# African and Asian Conflict Prediction: Final Individual Choice Project

*Allison Patch*

*6/17/2019*

## Introduction

Perhaps one of the biggest hurdles to conflict management processes is not knowing how destructive conflicts are likely to be to human life. This project explores the applicability of machine learning techniques to understanding the likelihood of accurate contflict fatality rate prediction using data from Asia and Africa found in The Armed Conflict Location & Event Data Project (ACLED). Using the system of parameter building outlined in the EdX Machine Learning course, through this project I will explore the particular variables included in the ACLED dataset that produce the lowest Root Mean Squared Error (RMSE) for my model. RMSE is important because it is a measurement of the average error our algorithm produces, which is helpful because it gives us an idea of how reliable our algorithm is.

The process of algorithm selection I used in my analysis was based on Matrix factorization. I chose this method because it allowed me to evaluate the success of iterative models with new parameter vectors included in each model iteration. This allows for easy comparison between models and shows how each new parameter influences the RMSE. Additionally, I used a regulariation process to see if I could improve upon my RMSE, but this did not help to lower it.

However, despite these efforts to model a reliable prediction algorithm for conflict, I was unable to get an RMSE below about 3.33. These dismal results are likely rooted in the limited data I had available as well as the relative infrequency of conflict events more generally. The dataset I used only included about 120,000 unique events across 21 years. This limits the

## Methods

The first step to the process was downloading the data, cleaning it, and splitting it into a training (data name: train) and test set (data name: test). For my project I adapting some code from a kaggle kernal (https://www.kaggle.com/ambarish/eda-african-conflicts-with-leaflets/code). Following this, I conducted a bit of data exploration guided by my expectations of variables that would be important to predicting the number of fatalities. Finally, I developed several iterative models using the training set and measuring the quality of the model by calculating the RMSE using the test set for each model. The first two sections of code can be found together below. The remainder of the report will discuss the results of my recommendation system algorithm exploration.

```
#####################################################################################################
# Create train and test set #########################################################################
#####################################################################################################

# Note: this process could take a couple of minutes


if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

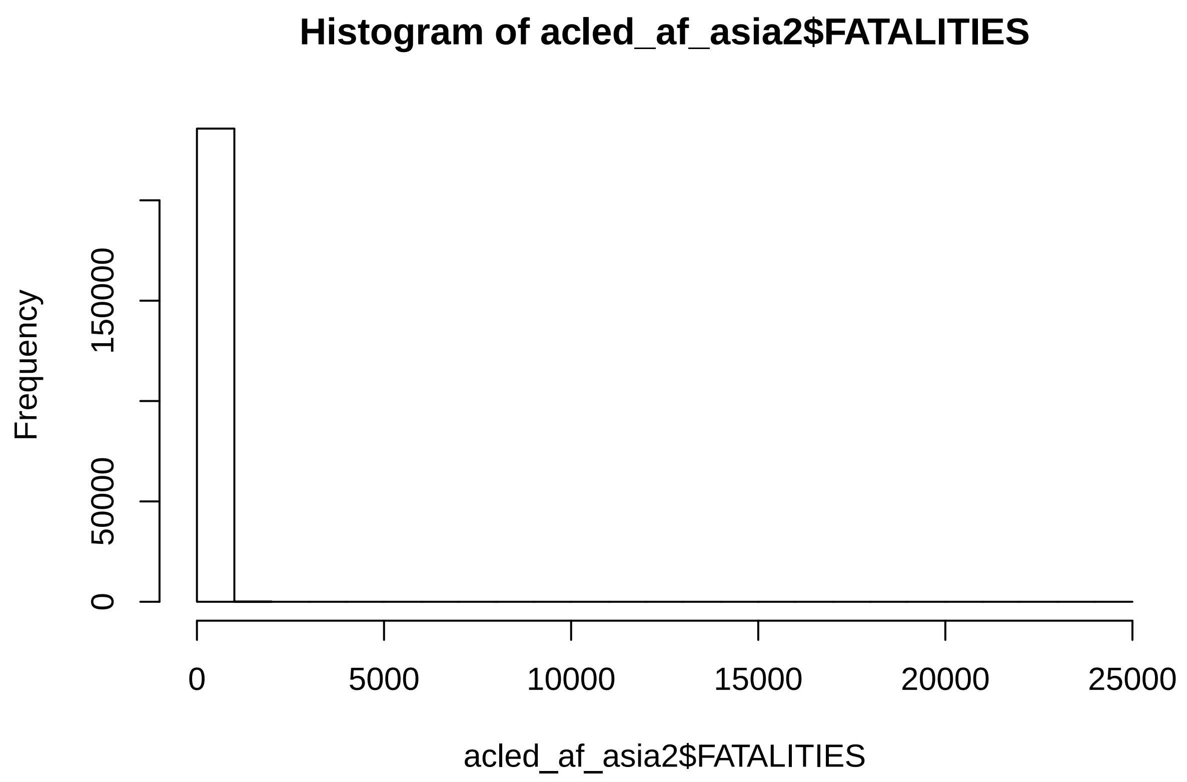
```
## Loading required package: tidyverse

## -- Attaching packages --------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.1.1       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0

