Bayesian Data Analysis Project

Analysis of Candy Crush First 15 Levels Data

anonymous

1 Introduction

The number of attempts to a level by players is an important metric for game designers to evaluate how balanced their game is. Since the levels are designed by teams with an expected attempt number in mind, it is only possible to observe if it matches with the player behaviour when the game goes live. Then, adjustments to level can be made by game designers if necessary. In our project, we worked on modeling the number of attempts by players for the first 15 levels of Candy Crush. Due to the nature of our data being right-skewed as shown below, we worked with log-normal model. Additionally, we modeled our problem hierarchically by introducing levels as groups since each can have different dynamics.
Section 2 will begin with description of data, followed by description of the models in section 3, priors will be given in section 4, model code will be presented in section 5. Then, section 6 will give MCMC details, followed by convergence diagnostics in section 7. Posterior predictive checks will be presented in section 8 and prior sensitivity analysis will be shown in section 9. In the following sections, discussion, conclusion and self reflection will be given.

2 Description of Data and Analysis Problem

The data consists of 16865 observations and has 5 columns: player_id, dt, level, num_attempts, num_success. Player ID is hashed and each rows describes at what date the player played which level. Also, how many attempts and wins that player has in the corresponding level is presented. The dates are only from the first 7 days of 2014 and first 15 levels of the game are included. During preprocessing, we combined the number of attempts and wins together if the player has played the same level on different days.

We found our dataset on Kaggle. There are some studies on the dataset, however they are focused on estimating the expected probability of wins and based on Frequentist approaches. To differ from existing studies, we decided to work on modeling to number of attempts for levels with Bayesian analysis.

```
library(bayesplot)
library(cmdstanr)
library(dplyr)
library(ggplot2)
library(ggdist)
library(posterior)
```

```r
library(brms)
library(tinytex)
options(brms.backend="cmdstanr")
options(brms.file_refit="on_change")
ggplot2::theme_set(theme_minimal(base_size = 14))
bayesplot::bayesplot_theme_set(theme_minimal(base_size = 14))
```

```r
candy_crush_data <- read.csv("./candy_crush.csv",
                header = TRUE, sep = ",")

head(candy_crush_data)
```

```
                          player_id         dt level num_attempts num_success
1 6dd5af4c7228fa353d505767143f5815 2014-01-04     4            3           1
2 c7ec97c39349ab7e4d39b4f74062ec13 2014-01-01     8            4           1
3 c7ec97c39349ab7e4d39b4f74062ec13 2014-01-05    12            6           0
4 a32c5e9700ed356dc8dd5bb3230c5227 2014-01-03    11            1           1
5 a32c5e9700ed356dc8dd5bb3230c5227 2014-01-07    15            6           0
6 b94d403ac4edf639442f93eeffdc7d92 2014-01-01     8            8           1
```

```r
#Combine the num_attemps and num_success if player plays the same level on different days
candy_crush_data <- candy_crush_data %>%
  group_by(player_id, level) %>%
  summarize(
    num_attempts = sum(num_attempts),
    num_success = sum(num_success)
  ) %>%
  ungroup()
```

```r
day_value_counts <- table(candy_crush_data$dt)
level_value_counts <- table(candy_crush_data$level)
print(day_value_counts)
```

```
< table of extent 0 >
```

```r
print(level_value_counts)
```

```
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15 
 676  680  673  705 1200  670  966 1839 1216  879  994 1111  687  844 2836 
```

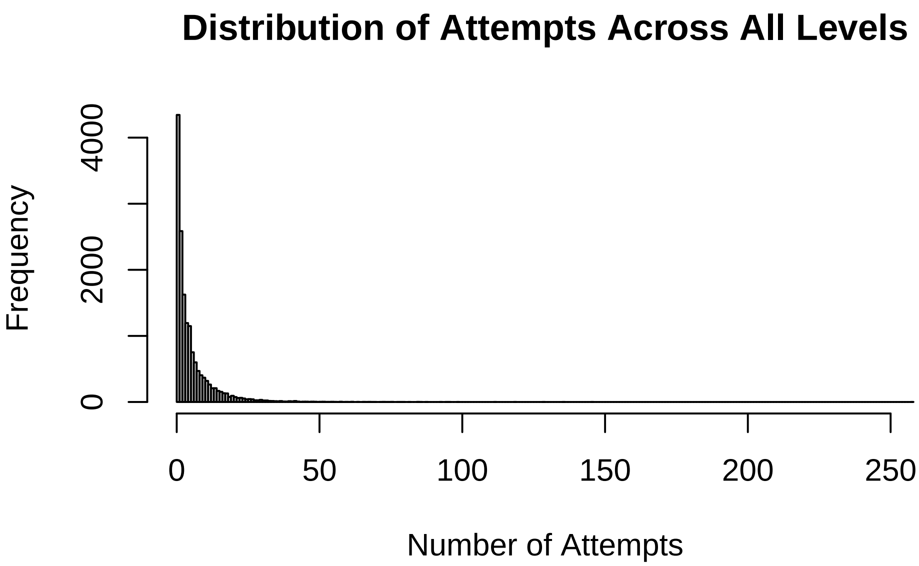
```r
print(min(candy_crush_data$num_attempts))
```

```
[1] 0
```

