

# Models

2025-04-28

```
library(tidyverse)
library(janitor)
library(rstan)
library(rstanarm)
library(bayesplot)
library(MCMCpack)
library(lme4)
student_data <- read.csv("student-scores.csv");
clean_data <- read.csv("student-scores-clean.csv")
head(student_data)
```

##	id	first_name	last_name	email	gender
## 1	1	Paul	Casey	paul.casey.1@gslingacademy.com	male
## 2	2	Danielle	Sandoval	danielle.sandoval.2@gslingacademy.com	female
## 3	3	Tina	Andrews	tina.andrews.3@gslingacademy.com	female
## 4	4	Tara	Clark	tara.clark.4@gslingacademy.com	female
## 5	5	Anthony	Campos	anthony.campos.5@gslingacademy.com	male
## 6	6	Kelly	Wade	kelly.wade.6@gslingacademy.com	female

##	part_time_job	absence_days	extracurricular_activities	weekly_self_study_hours
## 1	False	3	False	27
## 2	False	2	False	47
## 3	False	9	True	13
## 4	False	5	False	3
## 5	False	5	False	10
## 6	False	2	False	26

##	career_aspiration	math_score	history_score	physics_score	chemistry_score
## 1	Lawyer	73	81	93	97
## 2	Doctor	90	86	96	100
## 3	Government Officer	81	97	95	96
## 4	Artist	71	74	88	80
## 5	Unknown	84	77	65	65
## 6	Unknown	93	100	67	78

##	biology_score	english_score	geography_score
## 1	63	80	87
## 2	90	88	90
## 3	65	77	94
## 4	89	63	86
## 5	80	74	76
## 6	72	80	84

```
head(clean_data)
```

##	id	first_name	last_name	email	gender
## 1	1	Paul	Casey	paul.casey.1@gslingacademy.com	1
## 2	2	Danielle	Sandoval	danielle.sandoval.2@gslingacademy.com	0
## 3	3	Tina	Andrews	tina.andrews.3@gslingacademy.com	0

## 4	4	Tara	Clark	tara.clark.4@gslingacademy.com	0	
## 5	5	Anthony	Campos	anthony.campos.5@gslingacademy.com	1	
## 6	6	Kelly	Wade	kelly.wade.6@gslingacademy.com	0	
##		part_time_job	absence_days	extracurricular_activities	weekly_self_study_hours	
## 1		0	3	0	27	
## 2		0	2	0	47	
## 3		0	9	1	13	
## 4		0	5	0	3	
## 5		0	5	0	10	
## 6		0	2	0	26	
##		career_aspiration	math_score	history_score	physics_score	chemistry_score
## 1		Lawyer	73	81	93	97
## 2		Doctor	90	86	96	100
## 3	Government	Officer	81	97	95	96
## 4		Artist	71	74	88	80
## 5		Unknown	84	77	65	65
## 6		Unknown	93	100	67	78
##		biology_score	english_score	geography_score	average_score	
## 1		63	80	87	82.00	
## 2		90	88	90	91.43	
## 3		65	77	94	86.43	
## 4		89	63	86	78.71	
## 5		80	74	76	74.43	
## 6		72	80	84	82.00	

$Y_i|B_0, B_1, \sigma^2 \sim N(\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \beta_4 x_{4i}, \sigma^2)$  where:  $x_{1i}, x_{2i}, x_{3i}, x_{4i}$  are the predictors for observation  $i$