## -- Conflicts ------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

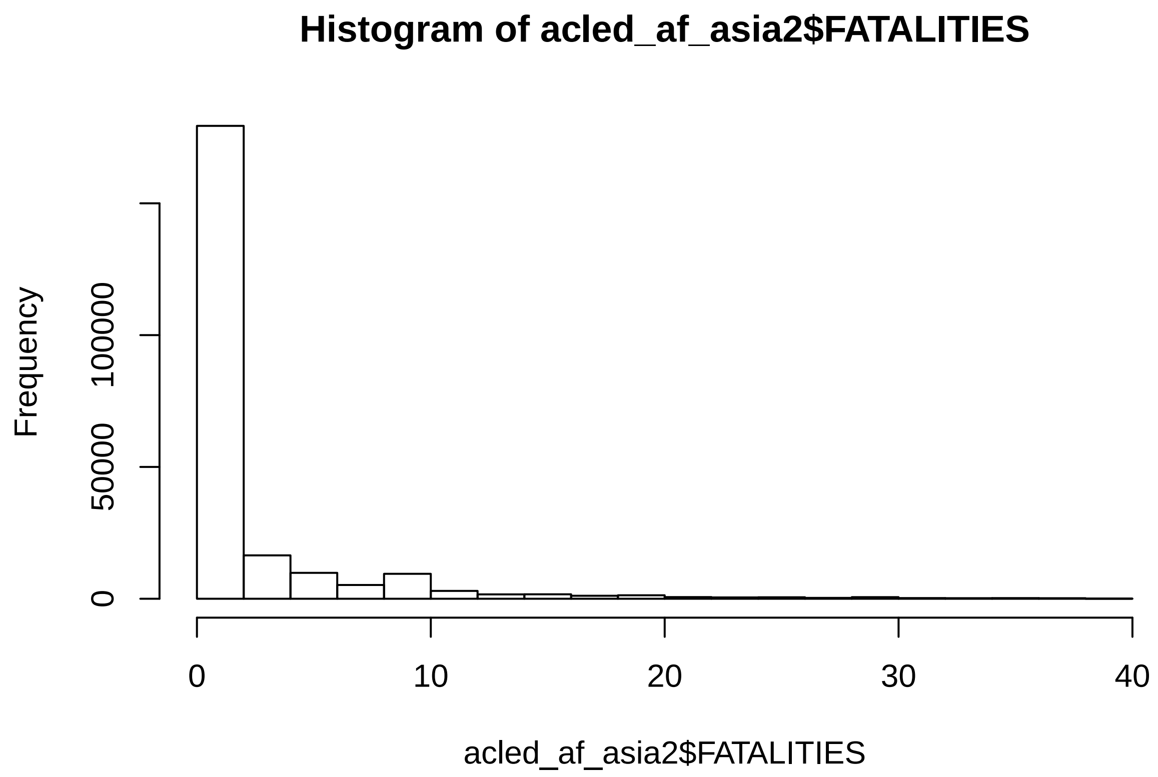
```
## Loading required package: caret

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(readr)
library(knitr)
library(caret)
library(tidyverse)

##The data used for this project can be found in my github through the following link

# https://github.com/akpatch/edx_fp_individual.git


##If you plan to run the code, please be sure to update the path for the data as it is not linked direc

# Note I used code from a kaggle kernal when cleaning the data
acled_af_asia<-read_csv("C:/Users/APATCH/OneDrive - UMUC/Data Science Course/African Conflict/asia_afri
  .default = col_character(),
  FATALITIES = col_integer(),
  GEO_PRECISION = col_integer(),
  GWNO = col_integer(),
  INTER1 = col_integer(),
  INTER2 = col_integer(),
  INTERACTION = col_integer(),
  LATITUDE = col_character(),
  LONGITUDE = col_character(),
  TIME_PRECISION = col_integer(),
  YEAR = col_integer()
))




acled_af_asia$LATITUDE[!grepl("^[0-9.]+$", acled_af_asia$LATITUDE)] <- NA
acled_af_asia$LONGITUDE[!grepl("^[0-9.]+$", acled_af_asia$LONGITUDE)] <- NA

acled_af_asia$LATITUDE = as.numeric(as.character(acled_af_asia$LATITUDE))
acled_af_asia$LONGITUDE = as.numeric(as.character(acled_af_asia$LONGITUDE))


acled_af_asia$EVENT_TYPE = trimws(acled_af_asia$EVENT_TYPE)
```

```
acled_af_asia$EVENT_TYPE = gsub("Strategic development","Strategic Development",acled_af_asia$EVENT_TYPE
acled_af_asia$EVENT_TYPE = gsub("Violence against civilians","Violence Against Civilians",acled_af_asia$
acled_af_asia$EVENT_TYPE = gsub("Battle-no change of territory","Battle-No change of territory",acled_af
acled_af_asia$EVENT_TYPE = gsub("Remote violence","Remote Violence",acled_af_asia$EVENT_TYPE)
acled_af_asia$EVENT_TYPE = gsub("Strategic development","Strategic Development",acled_af_asia$EVENT_TYPE
acled_af_asia$EVENT_TYPE = gsub("Violence against civilians","Violence Against Civilians",acled_af_asia$
acled_af_asia$EVENT_TYPE = gsub("Battle-no change of territory","Battle-No change of territory",acled_af
acled_af_asia$EVENT_TYPE = gsub("Remote violence","Remote Violence",acled_af_asia$EVENT_TYPE)


###We don't need all of the columns included in the dataset, let's keep just a few of interest
acled_af_asia2<-select(acled_af_asia, ACTOR1, ACTOR2, ADMIN1, ADMIN2, COUNTRY, EVENT_DATE, EVENT_TYPE, F
acled_af_asia2<-na.omit(acled_af_asia2)


###About how many fatalities are there across all of these events?
hist(acled_af_asia2$FATALITIES)
```



```
###We can see that there are very few events with more than 40 fatalities, so we will exclude these outl
acled_af_asia2<- subset(acled_af_asia2, acled_af_asia2$FATALITIES<40)

hist(acled_af_asia2$FATALITIES)
```



```
###Finally we split the data into our test and training models
set.seed(1)
test_index <- createDataPartition(y = acled_af_asia2$FATALITIES, times = 1, p = 0.1, list = FALSE)
train<- acled_af_asia2[-test_index,]
test <- acled_af_asia2[test_index,]
```

```
##########################################################################################################
## Exploring Data #########################################################################################
##########################################################################################################

#getting row and column length
nrow(train)
```

```
## [1] 209431
```

```
ncol(train)
```

```
## [1] 10
```

```
#frequency table of fatalities
table(train$FATALITIES)
```

```
##
##      0      1      2      3      4      5      6      7      8      9
## 113065  33855  14525   8819   5955   5440   3414   2664   2057   1447
##     10     11     12     13     14     15     16     17     18     19
##   7085   1133   1528    797    684    959    572    530    502    319
##     20     21     22     23     24     25     26     27     28     29
##    859    264    296    252    202    338    172    159    131    107
##     30     31     32     33     34     35     36     37     38     39
##    443    121    101    102     93    125     96     98     82     40
```

```
#number of unique countries
length(unique(train$COUNTRY))
```

```
## [1] 74
```

```
#number of unique locations
length(unique(train$LOCATION))
```

```
## [1] 29091
```

```
#number of unique Primary Actors
length(unique(train$ACTOR1))
```

```
## [1] 4788
```

```
#number of unique Second Actors
length(unique(train$ACTOR2))
```

```
## [1] 4242
```

```
#number of unique event types
table(train$EVENT_TYPE)
```

```
##
##       Battle-Government regains territory
##                                      5490
##             Battle-No change of territory
##                                     73594
## Battle-Non-state actor overtakes territory
##                                      3381
##           Headquarters or base established
##                                        39
##          Non-violent transfer of territory
##                                       662
##                            Remote Violence
##                                     42137
##                            Riots/Protests
##                                     23370
##                      Strategic Development
##                                      7515
##                 Violence against Civilians
##                                         3
##                 Violence Against Civilians
##                                     53240
```

```
#Country with highest fatality rates
highest<-train%>% group_by(COUNTRY)%>% summarize(count=length(FATALITIES)) %>% arrange(desc(count))
highest
```

```
## # A tibble: 74 x 2
##    COUNTRY                        count
##    <chr>                          <int>
##  1 Syria                          23063
##  2 Somalia                        20438
##  3 Yemen                          12148
##  4 Iraq                           10929
##  5 Afghanistan                   10553
##  6 Pakistan                      10517
##  7 Democratic Republic of Congo  8918
##  8 Sudan                         8855
##  9 India                         8079
## 10 Nigeria                       7451
## # ... with 64 more rows
```

### Methods–Algorithm Exploration

In thinking of how to create the most accurate prediction for fatality rates of future conflicts, we first have to reflect on the factors that go into the fatalty rates. First, there is a general average fatality count that conflicts have. Second, we can think of how the rate of fatalities in different countries compare to the average—are they better, worse, the same? Third, we need to account for the type of conflict event that we are discussing because some types of conflicts are likely to see more fatalities occur than others. Fourth, we need to account for the location of the conflict. Similar rates of fatalities are likely to occur in particular locations because people are conditioned to expect a similar rate of violence in confrontations. Finally, we need to account for the primary and secondary actors involved in the conflict. Particular actors will likely engage in similar levels of violence across all events that they participate in.