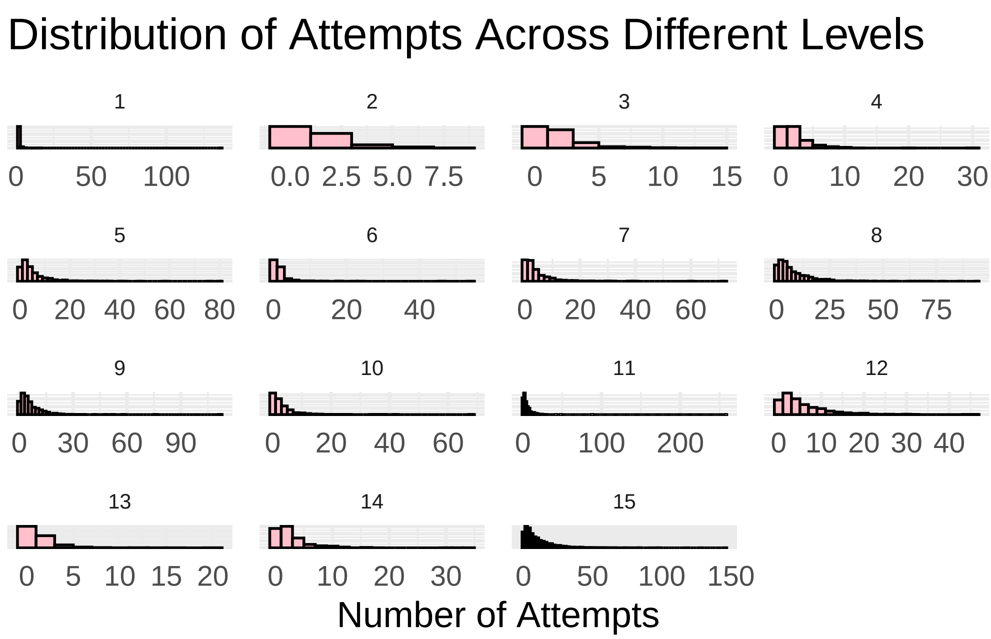
```r
print(max(candy_crush_data$num_attempts))
```

```
[1] 258
```

```r
hist(candy_crush_data$num_attempts, freq = TRUE, breaks = 200, xlab = "Number of Attempts"
     , main = "Distribution of Attempts Across All Levels")
```




```r
ggplot(candy_crush_data, aes(x = num_attempts)) +
  geom_histogram(binwidth = 2, fill = "pink", color = "black") +
  facet_wrap(~ level, scales = "free") +
  labs(x = "Number of Attempts", y = NULL, title = "Distribution of Attempts Across Different Levels
  theme(strip.text = element_text(size = 8),
        axis.text.y = element_blank(),
        axis.title.y = element_blank())
```



3 Description of Models

- Hurdle Log-normal Model: Due to the right-skewed structure of our data, we decided to use Log-normal observation model. Since our observations can also include 0 for response if the player hasn't played the level (num_attempts = 0), we were not able to use log-normal normal directly (it only allows positive values). Instead, we used the family "hurdle_lognormal" which is another variation of it that allows log-normal modeling when the data has 0 response. The level number and number of wins by the player are used as covariates.
- Hierarchical Model: In the hierarchical model, we assumed Hurdle Log-normal distribution for the number of attempts once again. Number of successes by the player is a covarite as well. However, the level number is used to define a hierarchical structure instead of a numerical value covariate. This way, the other covariates can vary between levels which can model the fluctuating difficulty of levels in a more convenient way.

