ProjectSubmission

HamzaAfzalAshraf

2025-03-07

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## 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
```

```
## Warning: package 'factoextra' was built under R version 4.4.3
```

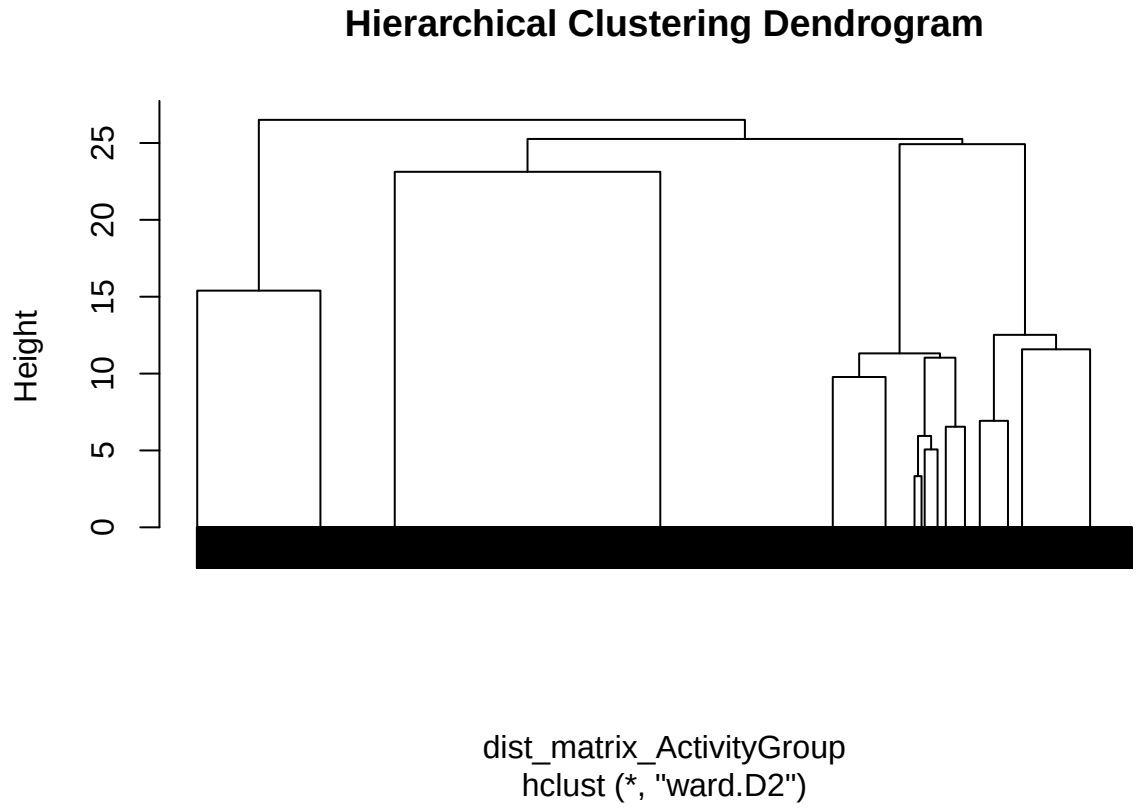
```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
## Loading required package: Rcpp
## Loading 'brms' package (version 2.22.0). Useful instructions
## can be found by typing help('brms'). A more detailed introduction
## to the package is available through vignette('brms_overview').
##
## Attaching package: 'brms'
##
## The following object is masked from 'package:stats':
##
##     ar
##
## This is cmdstanr version 0.8.0
## - CmdStanR documentation and vignettes: mc-stan.org/cmdstanr
## - CmdStan path: C:/Users/HamzaPC/.cmdstan/cmdstan-2.36.0
## - CmdStan version: 2.36.0
```

Developing Clusters

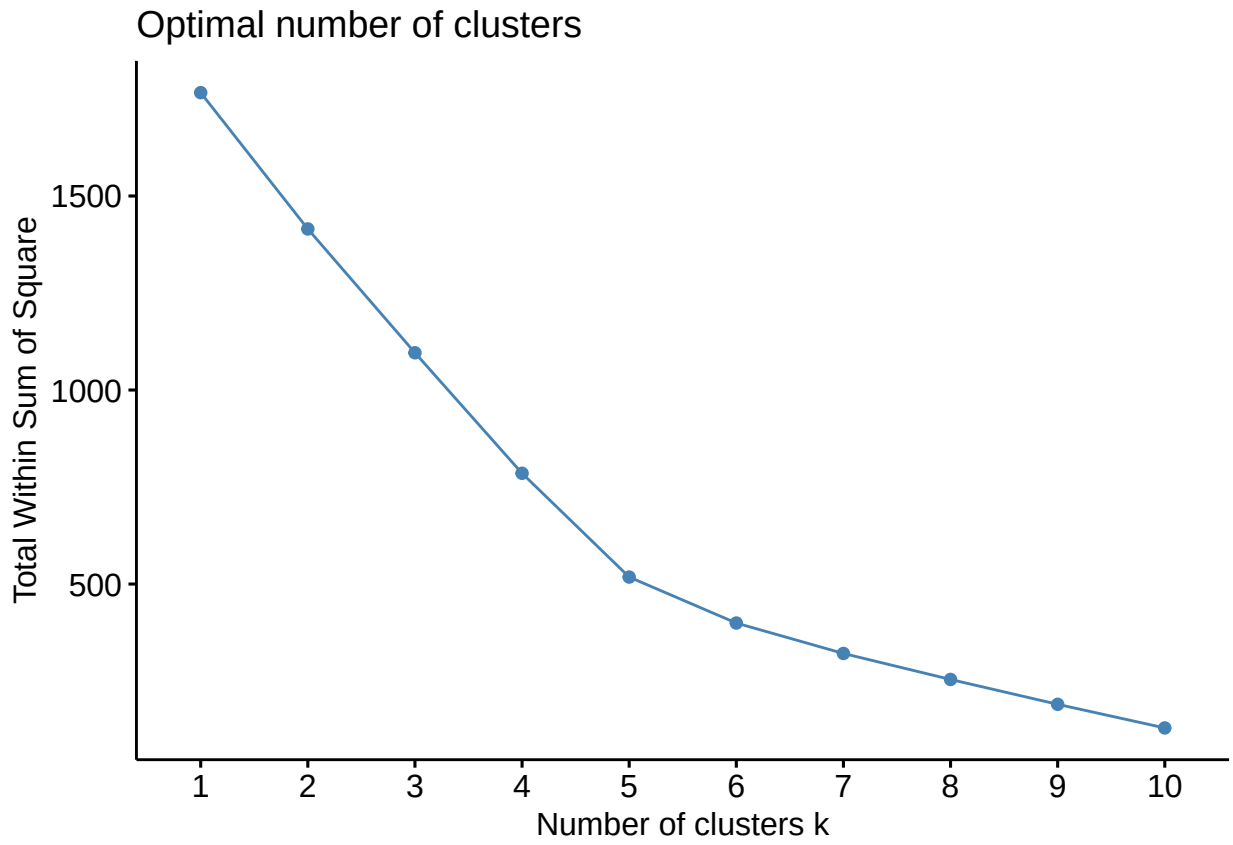
ActivityGroupCluster

```
data <- read.csv("Student_performance.csv")

df_cluster_ActivityGroup <- data %>%
  select(Extracurricular, Sports, Music, Volunteering)

dist_matrix_ActivityGroup <- dist(df_cluster_ActivityGroup, method = "euclidean")

hc_ActivityGroup <- hclust(dist_matrix_ActivityGroup, method = "ward.D2")
```

```r
plot(hc_ActivityGroup, labels = FALSE, main = "Hierarchical Clustering Dendrogram")
```

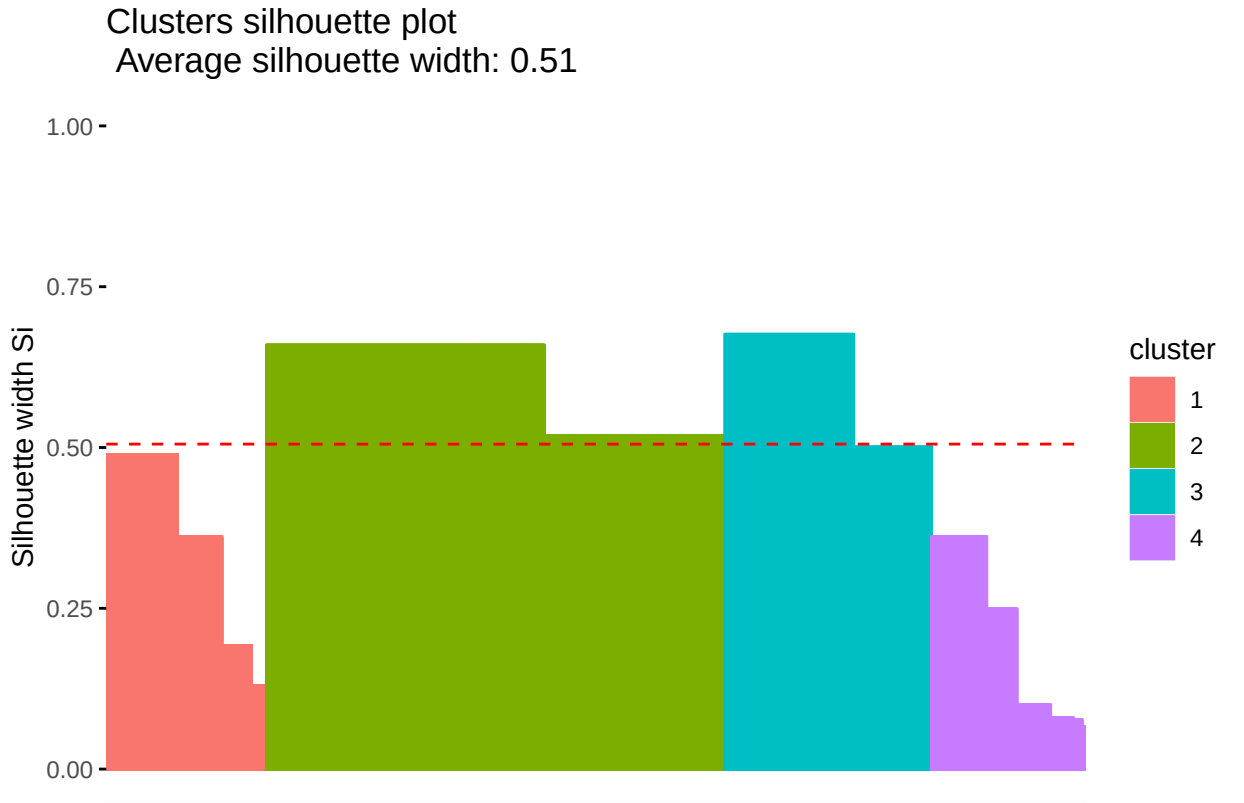



```r
library(factoextra)
fviz_nbclust(df_cluster_ActivityGroup, hcut, method = "wss")
```



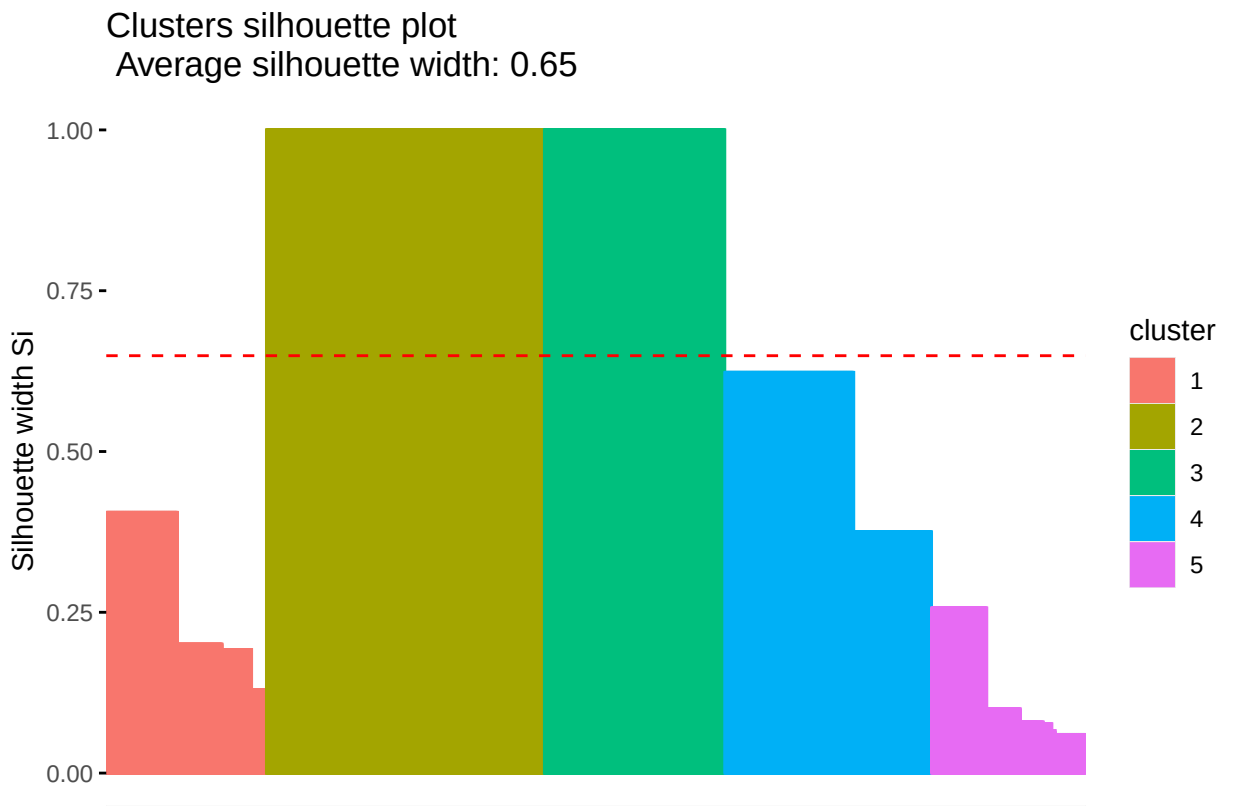

```
library(cluster)
sil_ActivityGroup <- silhouette(cutree(hc_ActivityGroup, k = 4), dist_matrix_ActivityGroup)
fviz_silhouette(sil_ActivityGroup)
```

```
##   cluster size ave.sil.width
## 1       1  391          0.37
## 2       2 1120          0.60
## 3       3  505          0.61
## 4       4  376          0.22
```




```
library(cluster)
sil_ActivityGroup <- silhouette(cutree(hc_ActivityGroup, k = 5), dist_matrix_ActivityGroup)
fviz_silhouette(sil_ActivityGroup)
```

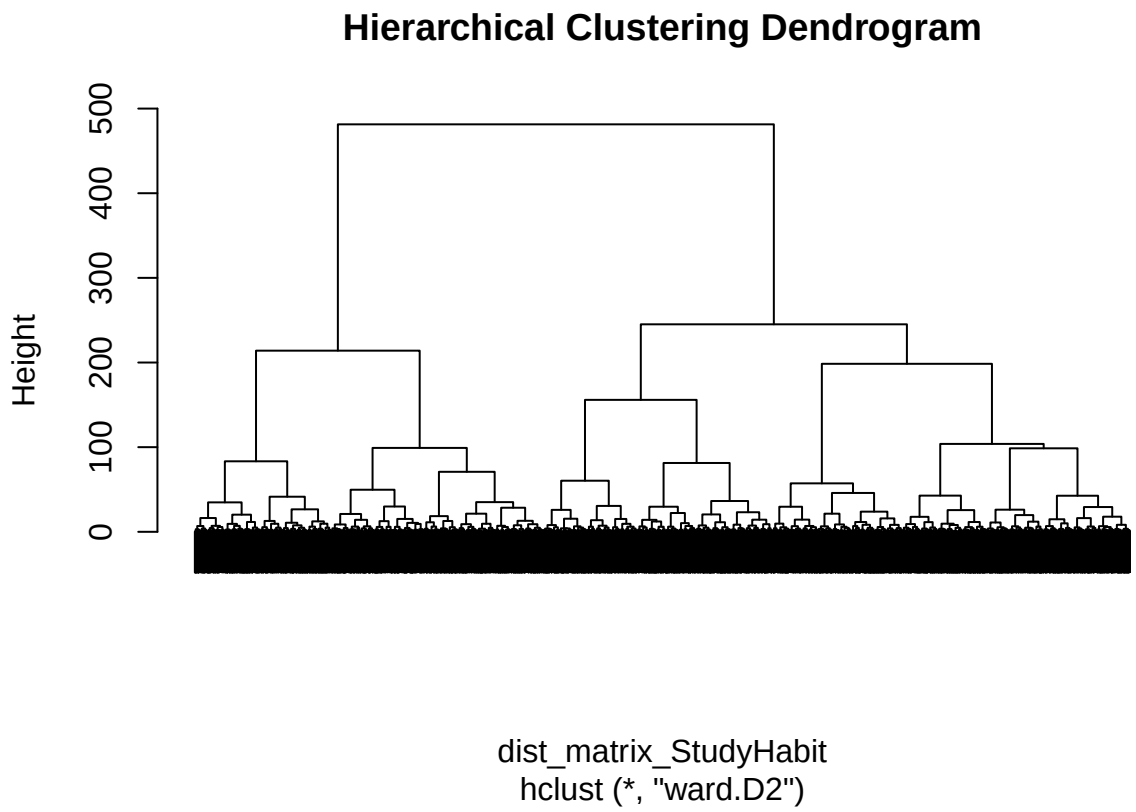
```
##   cluster size ave.sil.width
## 1       1  391          0.28
## 2       2  679          1.00
## 3       3  441          1.00
## 4       4  505          0.53
## 5       5  376          0.14
```



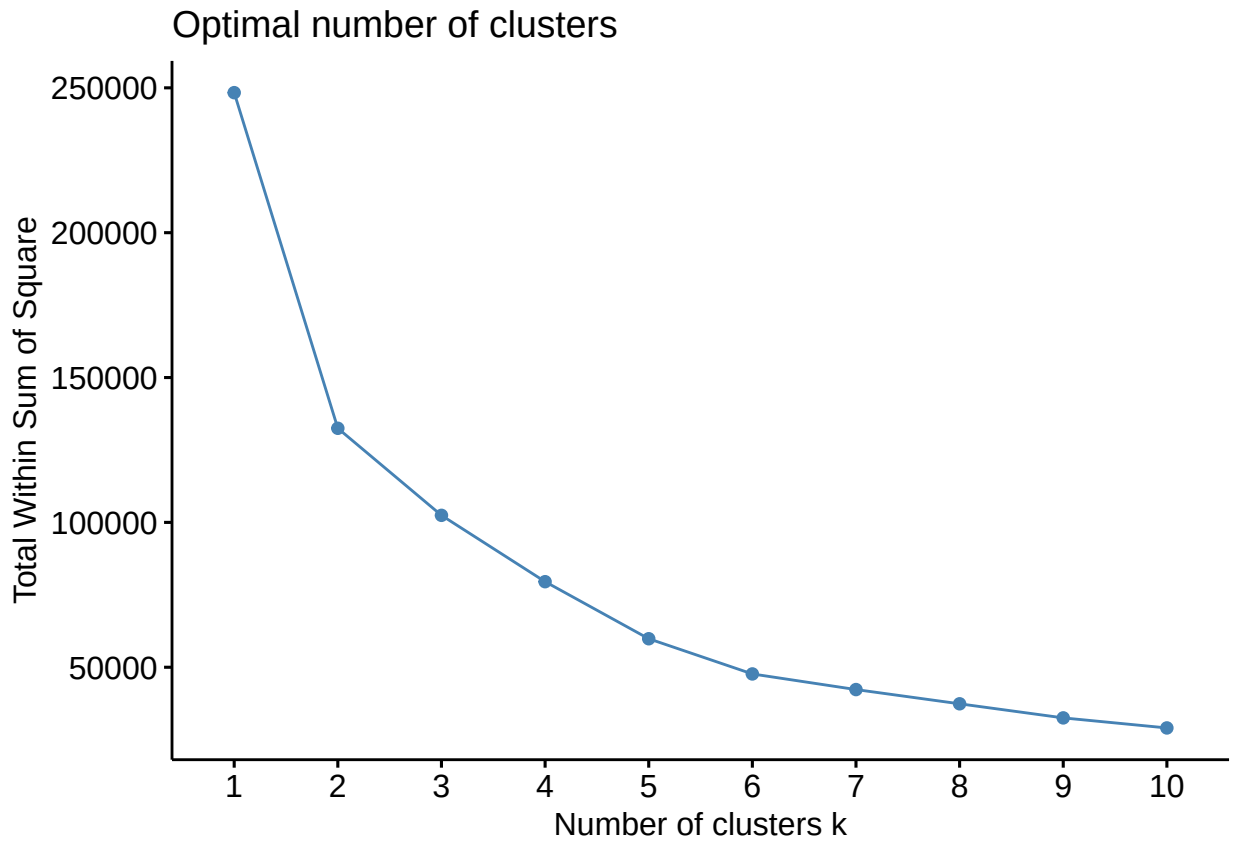
Since AWS values lies between [-1,1] and Higher values are better, we will go with k=5 which is also taken from the WSS Elbow

StudyHabitCluster

```
data <- read.csv("Student_performance.csv")

df_cluster_StudyHabit <- data %>%
  select(StudyTimeWeekly, Absences, Tutoring)

dist_matrix_StudyHabit <- dist(df_cluster_StudyHabit, method = "euclidean")

hc_StudyHabit <- hclust(dist_matrix_StudyHabit, method = "ward.D2")


plot(hc_StudyHabit, labels = FALSE, main = "Hierarchical Clustering Dendrogram")
```

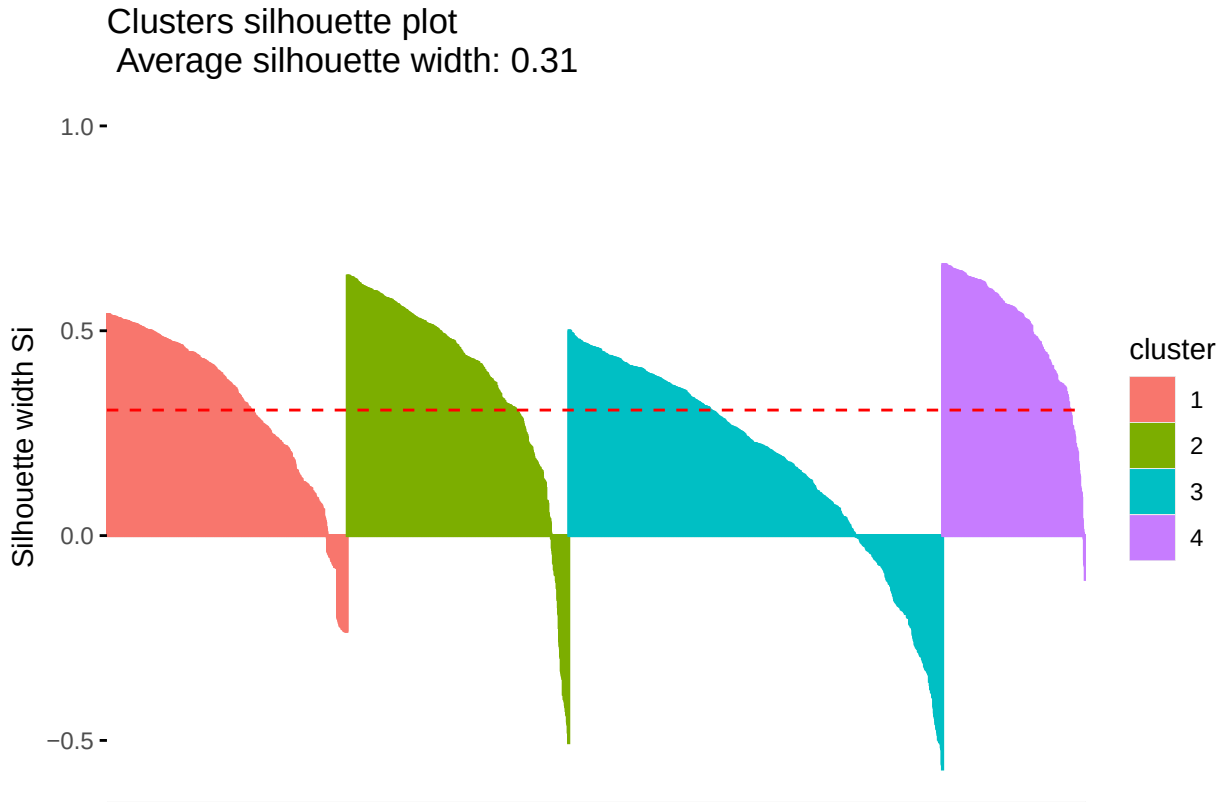