## Results

The following sections of this report exmaine each of these factors in turn, building to the final algortihm for fatality prediction that I produce.

To begin, I created a fuction which would calculate the RMSE for each of the models.

```r
###Since the outcome of interest is RMSE, create a function to calculate RMSE
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
}
```

The overall average fatality count is the staring point to the analysis for my algorithms. The overall average was:

```
## [1] 2.315908
```

Using this overall average, the first model I ran used a naive bayes approach simply looking at the overall average fatalities given and how well that could predict what fatality rates in the test set would be.

```r
###calculate RMSE for the overall average
overall_average_rmse <- RMSE(test$FATALITIES, mu_hat)
overall_average_rmse
```

```
## [1] 4.814038
```

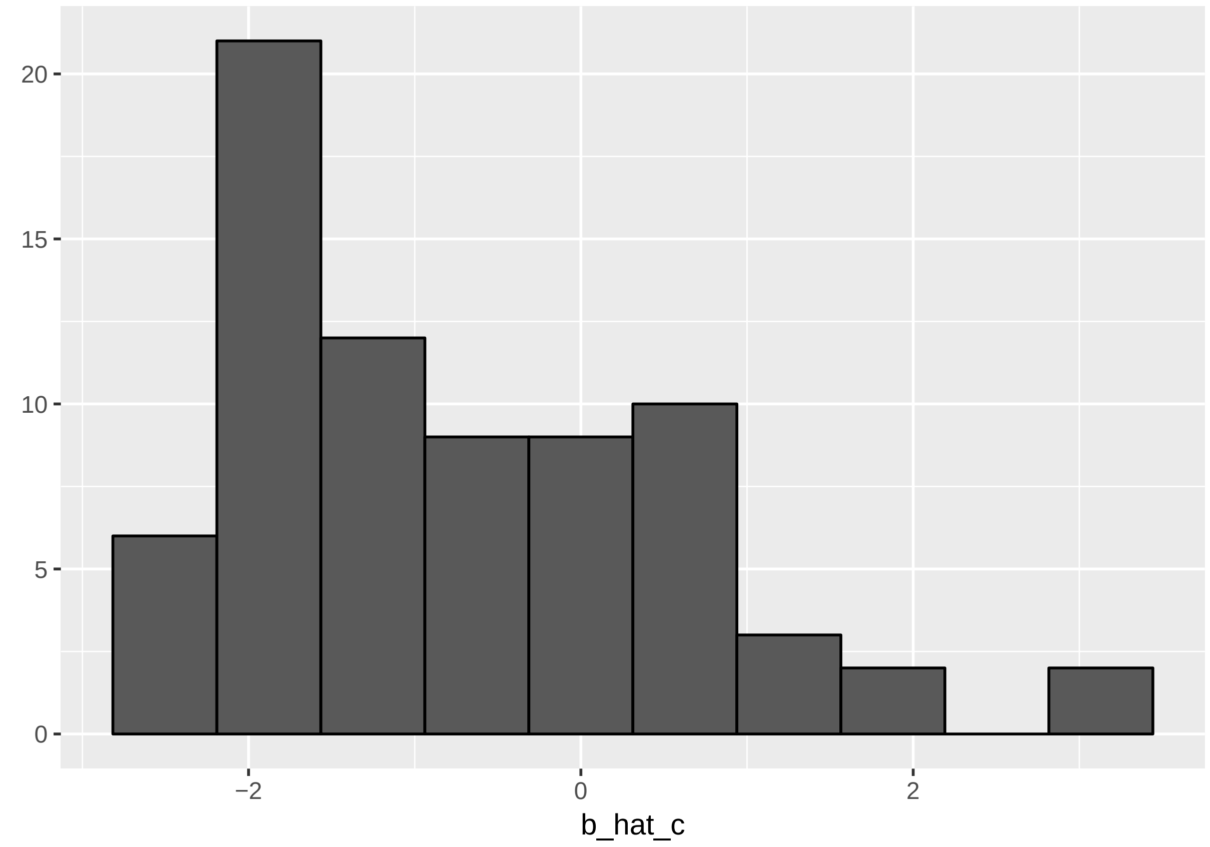
```r
###Here I am creating a tibble to hold the results of the models to compare RMSE
rmse_results <- tibble(method = "Overall average", RMSE = overall_average_rmse)
```

| method | RMSE |
|---|---:|
| Overall average | 4.814038 |

The RMSE result for the Overall Average Model is almost 5, which is not very good. To improve this we will add another parameter to the model: Country Effects. This will account for the difference in each country's average fatality count from the overall average.

```
## b_hat_c is the average of the diffence between the fatality rate within a specific country minus the
country_avgs <- train %>%
  group_by(COUNTRY) %>%
  summarize(b_hat_c = mean(FATALITIES - mu_hat))
```

A histogram of this difference shows a slightly negatively skewed distribution.



I then ran a second model including country effects.

```
##Now we use the validation data to check the RSME for the mode

#first we have to calculate b_hat_c for the validation set and calculate the predicted ratings for the
predicted_fatalities <- mu_hat + test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  .$b_hat_c

#next we test to see the RMSE
model_country_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Country Effect Model",
                                     RMSE = model_country_rmse ))
```

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

| method | RMSE |
|---|---|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |

The RMSE result for the Country Effects is still not very low so I added another parameter to the model: Event Type Effects. The summary of average fatalities across event types shows that there is variation in the average fatality counts over the different types of conflict.

```
## b_hat_et is the average fatality observed for each type of event recorded in the data
train %>%
  group_by(EVENT_TYPE) %>%
  summarize(b_hat_et= mean(FATALITIES))
```

```
## # A tibble: 10 x 2
##    EVENT_TYPE                               b_hat_et
##    <chr>                                       <dbl>
##  1 Battle-Government regains territory       3.43
##  2 Battle-No change of territory             3.37
##  3 Battle-Non-state actor overtakes territory  2.69
##  4 Headquarters or base established          0.103
##  5 Non-violent transfer of territory         0.00604
##  6 Remote Violence                           2.54
##  7 Riots/Protests                            0.259
##  8 Strategic Development                     0.0274
##  9 Violence against Civilians                0
## 10 Violence Against Civilians                1.81
```

I then ran a third model including Country Effects and Event Effects.

