.5711)
##     No Information Rate : 0.5778
##     P-Value [Acc > NIR] : 0.99011
##
##                   Kappa : 0.0368
##
##  Mcnemar's Test P-Value : 0.06386
##
##             Sensitivity : 0.5251
##             Specificity : 0.5125
##          Pos Pred Value : 0.5959
##          Neg Pred Value : 0.4409
##              Prevalence : 0.5778
##          Detection Rate : 0.3034
##    Detection Prevalence : 0.5092
##       Balanced Accuracy : 0.5188
##
##        'Positive' Class : CT
##
```

###discuss whether performance and feature importance is as expected