$\beta_j \sim N(\mu, \tau^2)$  where  $j=0, 1, 2, 3, 4$

$\sigma^2 \sim InvGamma(\alpha_1, \alpha_2)$

```
#Block Gibbs Sampler
set.seed(4889)
clean_data <- read.csv("student-scores-clean.csv")

set.seed(8451)
y <- clean_data$average_score
x1 <- clean_data$part_time_job
x2 <- clean_data$absence_days
x3 <- clean_data$extracurricular_activities
x4 <- clean_data$weekly_self_study_hours

# Design matrix
X <- cbind(1, x1, x2, x3, x4)
n <- length(y)
p <- ncol(X)

# Hyperparameters
tau2 <- 10000^2
```

```

a <- b <- 1
mu0 <- rep(0, p)

S <- 2.5e4

#place to store data
posterior_beta <- matrix(NA, S, p)
posterior_sig2 <- rep(NA, S)

beta <- rep(0, p)
sig2 <- 1

XX <- t(X) %*% X
Xy <- t(X) %*% y

# block Gibbs sampler
for (s in 1:S) {

  # Update beta0
  v <- solve(XX / sig2 + diag(rep(1/tau2, p)))
  m <- v %*% (Xy / sig2 + mu0 / tau2)
  beta <- m + t(chol(v)) %*% rnorm(p)

  # Update sig2 (variance)
  sig2 <- rinvgamma(1, a + n/2,
                    b + t(y - X %*% beta) %*% (y - X %*% beta) / 2)

  # Store results
  posterior_beta[s, ] <- beta