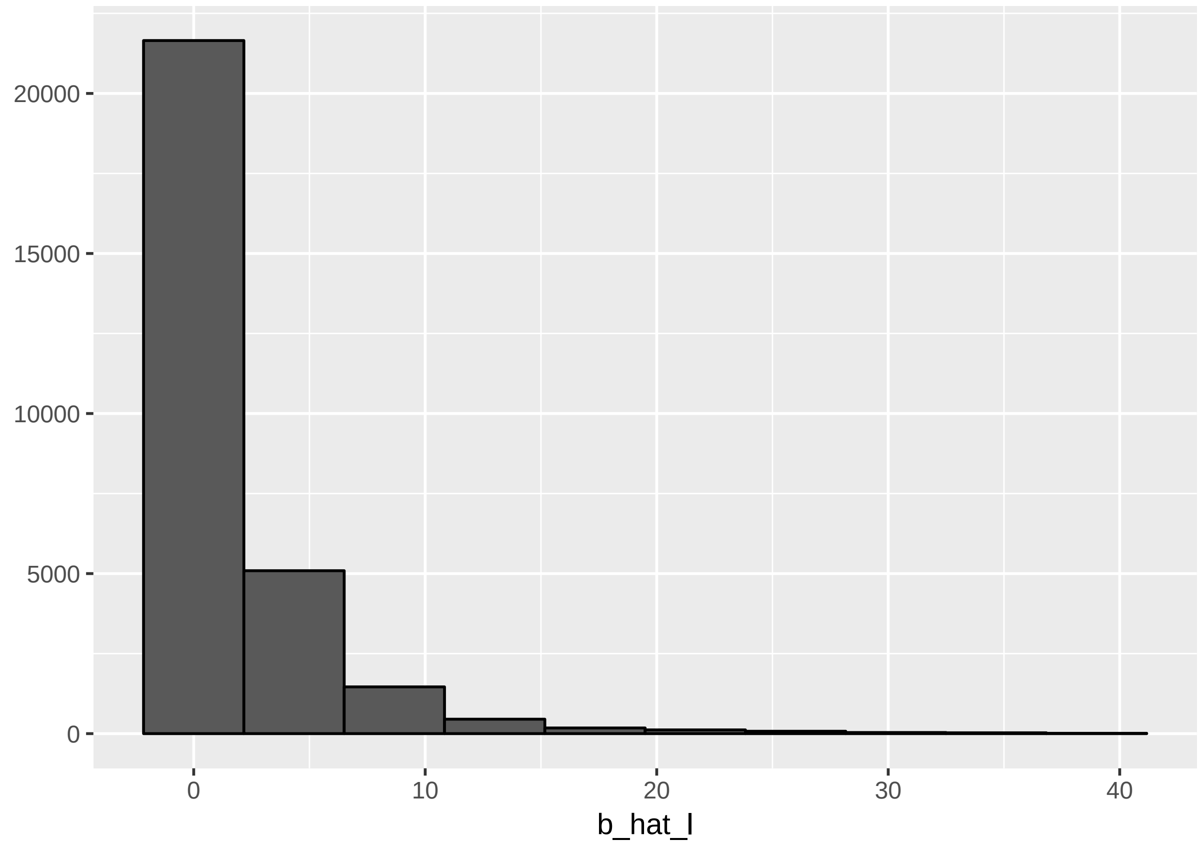
```
#first we have to create user averages for the test set so that we can make our predictions
event_avgs <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  group_by(EVENT_TYPE) %>%
  summarize(b_hat_et= mean(FATALITIES - mu_hat - b_hat_c))

## Then we have to calculate the predicted ratings for the test model
predicted_fatalities <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(event_avgs, by='EVENT_TYPE') %>%
  mutate(pred = mu_hat + b_hat_c + b_hat_et) %>%
  .$pred

#next we test to see the RMSE
model_eventtype_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Country + Event Type Effects Model",
                                     RMSE = model_eventtype_rmse ))
```

| method | RMSE |
|---|---|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |
| Country + Event Type Effects Model | 4.576674 |

The RMSE result for the Country and Events Effects is still not very low so I added another parameter to the model: Location Effects. The summary of average fatalities across event types shows that there is variation in the average fatality counts over the different types of conflict.

```
## b_hat_l is the average factality rate for each specific location observed in the dataset
location_avgs<-train %>%
  group_by(LOCATION) %>%
  summarize(b_hat_l = mean(FATALITIES))
```

A histogram of this difference shows a slightly positively skewed distribution.



I then ran a third model including Country, Event, and Location Effects.