```
library(factoextra)
fviz_nbclust(df_cluster_StudyHabit, hcut, method = "wss")
```



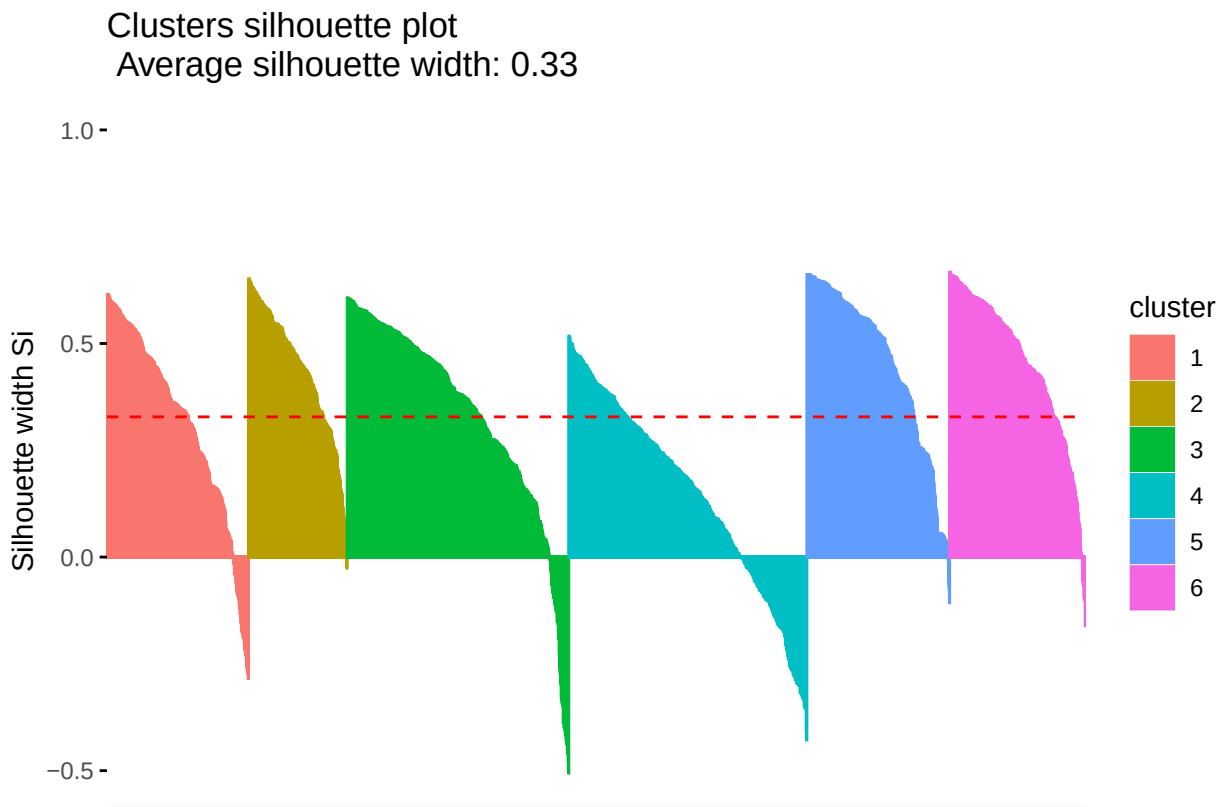

```
library(cluster)
sil_StudyHabit <- silhouette(cutree(hc_StudyHabit, k = 4), dist_matrix_StudyHabit)
fviz_silhouette(sil_StudyHabit)
```

```
##   cluster size ave.sil.width
## 1       1  587          0.32
## 2       2  542          0.39
## 3       3  914          0.18
## 4       4  349          0.50
```



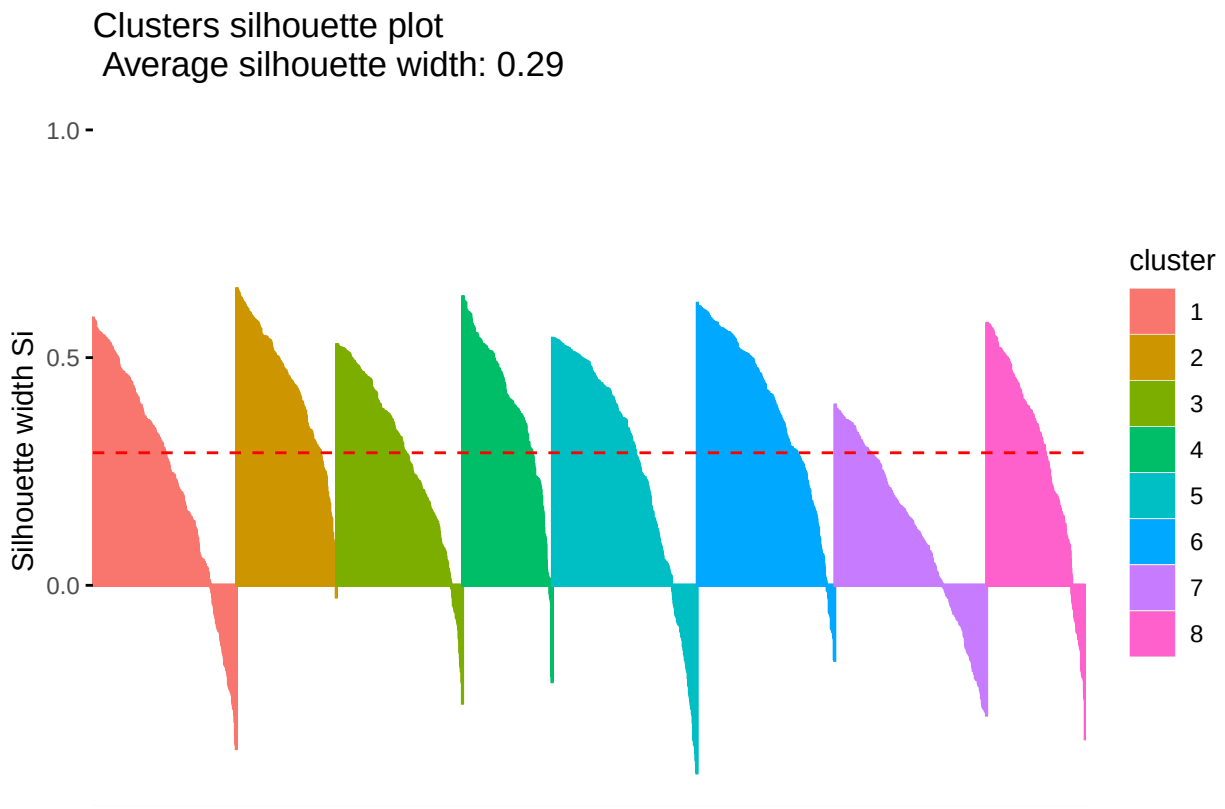

```
library(cluster)
sil_StudyHabit <- silhouette(cutree(hc_StudyHabit, k = 6), dist_matrix_StudyHabit)
fviz_silhouette(sil_StudyHabit)
```

```
##   cluster size ave.sil.width
## 1       1  346          0.31
## 2       2  241          0.44
## 3       3  542          0.33
## 4       4  582          0.14
## 5       5  349          0.45
## 6       6  332          0.45
```



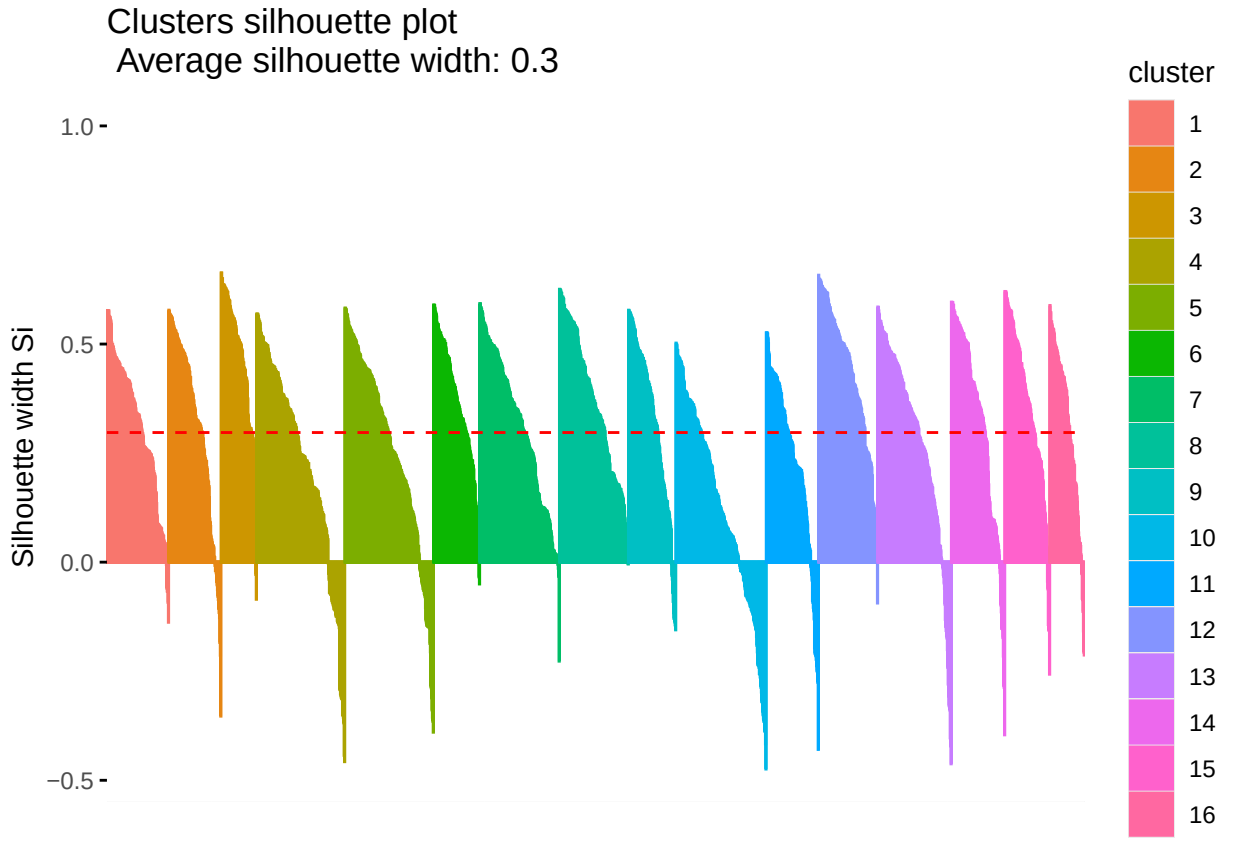

```
library(cluster)
sil_StudyHabit <- silhouette(cutree(hc_StudyHabit, k = 8), dist_matrix_StudyHabit)
fviz_silhouette(sil_StudyHabit)
```

```
##   cluster size ave.sil.width
## 1       1  346          0.24
## 2       2  241          0.44
## 3       3  304          0.29
## 4       4  216          0.39
## 5       5  349          0.27
## 6       6  332          0.38
## 7       7  366          0.12
## 8       8  238          0.29
```




```
library(cluster)
sil_StudyHabit <- silhouette(cutree(hc_StudyHabit, k = 16), dist_matrix_StudyHabit)
fviz_silhouette(sil_StudyHabit)
```

```
##    cluster size ave.sil.width
## 1        1  151          0.31
## 2        2  128          0.31
## 3        3   87          0.47
## 4        4  216          0.23
## 5        5  217          0.26
## 6        6  113          0.37
## 7        7  195          0.31
## 8        8  169          0.37
## 9        9  116          0.32
## 10      10  221          0.12
## 11      11  128          0.23
## 12      12  145          0.43
## 13      13  180          0.28
## 14      14  131          0.31
## 15      15  110          0.36
## 16      16   85          0.31
```



#1) Prior Sensitivity Analysis for StudyHabits Cluster

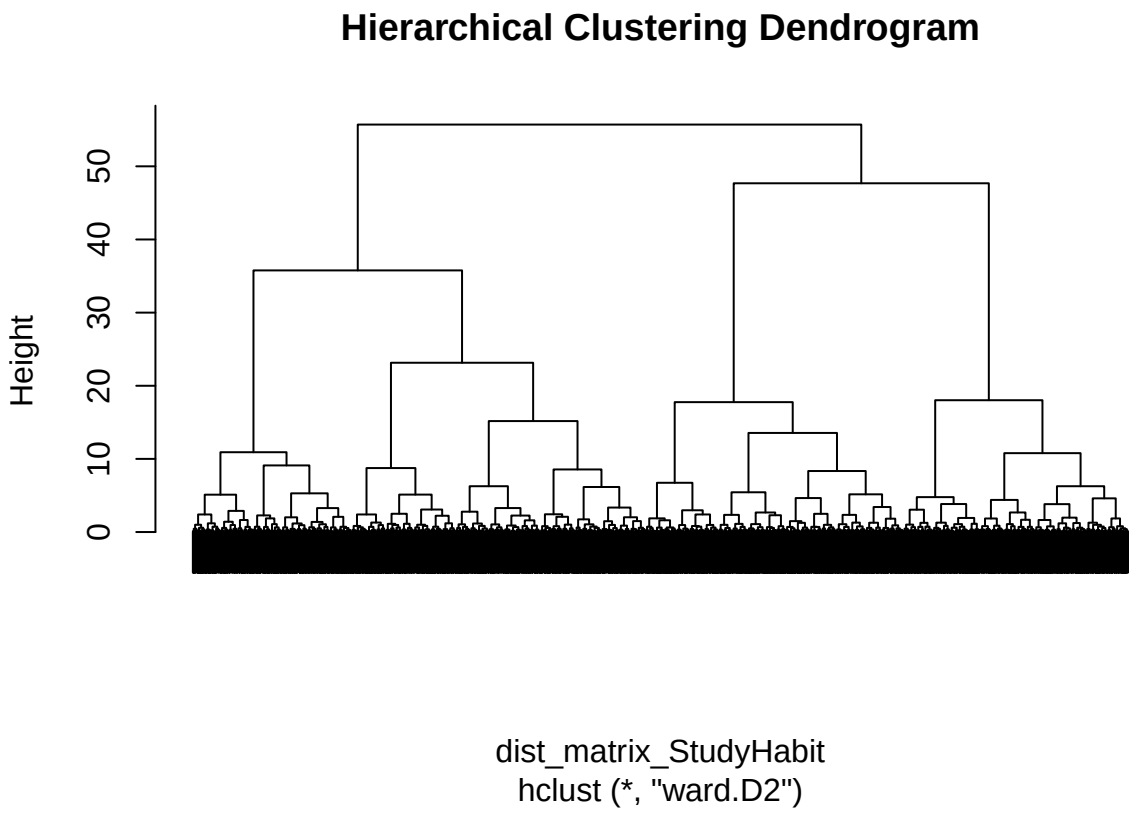
```
data <- read.csv("Student_performance.csv")

data$GradeClass <- factor(data$GradeClass,
                          levels = c(0, 1, 2, 3, 4),
                          labels = c("A", "B", "C", "D", "F"),
                          ordered = TRUE)
```

We focus on changing the prior for regression coefficients while keeping the prior for random effects constant. This way, we isolate the impact of different priors on the regression coefficients.

```
# Narrower Prior (More informative):
priors_alt1 <- c(
  set_prior("normal(0, 1)", class = "b"),  # Narrow prior for regression coefficients
  set_prior("cauchy(0, 1)", class = "sd"),  # Same prior for random effects (Level)
  set_prior("student_t(3, 0, 2)", class = "Intercept")  # Informative prior for intercept
)

# Broader Prior (Weakly informative):
priors_alt2 <- c(
  set_prior("normal(0, 10)", class = "b"),  # Broad prior for regression coefficients
  set_prior("cauchy(0, 1)", class = "sd"),  # Same prior for random effects (Level)
  set_prior("student_t(3, 0, 2)", class = "Intercept")  # Informative prior for intercept
)
```

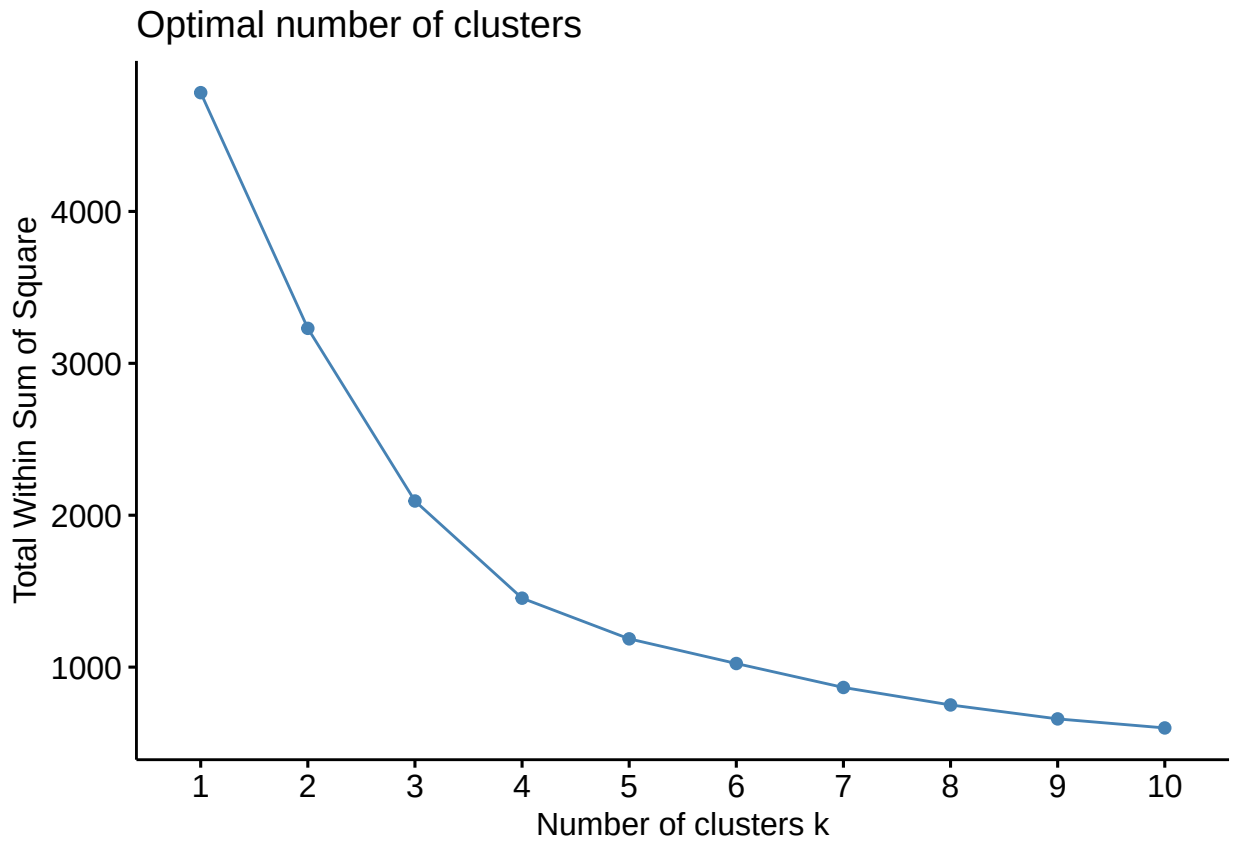
Apply clustering

```r
data <- data %>%
  mutate(
    StudyTimeWeekly = scale(StudyTimeWeekly),
    Absences = scale(Absences)
  )

df_cluster_StudyHabit <- data %>%
  select(StudyTimeWeekly, Absences) %>%
  na.omit()  # Ensure no missing values

# Compute Euclidean distance
dist_matrix_StudyHabit <- dist(df_cluster_StudyHabit, method = "euclidean")

# Perform hierarchical clustering using Ward's method
hc <- hclust(dist_matrix_StudyHabit, method = "ward.D2")

# Plot dendrogram
plot(hc, labels = FALSE, main = "Hierarchical Clustering Dendrogram")
```



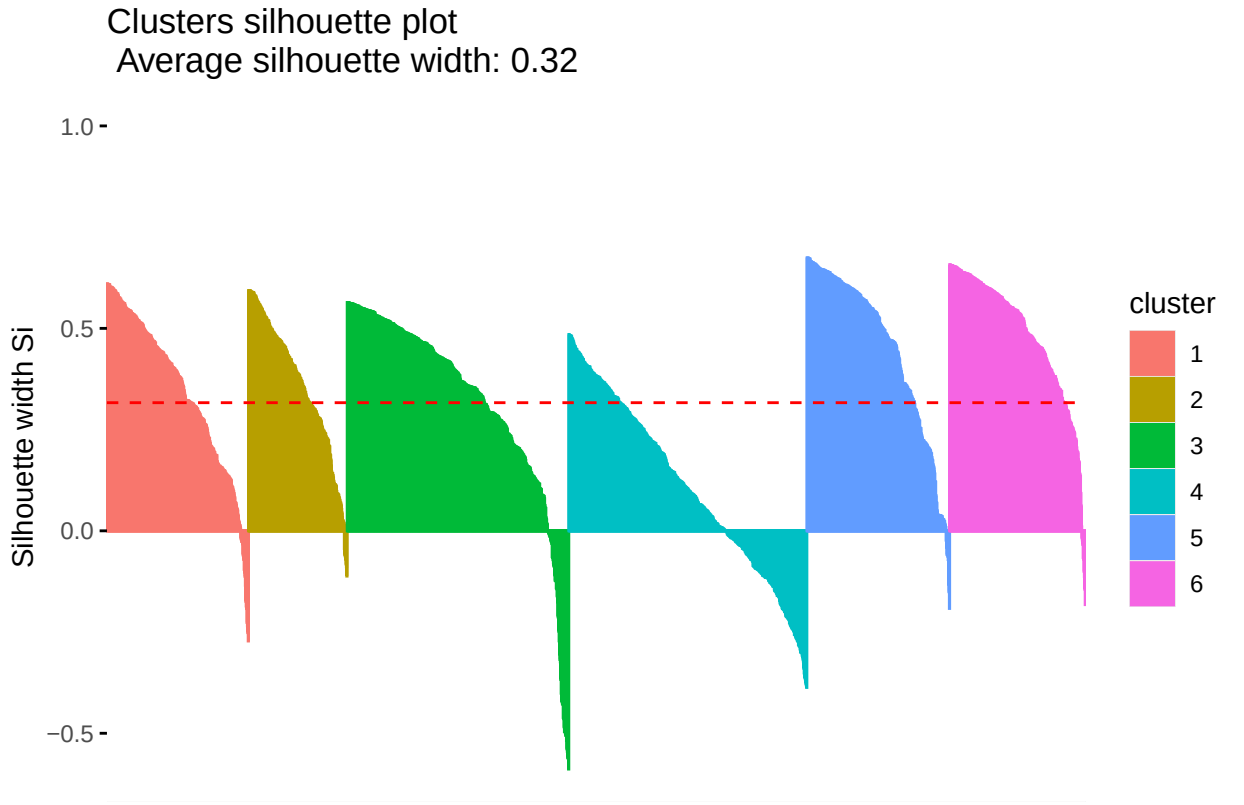
Check Results

```r
library(factoextra)
fviz_nbclust(df_cluster_StudyHabit, hcut, method = "wss")
```




```
library(cluster)
sil_StudyHabit <- silhouette(cutree(hc_StudyHabit, k = 6), dist_matrix_StudyHabit)
fviz_silhouette(sil_StudyHabit)
```

```
##   cluster size ave.sil.width
## 1       1  346          0.34
## 2       2  241          0.36
## 3       3  542          0.32
## 4       4  582          0.11
## 5       5  349          0.45
## 6       6  332          0.48
```




```
data$Cluster_HC <- cutree(hc, k = 6)

data$Cluster_HC <- as.factor(data$Cluster_HC)
```

Fit two versions of the model, keeping all settings the same except for the prior on the regression coefficients.