4 Description of Priors

Since the data we use is game specific, we were unable to find a suitable prior reference through research. Then, because of having large number observations, we decided to use weakly informative priors. The priors are adjusted represent the change in log scale.

```
lognormal_model_priors <- c(
  prior(normal(0, log(5)), coef = "level"),
  prior(normal(0, log(5)), coef = "num_success")
)

hierarchical_model_priors <- c(
  prior(normal(0, log(5)), coef = "num_success")
)
```

5 Model Code

```
lognormal_model <- brm(num_attempts ~ 1 + level + num_success,
                       data = candy_crush_data,
                       family = hurdle_lognormal(),
                       prior = lognormal_model_priors,
                       chains = 4,
                       cores = 16)
```

```
hierarchical_model <- brm(num_attempts ~ 1 + num_success + (1 + num_success | level),
             data = candy_crush_data,
             family = hurdle_lognormal(),
             prior = hierarchical_model_priors,
             chains = 4,
             cores = 16)
```

6 MCMC

We used the default MCMC algorithm of brm function. By default, Stan uses the No-U-Turn Sampler (NUTS) which is a Hamiltonian Monte Carlo (HMC) method.

7 Convergence Diagnostics

- For both of the models, Rhat values are close to 1 which depicts convergence of chains.
- For both of the models, no divergent transitions are found which means all chains in the NUTS algorithm were able to converge.
- Effective Sample Size (ESS) values are once again very high in both models which shows parameters space is explored widely and parameters are estimated confidently.

```r
summary(lognormal_model)
```

```
 Family: hurdle_lognormal
  Links: mu = identity; sigma = identity; hu = identity
Formula: num_attempts ~ 1 + level + num_success
   Data: candy_crush_data (Number of observations: 15976)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Population-Level Effects:
            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept       0.62      0.02     0.59     0.66 1.00     5589     3406
level           0.07      0.00     0.06     0.07 1.00     5728     3224
num_success    -0.04      0.01    -0.05    -0.02 1.00     4680     3118

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.96      0.01     0.95     0.97 1.00     5297     2798
hu        0.01      0.00     0.00     0.01 1.00     4411     2964

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).
```

```r
np <- nuts_params(lognormal_model)
str(np)
```

```
'data.frame':	24000 obs. of  4 variables:
 $ Chain    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Iteration: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Parameter: Factor w/ 6 levels "accept_stat__",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ Value    : num  0.747 0.991 0.953 0.997 0.989 ...
```

```r
cat("\nNumber of divergent transitions for the log-normal model:",
    sum(subset(np, Parameter == "divergent__")$Value))
```

```
Number of divergent transitions for the log-normal model: 0
```

```r
summary(hierarchical_model)
```

```
 Family: hurdle_lognormal
  Links: mu = identity; sigma = identity; hu = identity
Formula: num_attempts ~ 1 + num_success + (1 + num_success | level)
   Data: candy_crush_data (Number of observations: 15976)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Group-Level Effects:
~level (Number of levels: 15)
                           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
sd(Intercept)                  0.64      0.13     0.44     0.95 1.00      956
sd(num_success)                0.19      0.04     0.12     0.28 1.00     1292
cor(Intercept,num_success)    -0.62      0.18    -0.87    -0.20 1.00     1969
                           Tail_ESS
sd(Intercept)                  1260
sd(num_success)                2050
cor(Intercept,num_success)     2228

Population-Level Effects:
            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept       0.97      0.17     0.62     1.31 1.00      872     1201
num_success     0.03      0.05    -0.07     0.13 1.00     1445     1774

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.87      0.00     0.86     0.88 1.00     4728     2938
hu        0.01      0.00     0.00     0.01 1.00     4202     2408

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).
```

```r
np <- nuts_params(hierarchical_model)
str(np)
```

```
'data.frame':   24000 obs. of  4 variables:
 $ Chain    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Iteration: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Parameter: Factor w/ 6 levels "accept_stat__",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ Value    : num  0.955 0.943 0.988 0.924 0.804 ...
```