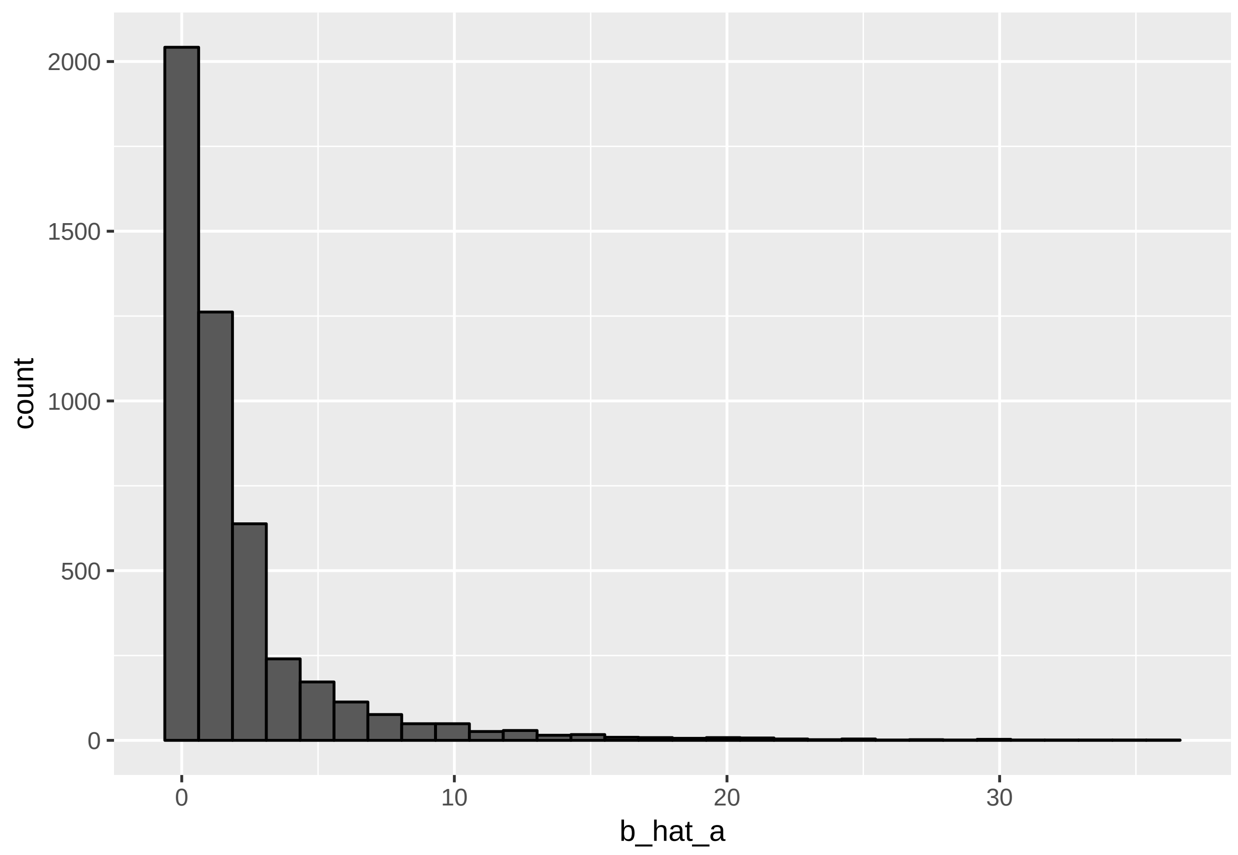
```
##Now we use the test data to check the RSME for the model
#first we have to create user averages for the test set so that we can make our predictions
location_avgs <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(event_avgs, by='EVENT_TYPE') %>%
  group_by(LOCATION) %>%
  summarize(b_hat_l = mean(FATALITIES - mu_hat - b_hat_c- b_hat_et))

## Then we have to calculate the predicted ratings for the test model
predicted_fatalities <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(event_avgs, by='EVENT_TYPE') %>%
  left_join(location_avgs, by='LOCATION') %>%
  mutate(pred = mu_hat + b_hat_c + b_hat_et+b_hat_l) %>%
  .$pred
```

```
#next we test to see the RMSE
model_location_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Country + Event + Location Effects Model",
                                     RMSE = model_location_rmse ))
```

| method | RMSE |
|---|---|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |
| Country + Event Type Effects Model | 4.576674 |
| Country + Event + Location Effects Model | 3.559413 |

The RMSE result for the Country, Events, and Location Effects is still not very low so I added two more parameters to the model: Primary and Secondary Actor Effects. I first added the Primary Actor and then the Secondary Actor to see if there were independent effects and there appeared to be a slight difference. The summary of average fatalities across event types shows that there is variation in the average fatality counts over the different actors involved in the conflicts. Below is the model including just the primary actor followed by the model including both Actors.

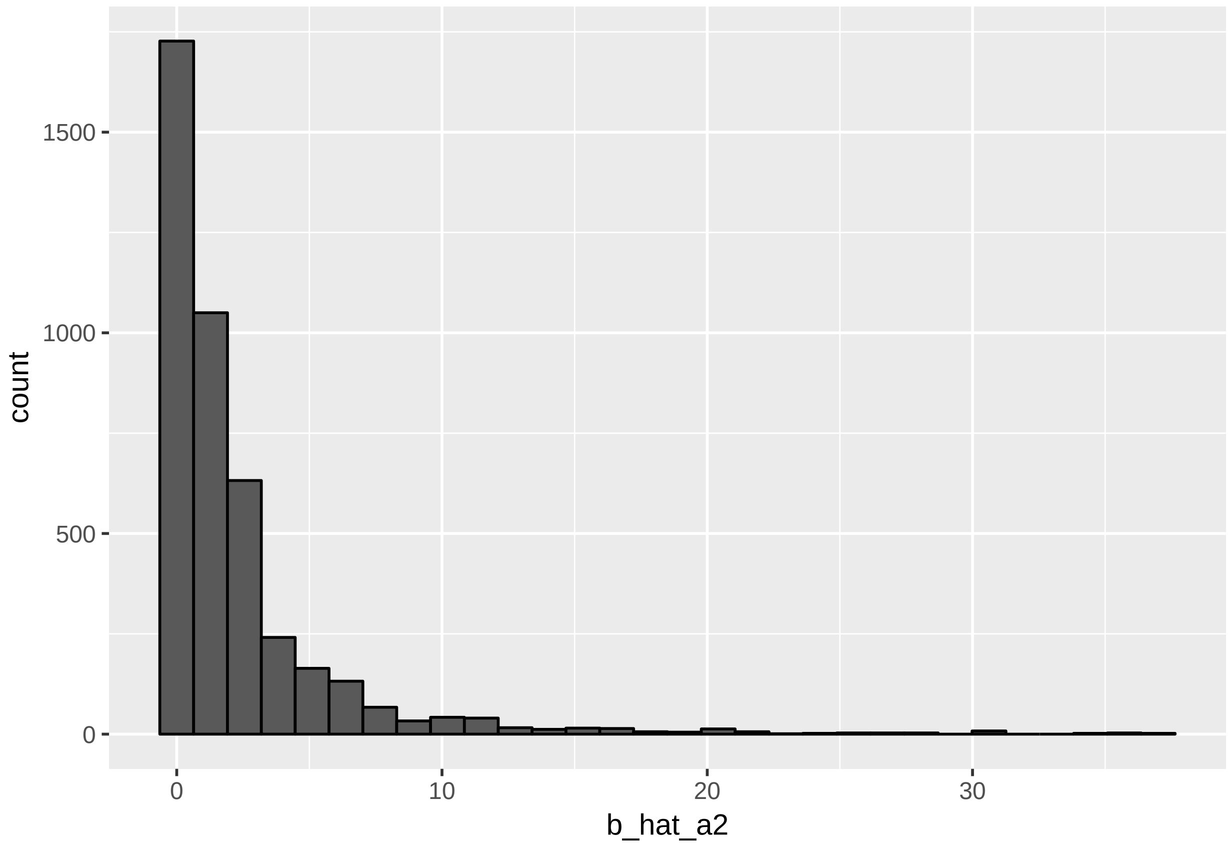
```
## b_hat_a is the average factality rate for each specific actor1 observed in the dataset
train %>%
  group_by(ACTOR1) %>%
  summarize(b_hat_a = mean(FATALITIES)) %>%
  ggplot(aes(b_hat_a)) +
  geom_histogram(bins = 30, color = "black")
```



```
##Now we use the test data to check the RSME for the model
#first we have to create user averages for the test set so that we can make our predictions
actor_avgs <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(event_avgs, by='EVENT_TYPE') %>%
  left_join(location_avgs, by='LOCATION') %>%
  group_by(ACTOR1) %>%
  summarize(b_hat_a = mean(FATALITIES - mu_hat - b_hat_c- b_hat_et- b_hat_l))

## Then we have to calculate the predicted ratings for the test model
predicted_fatalities <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(event_avgs, by='EVENT_TYPE') %>%
  left_join(location_avgs, by='LOCATION') %>%
  left_join(actor_avgs, by='ACTOR1') %>%
  mutate(pred = mu_hat + b_hat_c + b_hat_et+b_hat_l+ b_hat_a) %>%
  .$pred

#next we test to see the RMSE
model_actor_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Country + Event + Location + Actor 1 Effects Model",
                                     RMSE = model_actor_rmse ))
```

