Final Project

Onyul Haque

4/25/2022

```r
library(tidyr)
library(dplyr)
library(reshape2)
library(car)
library(compute.es)
library(effects)
library(multcomp)
library(pastecs)
library(psych)
library(ggplot2)
library(gmodels)
library(mlogit)
library(Hmisc)
library(brms)
library(here)  # for easier management of file paths
library(rstan)
rstan_options(auto_write = TRUE)  # save compiled Stan object
library(brms)  # simplify fitting Stan GLM models
library(shinystan)  # graphical exploration
library(posterior)  # for summarizing draws
library(bayesplot)  # for plotting
library(modelsummary)  # table for brms
library(ProbBayes)
theme_set(theme_classic() +
    theme(panel.grid.major.y = element_line(color = "grey92")))
```

Research Question

Do people trust less when there is a timer that can cut them off from making a and is this uniquely due to the fact that the origin of risk is a person and not nature?

Bayesian Analysis - Give vs Do not Give

Model Equation:

$$Decision_i \sim Bin(N, \theta)$$

$$\theta = \beta_o + \beta_1 game_i + \beta_2 timer_i + \beta_3 game_i * timer_i$$

Prior:

Lottery No Timer

$$\beta_0 \sim N(0, 1)$$

Difference between Lottery No timer and Trust No Timer

$$\beta_1 \sim N(0, 1)$$

Difference between Lottery No Timer - Lottery Timer

$$\beta_2 \sim N(0, 1)$$

Difference between Lottery Timer - Trust Timer beyond the difference of Lottery no Timer and Trust No Timer

$$\beta_3 \sim N(0, 1)$$

```
m1 <-brm(
    # Y (vote) = beta0 + beta1 (growth)
    decision ~ game * timer,
    data = trustlong,
    # Normal distribution with identity link
    family = binomial,
    # Overwrite the default priors
    prior = c(
        # prior for beta0
        prior(normal(0,1), class = "Intercept"),
        # prior for beta1 game
        prior(normal(0,1), class = "b", coef = "gameTrust"),
        #prior Beta 2 Time
        prior(normal(0,1), class = "b", coef = "timerTimer"),
        #Beta 3 Interaction Term
        prior(normal(0,1), class = "b", coef = "gameTrust:timerTimer")
    ),
    sample_prior = TRUE,  # also sample the prior distributions
    iter = 4000,  # default is 4 chains, 2000 iterations each
    seed = 21
)
```

```
## Using the maximum response value as the number of trials.
```