```
## Start sampling
```

```
## Running MCMC with 4 chains, at most 8 in parallel...
##
## Chain 1 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 3 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 2 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 1 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 3 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 2 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 1 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 3 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 1 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4 Iteration:  300 / 4000 [  7%]  (Warmup)
```

```
## Chain 2 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 3 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 4 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 2 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 4 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 4 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 1 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 2 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 1 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 4 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 3 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 2 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 1 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 4 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 3 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 2 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 1 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 3 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 2 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 3 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 2 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 1 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 2 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 4 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 1 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 3 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 2 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 1 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 3 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 2 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 4 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 3 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 3 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 1 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 2 Iteration: 1700 / 4000 [ 42%]  (Warmup)
```

```
## Chain 4 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 3 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 1 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 4 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 3 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 2 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 1 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 4 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 4 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 3 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 4 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 1 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 2 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 1 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 4 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 1 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 1 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 4 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 2 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 3 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 1 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 1 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 4 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 2 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 3 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 1 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 4 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 3 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 4 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1 Iteration: 3500 / 4000 [ 87%]  (Sampling)
```

```
## Chain 2 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 1 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 3 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 4 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 3 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 1 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 1 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1 finished in 215.1 seconds.
## Chain 4 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 2 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 4 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 3 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 2 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 3 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 3 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 4 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 4 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3 finished in 259.7 seconds.
## Chain 2 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 4 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 4 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 2 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4 finished in 290.1 seconds.
## Chain 2 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 2 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 2 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 2 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 2 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 2 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 2 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2 finished in 395.1 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 290.0 seconds.
## Total execution time: 395.3 seconds.

## Loading required package: rstan

## Loading required package: StanHeaders
```

```
##
## rstan version 2.32.6 (Stan version 2.32.2)

## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
## For within-chain threading using `reduce_sum()` or `map_rect()` Stan functions,
## change `threads_per_chain` option:
## rstan_options(threads_per_chain = 1)

## Do not specify '-march=native' in 'LOCAL_CPPFLAGS' or a Makevars file

##
## Attaching package: 'rstan'

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

## Start sampling

## Running MCMC with 4 chains, at most 8 in parallel...
##
## Chain 1 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 1 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 3 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 2 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 1 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 2 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 3 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 4 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 2 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 4 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 2 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 4 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 1 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 4 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 3 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  700 / 4000 [ 17%]  (Warmup)
```

```
## Chain 4 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 2 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 4 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 1 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 3 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 4 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 2 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 1 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 3 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 4 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 2 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 1 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 3 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 2 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 1 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 3 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 2 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 1 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 1 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 2 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 3 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 1 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 4 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 3 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 2 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 1 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 3 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 1 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 3 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 2 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 1 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 4 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 3 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 1 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 4 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 1 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2 Iteration: 2001 / 4000 [ 50%]  (Sampling)
```

```
## Chain 3 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 4 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 3 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 1 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 2 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 3 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 1 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 2 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 3 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 4 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 1 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 1 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 4 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 3 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 4 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 3 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 2 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 3 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 1 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 4 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 3 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 2 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 3 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 4 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 2 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 1 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 3 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 2 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 3 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 1 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 2 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 3 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 1 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 2 Iteration: 3400 / 4000 [ 85%]  (Sampling)
```

```
## Chain 3 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 2 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 1 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3 finished in 254.6 seconds.
## Chain 2 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 2 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 1 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 1 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 2 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 1 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1 finished in 284.3 seconds.
## Chain 4 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2 finished in 286.3 seconds.
## Chain 4 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 4 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 4 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 4 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 4 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4 finished in 395.3 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 305.1 seconds.
## Total execution time: 395.4 seconds.
```

Extract and compare posterior summaries for key parameters across the two models.

```r
# Summaries for the hierarchical models
summary_alt1 <- summary(mdl_cnt_sh_prior_def_alt1)$fixed
summary_alt2 <- summary(mdl_cnt_sh_prior_def_alt2)$fixed

# Combine summaries into a single table
sensitivity_summary <- bind_rows(
  Alt1 = summary_alt1,
  Alt2 = summary_alt2,
  .id = "Model"
)

# Print the summary
print(sensitivity_summary)
```

```
##                       Model     Estimate   Est.Error     l-95% CI   u-95% CI
## Intercept...1          Alt1  1.130747998 0.441233062  0.188033800 1.96558550
## Gender...2             Alt1 -0.007241511 0.018507457 -0.043612955 0.02885253
```

```
## ParentalEducation...3   Alt1  0.004788063 0.009103372 -0.013034140 0.02242323
## ParentalSupport...4     Alt1  0.147923409 0.008042503  0.132113425 0.16358035
## Extracurricular...5     Alt1  0.187924230 0.018773950  0.151277150 0.22489862
## Sports...6              Alt1  0.150302632 0.019508119  0.112722150 0.18829682
## Music...7               Alt1  0.152566502 0.023387377  0.105654075 0.19891362
## Volunteering...8        Alt1  0.006789584 0.024536431 -0.041268282 0.05430637
## Ethnicity...9           Alt1  0.012061005 0.008795168 -0.005334056 0.02897449
## Intercept...10          Alt2  1.130504017 0.469431584  0.172846300 2.00348400
## Gender...11             Alt2 -0.007797298 0.018061944 -0.042631992 0.02838433
## ParentalEducation...12  Alt2  0.004666828 0.008833302 -0.012778200 0.02204821
## ParentalSupport...13    Alt2  0.147808889 0.007988970  0.131937300 0.16354177
## Extracurricular...14    Alt2  0.188214001 0.018750734  0.151250800 0.22483657
## Sports...15             Alt2  0.150549027 0.019389172  0.112620875 0.18946662
## Music...16              Alt2  0.152347761 0.022745140  0.108158600 0.19731837
## Volunteering...17       Alt2  0.006593951 0.024411333 -0.041947850 0.05500246
## Ethnicity...18          Alt2  0.012087934 0.009079669 -0.005658465 0.03008897
##                                   Rhat  Bulk_ESS Tail_ESS
## Intercept...1          1.0030956  1885.051 2115.655
## Gender...2             1.0007899  9801.915 5789.913
## ParentalEducation...3  1.0007907 10547.615 5557.598
## ParentalSupport...4    1.0009751  9627.081 5746.659
## Extracurricular...5    1.0003961  9519.212 5417.066
## Sports...6             0.9998586 11065.626 5838.186
## Music...7              1.0003821 10837.561 5570.599
## Volunteering...8       1.0007351 11803.776 5719.026
## Ethnicity...9          1.0001410 10648.743 5884.175
## Intercept...10         1.0035405  1628.037 2427.763
## Gender...11            1.0002501 10905.601 5439.935
## ParentalEducation...12 1.0007283 11368.995 5449.873
## ParentalSupport...13   1.0000617 12620.226 6331.337
## Extracurricular...14   1.0009417 10369.993 5670.944
## Sports...15            1.0002949 10991.295 5239.965
## Music...16             1.0014668 10582.142 5837.944
## Volunteering...17      1.0008378 10582.230 6245.932
## Ethnicity...18         1.0011170 10620.436 5796.532
```

Visualize the posterior distributions of key regression coefficients (e.g., Absences, StudyTimeWeekly) across the two models.

```
# Extraction of posterior samples
posterior_alt1 <- posterior_samples(mdl_cnt_sh_prior_def_alt1)
```

```
## Warning: Method 'posterior_samples' is deprecated. Please see ?as_draws for
## recommended alternatives.
```

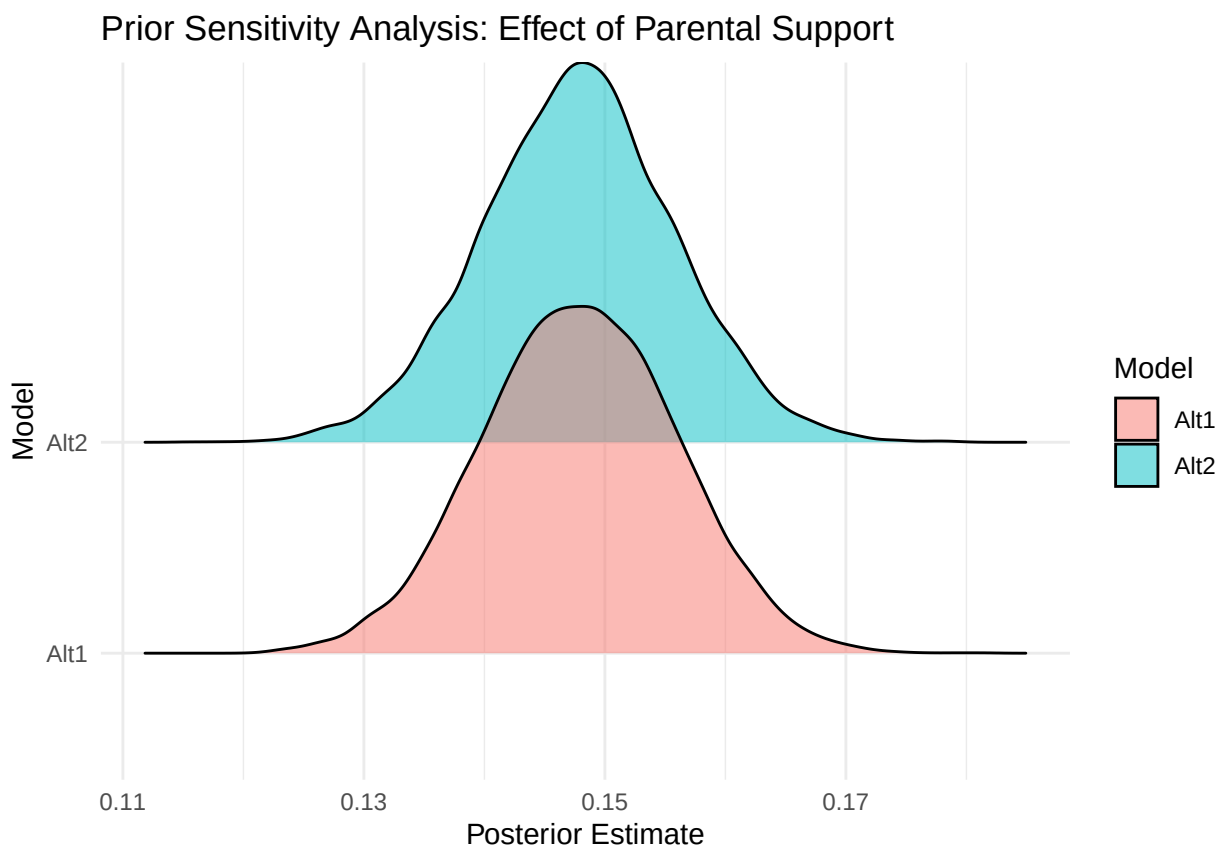
```
posterior_alt2 <- posterior_samples(mdl_cnt_sh_prior_def_alt2)
```

```
## Warning: Method 'posterior_samples' is deprecated. Please see ?as_draws for
## recommended alternatives.
```

```
# Combineed posterior samples for visualization
posterior_combined <- bind_rows(
  posterior_alt1 %>% mutate(Model = "Alt1"),
  posterior_alt2 %>% mutate(Model = "Alt2")
)
```

```r
# Plotposterior densities for a key parameter (e.g., "b_ParentalSupport")
library(ggridges)
ggplot(posterior_combined, aes(x = b_ParentalSupport, y = Model, fill = Model)) +
  geom_density_ridges(alpha = 0.5) +
  labs(title = "Prior Sensitivity Analysis: Effect of Parental Support",
       x = "Posterior Estimate", y = "Model") +
  theme_minimal()
```

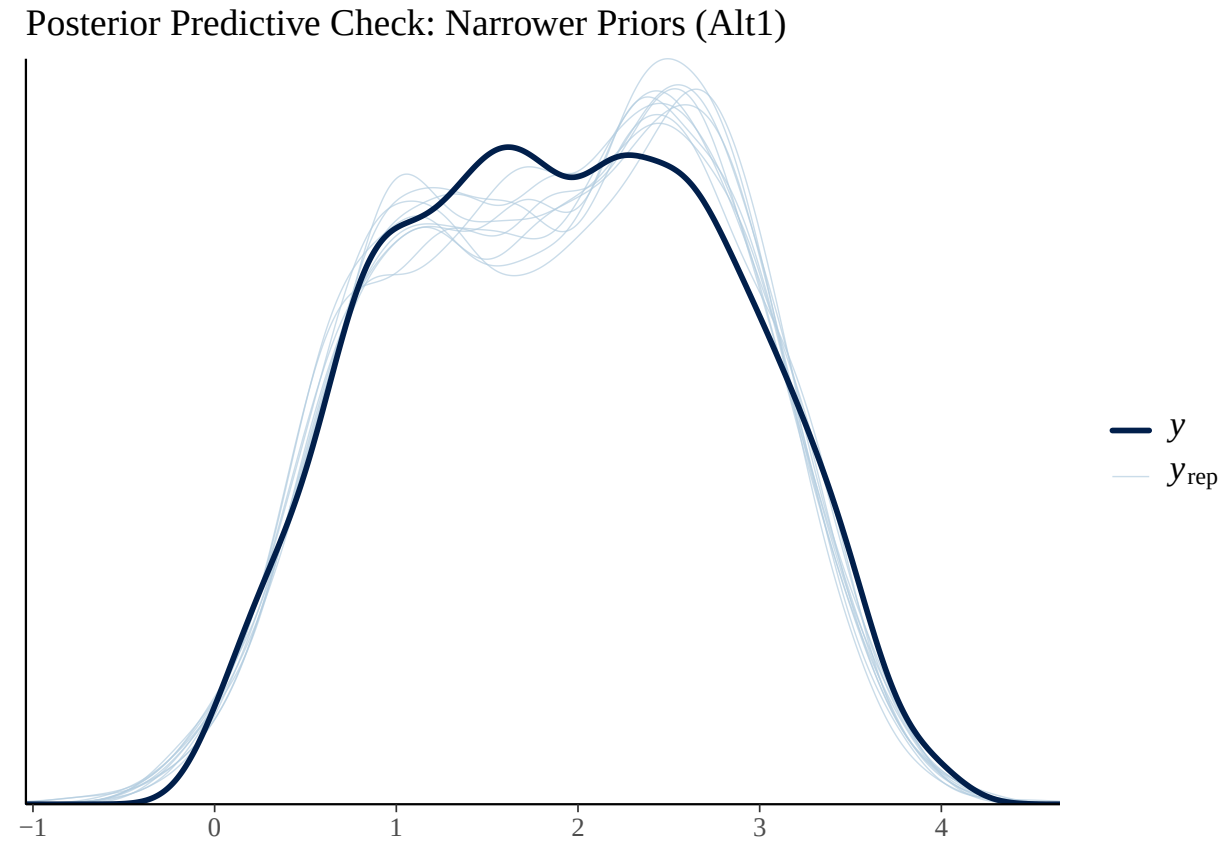
```
## Picking joint bandwidth of 0.00118
```



Run posterior predictive checks for both models to compare how well they predict the observed data.

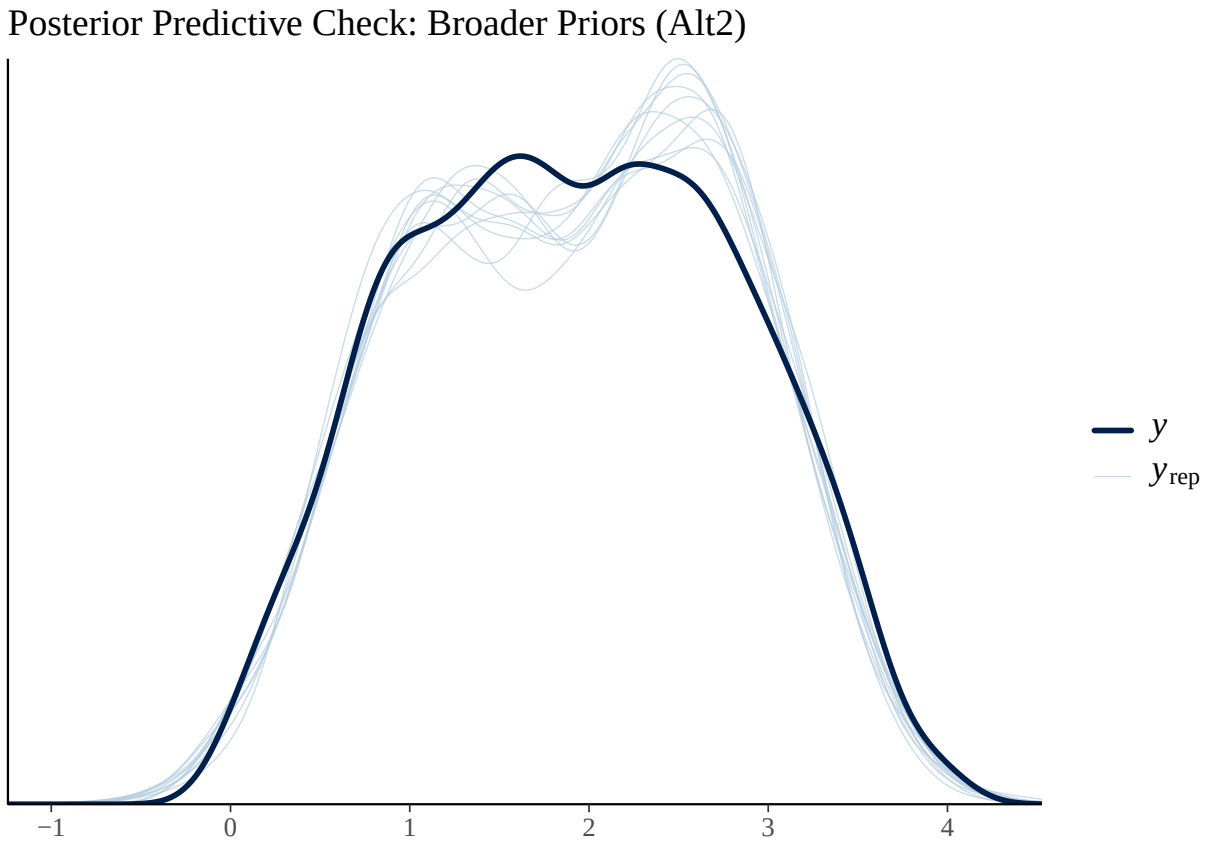
```r
# Posterior predictive checks for the first model
pp_check(mdl_cnt_sh_prior_def_alt1, type = "dens_overlay") +
  ggtitle("Posterior Predictive Check: Narrower Priors (Alt1)")
```

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



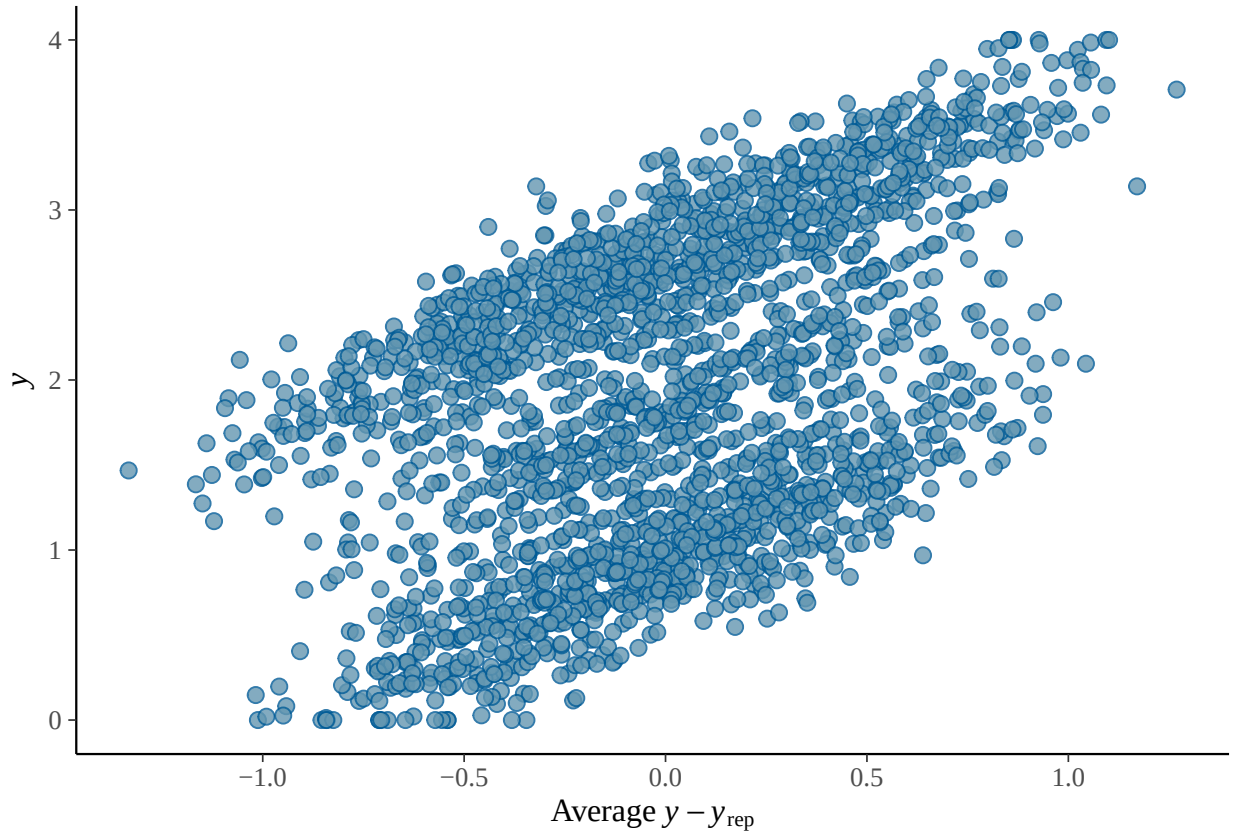

```
# Posterior predictive checks for the second model
pp_check(mdl_cnt_sh_prior_def_alt2, type = "dens_overlay") +
  ggtitle("Posterior Predictive Check: Broader Priors (Alt2)")
```

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



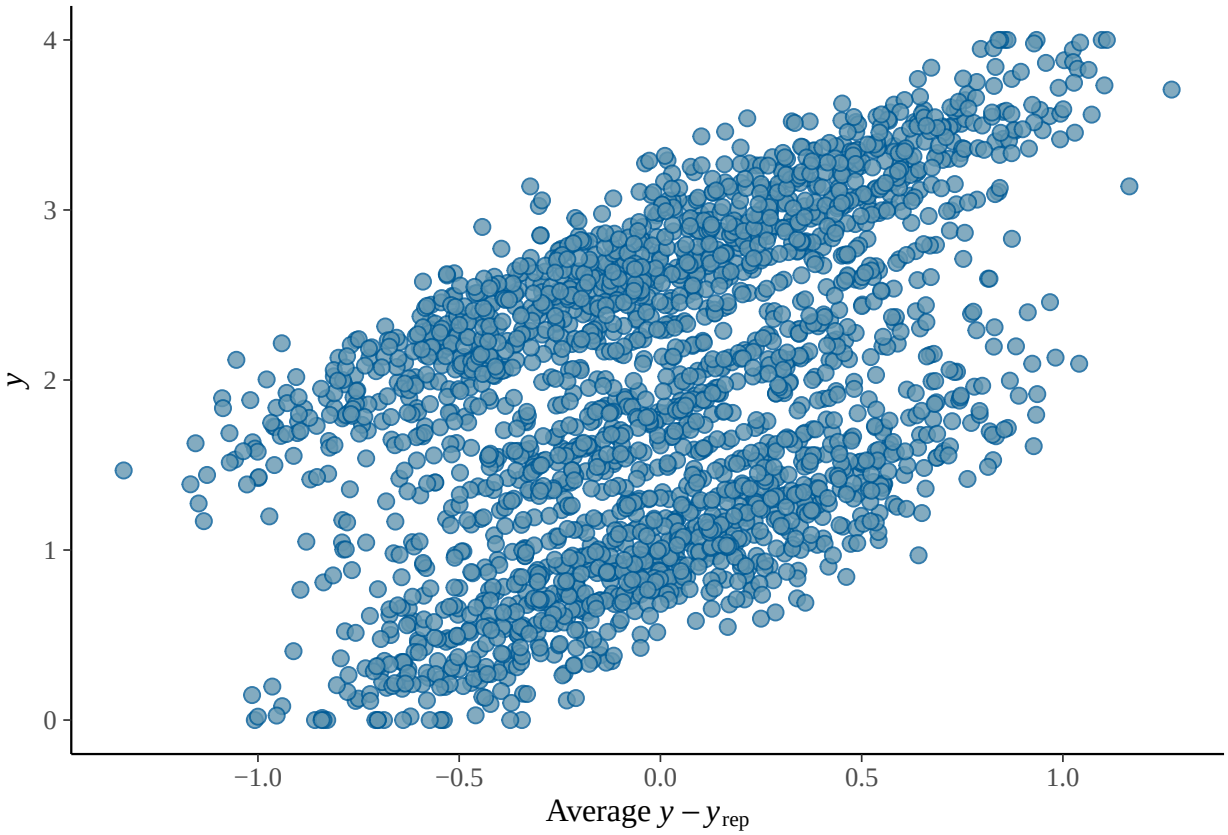