| method | RMSE |
|---|---|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |
| Country + Event Type Effects Model | 4.576674 |
| Country + Event + Location Effects Model | 3.559413 |
| Country + Event + Location + Actor 1 Effects Model | 3.390773 |

```
## b_hat_a2 is the average factality rate for each specific actor2 observed in the dataset
train %>%
  group_by(ACTOR2) %>%
  summarize(b_hat_a2 = mean(FATALITIES)) %>%
  ggplot(aes(b_hat_a2)) +
  geom_histogram(bins = 30, color = "black")
```

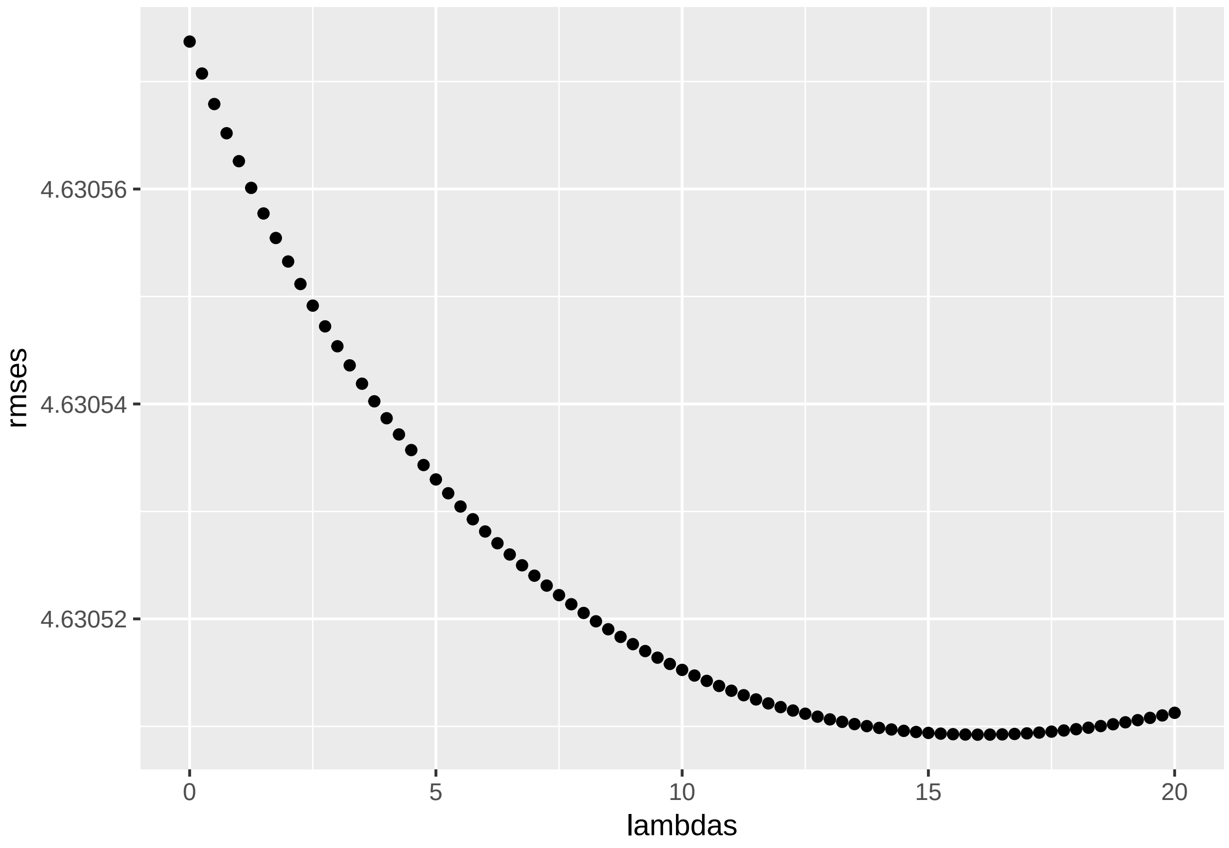


```
#first we have to create user averages for the test set so that we can make our predictions
actor2_avgs <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(event_avgs, by='EVENT_TYPE') %>%
  left_join(location_avgs, by='LOCATION') %>%
  left_join(actor_avgs, by='ACTOR1') %>%
  group_by(ACTOR2) %>%
  summarize(b_hat_a2 = mean(FATALITIES - mu_hat - b_hat_c- b_hat_et- b_hat_l-b_hat_a))

## Then we have to calculate the predicted ratings for the test model
predicted_fatalities <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(event_avgs, by='EVENT_TYPE') %>%
  left_join(location_avgs, by='LOCATION') %>%
  left_join(actor_avgs, by='ACTOR1') %>%
  left_join(actor2_avgs, by='ACTOR2') %>%
  mutate(pred = mu_hat + b_hat_c + b_hat_et+b_hat_l+ b_hat_a+ b_hat_a2) %>%
  .$pred

#next we test to see the RMSE
model_actor2_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Country + Event + Location + Actor 1 & 2Effects Model",
                                     RMSE = model_actor2_rmse ))
```

| method | RMSE |
|---|---|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |
| Country + Event Type Effects Model | 4.576674 |
| Country + Event + Location Effects Model | 3.559413 |
| Country + Event + Location + Actor 1 Effects Model | 3.390773 |
| Country + Event + Location + Actor 1 & 2Effects Model | 3.309326 |

The RMSE result for the Country, Events, Location, and Actor 1 & 2 Effects is still not very low so in lieu of adding any additional parameters, I added a regularization effect to attempt to account for differences in average fatality rates across countries allowing for a regularization of the rates for countries with very low fatality rates. To do so I first used cross validation to identify the most effective penalty term (lambda) for the regularization process.