  posterior_sig2[s] <- sig2
}

posterior2 <- cbind(posterior_beta, posterior_sig2)
colnames(posterior2) <- c("beta0", "beta1", "beta2",
                        "beta3", "beta4", "sigma")

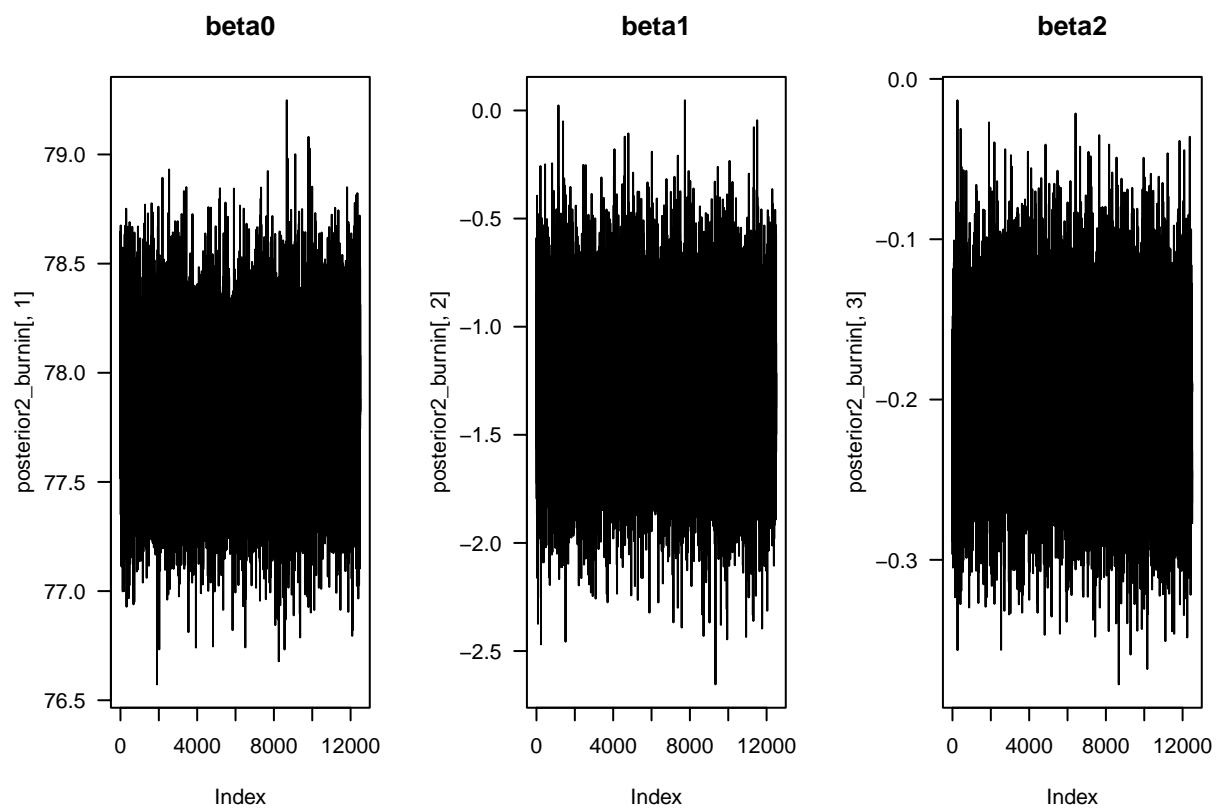
#remove burn-in
posterior2_burnin <- posterior2[1:round(s/2),]

head(posterior2_burnin)

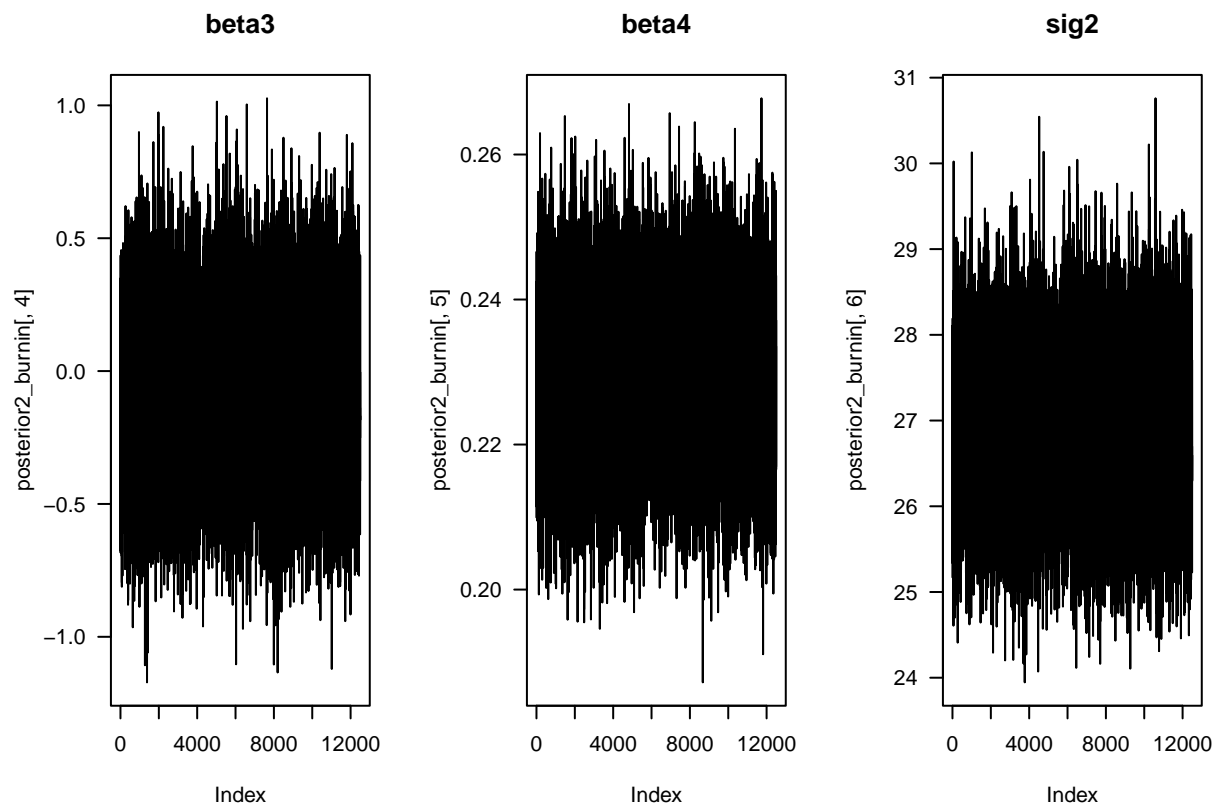
##          beta0      beta1      beta2      beta3      beta4      sigma
## [1,] 77.87654 -1.281166 -0.2003770 -0.1229279 0.2255845 27.69401
## [2,] 77.90033 -0.589276 -0.2092001 0.1524083 0.2279868 25.71107
## [3,] 78.04359 -1.461563 -0.2041925 -0.2506536 0.2197806 26.16819
## [4,] 77.76850 -1.559494 -0.2369275 -0.0571564 0.2424850 27.25814
## [5,] 78.24938 -1.271030 -0.2965047 -0.1193520 0.2139665 27.47846
## [6,] 78.64778 -1.082733 -0.2756962 -0.6835690 0.2114292 27.02233

# Block Gibbs Sampler Trace plots
par(mfrow=c(1,3))
plot(posterior2_burnin[,1], type="l", las=1, main="beta0")
plot(posterior2_burnin[,2], type="l", las=1, main="beta1")
plot(posterior2_burnin[,3], type="l", las=1, main="beta2")