```
pp_check(mdl_cnt_sh_prior_def_alt1, type = "error_scatter_avg")
```

```
## Using all posterior draws for ppc type 'error_scatter_avg' by default.
```




```
pp_check(mdl_cnt_sh_prior_def_alt2, type = "error_scatter_avg")
```

```
## Using all posterior draws for ppc type 'error_scatter_avg' by default.
```




```
loo_alt1 <- loo(mdl_cnt_sh_prior_def_alt1)
loo_alt2 <- loo(mdl_cnt_sh_prior_def_alt2)

loo_comparison <- loo_compare(loo_alt1, loo_alt2)
print(loo_comparison)
```

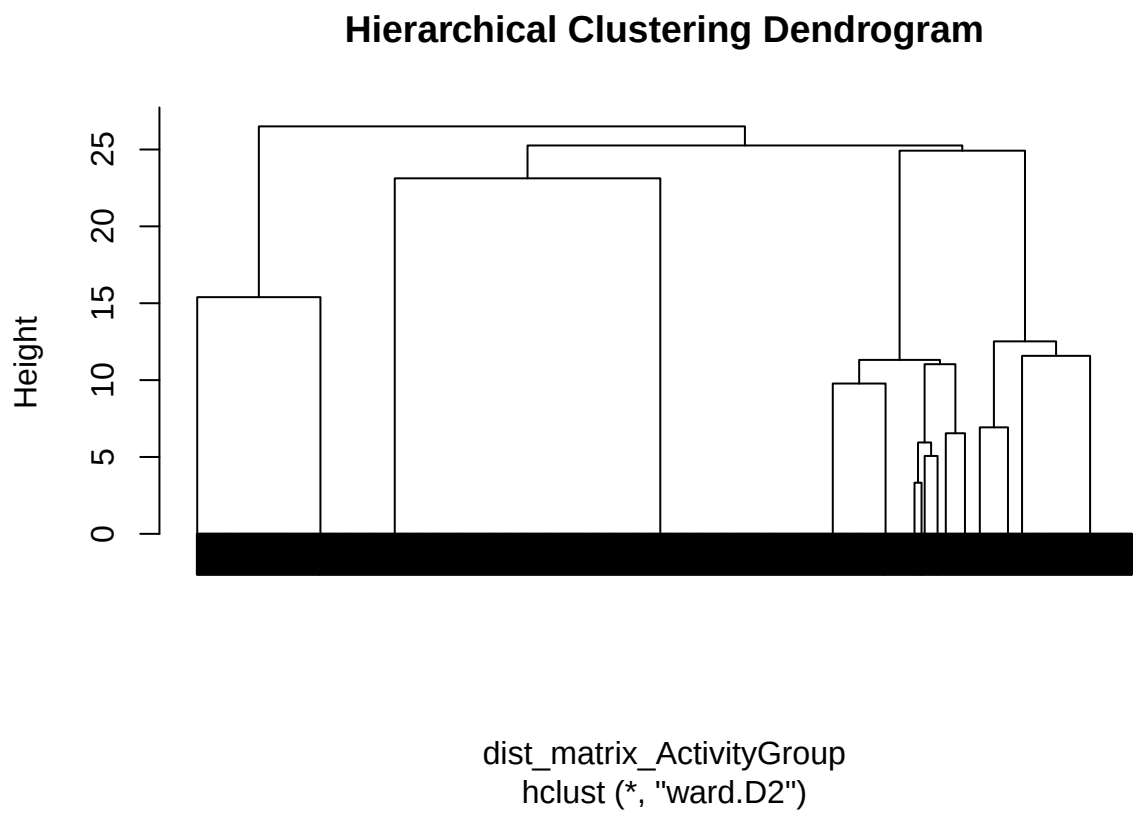
```
##                            elpd_diff se_diff
## mdl_cnt_sh_prior_def_alt2  0.0       0.0
## mdl_cnt_sh_prior_def_alt1 -0.1       0.1
```

#2) Prior Sensitivity Analysis for ActivityGroup Cluster

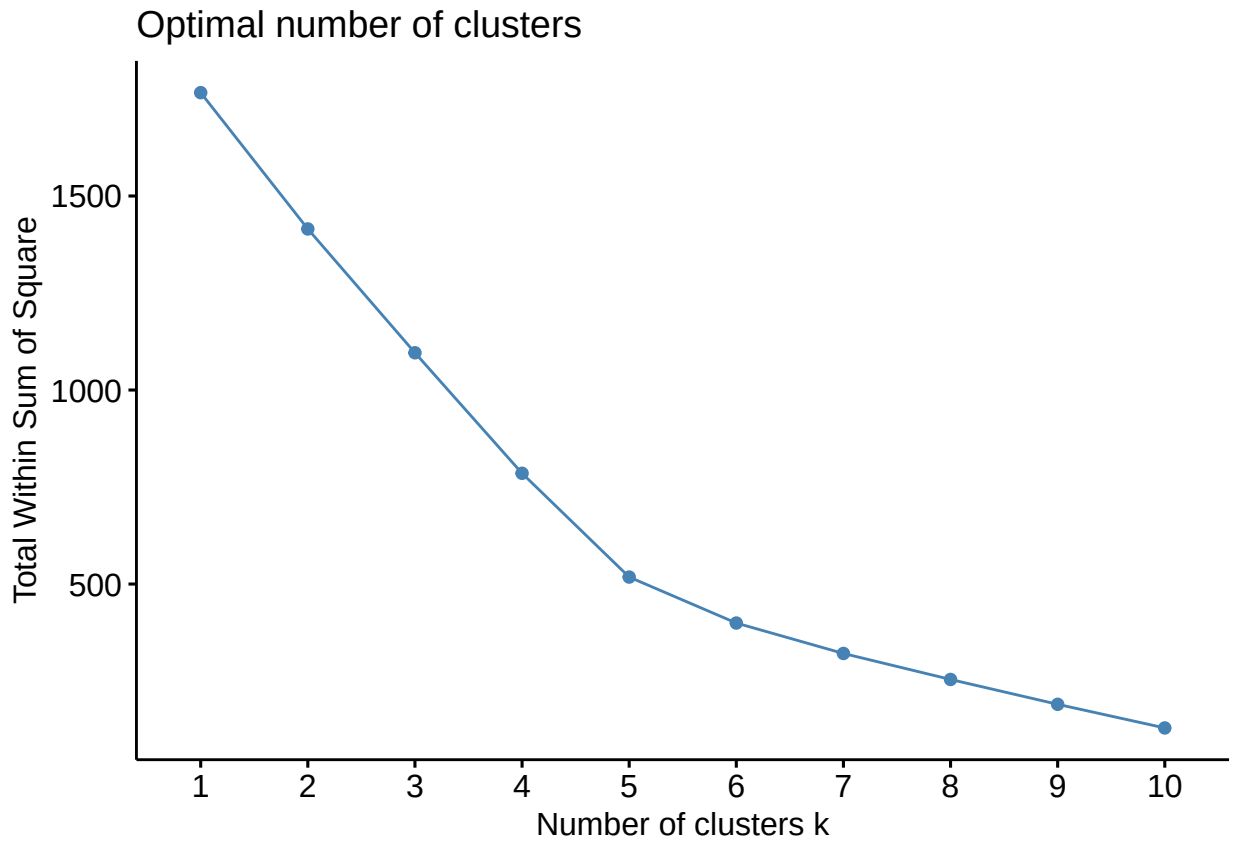
We should focus on changing the prior for regression coefficients while keeping the prior for random effects constant. This way, we isolate the impact of different priors on the regression coefficients.

```
data <- read.csv("Student_performance.csv")
# Scale continuous predictors
data$StudyTimeWeekly <- scale(data$StudyTimeWeekly)
data$Absences <- scale(data$Absences)

# Select variables for clustering
df_cluster_ActivityGroup <- data %>%
  select(Extracurricular, Sports, Music, Volunteering)

# Compute distance matrix
dist_matrix_ActivityGroup <- dist(df_cluster_ActivityGroup, method = "euclidean")

# Perform hierarchical clustering using Ward's method
```

```r
hc_ActivityGroup <- hclust(dist_matrix_ActivityGroup, method = "ward.D2")


# Plot dendrogram
plot(hc_ActivityGroup, labels = FALSE, main = "Hierarchical Clustering Dendrogram")
```

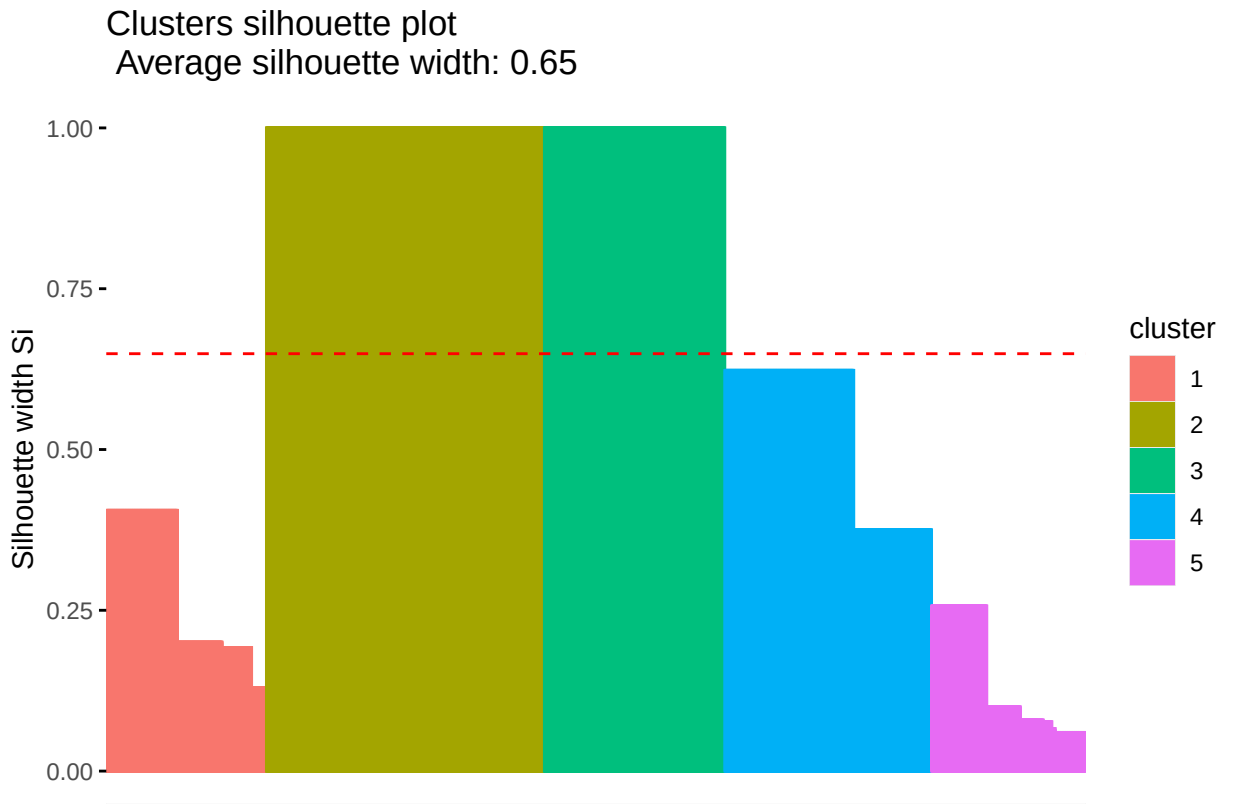



```r
library(factoextra)
fviz_nbclust(df_cluster_ActivityGroup, hcut, method = "wss")
```



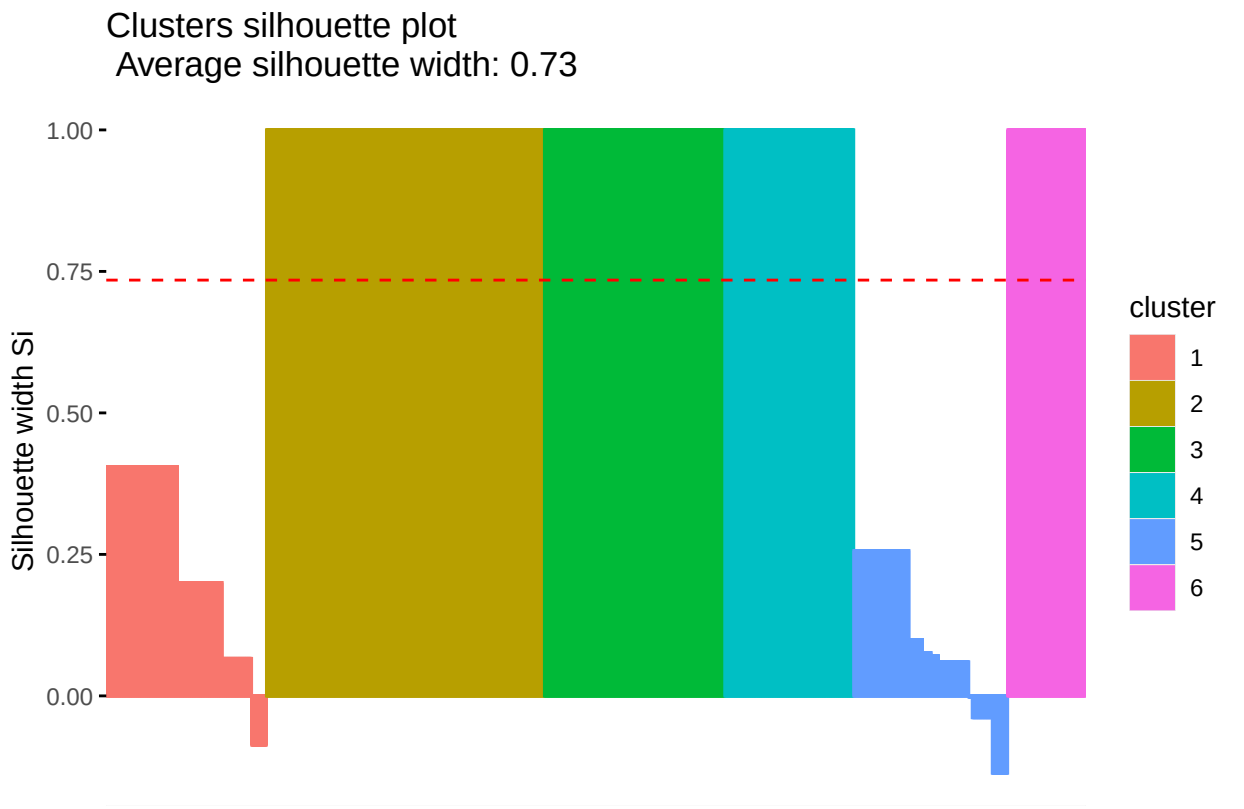

```
library(cluster)
sil_ActivityGroup <- silhouette(cutree(hc_ActivityGroup, k = 5), dist_matrix_ActivityGroup)
fviz_silhouette(sil_ActivityGroup)
```

```
##   cluster size ave.sil.width
## 1       1  391          0.28
## 2       2  679          1.00
## 3       3  441          1.00
## 4       4  505          0.53
## 5       5  376          0.14
```




```
library(cluster)
sil_ActivityGroup <- silhouette(cutree(hc_ActivityGroup, k = 6), dist_matrix_ActivityGroup)
fviz_silhouette(sil_ActivityGroup)
```

```
##   cluster size ave.sil.width
## 1       1  391          0.24
## 2       2  679          1.00
## 3       3  441          1.00
## 4       4  315          1.00
## 5       5  376          0.10
## 6       6  190          1.00
```




```
priors_alt1 <- c(
  set_prior("normal(0, 1)", class = "b"),  # Tight prior for regression coefficients
  set_prior("cauchy(0, 1)", class = "sd"),  # Random effects prior (same as previous)
  set_prior("student_t(3, 0, 2)", class = "Intercept")  # Informative prior for intercept
)

priors_alt2 <- c(
  set_prior("normal(0, 10)", class = "b"),  # Broad prior for regression coefficients
  set_prior("cauchy(0, 1)", class = "sd"),  # Random effects prior (same as previous)
  set_prior("student_t(3, 0, 10)", class = "Intercept")  # Weakly informative prior for intercept
)

# Cut tree to create 5 clusters
data$Cluster_HC <- cutree(hc_ActivityGroup, k = 5)

# Convert to factor
data$Cluster_HC <- as.factor(data$Cluster_HC)
```

Fit two versions of the model, keeping all settings the same except for the prior on the regression coefficients.

```
## Start sampling
```

```
## Running MCMC with 4 chains, at most 8 in parallel...
##
## Chain 1 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3 Iteration:    1 / 4000 [  0%]  (Warmup)
```

```
## Chain 4 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 3 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 1 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 2 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 3 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 1 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 3 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 4 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 2 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 2 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 2 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 3 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 4 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 3 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 4 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 1 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 4 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 1 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 1 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 2 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 3 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 4 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 3 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 2 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 1 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 4 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 2 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 2 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 3 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 1 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 2 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 3 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 1 Iteration: 1200 / 4000 [ 30%]  (Warmup)
```

```
## Chain 4 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 2 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 1 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 3 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 2 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 1 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 3 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 4 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 2 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 1 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 3 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 4 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 1 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 2 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 4 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 3 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 4 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 1 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 3 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 2 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 1 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 4 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 2 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 3 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 1 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 3 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 3 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 2 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 2 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 3 Iteration: 3300 / 4000 [ 82%]  (Sampling)
```

```
## Chain 1 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 4 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 2 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 3 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 1 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 2 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 3 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 1 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 3 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 3 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3 finished in 80.5 seconds.
## Chain 1 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 4 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 1 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 2 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 4 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 2 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 2 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 1 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 4 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 1 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 2 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 4 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 2 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 1 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 4 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 2 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 1 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 2 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 4 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 2 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2 finished in 116.7 seconds.
## Chain 1 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 1 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 1 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1 finished in 136.7 seconds.
## Chain 4 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 4 Iteration: 3400 / 4000 [ 85%]  (Sampling)
```

```
## Chain 4 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 4 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 4 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4 finished in 180.2 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 128.5 seconds.
## Total execution time: 180.3 seconds.
```

```
## Start sampling
```

```
## Running MCMC with 4 chains, at most 8 in parallel...
##
## Chain 1 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 3 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 2 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 1 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 1 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 4 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 3 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 1 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 2 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 4 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 2 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 4 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 2 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 3 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 4 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 2 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 1 Iteration: 1000 / 4000 [ 25%]  (Warmup)
```

```
## Chain 4 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 1 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 4 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 2 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 3 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 1 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 2 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 3 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 1 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 4 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 2 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 3 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 1 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 1 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 2 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 1 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 3 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 1 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 4 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 2 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 1 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 3 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 2 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 3 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 4 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 3 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 2 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 1 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 3 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 4 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 1 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 4 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 2 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 4 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 3 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3 Iteration: 2001 / 4000 [ 50%]  (Sampling)
```

```
## Chain 2 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 1 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 1 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 4 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 1 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 2 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 3 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 1 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 4 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 1 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 3 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 2 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 1 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 3 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 1 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 4 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 2 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 4 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 1 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 2 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 3 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 4 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 2 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 4 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 3 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 4 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 1 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 2 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 1 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 2 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 3 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 1 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 3900 / 4000 [ 97%]  (Sampling)
```

```
## Chain 3 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 2 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 1 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 4 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4 finished in 116.9 seconds.
## Chain 1 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1 finished in 118.8 seconds.
## Chain 2 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 2 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 3 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 2 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 2 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2 finished in 142.8 seconds.
## Chain 3 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 3 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 3 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3 finished in 177.0 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 138.9 seconds.
## Total execution time: 177.1 seconds.
```

Extract and compare posterior summaries for key parameters across the two models.

```
# Summaries for the hierarchical models
summary_alt1 <- summary(mdl_cnt_ag_prior_alt1)$fixed
summary_alt2 <- summary(mdl_cnt_ag_prior_alt2)$fixed

# Combine summaries into a single table
sensitivity_summary <- bind_rows(
  Alt1 = summary_alt1,
  Alt2 = summary_alt2,
  .id = "Model"
)

# Print the summary
print(sensitivity_summary)
```

```
##                        Model     Estimate   Est.Error     l-95% CI     u-95% CI
## Intercept...1           Alt1  1.507951048 0.085111760  1.334571250  1.66770625
## Gender...2              Alt1  0.009597294 0.008903621 -0.007911414  0.02729364
## ParentalEducation...3   Alt1  0.003947076 0.004458760 -0.004809497  0.01279941
## ParentalSupport...4     Alt1  0.149715137 0.003876613  0.142149950  0.15718417
## StudyTimeWeekly...5     Alt1  0.162304510 0.004490713  0.153563900  0.17096347
## Absences...6            Alt1 -0.842549376 0.004481779 -0.851220175 -0.83370540
## Tutoring...7            Alt1  0.251197239 0.009705416  0.232296900  0.27049603
## Ethnicity...8           Alt1  0.004253856 0.004365789 -0.004309197  0.01270826
```

```
## Intercept...9           Alt2  1.512561393 0.090255182  1.331875250  1.67987525
## Gender...10             Alt2  0.009757952 0.008995880 -0.008033158  0.02720816
## ParentalEducation...11  Alt2  0.003833684 0.004432292 -0.004780839  0.01245554
## ParentalSupport...12    Alt2  0.149674357 0.003971937  0.141878950  0.15753508
## StudyTimeWeekly...13    Alt2  0.162185037 0.004429622  0.153515750  0.17090907
## Absences...14           Alt2 -0.842602717 0.004501463 -0.851500150 -0.83378697
## Tutoring...15           Alt2  0.251201502 0.009764267  0.231775975  0.27010515
## Ethnicity...16          Alt2  0.004289845 0.004315976 -0.004195426  0.01286272
##                            Rhat  Bulk_ESS Tail_ESS
## Intercept...1          1.001178  1993.805 2490.093
## Gender...2             1.000156  9037.838 5692.844
## ParentalEducation...3  1.000773  9340.706 5714.710
## ParentalSupport...4    1.000379 11026.939 5388.002
## StudyTimeWeekly...5    1.000385 10756.103 5373.483
## Absences...6           1.000272 10424.424 5497.483
## Tutoring...7           1.000608  9201.323 5567.168
## Ethnicity...8          1.001969 10759.339 5580.703
## Intercept...9          1.004004  1486.902 1678.582
## Gender...10            1.000619  9284.651 5665.352
## ParentalEducation...11 1.000337  9787.108 6035.248
## ParentalSupport...12   1.000644 10376.701 5900.712
## StudyTimeWeekly...13   1.000477 10624.214 5666.704
## Absences...14          1.000322  9653.439 5002.948
## Tutoring...15          1.000667  7869.994 5459.552
## Ethnicity...16         1.000670 11144.315 5805.781
```

Visualize the posterior distributions of key regression coefficients (e.g., Absences, StudyTimeWeekly) across the two models.