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

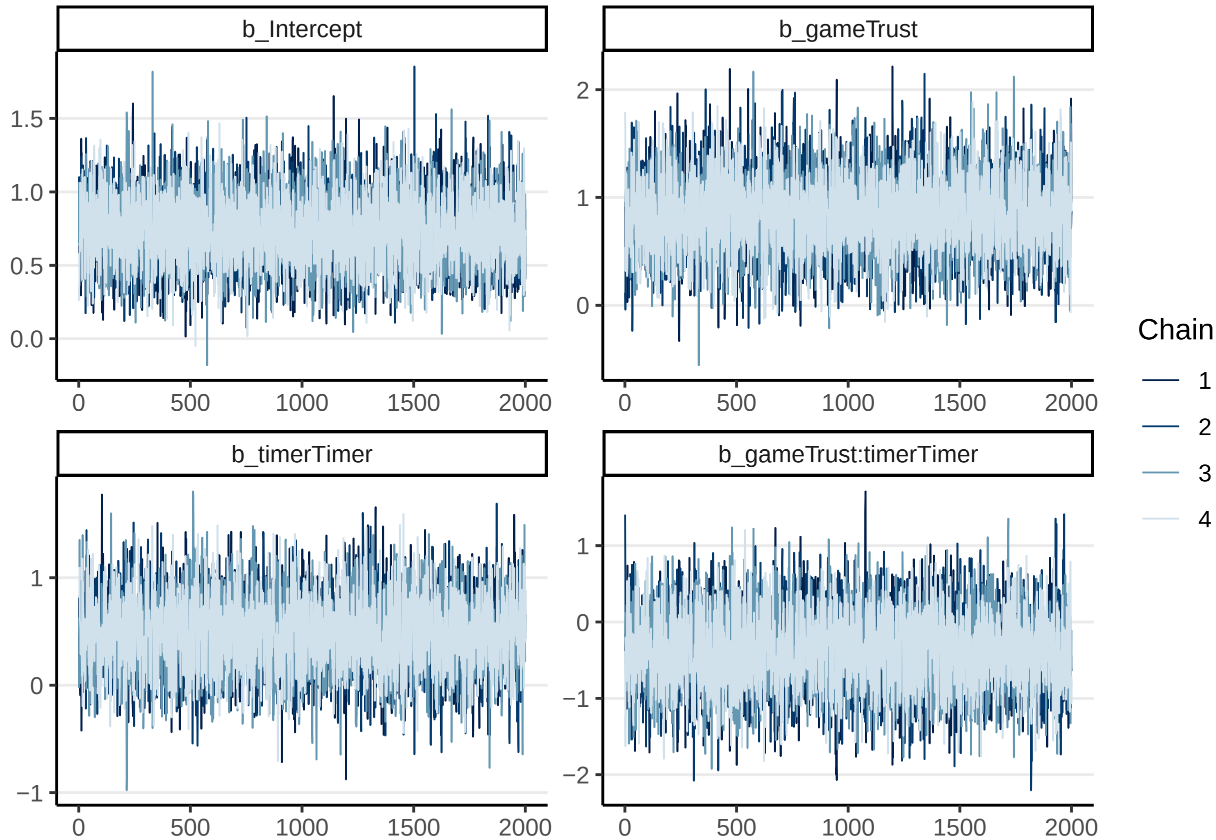
```
## Start sampling
```

```
##
## SAMPLING FOR MODEL 'd4855645b1a7dd8d4ba3d3de88379601' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
```