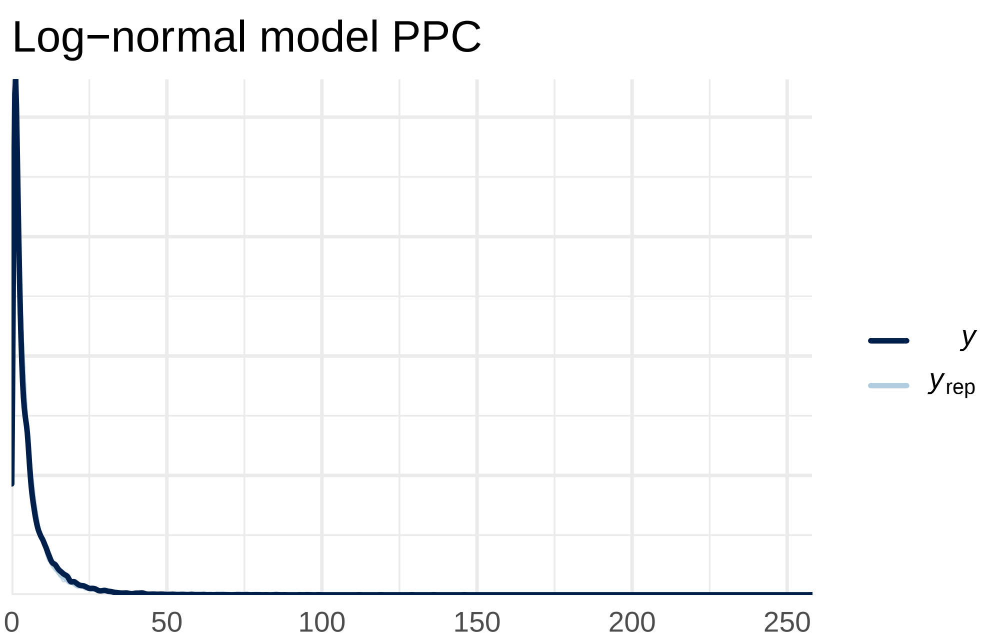
```
cat("\nNumber of divergent transitions for the hierarchical model:"
    , sum(subset(np, Parameter == "divergent__")$Value))
```

```
Number of divergent transitions for the hierarchical model: 0
```

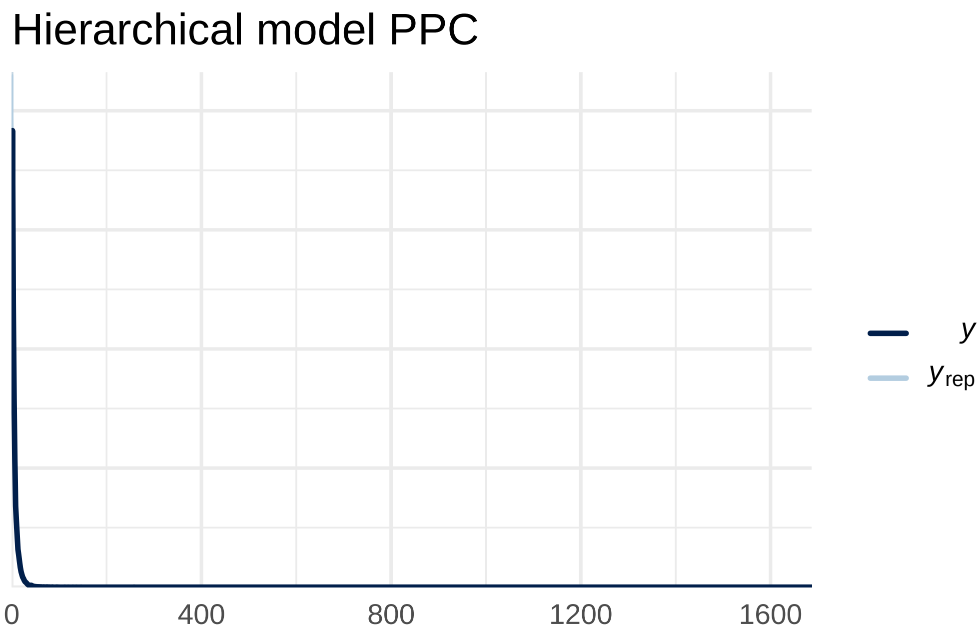
8 Posterior Predictive Checks

With ppc_check for both models, it is a bit hard to understand since the observation is right-skewed and ranges are very large. Therefore, we also included plots for the leftmost part of the ppc_check plot between attemps of 0-75. Both models looks like a convenient fit. However, to understand which one has better predictive performance, further analysis must be made.