```
# Extract posterior samples
posterior_alt1 <- posterior_samples(mdl_cnt_ag_prior_alt1)
```

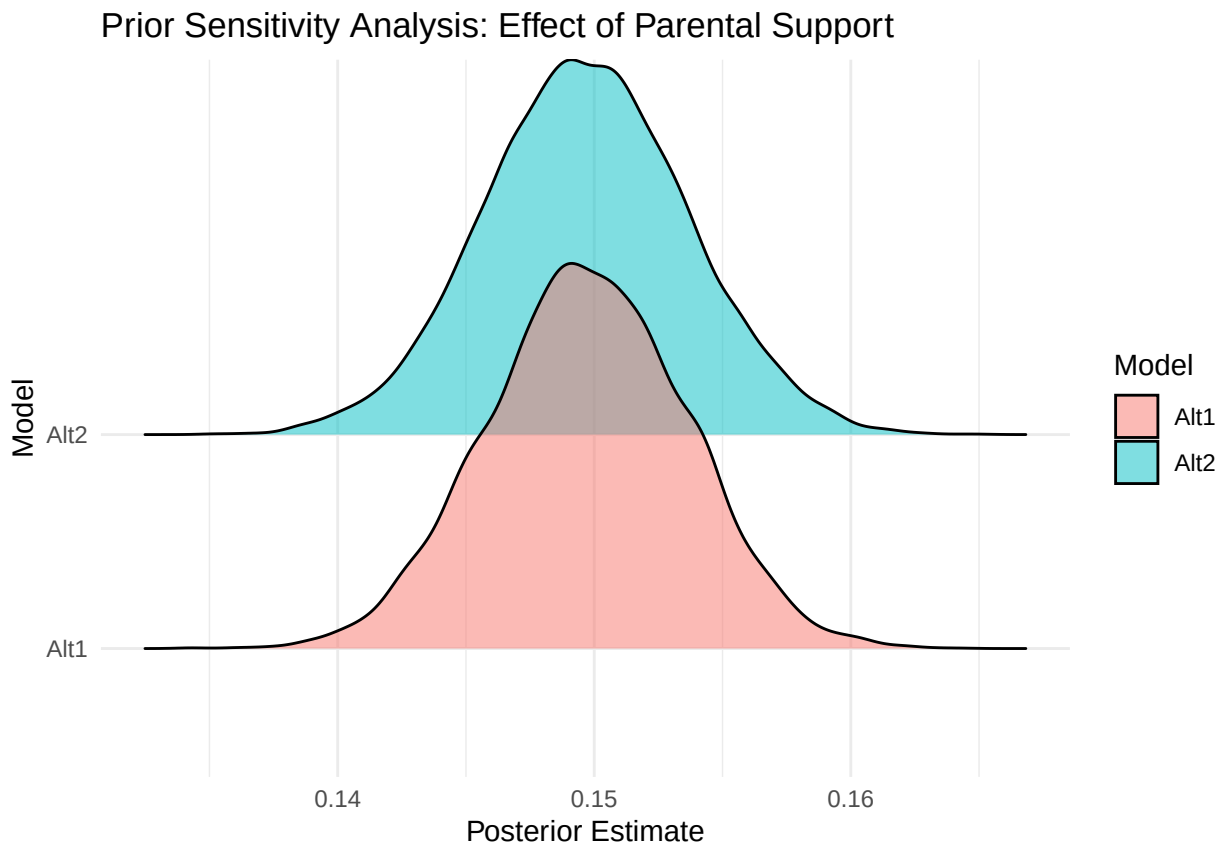
```
## Warning: Method 'posterior_samples' is deprecated. Please see ?as_draws for
## recommended alternatives.
```

```
posterior_alt2 <- posterior_samples(mdl_cnt_ag_prior_alt2)
```

```
## Warning: Method 'posterior_samples' is deprecated. Please see ?as_draws for
## recommended alternatives.
```

```
# Combine posterior samples for visualization
posterior_combined <- bind_rows(
  posterior_alt1 %>% mutate(Model = "Alt1"),
  posterior_alt2 %>% mutate(Model = "Alt2")
)

# Plot posterior densities for a key parameter (e.g., "b_ParentalSupport")
library(ggridges)
ggplot(posterior_combined, aes(x = b_ParentalSupport, y = Model, fill = Model)) +
  geom_density_ridges(alpha = 0.5) +
  labs(title = "Prior Sensitivity Analysis: Effect of Parental Support",
       x = "Posterior Estimate", y = "Model") +
  theme_minimal()
```

```
## Picking joint bandwidth of 0.000585
```

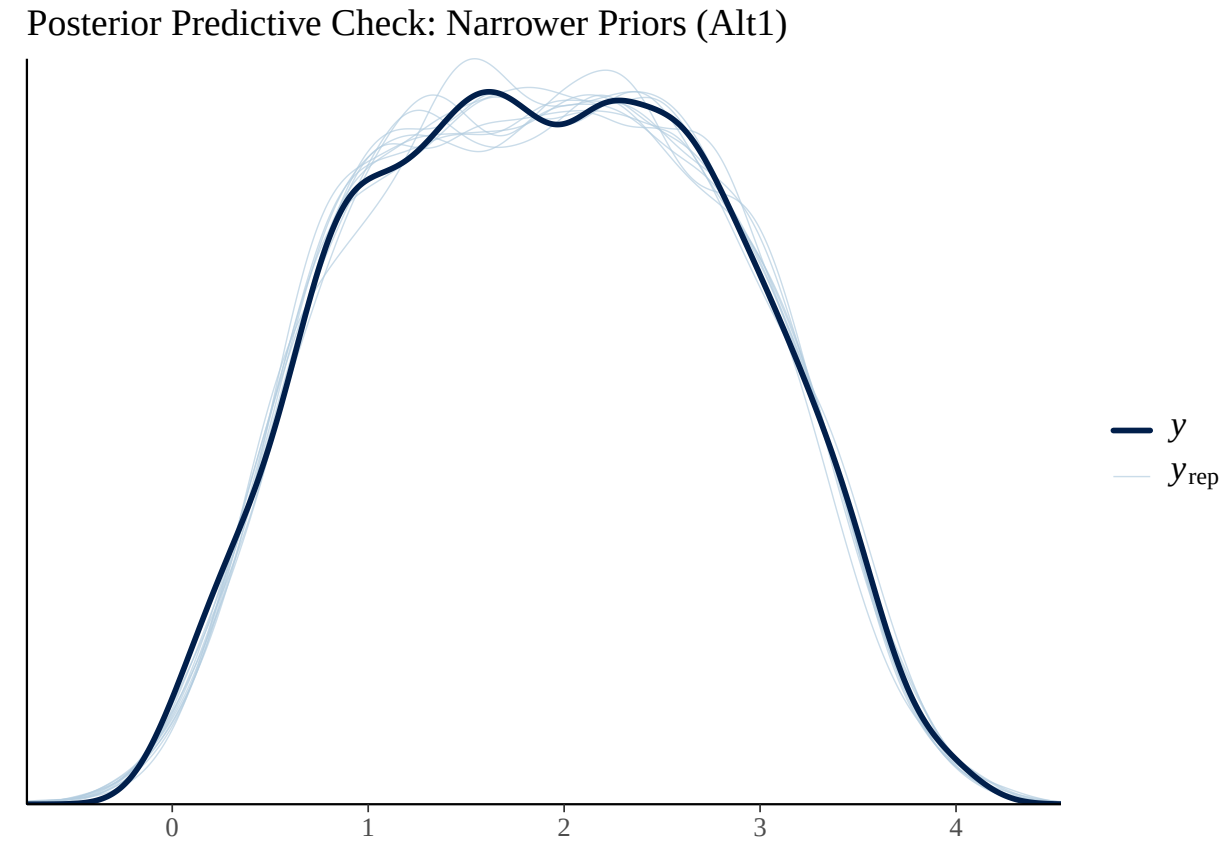



```
# Repeat for other predictors if needed
```

Run posterior predictive checks for both models to compare how well they predict the observed data.

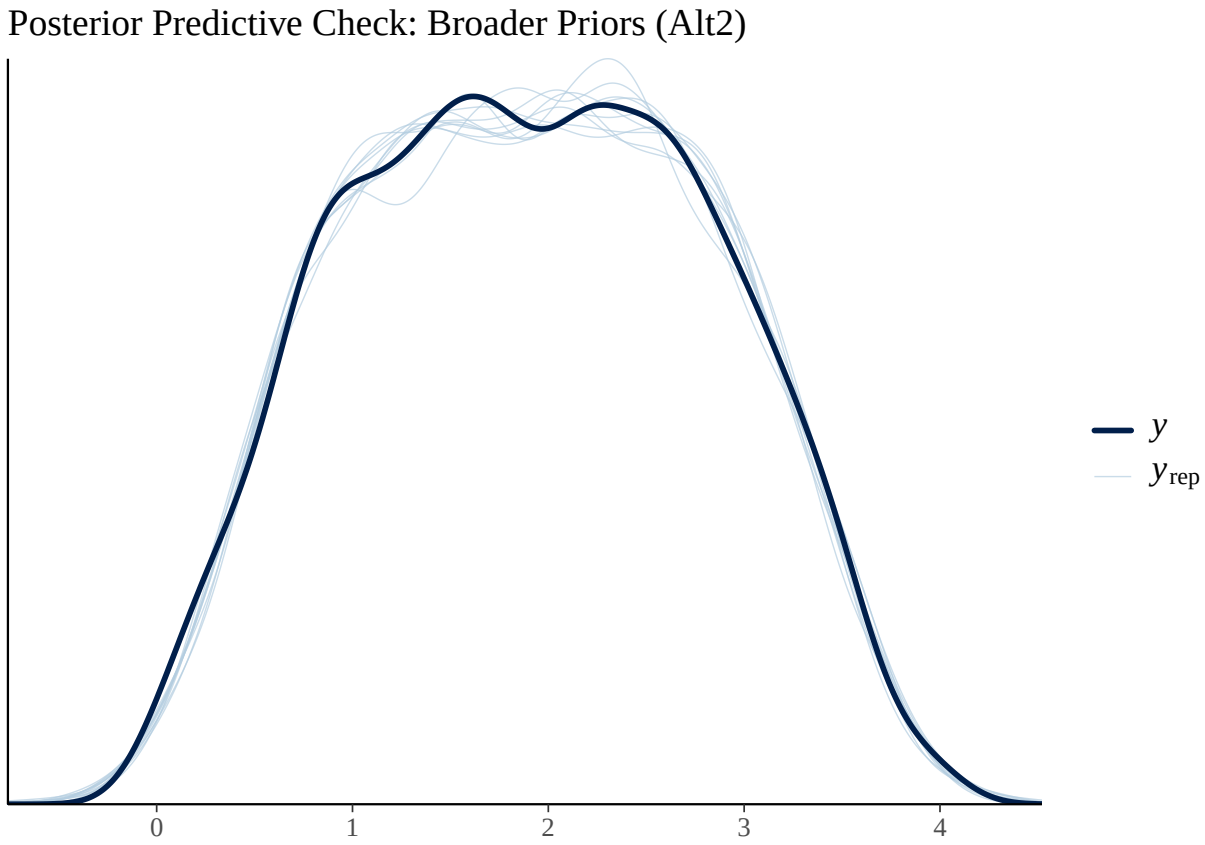
```
# Posterior predictive checks for the first model
pp_check(mdl_cnt_ag_prior_alt1, type = "dens_overlay") +
  ggtitle("Posterior Predictive Check: Narrower Priors (Alt1)")
```

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



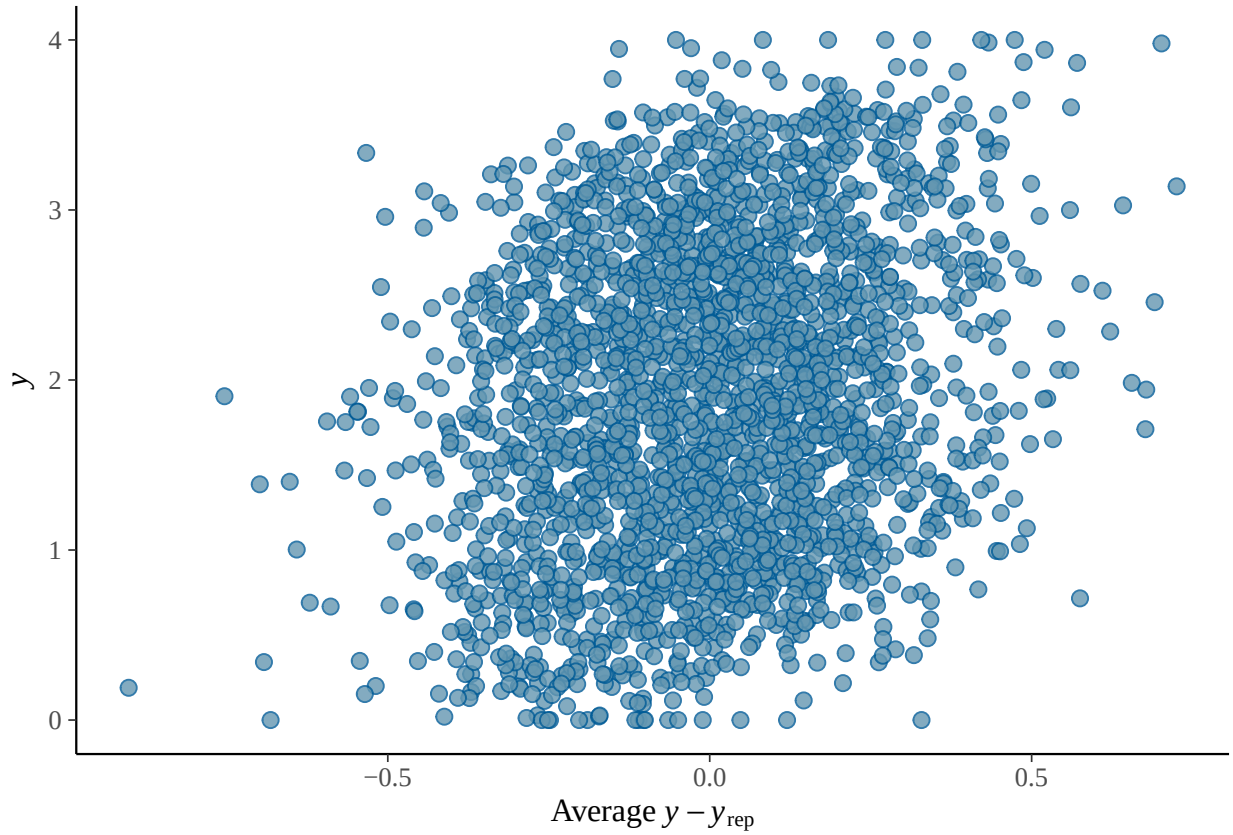

```
# Posterior predictive checks for the second model
pp_check(mdl_cnt_ag_prior_alt2, type = "dens_overlay") +
  ggtitle("Posterior Predictive Check: Broader Priors (Alt2)")
```

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



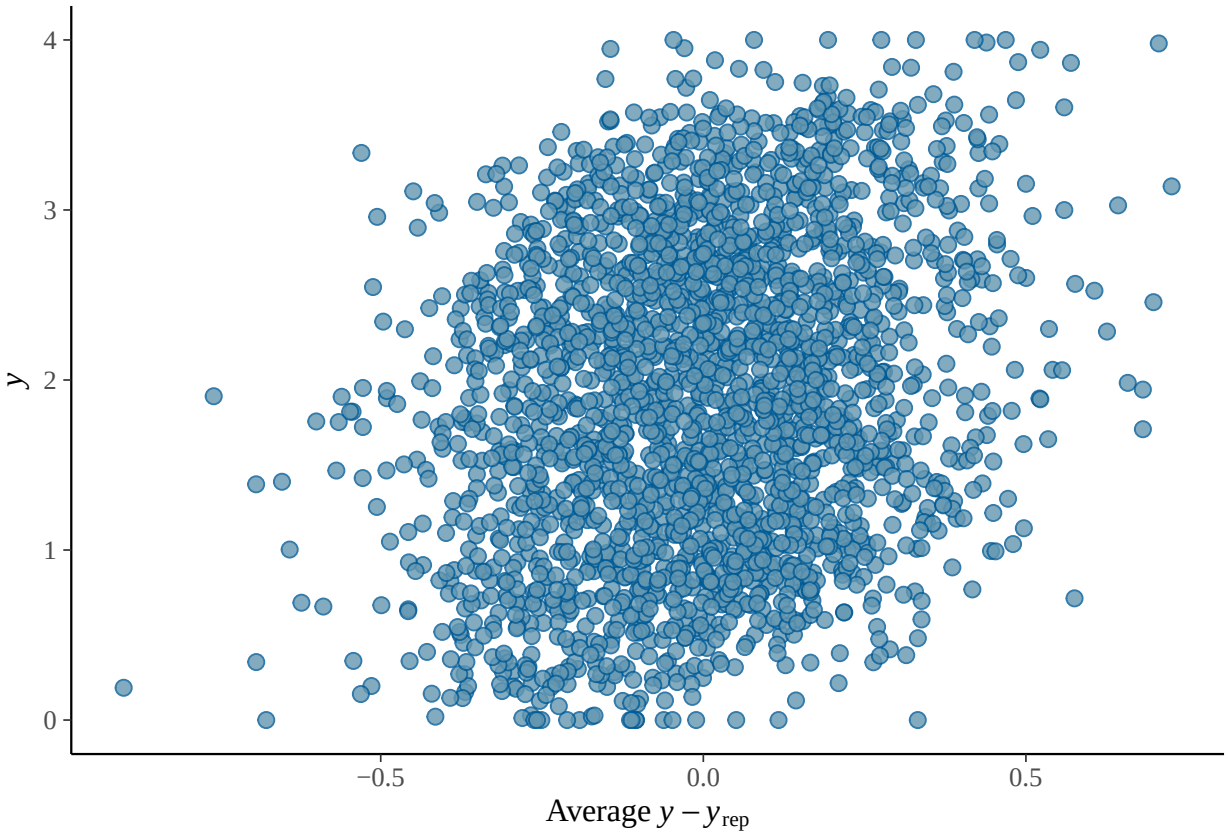

```
pp_check(mdl_cnt_ag_prior_alt1, type = "error_scatter_avg")
```

```
## Using all posterior draws for ppc type 'error_scatter_avg' by default.
```




```
pp_check(mdl_cnt_ag_prior_alt2, type = "error_scatter_avg")
```

```
## Using all posterior draws for ppc type 'error_scatter_avg' by default.
```



Use Leave-One-Out Cross-Validation (LOO) to compare predictive performance between the two models.

```
loo_alt1 <- loo(mdl_cnt_ag_prior_alt1)
loo_alt2 <- loo(mdl_cnt_ag_prior_alt2)

loo_comparison <- loo_compare(loo_alt1, loo_alt2)
print(loo_comparison)
```

```
##                        elpd_diff se_diff
## mdl_cnt_ag_prior_alt2  0.0       0.0
## mdl_cnt_ag_prior_alt1 -0.1       0.1
```

Extract and compare posterior summaries for key parameters across the two models.

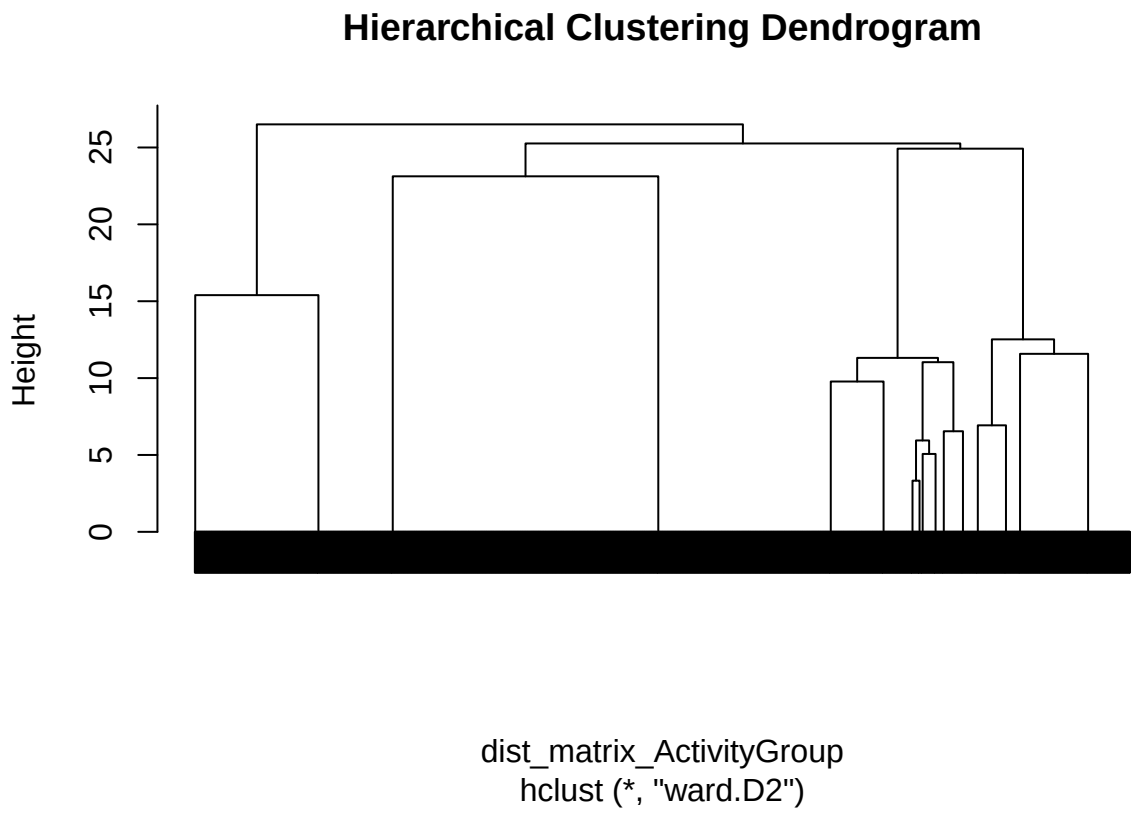
```
# Summaries for the hierarchical models
summary_alt1 <- summary(mdl_cnt_ag_prior_alt1)$fixed
summary_alt2 <- summary(mdl_cnt_ag_prior_alt2)$fixed

# Combine summaries into a single table
sensitivity_summary <- bind_rows(
  Alt1 = summary_alt1,
  Alt2 = summary_alt2,
  .id = "Model"
)

# Print the summary
print(sensitivity_summary)
```

```
##                         Model     Estimate   Est.Error     l-95% CI     u-95% CI
## Intercept...1            Alt1  1.507951048 0.085111760  1.334571250  1.66770625
## Gender...2               Alt1  0.009597294 0.008903621 -0.007911414  0.02729364
## ParentalEducation...3    Alt1  0.003947076 0.004458760 -0.004809497  0.01279941
## ParentalSupport...4      Alt1  0.149715137 0.003876613  0.142149950  0.15718417
## StudyTimeWeekly...5      Alt1  0.162304510 0.004490713  0.153563900  0.17096347
## Absences...6             Alt1 -0.842549376 0.004481779 -0.851220175 -0.83370540
## Tutoring...7             Alt1  0.251197239 0.009705416  0.232296900  0.27049603
## Ethnicity...8            Alt1  0.004253856 0.004365789 -0.004309197  0.01270826
## Intercept...9            Alt2  1.512561393 0.090255182  1.331875250  1.67987525
## Gender...10              Alt2  0.009757952 0.008995880 -0.008033158  0.02720816
## ParentalEducation...11   Alt2  0.003833684 0.004432292 -0.004780839  0.01245554
## ParentalSupport...12     Alt2  0.149674357 0.003971937  0.141878950  0.15753508
## StudyTimeWeekly...13     Alt2  0.162185037 0.004429622  0.153515750  0.17090907
## Absences...14            Alt2 -0.842602717 0.004501463 -0.851500150 -0.83378697
## Tutoring...15            Alt2  0.251201502 0.009764267  0.231775975  0.27010515
## Ethnicity...16           Alt2  0.004289845 0.004315976 -0.004195426  0.01286272
##                             Rhat  Bulk_ESS Tail_ESS
## Intercept...1           1.001178  1993.805 2490.093
## Gender...2              1.000156  9037.838 5692.844
## ParentalEducation...3   1.000773  9340.706 5714.710
## ParentalSupport...4     1.000379 11026.939 5388.002
## StudyTimeWeekly...5     1.000385 10756.103 5373.483
## Absences...6            1.000272 10424.424 5497.483
## Tutoring...7            1.000608  9201.323 5567.168
## Ethnicity...8           1.001969 10759.339 5580.703
## Intercept...9           1.004004  1486.902 1678.582
## Gender...10             1.000619  9284.651 5665.352
## ParentalEducation...11  1.000337  9787.108 6035.248
## ParentalSupport...12    1.000644 10376.701 5900.712
## StudyTimeWeekly...13    1.000477 10624.214 5666.704
## Absences...14           1.000322  9653.439 5002.948
## Tutoring...15           1.000667  7869.994 5459.552
## Ethnicity...16          1.000670 11144.315 5805.781
```

#3) Ordinal Model for ActivityGroup:

```r
# Read in your dataset
data <- read.csv("Student_performance.csv")

# Scale continuous predictors
data$StudyTimeWeekly <- scale(data$StudyTimeWeekly)
data$Absences <- scale(data$Absences)

# Convert GradeClass to an ordered factor
data$GradeClass <- factor(data$GradeClass,
                                levels = c(0, 1, 2, 3, 4),
                                labels = c("A", "B", "C", "D", "F"),
                                ordered = TRUE) # Ordinal


# Select variables for clustering
df_cluster_ActivityGroup <- data %>%
  select(Extracurricular, Sports, Music, Volunteering)
```

```r
# Compute distance matrix
dist_matrix_ActivityGroup <- dist(df_cluster_ActivityGroup, method = "euclidean")

# Perform hierarchical clustering using Ward's method
hc_ActivityGroup <- hclust(dist_matrix_ActivityGroup, method = "ward.D2")


# Plot dendrogram
plot(hc_ActivityGroup, labels = FALSE, main = "Hierarchical Clustering Dendrogram")
```




```r
# Cut tree to create 5 clusters
data$Cluster_HC <- cutree(hc_ActivityGroup, k = 5)

# Convert to factor
data$Cluster_HC <- as.factor(data$Cluster_HC)



priors <- c(
  set_prior("normal(0, 1)", class = "b"),  # Tight prior for regression coefficients
  set_prior("cauchy(0, 1)", class = "sd"),  # Random effects prior (same as previous)
  set_prior("student_t(3, 0, 2)", class = "Intercept")  # Informative prior for intercept
)
```

```
## Start sampling
```

```
## Running MCMC with 4 chains, at most 8 in parallel...
```

```
##
## Chain 1 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 2 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 3 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 1 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 1 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 3 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 2 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 1 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 3 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 1 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 2 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 4 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 2 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 4 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 2 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 4 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 1 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 3 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 1 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 4 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 2 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 1 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 4 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 2 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 1 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 4 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 2 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 1 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 2 Iteration: 1200 / 4000 [ 30%]  (Warmup)
```

```
## Chain 1 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 4 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 2 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 1 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 3 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 2 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 1 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 3 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 2 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 4 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 3 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 2 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 1 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 4 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 1 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 3 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 4 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 2 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 4 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 4 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 3 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 1 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 4 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 2 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 4 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 1 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 2 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 4 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 4 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 1 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 1 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 4 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 4 Iteration: 2800 / 4000 [ 70%]  (Sampling)
```

```
## Chain 1 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 4 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 3 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 4 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 4 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 2 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 3 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 2 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 4 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 1 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 4 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 2 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 4 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 4 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 1 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 2 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 2 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 3 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 4 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 1 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 2 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 4 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4 finished in 473.8 seconds.
## Chain 1 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 2 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 1 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 2 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 3 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 2 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 1 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 2 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 2 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2 finished in 544.5 seconds.
## Chain 1 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 3 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1 finished in 596.0 seconds.
```

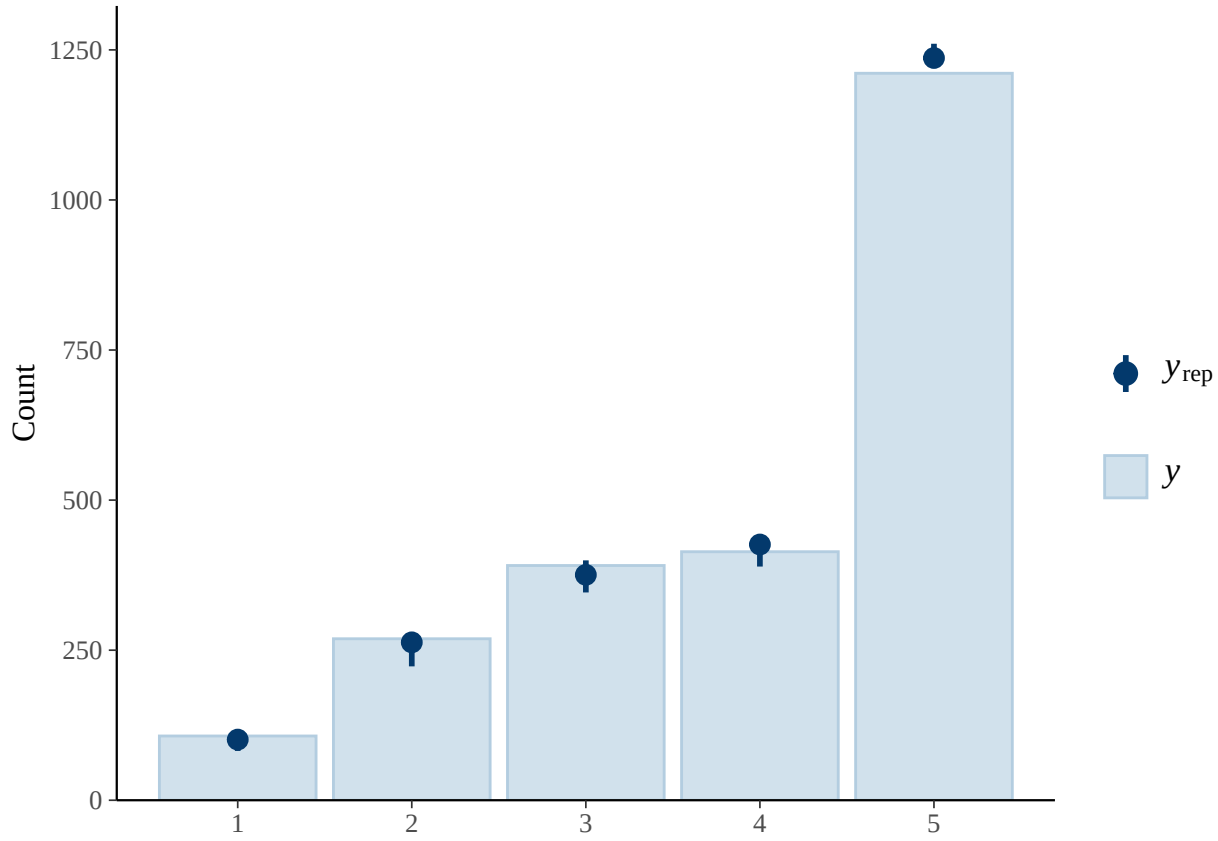
```
## Chain 3 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 3 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 3 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 3 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 3 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 3 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3 finished in 870.7 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 621.3 seconds.
## Total execution time: 870.8 seconds.
```

```
summary(mdl_ord_ag_prior)
```

```
##  Family: cumulative
##   Links: mu = logit; disc = identity
## Formula: GradeClass ~ Gender + ParentalEducation + ParentalSupport + StudyTimeWeekly + Absences + Tu
##    Data: data (Number of observations: 2392)
##   Draws: 4 chains, each with iter = 4000; warmup = 2000; thin = 1;
##          total post-warmup draws = 8000
##
## Multilevel Hyperparameters:
## ~Cluster_HC (Number of levels: 5)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     0.50      0.29     0.21     1.24 1.00     2088     3111
##
## Regression Coefficients:
##                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept[1]         -7.78      0.36    -8.46    -7.03 1.00     2699     3208
## Intercept[2]         -5.58      0.33    -6.18    -4.87 1.00     2716     3353
## Intercept[3]         -3.55      0.32    -4.11    -2.83 1.00     2724     3161
## Intercept[4]         -1.54      0.31    -2.09    -0.86 1.00     2765     3207
## Gender               -0.05      0.09    -0.23     0.12 1.00    10010     5814
## ParentalEducation     0.03      0.05    -0.06     0.11 1.00    10271     5848
## ParentalSupport      -0.56      0.04    -0.64    -0.47 1.00     8011     5915
## StudyTimeWeekly      -0.59      0.05    -0.69    -0.50 1.00     7091     5659
## Absences              3.20      0.09     3.02     3.39 1.00     4796     5886
## Tutoring             -0.94      0.10    -1.13    -0.74 1.00     8469     5854
## Ethnicity            -0.05      0.04    -0.14     0.03 1.00    10260     6206
##
## Further Distributional Parameters:
##      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## disc     1.00      0.00     1.00     1.00   NA       NA       NA
##
## Draws were sampled using sample(hmc). 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).
```

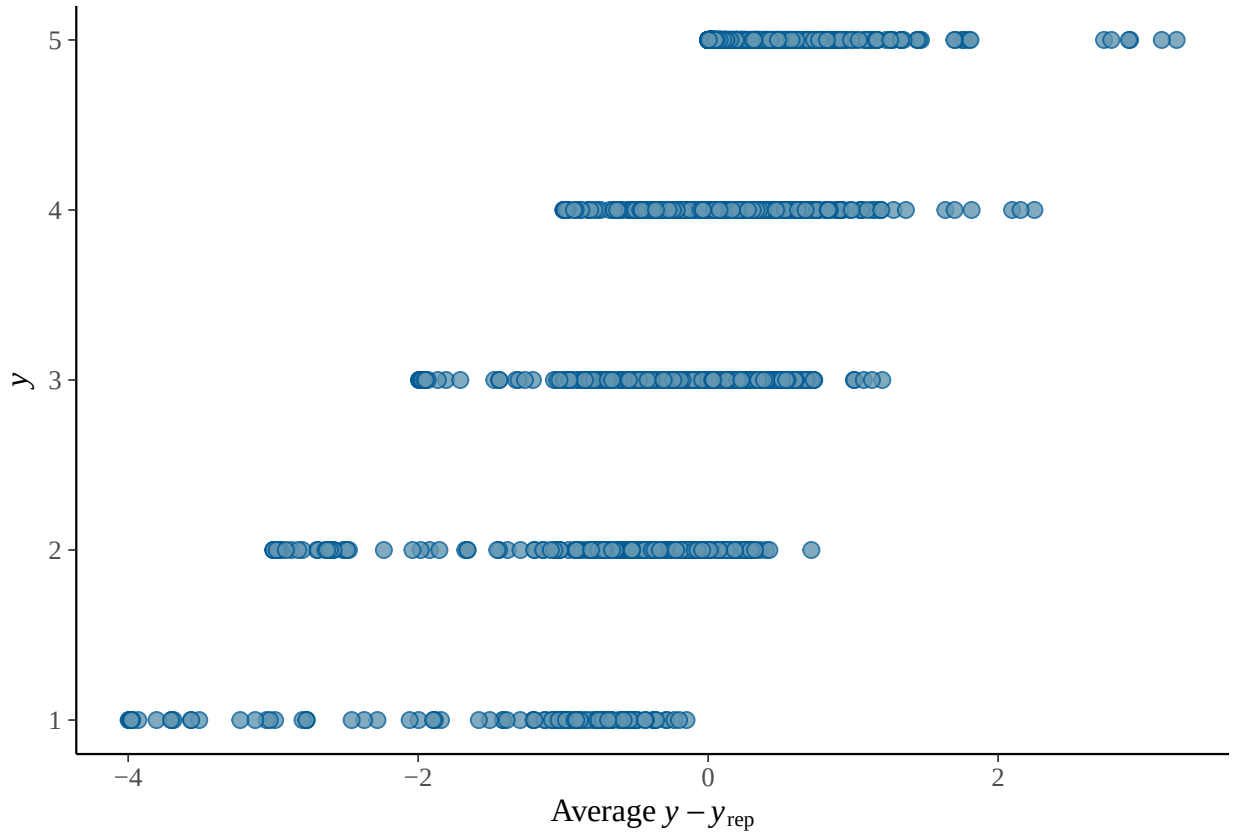
```
pp_check(mdl_ord_ag_prior, type = "bars")
```

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




```
pp_check(mdl_ord_ag_prior, type = "error_scatter_avg")
```

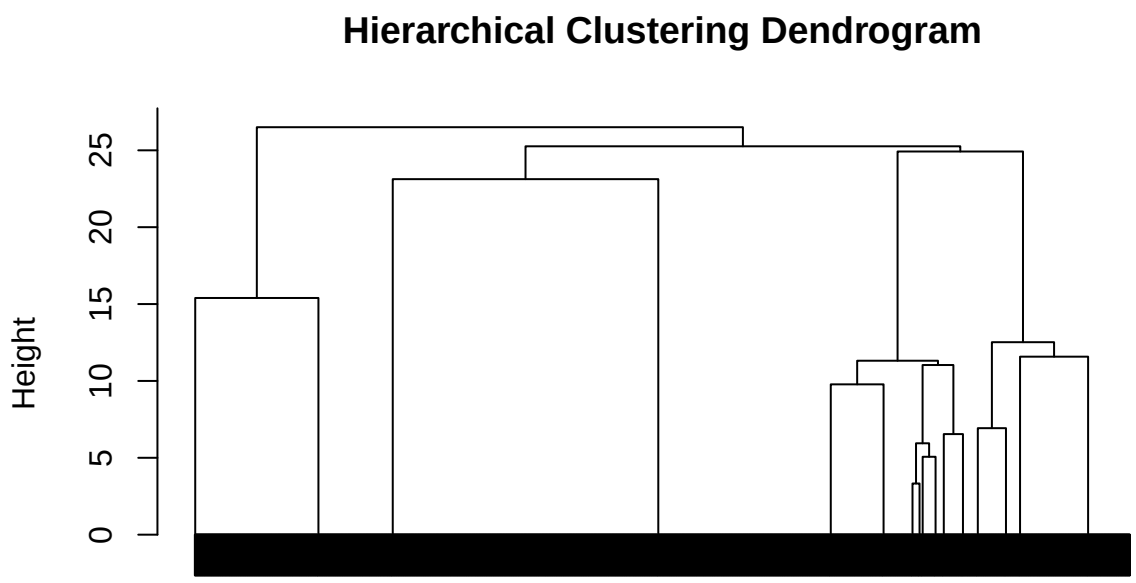
```
## Using all posterior draws for ppc type 'error_scatter_avg' by default.
```



#4) Ordinal model for ActivityGroup Refined:

```
# Read in your dataset
data <- read.csv("Student_performance.csv")

# Scale continuous predictors
data$StudyTimeWeekly <- scale(data$StudyTimeWeekly)
data$Absences <- scale(data$Absences)

# Convert GradeClass to an ordered factor
data$GradeClass <- factor(data$GradeClass,
                                levels = c(0, 1, 2, 3, 4),
                                labels = c("A", "B", "C", "D", "F"),
                                ordered = TRUE) # Ordinal


# Select variables for clustering
df_cluster_ActivityGroup <- data %>%
  select(Extracurricular, Sports, Music, Volunteering)

# Compute distance matrix
dist_matrix_ActivityGroup <- dist(df_cluster_ActivityGroup, method = "euclidean")

# Perform hierarchical clustering using Ward's method
hc_ActivityGroup <- hclust(dist_matrix_ActivityGroup, method = "ward.D2")
```

```r
# Plot dendrogram
plot(hc_ActivityGroup, labels = FALSE, main = "Hierarchical Clustering Dendrogram")
```




```r
# Cut tree to create 5 clusters
data$Cluster_HC <- cutree(hc_ActivityGroup, k = 5)

# Convert to factor
data$Cluster_HC <- as.factor(data$Cluster_HC)



priors <- c(
  set_prior("normal(0, 1)", class = "b"),  # Tight prior for regression coefficients
  set_prior("cauchy(0, 1)", class = "sd"),  # Random effects prior (same as previous)
  set_prior("student_t(3, 0, 2)", class = "Intercept")  # Informative prior for intercept
)
```

```
## Start sampling
```

```
## Running MCMC with 4 chains, at most 8 in parallel...
##
## Chain 1 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 2 Iteration:  100 / 4000 [  2%]  (Warmup)
```

```
## Chain 3 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 1 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 2 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 3 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 1 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 1 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 4 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 3 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 2 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 4 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 1 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 4 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 1 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 2 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 3 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 4 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 1 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 2 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 4 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 1 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 3 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 2 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 4 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 1 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 2 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 1 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 4 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 2 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 1 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 2 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 1 Iteration: 1500 / 4000 [ 37%]  (Warmup)
```

```
## Chain 3 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 2 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 1 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 3 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 4 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 1 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 4 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 3 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 2 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 4 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 3 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 2 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 4 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 2 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 4 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 2 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 1 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 4 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 1 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 2 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 4 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 3 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 2 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 4 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 2 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 4 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 1 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 4 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 3400 / 4000 [ 85%]  (Sampling)
```

```
## Chain 1 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 2 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 4 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 2 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 1 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 4 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 3 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 2 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 4 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 2 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2 finished in 471.9 seconds.
## Chain 1 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 3 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 4 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 4 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 3 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 4 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 1 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 4 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 3 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 1 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 4 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 1 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 3 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 4 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 1 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 4 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4 finished in 669.2 seconds.
## Chain 3 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 1 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 1 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 3 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 1 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 1 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 1 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1 finished in 816.3 seconds.
## Chain 3 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 3700 / 4000 [ 92%]  (Sampling)
```

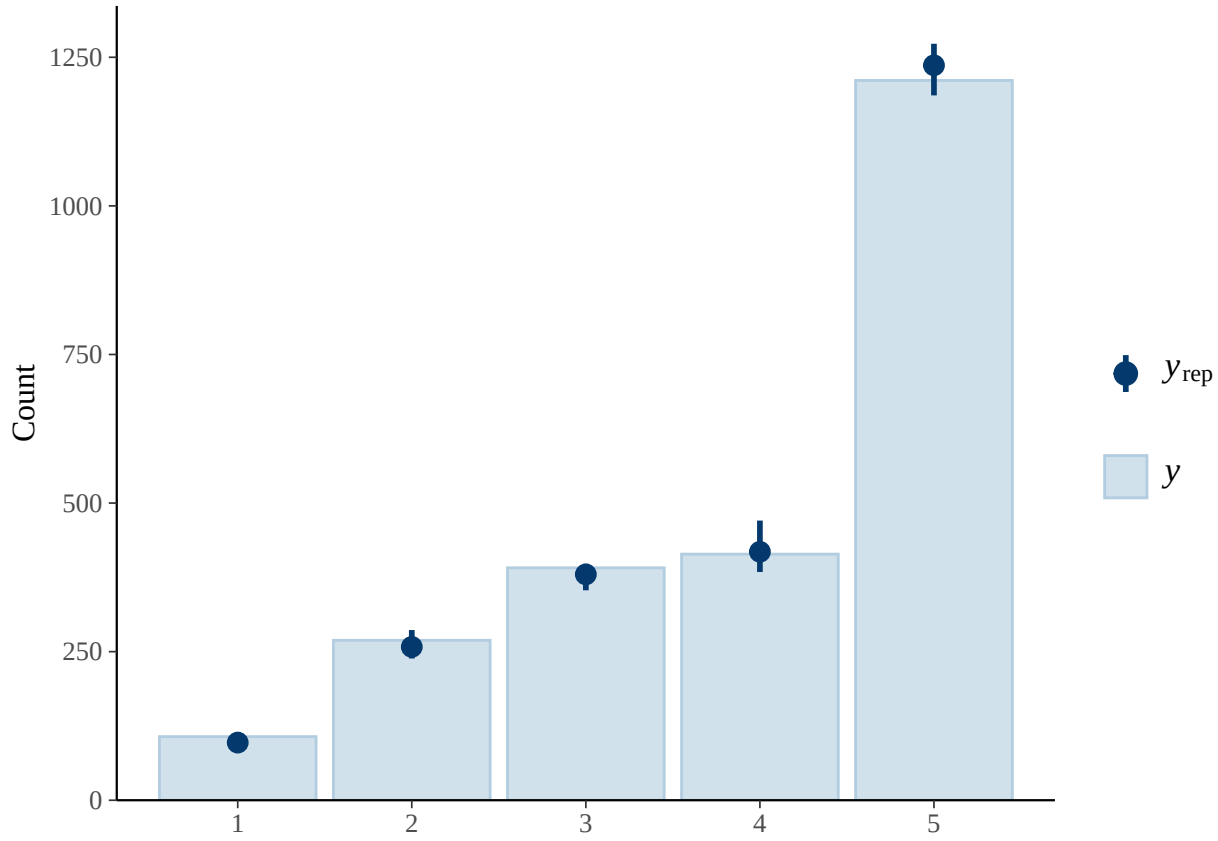
```
## Chain 3 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 3 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3 finished in 970.6 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 732.0 seconds.
## Total execution time: 970.7 seconds.
```

```
summary(mdl_ord_ag_prior_refined)
```

```
##  Family: cumulative
##   Links: mu = logit; disc = identity
## Formula: GradeClass ~ Tutoring + ParentalSupport + StudyTimeWeekly + Absences + Ethnicity + (1 | Clus
##    Data: data (Number of observations: 2392)
##   Draws: 4 chains, each with iter = 4000; warmup = 2000; thin = 1;
##          total post-warmup draws = 8000
##
## Multilevel Hyperparameters:
## ~Cluster_HC (Number of levels: 5)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     0.49      0.27     0.21     1.21 1.00     2110     3056
##
## Regression Coefficients:
##                 Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept[1]       -7.80      0.35    -8.43    -7.05 1.00     2963     3334
## Intercept[2]       -5.61      0.33    -6.18    -4.91 1.00     2881     3144
## Intercept[3]       -3.57      0.31    -4.11    -2.92 1.00     2874     3216
## Intercept[4]       -1.57      0.30    -2.08    -0.90 1.00     2851     3255
## Tutoring           -0.94      0.10    -1.13    -0.74 1.00     7300     6202
## ParentalSupport    -0.56      0.04    -0.64    -0.47 1.00     7150     5358
## StudyTimeWeekly    -0.59      0.05    -0.69    -0.50 1.00     7486     5913
## Absences            3.20      0.09     3.02     3.38 1.00     5025     5757
## Ethnicity          -0.05      0.04    -0.14     0.04 1.00     8067     5874
##
## Further Distributional Parameters:
##      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## disc     1.00      0.00     1.00     1.00   NA       NA       NA
##
## Draws were sampled using sample(hmc). 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).
```

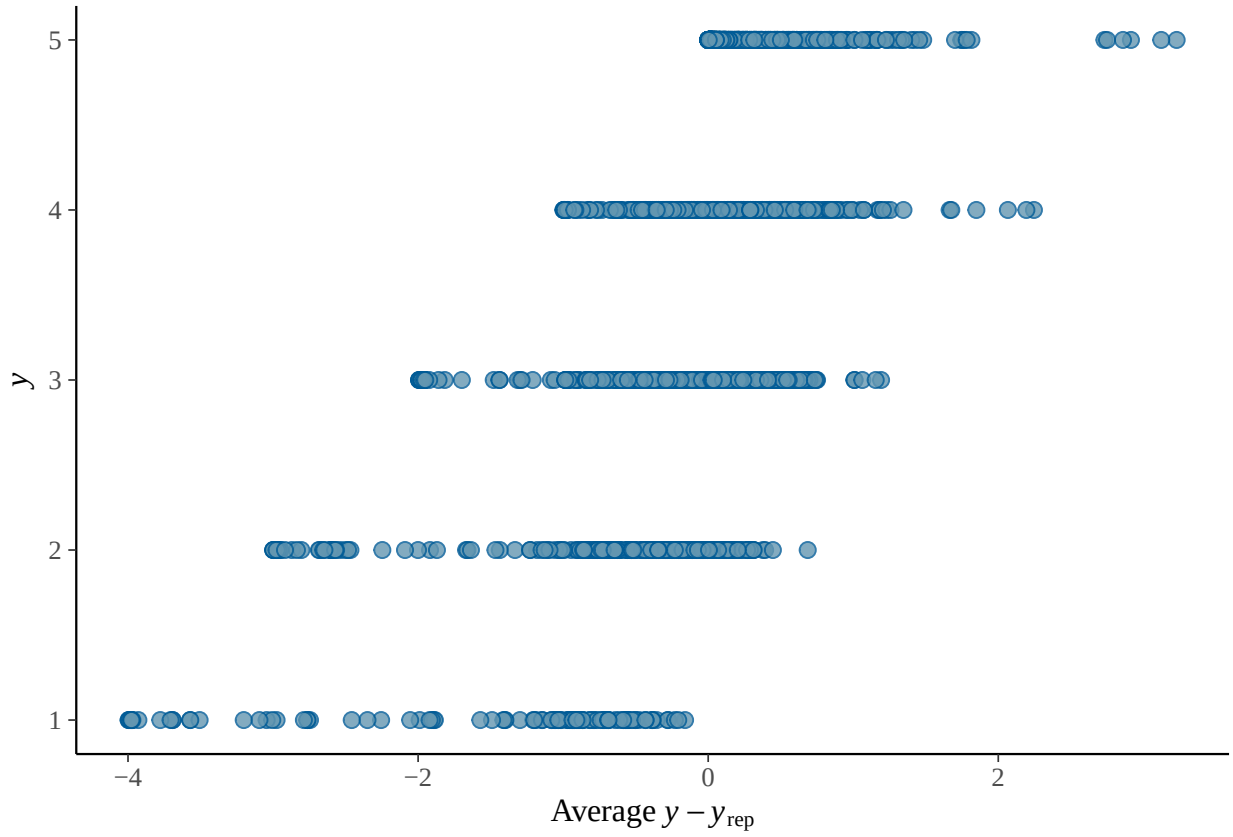
```
pp_check(mdl_ord_ag_prior_refined, type = "bars")
```

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