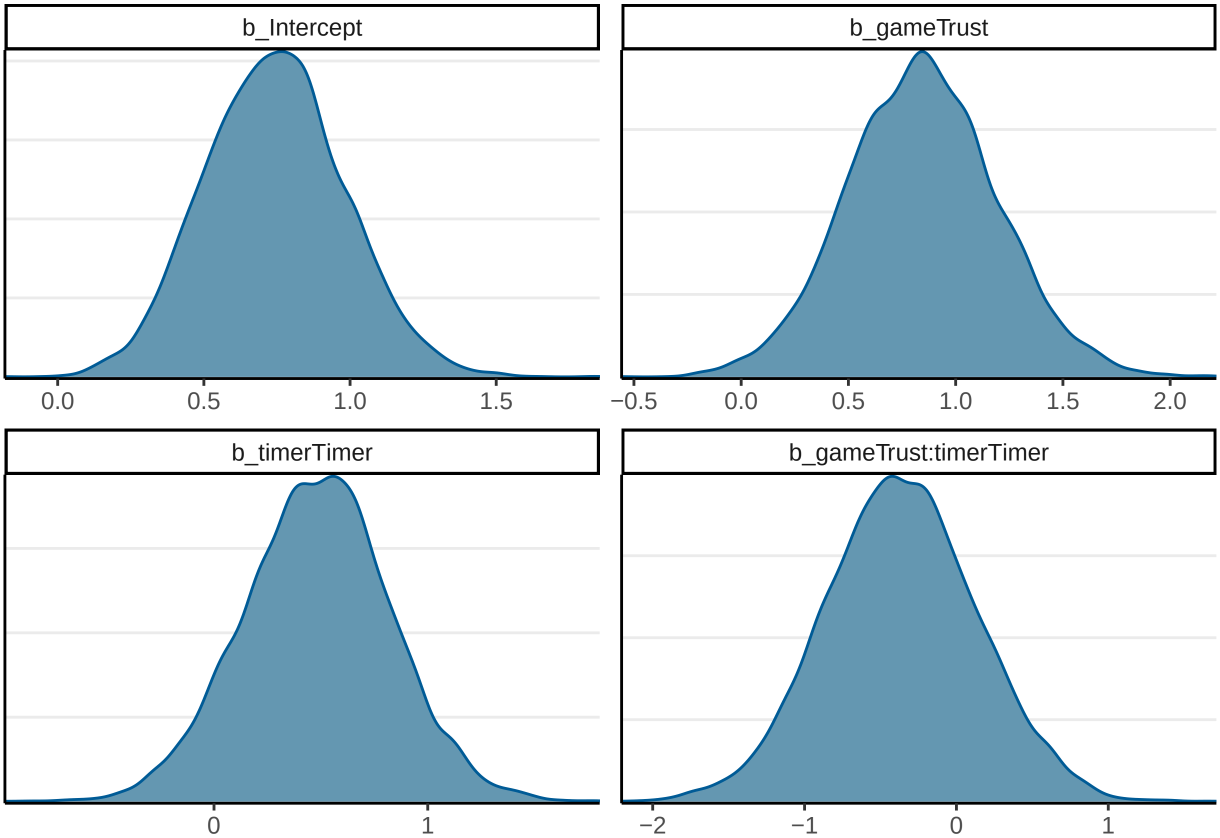
```
## Chain 1:
## Chain 1: Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 1: Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 1: Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 1: Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 1: Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 1: Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 1: Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 1: Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 1: Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 1: Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 1: Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 1: Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.57 seconds (Warm-up)
## Chain 1:                1.512 seconds (Sampling)
## Chain 1:                3.082 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'd4855645b1a7dd8d4ba3d3de88379601' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.001 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 2: Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 2: Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 2: Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 2: Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 2: Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 2: Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 2: Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 2: Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 2: Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 2: Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 2: Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 1.549 seconds (Warm-up)
## Chain 2:                1.824 seconds (Sampling)
## Chain 2:                3.373 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'd4855645b1a7dd8d4ba3d3de88379601' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 3: Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 3: Iteration:  800 / 4000 [ 20%]  (Warmup)
```

```
## Chain 3: Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 3: Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 3: Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 3: Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 3: Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 3: Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 3: Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 3: Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 3: Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 1.603 seconds (Warm-up)
## Chain 3:                1.659 seconds (Sampling)
## Chain 3:                3.262 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'd4855645b1a7dd8d4ba3d3de88379601' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 4000 [  0%]  (Warmup)
## Chain 4: Iteration:  400 / 4000 [ 10%]  (Warmup)
## Chain 4: Iteration:  800 / 4000 [ 20%]  (Warmup)
## Chain 4: Iteration: 1200 / 4000 [ 30%]  (Warmup)
## Chain 4: Iteration: 1600 / 4000 [ 40%]  (Warmup)
## Chain 4: Iteration: 2000 / 4000 [ 50%]  (Warmup)
## Chain 4: Iteration: 2001 / 4000 [ 50%]  (Sampling)
## Chain 4: Iteration: 2400 / 4000 [ 60%]  (Sampling)
## Chain 4: Iteration: 2800 / 4000 [ 70%]  (Sampling)
## Chain 4: Iteration: 3200 / 4000 [ 80%]  (Sampling)
## Chain 4: Iteration: 3600 / 4000 [ 90%]  (Sampling)
## Chain 4: Iteration: 4000 / 4000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 1.674 seconds (Warm-up)
## Chain 4:                1.884 seconds (Sampling)
## Chain 4:                3.558 seconds (Total)
## Chain 4:
```

```r
mcmc_trace(m1,
           pars = c("b_Intercept", "b_gameTrust",
                    "b_timerTimer", "b_gameTrust:timerTimer"))
```




```
mcmc_dens(m1,
          pars = c("b_Intercept", "b_gameTrust",
                   "b_timerTimer", "b_gameTrust:timerTimer"))
```




```
exp(fixef(m1))
```

```
##                        Estimate Est.Error      Q2.5    Q97.5
## Intercept             2.1009491  1.269133 1.3422306 3.386030
## gameTrust             2.3244824  1.418419 1.1742328 4.717003
## timerTimer            1.6196001  1.410928 0.8249838 3.169089
## gameTrust:timerTimer 0.6942237  1.650363 0.2625083 1.858538
```

I exponetiated the results of the model to obtain the odds ratio. The only the only paramters to obtain a confidence interval not including 1 are intercept and the game effect. Thus, the odds that somebody in the trust game would give $5 is 2.32 (95%c[1.17,4.17]) higher than somebody in the lottery game.