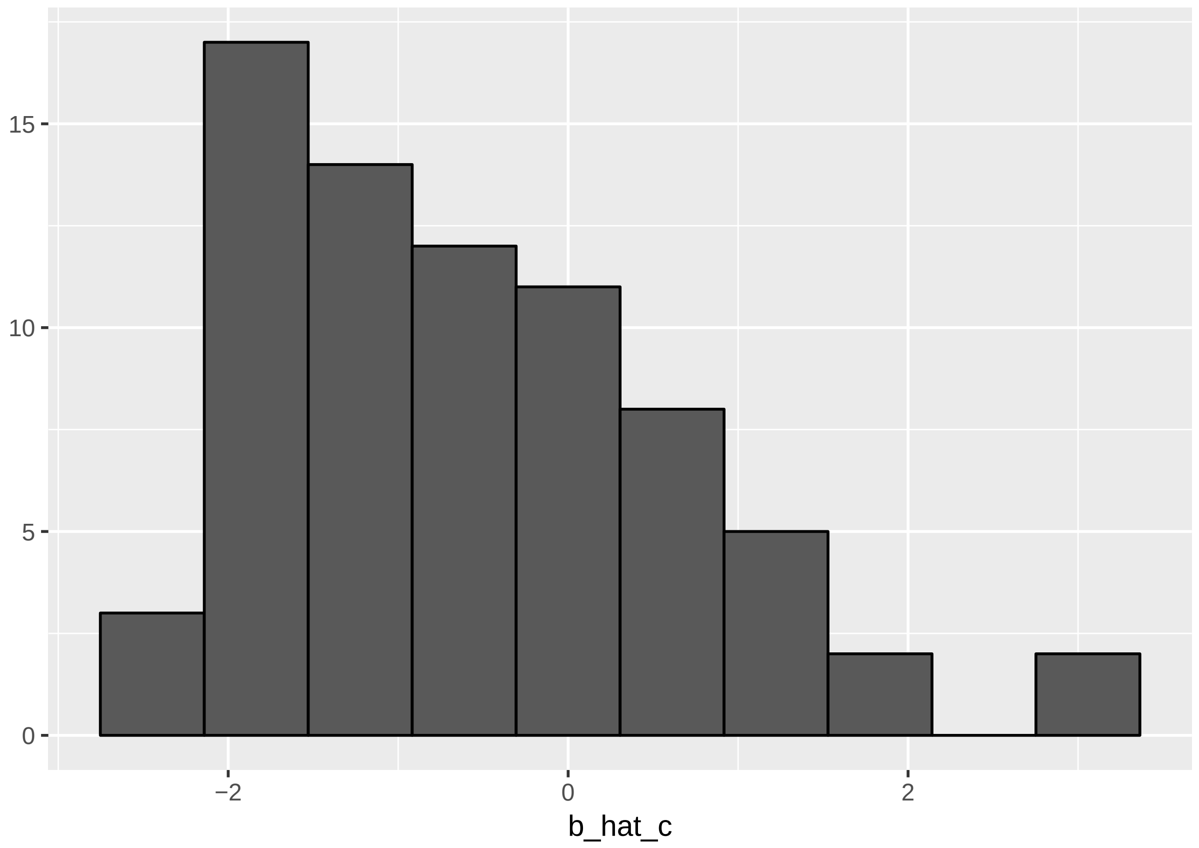
```
################################REGULARIZATION##########################################################
##In oder to ensure that states with different number of events are not causing interference in our mod

##Here we use cross-validation to identify the lambda (regularization penalty term) that will be most u

lambdas <- seq(0, 20, 0.25)

rmses <- sapply(lambdas, function(l){

  mu <- mean(train$FATALITIES)

  b_c <- train %>%
    group_by(COUNTRY) %>%
    summarize(b_c = sum(FATALITIES - mu)/(n()+l))



  predicted_fatalities <-
    test%>%
    left_join(b_c, by = 'COUNTRY') %>%
    mutate(pred = mu + b_c) %>%
    pull(pred)
  return(RMSE(predicted_fatalities , test$FATALITIES))
})

##Next we plot the rmses across lambdas to help identify which will be the most useful for our model
qplot(lambdas, rmses)
```



```
## [1] 16
```

With this lambda, 16, I re-ran all of the previous models (excluding event tye as that had a very low effect) and find that regularization actually makes my model perform worse for almost all model types

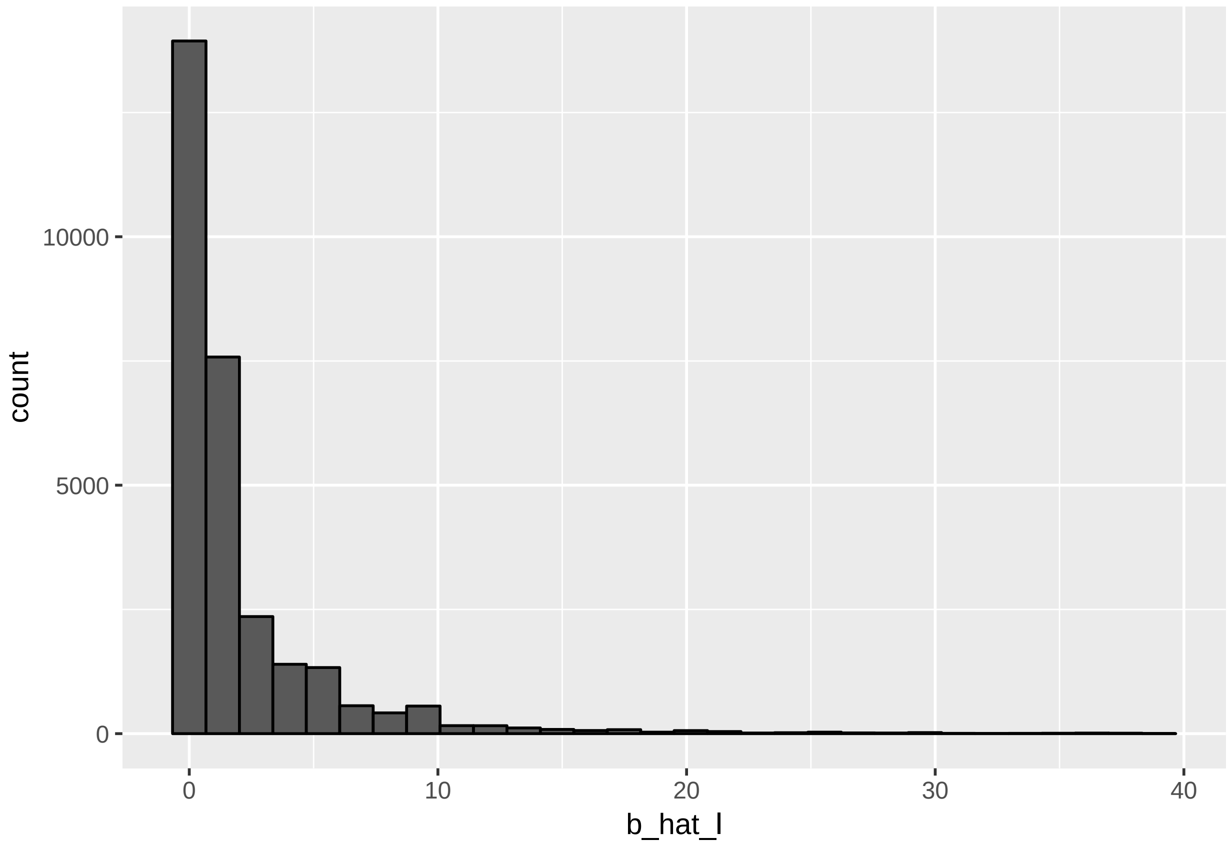
```
## b_hat_c is the average of the diffence between the fatality rate within a specific country minus the
country_avgs <- train %>%
  group_by(COUNTRY) %>%
  summarize(b_hat_c = sum(FATALITIES - mu_hat)/(n()+lambda), n_i = n())


##Here we visualize the distribution of b_hat_c
country_avgs %>% qplot(b_hat_c, geom ="histogram", bins = 10, data = ., color = I("black"))
```



```
##Now we use the validation data to check the RSME for the mode

#first we have to calculate b_hat_c for the validation set and calculate the predicted ratings for the
predicted_fatalities <- mu_hat + test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  .$b_hat_c

#next we test to see the RMSE
model_country_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Regularized Country Effect Model",
                                     RMSE = model_country_rmse ))
#View the table
rmse_results %>% knitr::kable()
```