```
pp_check(mdl_ord_ag_prior_refined, type = "error_scatter_avg")
```

```
## Using all posterior draws for ppc type 'error_scatter_avg' by default.
```

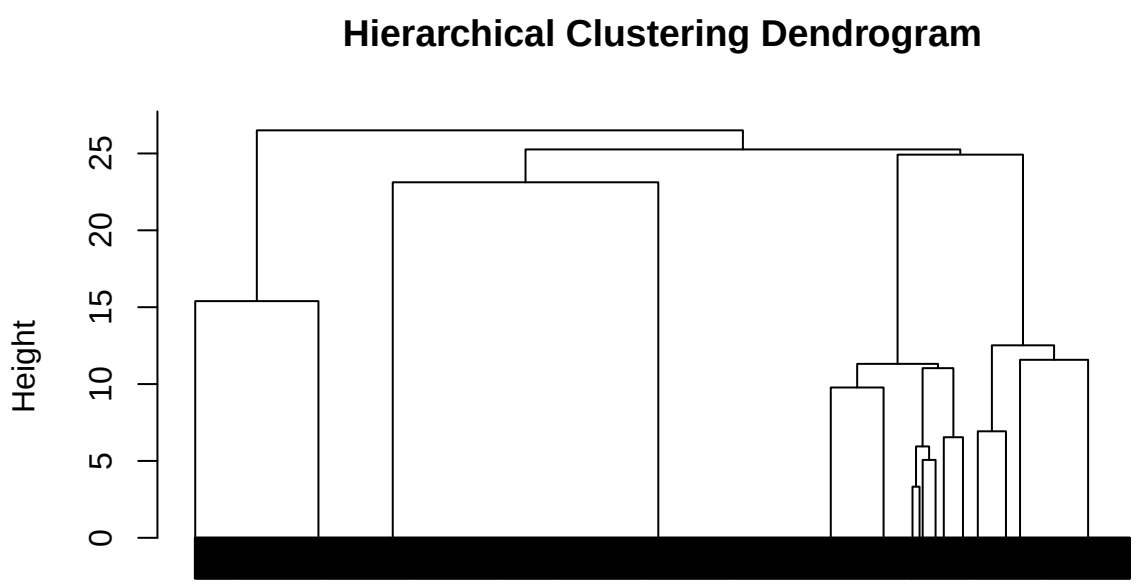


#5) Continuous model for ActivityGroup:

```
# Read in your dataset
data <- read.csv("Student_performance.csv")

sum(is.na(data$ActivityGroup))
```

```
## [1] 0
```

```
# Scale continuous predictors
data$StudyTimeWeekly <- scale(data$StudyTimeWeekly)
data$Absences <- scale(data$Absences)


df_cluster_ActivityGroup <- data %>%
  select(Extracurricular, Sports, Music, Volunteering)

# Compute distance matrix
dist_matrix_ActivityGroup <- dist(df_cluster_ActivityGroup, method = "euclidean")

# Perform hierarchical clustering using Ward's method
hc_ActivityGroup <- hclust(dist_matrix_ActivityGroup, method = "ward.D2")


# Plot dendrogram
plot(hc_ActivityGroup, labels = FALSE, main = "Hierarchical Clustering Dendrogram")
```




```
# Cut tree to create 5 clusters
data$Cluster_HC <- cutree(hc_ActivityGroup, k = 5)

# Convert to factor
data$Cluster_HC <- as.factor(data$Cluster_HC)


priors <- c(
  set_prior("normal(0, 1)", class = "b"),  # Tight prior for regression coefficients
  set_prior("cauchy(0, 1)", class = "sd"),  # Random effects prior (same as previous)
  set_prior("student_t(3, 0, 2)", class = "Intercept")  # Informative prior for intercept
)
```

```
## Start sampling
## Running MCMC with 4 chains, at most 8 in parallel...
##
## Chain 1 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4 Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 3 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 1 Iteration:  100 / 4000 [  2%]  (Warmup)
## Chain 4 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 2 Iteration:  200 / 4000 [  5%]  (Warmup)
```

```
## Chain 3 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 1 Iteration:  200 / 4000 [  5%]  (Warmup)
## Chain 4 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 1 Iteration:  300 / 4000 [  7%]  (Warmup)
## Chain 2 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1 Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 1 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 2 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 4 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 1 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 3 Iteration:  500 / 4000 [ 12%]  (Warmup)
## Chain 2 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 4 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 1 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 2 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 3 Iteration:  600 / 4000 [ 15%]  (Warmup)
## Chain 4 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 1 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 3 Iteration:  700 / 4000 [ 17%]  (Warmup)
## Chain 1 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 4 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 3 Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 1 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 2 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 4 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 1 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 3 Iteration:  900 / 4000 [ 22%]  (Warmup)
## Chain 2 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 4 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 1 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3 Iteration: 1000 / 4000 [ 25%]  (Warmup)
## Chain 2 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 4 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 1 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 3 Iteration: 1100 / 4000 [ 27%]  (Warmup)
## Chain 4 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 2 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 1 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 3 Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 2 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 1 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 3 Iteration: 1300 / 4000 [ 32%]  (Warmup)
## Chain 4 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 1 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 3 Iteration: 1400 / 4000 [ 35%]  (Warmup)
## Chain 4 Iteration: 1600 / 4000 [ 40%]  (Warmup)
```

```
## Chain 1 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 3 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 2 Iteration: 1500 / 4000 [ 37%]  (Warmup)
## Chain 4 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 1 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 3 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 2 Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 3 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 4 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 2 Iteration: 1700 / 4000 [ 42%]  (Warmup)
## Chain 3 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 1 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2 Iteration: 1800 / 4000 [ 45%]  (Warmup)
## Chain 3 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 4 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 1900 / 4000 [ 47%]  (Warmup)
## Chain 4 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 3 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2 Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 2 Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 1 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 4 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 2 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 3 Iteration: 2100 / 4000 [ 52%]  (Sampling)
## Chain 2 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 1 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 4 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 2 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 1 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 2200 / 4000 [ 55%]  (Sampling)
## Chain 2 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 4 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 3 Iteration: 2300 / 4000 [ 57%]  (Sampling)
## Chain 1 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 2 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 4 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 1 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 2 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2 Iteration: 3200 / 4000 [ 80%]  (Sampling)
```

```
## Chain 4 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 1 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 2 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 2 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 3 Iteration: 2500 / 4000 [ 62%]  (Sampling)
## Chain 4 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 2 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 2 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 1 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 3 Iteration: 2600 / 4000 [ 65%]  (Sampling)
## Chain 2 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 1 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 2 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 1 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 3 Iteration: 2700 / 4000 [ 67%]  (Sampling)
## Chain 2 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2 finished in 80.7 seconds.
## Chain 4 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 1 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 4 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 3 Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 4 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 1 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 2900 / 4000 [ 72%]  (Sampling)
## Chain 4 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 1 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 4 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3000 / 4000 [ 75%]  (Sampling)
## Chain 1 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 1 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 3100 / 4000 [ 77%]  (Sampling)
## Chain 4 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 1 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1 finished in 101.8 seconds.
## Chain 3 Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 4 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 3300 / 4000 [ 82%]  (Sampling)
## Chain 4 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4 finished in 111.4 seconds.
## Chain 3 Iteration: 3400 / 4000 [ 85%]  (Sampling)
## Chain 3 Iteration: 3500 / 4000 [ 87%]  (Sampling)
## Chain 3 Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3 Iteration: 3700 / 4000 [ 92%]  (Sampling)
## Chain 3 Iteration: 3800 / 4000 [ 95%]  (Sampling)
## Chain 3 Iteration: 3900 / 4000 [ 97%]  (Sampling)
## Chain 3 Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3 finished in 141.6 seconds.
```

```
##
## All 4 chains finished successfully.
## Mean chain execution time: 108.9 seconds.
## Total execution time: 141.7 seconds.
```

```
## Warning: 1 of 8000 (0.0%) transitions ended with a divergence.
## See https://mc-stan.org/misc/warnings for details.
```

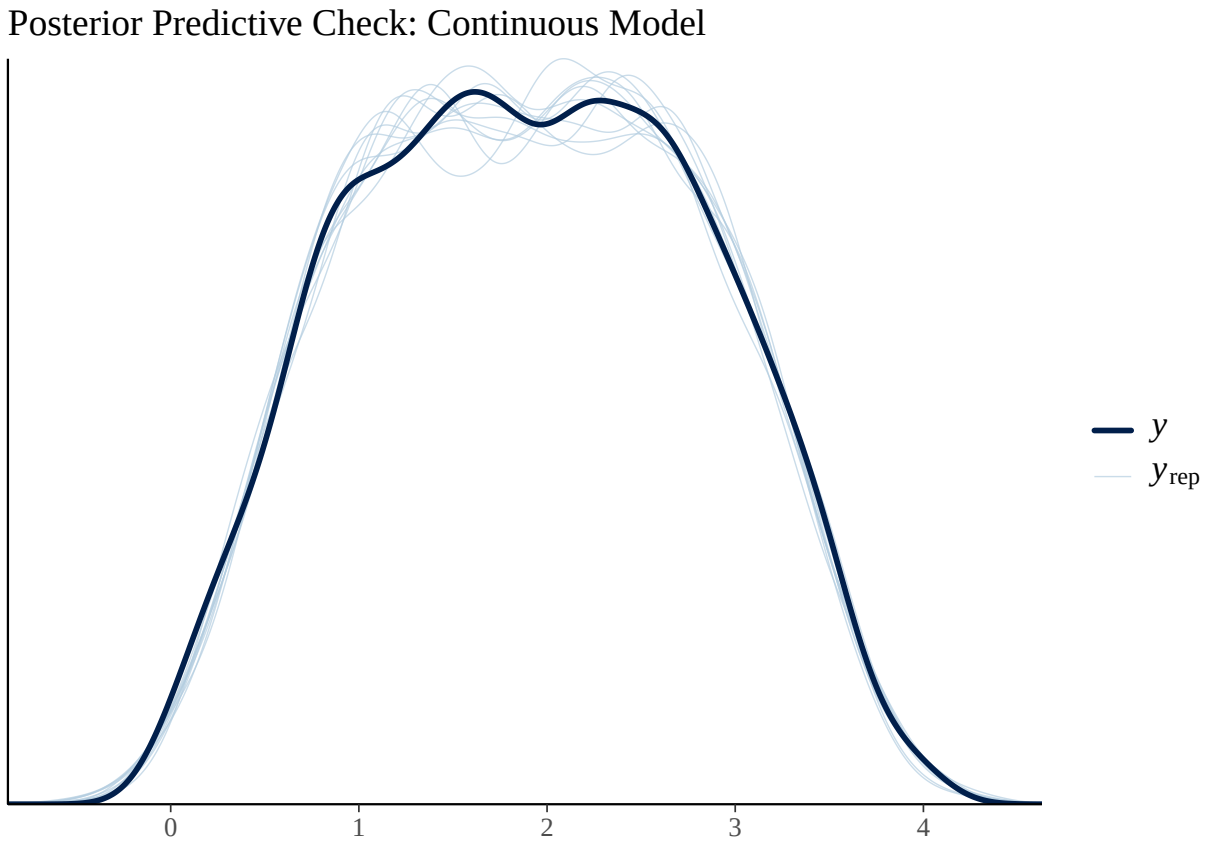
```
summary(mdl_cnt_ag_prior_refined)
```

```
## Warning: There were 1 divergent transitions after warmup. Increasing
## adapt_delta above 0.999 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: GPA ~ ParentalSupport + StudyTimeWeekly + Absences + Tutoring + Ethnicity + (1 | Cluster_HC)
##    Data: data (Number of observations: 2392)
##   Draws: 4 chains, each with iter = 4000; warmup = 2000; thin = 1;
##          total post-warmup draws = 8000
##
## Multilevel Hyperparameters:
## ~Cluster_HC (Number of levels: 5)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     0.16      0.09     0.07     0.41 1.00     2003     2696
##
## Regression Coefficients:
##                 Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept           1.52      0.08     1.35     1.67 1.00     1987     2188
## ParentalSupport     0.15      0.00     0.14     0.16 1.00     8942     4895
## StudyTimeWeekly     0.16      0.00     0.15     0.17 1.00    10218     5189
## Absences           -0.84      0.00    -0.85    -0.83 1.00     9353     5782
## Tutoring            0.25      0.01     0.23     0.27 1.00     9104     5483
## Ethnicity           0.00      0.00    -0.00     0.01 1.00     9032     5398
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.22      0.00     0.21     0.22 1.00     8582     4878
##
## Draws were sampled using sample(hmc). 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(mdl_cnt_ag_prior_refined, type = "dens_overlay") +
  ggtitle("Posterior Predictive Check: Continuous Model")
```

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