```



```
plot(posterior2_burnin[,4], type="l", las=1, main="beta3")
plot(posterior2_burnin[,5], type="l", las=1, main="beta4")
plot(posterior2_burnin[,6], type="l", las=1, main="sig2")
```



```
# fit model in rstanarm
grades_lmer <- stan_lmer(average_score ~ part_time_job +
  absence_days + extracurricular_activities +
  weekly_self_study_hours + career_aspiration + (1|gender),
  data = clean_data)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000213 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 2.13 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [ 0%] (Warmup)
## Chain 1: Iteration:  200 / 2000 [10%] (Warmup)
## Chain 1: Iteration:  400 / 2000 [20%] (Warmup)
## Chain 1: Iteration:  600 / 2000 [30%] (Warmup)
## Chain 1: Iteration:  800 / 2000 [40%] (Warmup)
## Chain 1: Iteration: 1000 / 2000 [50%] (Warmup)
## Chain 1: Iteration: 1001 / 2000 [50%] (Sampling)
## Chain 1: Iteration: 1200 / 2000 [60%] (Sampling)
## Chain 1: Iteration: 1400 / 2000 [70%] (Sampling)
## Chain 1: Iteration: 1600 / 2000 [80%] (Sampling)
## Chain 1: Iteration: 1800 / 2000 [90%] (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%] (Sampling)
```

```

## Chain 1:
## Chain 1: Elapsed Time: 25.815 seconds (Warm-up)
## Chain 1: 15.024 seconds (Sampling)
## Chain 1: 40.839 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.000133 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1.33 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration: 1 / 2000 [ 0%] (Warmup)
## Chain 2: Iteration: 200 / 2000 [ 10%] (Warmup)
## Chain 2: Iteration: 400 / 2000 [ 20%] (Warmup)
## Chain 2: Iteration: 600 / 2000 [ 30%] (Warmup)
## Chain 2: Iteration: 800 / 2000 [ 40%] (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%] (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%] (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%] (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%] (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%] (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%] (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%] (Sampling)
## Chain 2:
## Chain 2: Elapsed Time: 22.691 seconds (Warm-up)
## Chain 2: 7.892 seconds (Sampling)
## Chain 2: 30.583 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0.000131 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 1.31 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration: 1 / 2000 [ 0%] (Warmup)
## Chain 3: Iteration: 200 / 2000 [ 10%] (Warmup)
## Chain 3: Iteration: 400 / 2000 [ 20%] (Warmup)
## Chain 3: Iteration: 600 / 2000 [ 30%] (Warmup)
## Chain 3: Iteration: 800 / 2000 [ 40%] (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%] (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%] (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%] (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%] (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%] (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%] (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%] (Sampling)
## Chain 3:
## Chain 3: Elapsed Time: 16.576 seconds (Warm-up)
## Chain 3: 24.745 seconds (Sampling)
## Chain 3: 41.321 seconds (Total)