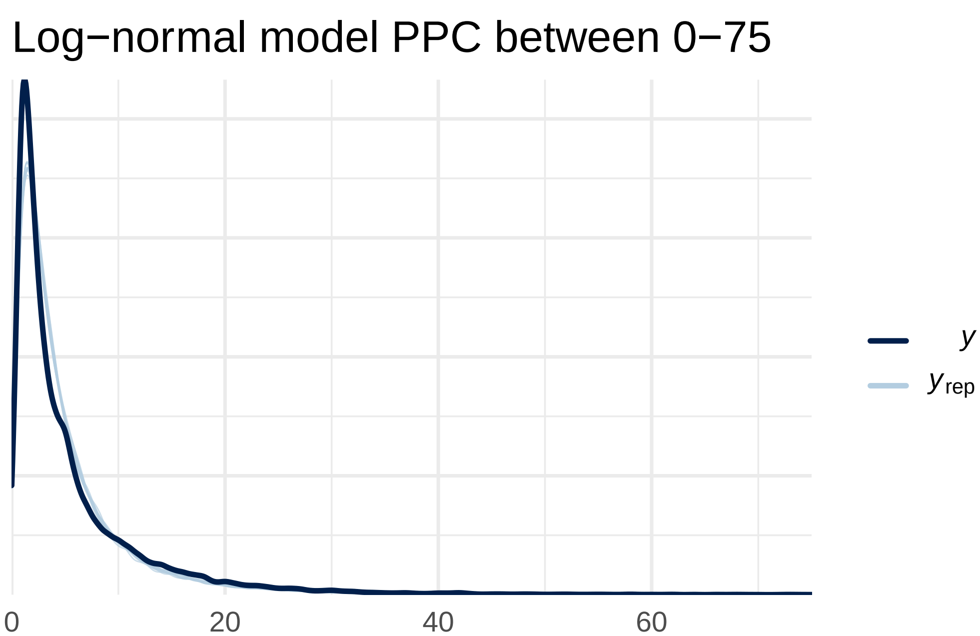
```
ppc_lognormal_model <- pp_check(lognormal_model)
plot(ppc_lognormal_model + labs(title = "Log-normal model PPC"))
```




```
ppc_hierarchical_model <- pp_check(hierarchical_model)
plot(ppc_hierarchical_model + labs(title = "Hierarchical model PPC"))
```

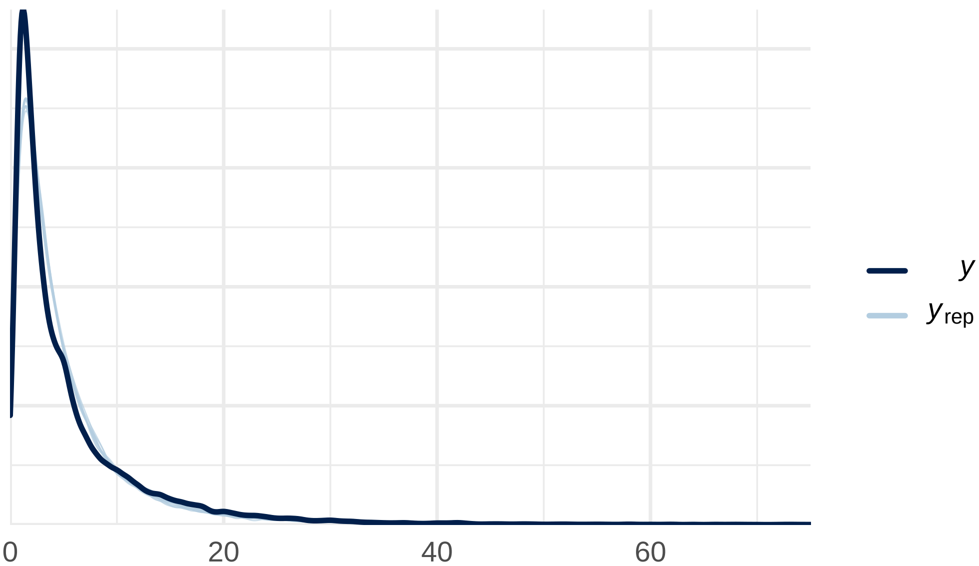



```
ppc_lognormal_model <- pp_check(lognormal_model)
plot(ppc_lognormal_model + xlim(0, 75) + labs(title = "Log-normal model PPC between 0-75"))
```




```
ppc_hierarchical_model <- pp_check(hierarchical_model)
plot(ppc_hierarchical_model + xlim(0, 75) + labs(title = "Hierarchical model PPC between 0-75"))
```