```
bayes_R2(mdl_ord_ag_prior_refined)
```

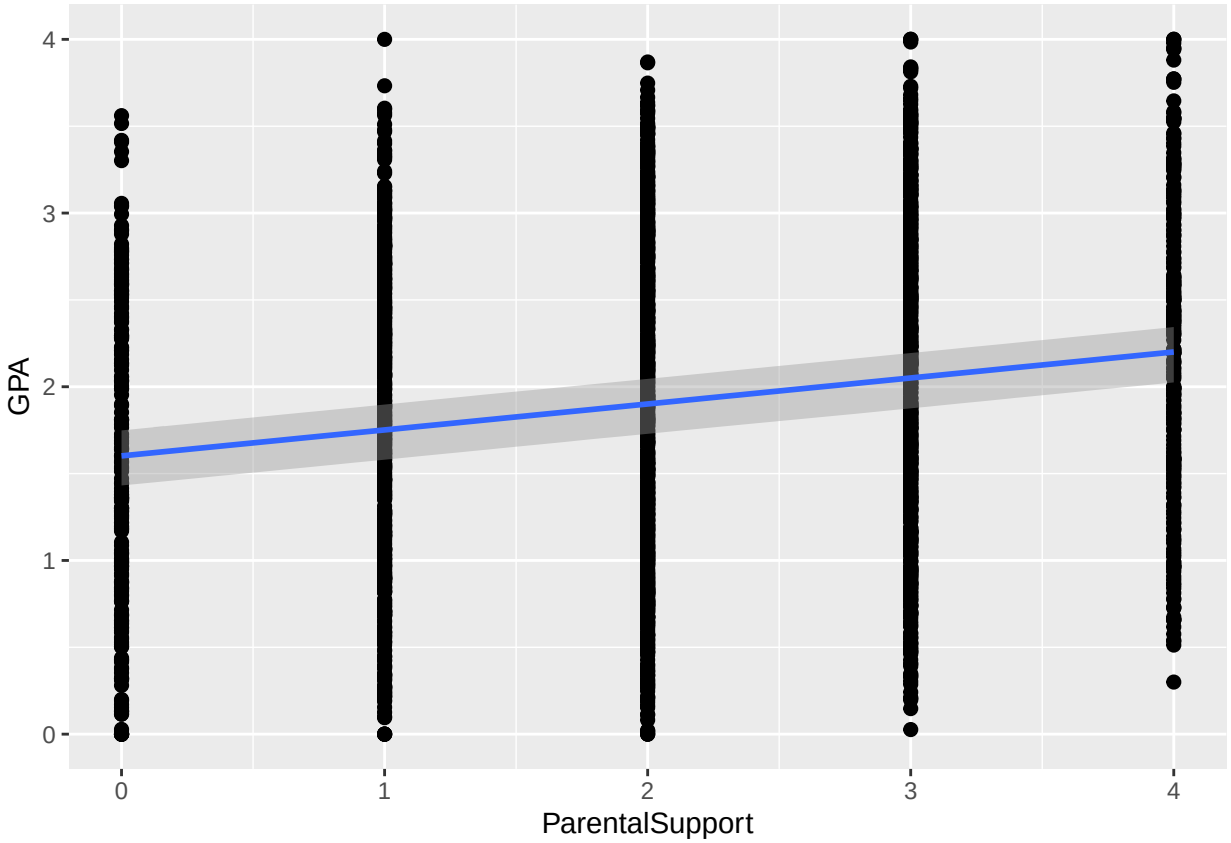
```
## Warning: Predictions are treated as continuous variables in 'bayes_R2' which is
## likely invalid for ordinal families.
```

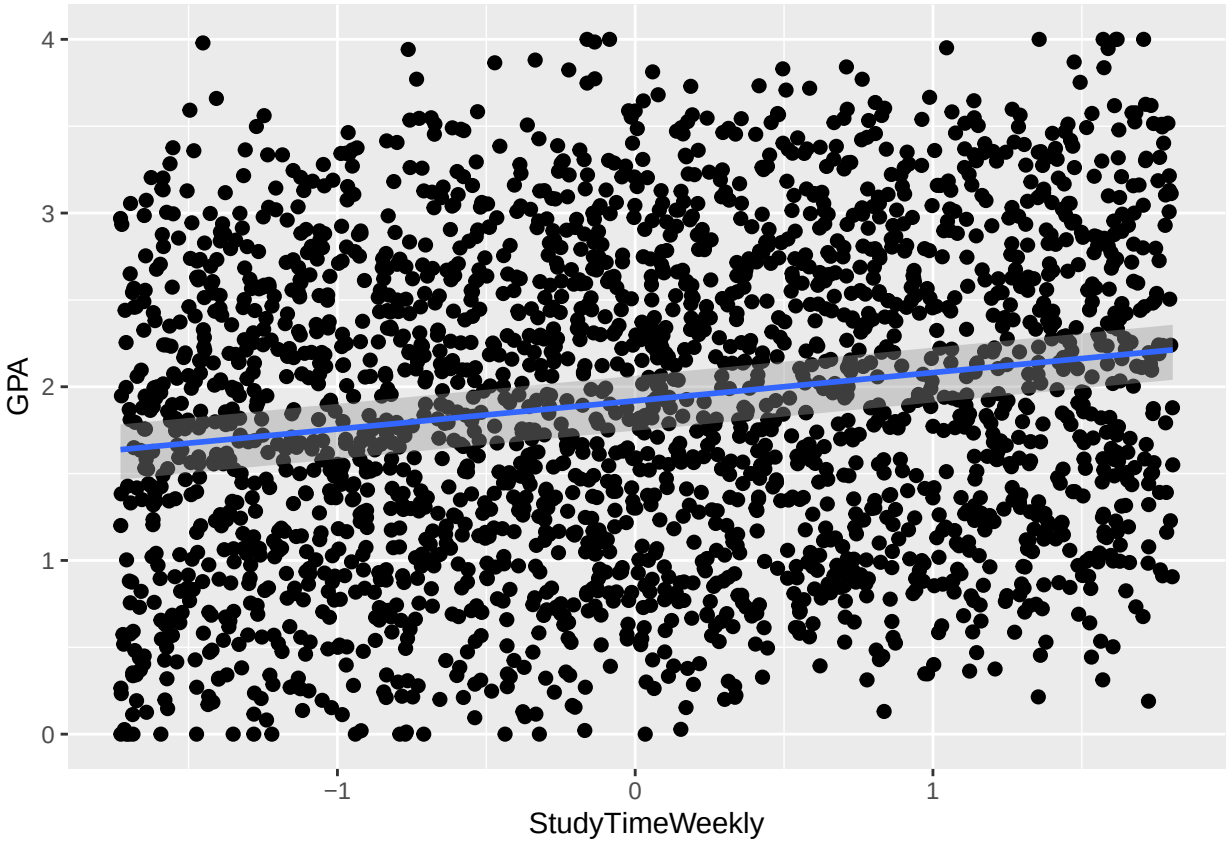
```
##     Estimate   Est.Error      Q2.5     Q97.5
## R2 0.6816029 0.005400606 0.6705197 0.6918628
```

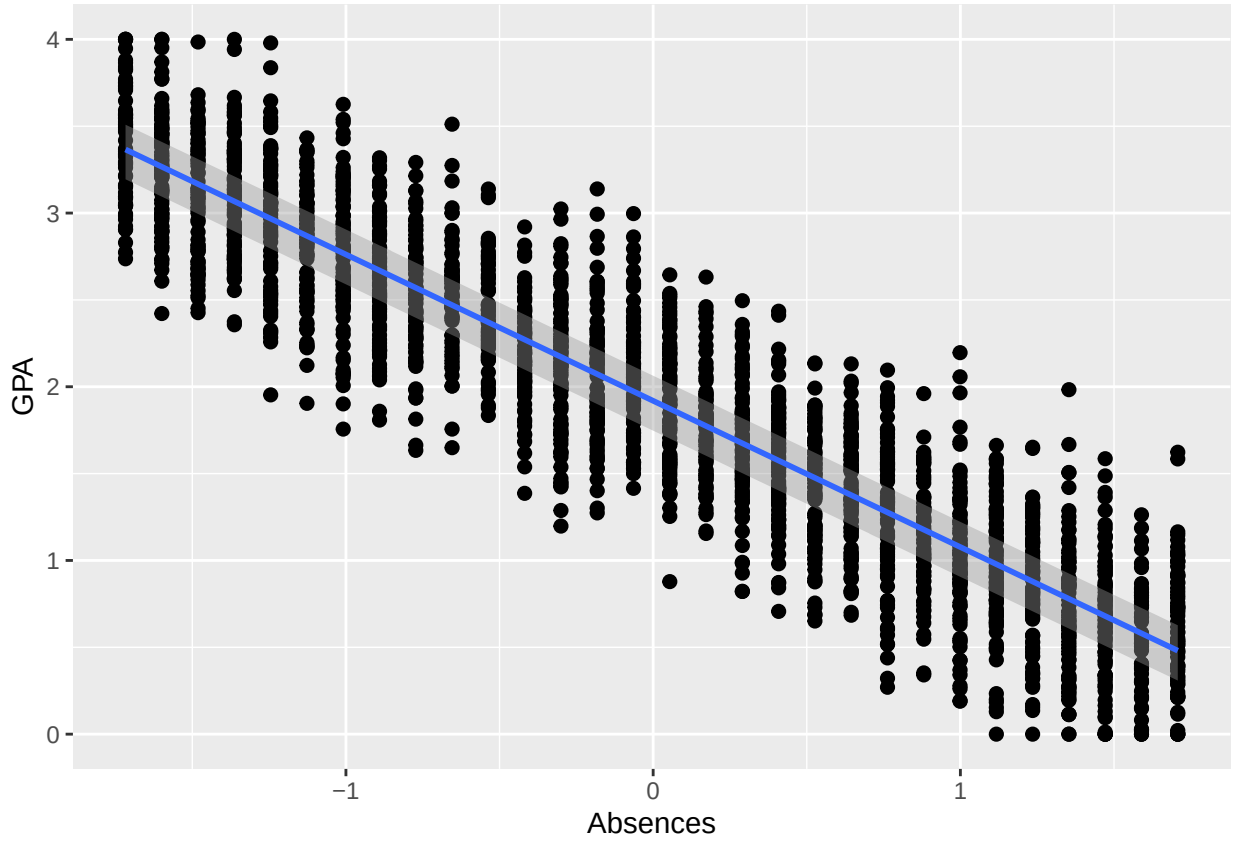
```
bayes_R2(mdl_cnt_ag_prior_refined)
```

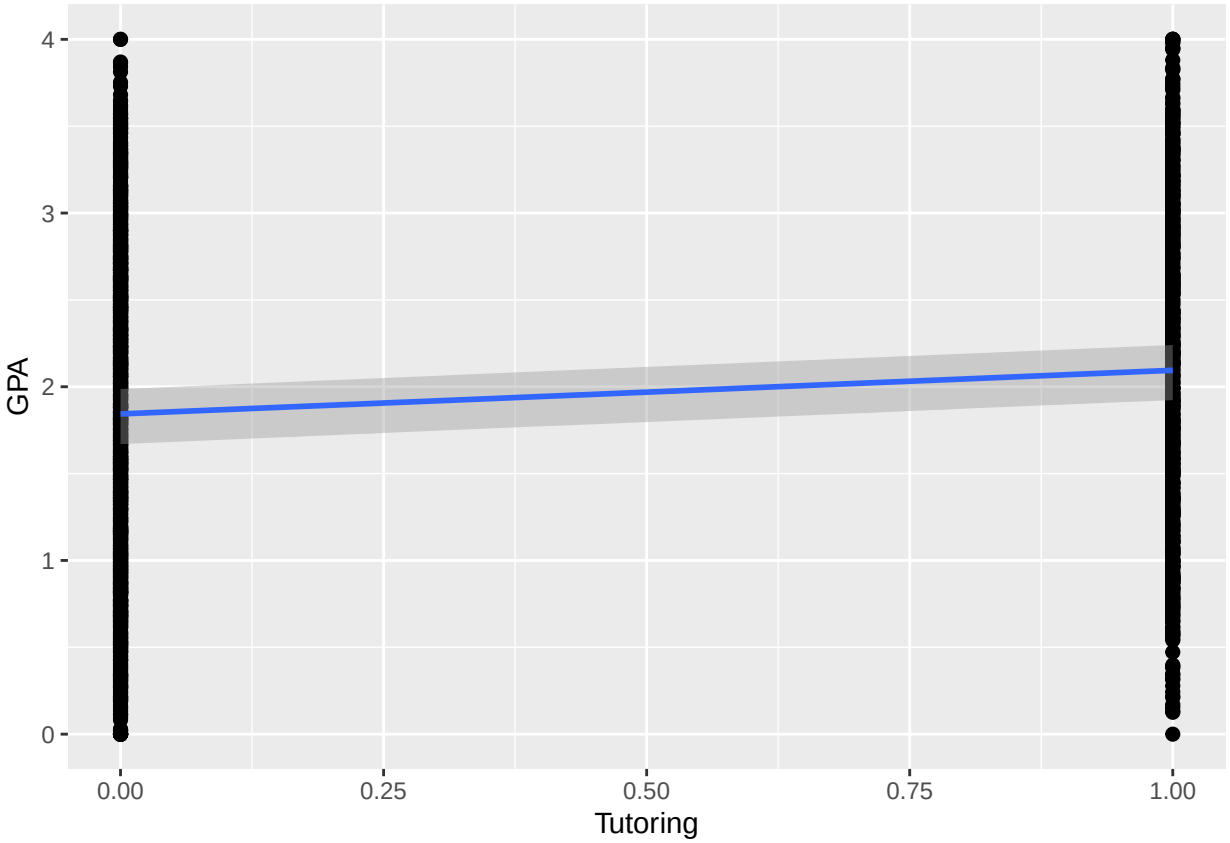
```
##     Estimate    Est.Error      Q2.5     Q97.5
## R2 0.9432906 0.0005496363 0.9421515 0.9442746
```

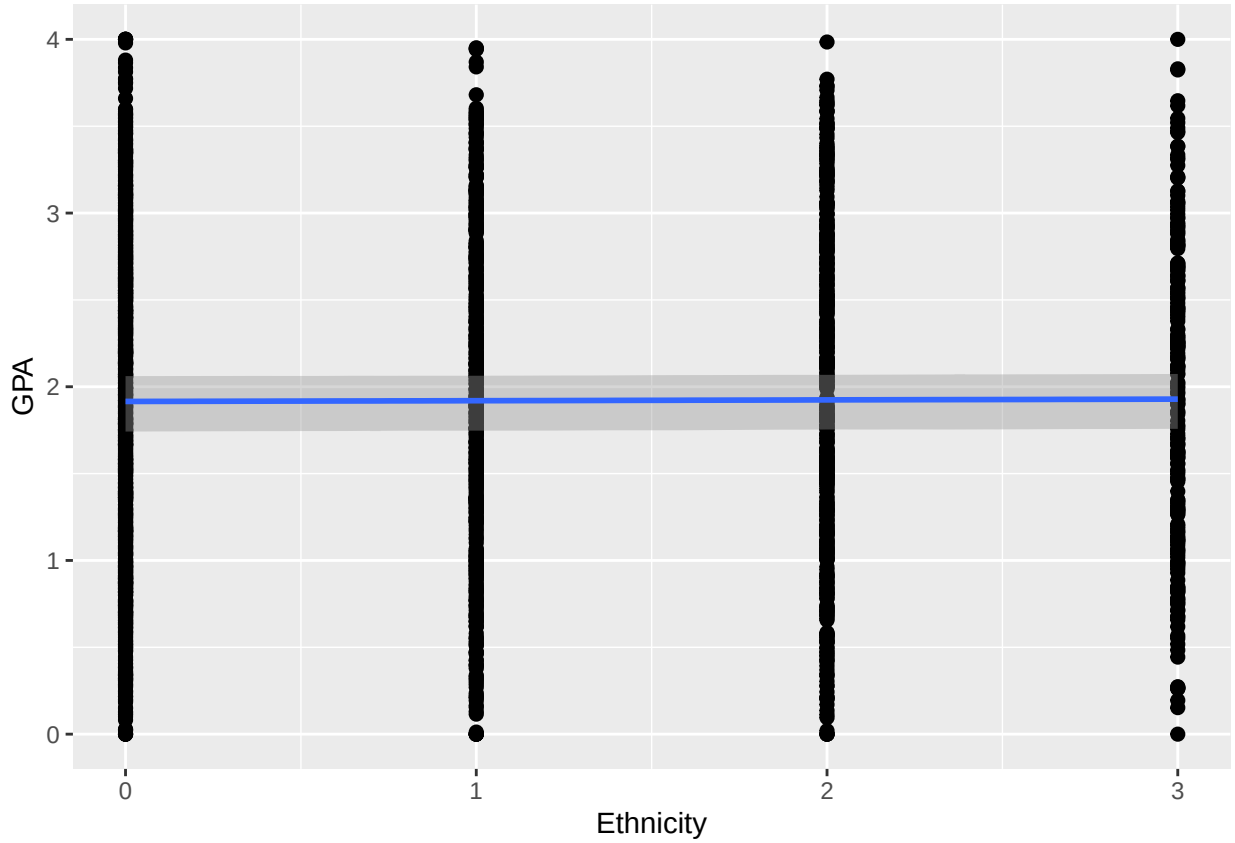
```
plot(conditional_effects(mdl_cnt_ag_prior_refined), points = TRUE)
```


GPA
4
3
2
1
0
0
1
2
3
4
ParentalSupport

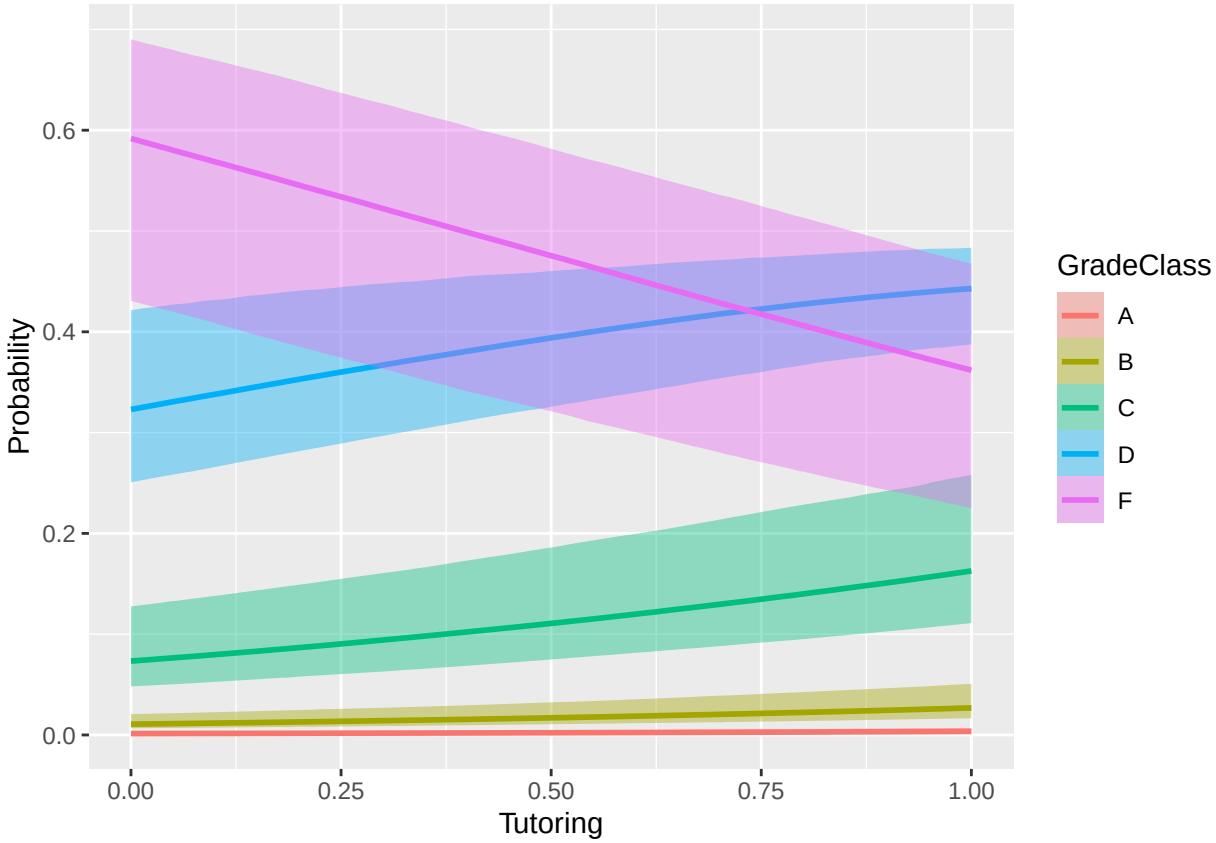

GPA
4
3
2
1
0
−1
0
1
StudyTimeWeekly

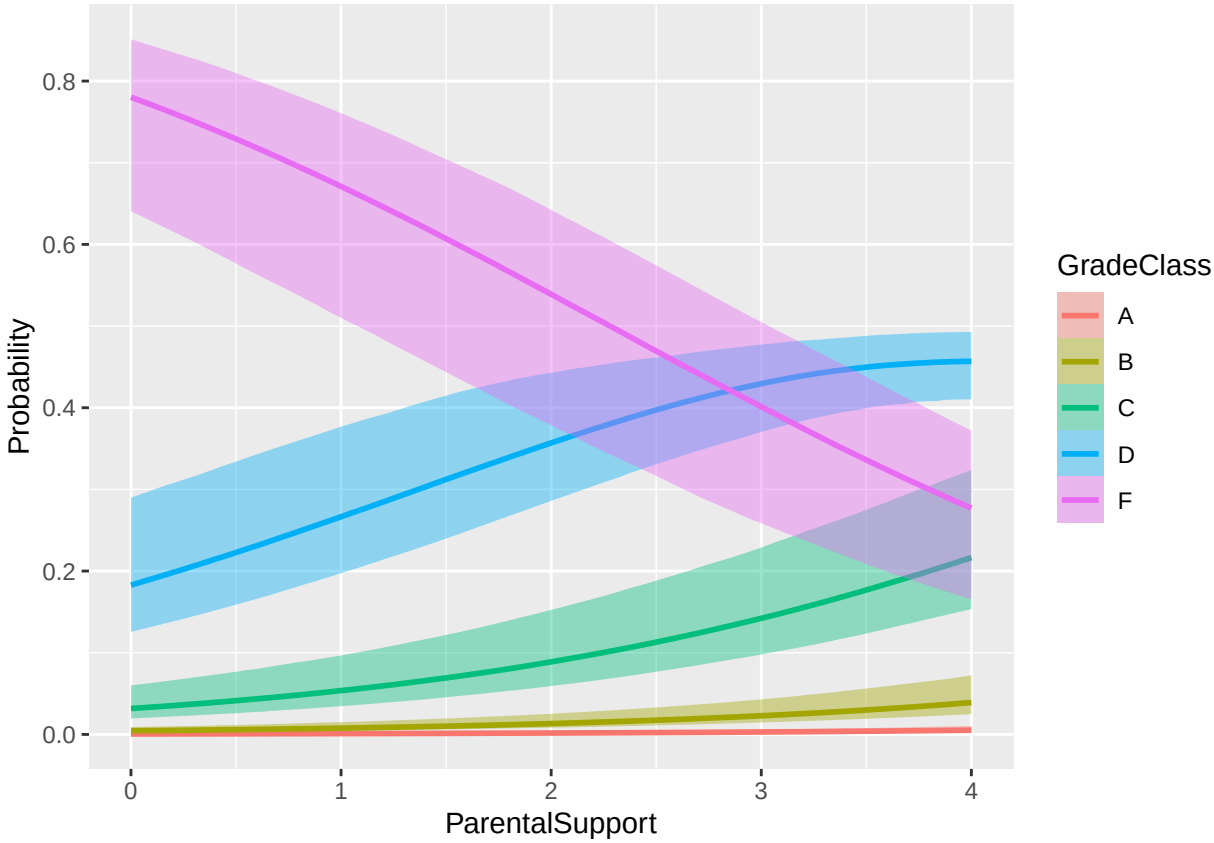

GPA
4
3
2
1
0
−1
0
1
Absences

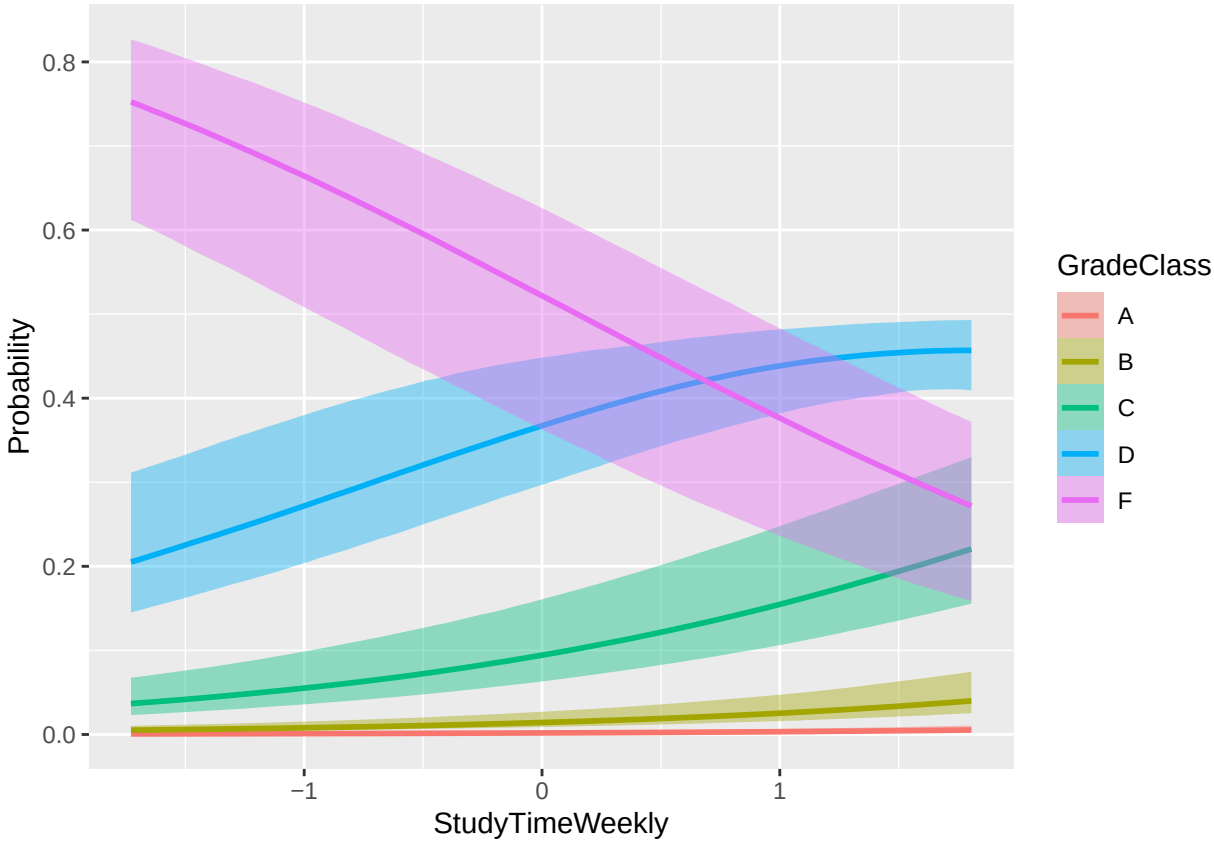

GPA
4
3
2
1
0
0.00
0.25
0.50
0.75
1.00
Tutoring

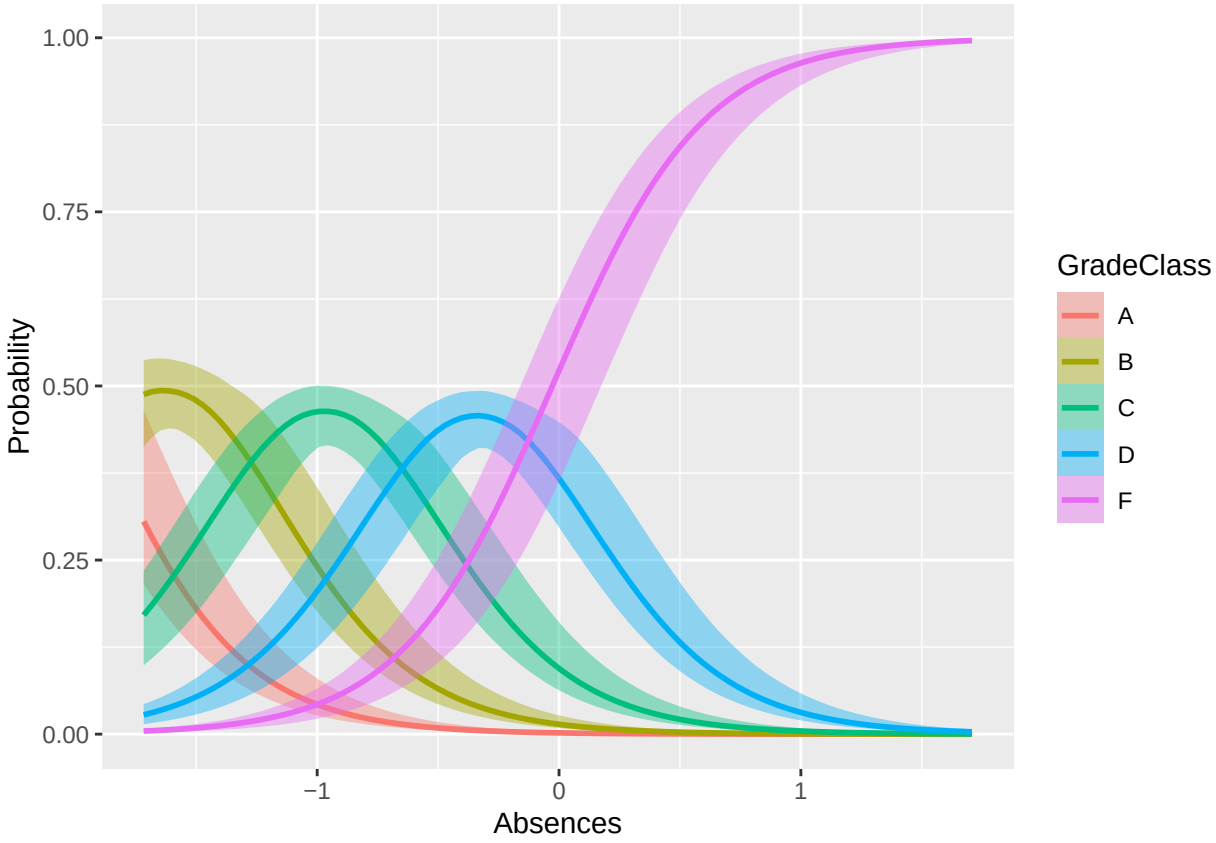


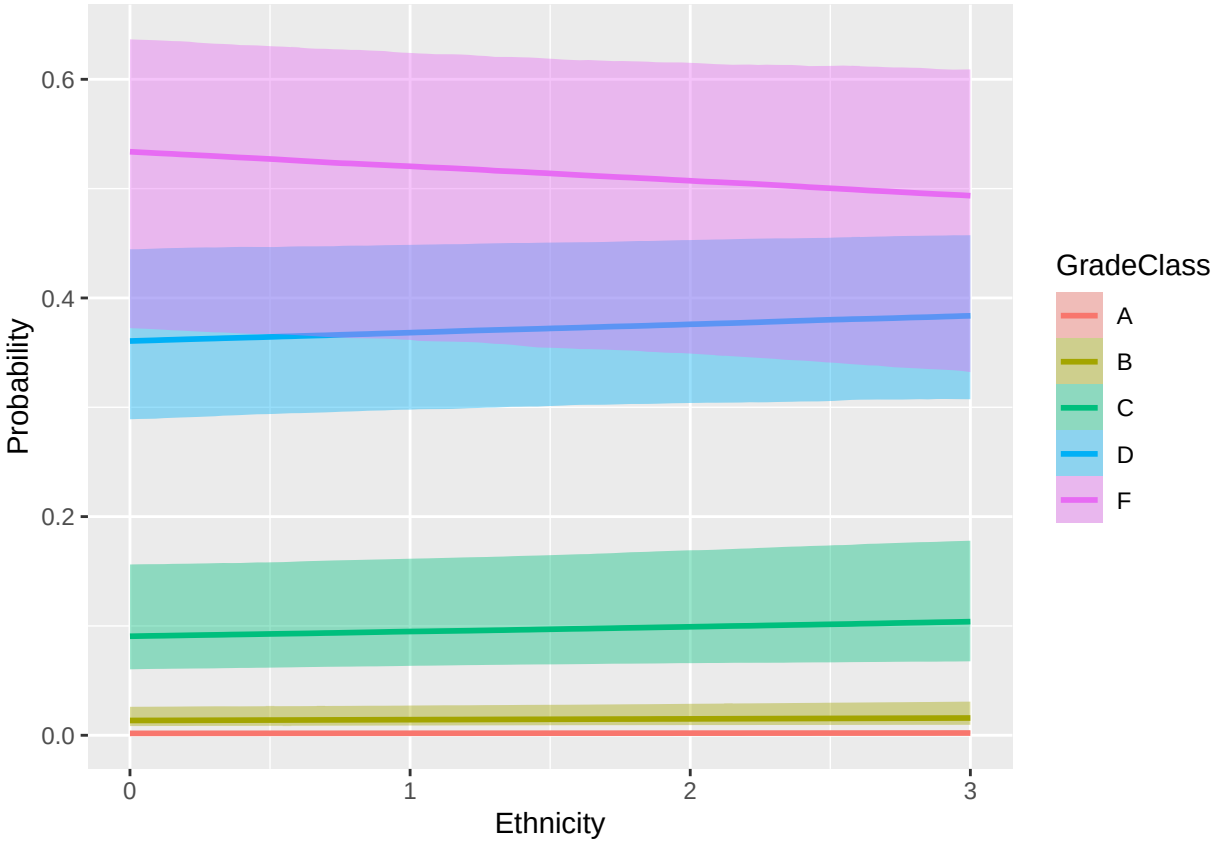

```
plot(conditional_effects(mdl_ord_ag_prior_refined, categorical = TRUE), points = TRUE)
```


Probability
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
Tutoring
GradeClass
A
B
C
D
F


0.8
0.6
0.4
0.2
0.0
Probability
0
1
2
3
4
ParentalSupport
GradeClass
A
B
C
D
F


0.8
0.6
0.4
0.2
0.0
Probability
−1
0
1
StudyTimeWeekly
GradeClass
A
B
C
D
F


1.00
0.75
0.50
0.25
0.00
Probability
−1
0
1
Absences
GradeClass
A
B
C
D
F


0.6
0.4
0.2
0.0
Probability
0
1
2
3
Ethnicity
GradeClass
A
B
C
D
F