```

```
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0.000139 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 1.39 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%] (Warmup)
## Chain 4: Iteration:   200 / 2000 [ 10%] (Warmup)
## Chain 4: Iteration:   400 / 2000 [ 20%] (Warmup)
## Chain 4: Iteration:   600 / 2000 [ 30%] (Warmup)
## Chain 4: Iteration:   800 / 2000 [ 40%] (Warmup)
## Chain 4: Iteration:  1000 / 2000 [ 50%] (Warmup)
## Chain 4: Iteration:  1001 / 2000 [ 50%] (Sampling)
## Chain 4: Iteration:  1200 / 2000 [ 60%] (Sampling)
## Chain 4: Iteration:  1400 / 2000 [ 70%] (Sampling)
## Chain 4: Iteration:  1600 / 2000 [ 80%] (Sampling)
## Chain 4: Iteration:  1800 / 2000 [ 90%] (Sampling)
## Chain 4: Iteration:  2000 / 2000 [100%] (Sampling)
## Chain 4:
## Chain 4: Elapsed Time: 23.37 seconds (Warm-up)
## Chain 4:                16.036 seconds (Sampling)
## Chain 4:                39.406 seconds (Total)
## Chain 4:
```

```
# show results
summary(grades_lmer, digits = 3)
```

```
##
## Model Info:
## function:      stan_lmer
## family:        gaussian [identity]
## formula:       average_score ~ part_time_job + absence_days + extracurricular_activities +
##               weekly_self_study_hours + career_aspiration + (1 | gender)
## algorithm:     sampling
## sample:        4000 (posterior sample size)
## priors:        see help('prior_summary')
## observations:  2000
## groups:        gender (2)
##
## Estimates:
```