Hierarchical model PPC between 0−75



9 Prior Sensitivity Analysis

We evaluated priors with a different mean and larger standard deviation in log scale. However, in both of the models estimates did not change significantly. We believe this is mostly due to having a high number of observations.

```
priors <- c(
  prior(normal(15, log(30)), coef = "level"),
  prior(normal(15, log(30)), coef = "num_success")
)

alternate_sd_priors_lognormal_model <- brm(num_attempts ~ 1 + level + num_success,
                       data = candy_crush_data,
                       family = hurdle_lognormal(),
                       prior = priors,
                       chains = 4,
                       cores = 16)
```

```
summary(alternate_sd_priors_lognormal_model)
```

```
 Family: hurdle_lognormal
  Links: mu = identity; sigma = identity; hu = identity
Formula: num_attempts ~ 1 + level + num_success
   Data: candy_crush_data (Number of observations: 15976)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Population-Level Effects:
            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept       0.62      0.02     0.59     0.66 1.00     5088     3411
level           0.07      0.00     0.06     0.07 1.00     5238     2974
num_success    -0.04      0.01    -0.05    -0.02 1.00     5144     3352
```

```
Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.96      0.01     0.95     0.97 1.00     5005     3163
hu        0.01      0.00     0.00     0.01 1.00     4121     2480

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).
```

```r
priors <- c(
  prior(normal(15, log(30)), coef = "num_success")
)

alternate_sd_hierarchical_model <- brm(num_attempts ~ 1 + num_success + (1 + num_success | level),
             data = candy_crush_data,
             family = hurdle_lognormal(),
             prior = priors,
             chains = 4,
             cores = 16)
```

```r
summary(alternate_sd_hierarchical_model)
```

```
 Family: hurdle_lognormal
  Links: mu = identity; sigma = identity; hu = identity
Formula: num_attempts ~ 1 + num_success + (1 + num_success | level)
   Data: candy_crush_data (Number of observations: 15976)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Group-Level Effects:
~level (Number of levels: 15)
                           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
sd(Intercept)                  0.64      0.14     0.43     0.96 1.01      827
sd(num_success)                0.19      0.04     0.12     0.28 1.01     1069
cor(Intercept,num_success)    -0.61      0.18    -0.88    -0.17 1.00     1553
                           Tail_ESS
sd(Intercept)                  1043
sd(num_success)                1793
cor(Intercept,num_success)     1892

Population-Level Effects:
            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept       0.96      0.16     0.61     1.27 1.00      943     1469
num_success     0.03      0.05    -0.07     0.13 1.00     1051     1384

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.87      0.00     0.86     0.88 1.00     5099     2917
hu        0.01      0.00     0.00     0.01 1.00     3743     2593

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).
```

10 Model Comparison

Since the hierarchical model has better predictive performance, it is taken as a reference and shown with 0 values of elpd_diff and se_diff. Log-normal model has an elpd_diff value of -1650.7 which is considerably lower, and standard error of 72. It can be safely concluded that, hierarchical models has better predictive performance.

```
loo_lognormal_model <- loo(lognormal_model, save_psis = TRUE)
loo_hierarchical_model <- loo(hierarchical_model, save_psis = TRUE)
```

```
loo_compare(loo_lognormal_model, loo_hierarchical_model)
```

```
                   elpd_diff se_diff
hierarchical_model     0.0       0.0
lognormal_model    -1650.3      71.4
```

11 Discussion of Problems and Potential Improvements

In general, we did not face any issues with convergence and modeling. We believe this was due to our structure of our data and having many observations. However, there are thing we have in mind that can be used for further improvement. For example, players can also be used to define hierarchical groups since each player has a different skillset which can change the number of attempts greatly. We have tried that at first, but model fitting takes way too long since there too many players which results with too many groups. Maybe categorizing players' skills (novice, medium, expert) and then using those as hierarchical groups could help. That could be done if there were more data about player's characteristics and demographics.

12 Conclusion

In a nutshell, we have evaluated the difference of treating the level number as a numeric covariate versus using it to define groups to estimate the number of attempts. Our hierarchical model clearly showed better performance than the former model. We learned that hierarchical modeling can help with real world cases where the groups' characteristics can change significantly such as game levels, assignments and diet differences.

13 Self Reflection

As a group, we learned to apply the processes we did during assignments in templates to real world data. Moreover, we grasped the idea of stating our problem on the observed data and come up with suitable modeling ideas accordingly.