	mean	sd	10%	50%	90%
## (Intercept)	76.155	1.354	74.839	76.126	77.534
## part_time_job	-0.138	0.322	-0.553	-0.142	0.275
## absence_days	0.052	0.048	-0.009	0.051	0.114
## extracurricular_activities	-0.092	0.262	-0.426	-0.086	0.245
## weekly_self_study_hours	0.134	0.016	0.113	0.133	0.155
## career_aspirationArtist	4.077	0.775	3.077	4.075	5.050
## career_aspirationBanker	2.689	0.557	1.980	2.691	3.399
## career_aspirationBusiness Owner	-2.229	0.625	-3.033	-2.235	-1.410
## career_aspirationConstruction Engineer	4.424	0.703	3.525	4.423	5.297
## career_aspirationDesigner	4.029	0.752	3.069	4.025	4.965
## career_aspirationDoctor	8.694	0.641	7.851	8.702	9.530

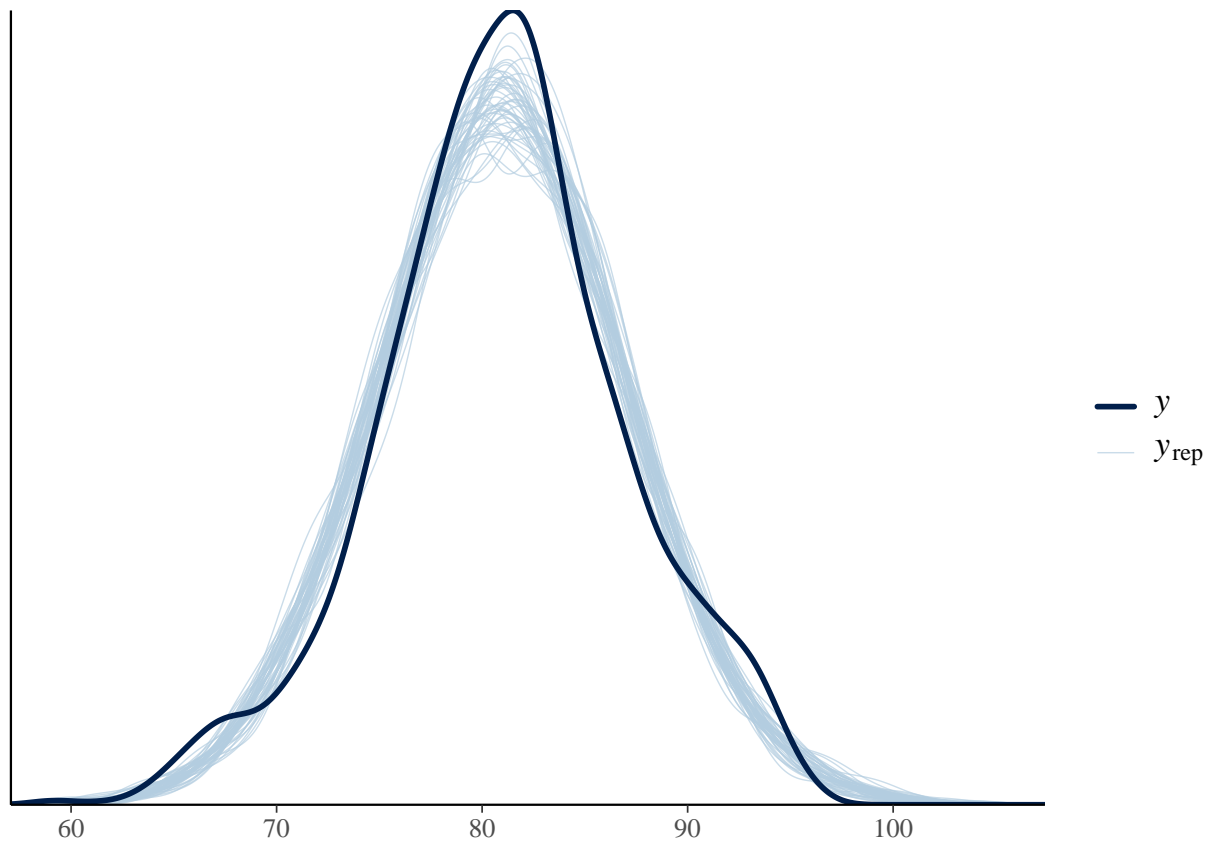
```

## career_aspirationGame Developer      5.147  0.807  4.118  5.141  6.200
## career_aspirationGovernment Officer   3.465  0.753  2.499  3.476  4.409
## career_aspirationLawyer                3.948  0.600  3.177  3.961  4.723
## career_aspirationReal Estate Developer 2.479  0.729  1.539  2.476  3.397
## career_aspirationScientist             5.900  0.886  4.773  5.920  7.039
## career_aspirationSoftware Engineer     2.732  0.495  2.081  2.739  3.352
## career_aspirationStock Investor        2.530  0.717  1.628  2.542  3.441
## career_aspirationTeacher               2.543  0.763  1.567  2.555  3.537
## career_aspirationUnknown               1.167  0.526  0.481  1.174  1.844
## career_aspirationWriter                4.587  0.945  3.357  4.579  5.812
## b[(Intercept) gender:0]                0.081  1.227 -0.981  0.082  1.138
## b[(Intercept) gender:1]               -0.205  1.233 -1.320 -0.109  0.809
## sigma                                  4.739  0.074  4.644  4.737  4.834
## Sigma[gender:(Intercept),(Intercept)] 4.357 11.847  0.018  0.570 11.062
##
## Fit Diagnostics:
##      mean    sd    10%    50%    90%
## mean_PPD 80.979  0.145 80.793 80.981 81.163
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##
##      mcse  Rhat  n_eff
## (Intercept)      0.069 1.014  386
## part_time_job      0.005 1.001 4064
## absence_days       0.001 1.000 3728
## extracurricular_activities 0.004 1.001 3857
## weekly_self_study_hours 0.000 1.002 1716
## career_aspirationArtist      0.021 1.002 1412
## career_aspirationBanker      0.018 1.003  968
## career_aspirationBusiness Owner 0.021 1.004  902
## career_aspirationConstruction Engineer 0.019 1.002 1436
## career_aspirationDesigner    0.020 1.003 1361
## career_aspirationDoctor      0.018 1.003 1302
## career_aspirationGame Developer 0.022 1.002 1308
## career_aspirationGovernment Officer 0.021 1.002 1293
## career_aspirationLawyer      0.018 1.004 1081
## career_aspirationReal Estate Developer 0.021 1.003 1218
## career_aspirationScientist    0.020 1.001 2007
## career_aspirationSoftware Engineer 0.017 1.005  812
## career_aspirationStock Investor 0.020 1.002 1254
## career_aspirationTeacher      0.019 1.003 1598
## career_aspirationUnknown      0.017 1.003  914
## career_aspirationWriter      0.021 1.001 2027
## b[(Intercept) gender:0]      0.064 1.012  372
## b[(Intercept) gender:1]      0.064 1.012  371
## sigma                      0.001 1.000 3147
## Sigma[gender:(Intercept),(Intercept)] 0.316 1.006 1404
## mean_PPD                    0.002 1.000 3881
## log-posterior                0.111 1.001 1125
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample

```



```
pp_check(grades_lmer)
```



```
set.seed(12344)
library(MCMCpack)
# Gibbs
data <- read.csv("student-scores-clean.csv")
y <- clean_data$average_score
x1 <- clean_data$part_time_job
x2 <- clean_data$absence_days
x3 <- clean_data$extracurricular_activities
x4 <- clean_data$weekly_self_study_hours
#design matrix
z <- model.matrix(~as.factor(career_aspiration)-1, data=data)

gibbs <- function(y,x1,x2,x3,x4,z,a,b,a_kappa,b_kappa,mu0,tau2,S) {

  n <- length(y)

  X <- cbind(1,x1,x2,x3,x4,z)

  p <- ncol(X)

  # hyperparameters
```

```

#fixed tau for the first 3 something
tau2 <- 100^2
beta <- rep(0,p)
kappa2 <- 1

a <- b <- 1 # complete
a_kappa <- b_kappa <- 1
mu0 <- 0

#draws
S <- 10

#place to store data
posterior_beta <- matrix(NA,S,p)
posterior_sig2 <- rep(NA,S)
posterior_kappa2 <- rep(NA, S)

#starting values
beta <- rep(0,p)
sig2 <- 1

# some useful quantities
# Do them here to avoid computing this S times
XX <- t(X)%*%X
Xy <- t(X)%*%y

for(s in 1:S){
  # update beta0
  prior_cov <- diag(c(rep(1/tau2,3),rep(1/kappa2,p-3)))
  v <- solve(XX/sig2 + prior_cov)
  m <- v %*% (Xy/sig2 + mu0*diag(prior_cov))
  beta <- m + t(chol(v)) %*% rnorm(p)
  #update sig2
  sig2 <- rinvgamma(1, a + n/2,
                    b + t(y-X%*%beta)%*%(y-X%*%beta)/2)
  #update kappa
  kappa2 <- rinvgamma(1, a_kappa + (p-3)/2,
                    b_kappa + 0.5* sum((beta[-c(1:3)])^2))

  #store results
  posterior_beta[s,] <- beta
  posterior_sig2[s] <- sig2
  posterior_kappa2[s] <- kappa2
}
return(cbind(posterior_beta, posterior_sig2, posterior_kappa2))
}
post_samples <- gibbs(y,x1,x2,x3,x4,z,a,b,a_kappa,b_kappa,mu0,tau2,S);

set.seed(222)
num_beta <- ncol(post_samples) - 2
post_samples_burnin <- post_samples[-c(1:5), ]

```

```

results <- data.frame(
  mean = colMeans(post_samples_burnin),
  sd = apply(post_samples_burnin, 2, sd),
  lower = apply(post_samples_burnin, 2, quantile, 0.025),
  upper = apply(post_samples_burnin, 2, quantile, 0.975)
)
row.names(results) <- c(paste0("beta_", 0:(num_beta - 1)), "kappa2", "sigma")

head(results)

```

```

##           mean           sd           lower           upper
## beta_0 78.96749223 0.299160497 78.54624841 79.28831058
## beta_1 -0.14123654 0.296238377 -0.47501084 0.21929513
## beta_2 0.06077318 0.025627039 0.02442782 0.08832263
## beta_3 -0.14743705 0.152208393 -0.29017099 0.02538010
## beta_4 0.14343141 0.008130231 0.13260699 0.15279240
## beta_5 -3.20906282 0.375928974 -3.69959162 -2.87588629

```

```

tail(results)

```

```

##           mean           sd           lower           upper
## beta_18 -0.74591712 1.0465019 -1.6912935 0.7876879
## beta_19 -0.04521435 0.7306792 -0.9197340 0.9409913
## beta_20 -1.86856519 0.5759868 -2.5344279 -1.2408696
## beta_21 1.95241149 1.2622726 0.0501249 3.0524233
## kappa2 22.05830132 0.3955220 21.5765232 22.5806877
## sigma 6.06573479 2.2087189 4.0559350 9.3204446

```