| method | RMSE |
|---|---|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |
| Country + Event Type Effects Model | 4.576674 |
| Country + Event + Location Effects Model | 3.559413 |
| Country + Event + Location + Actor 1 Effects Model | 3.390773 |
| Country + Event + Location + Actor 1 & 2Effects Model | 3.309326 |
| Regularized Country Effect Model | 4.630509 |

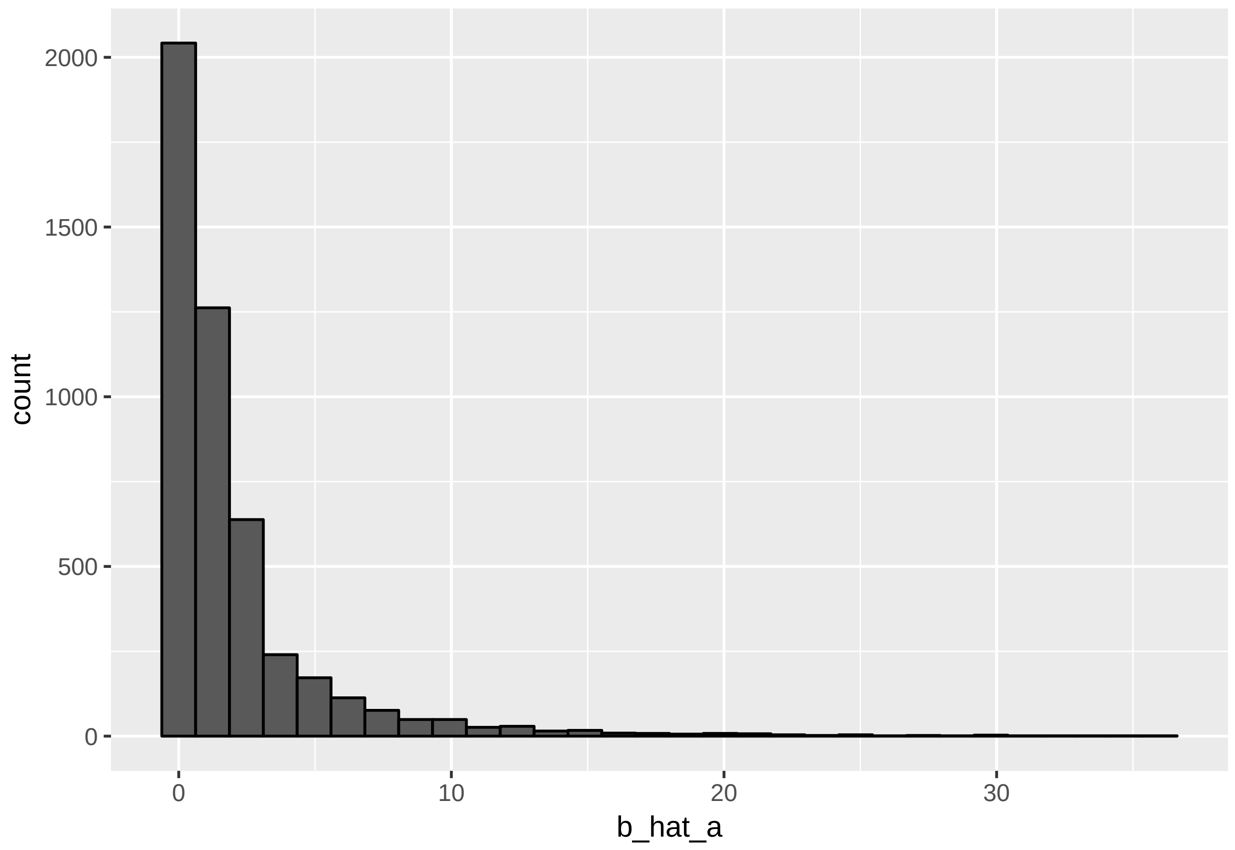
```r
## b_hat_l is the average factality rate for each specific location observed in the dataset
train %>%
  group_by(LOCATION) %>%
  summarize(b_hat_l = mean(FATALITIES)) %>%
  ggplot(aes(b_hat_l)) +
  geom_histogram(bins = 30, color = "black")
```



```r
##Now we use the test data to check the RSME for the model
#first we have to create user averages for the test set so that we can make our predictions
location_avgs <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  group_by(LOCATION) %>%
  summarize(b_hat_l = mean(FATALITIES - mu_hat - b_hat_c))

## Then we have to calculate the predicted ratings for the test model
predicted_fatalities <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(location_avgs, by='LOCATION') %>%
  mutate(pred = mu_hat + b_hat_c +b_hat_l) %>%
  .$pred

#next we test to see the RMSE
model_location_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Regularized Country + Location Effects Model",
                                     RMSE = model_location_rmse ))

#View the table
rmse_results %>% knitr::kable()
```

| method | RMSE |
|---|---|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |
| Country + Event Type Effects Model | 4.576674 |
| Country + Event + Location Effects Model | 3.559413 |
| Country + Event + Location + Actor 1 Effects Model | 3.390773 |
| Country + Event + Location + Actor 1 & 2Effects Model | 3.309326 |
| Regularized Country Effect Model | 4.630509 |
| Regularized Country + Location Effects Model | 3.592064 |

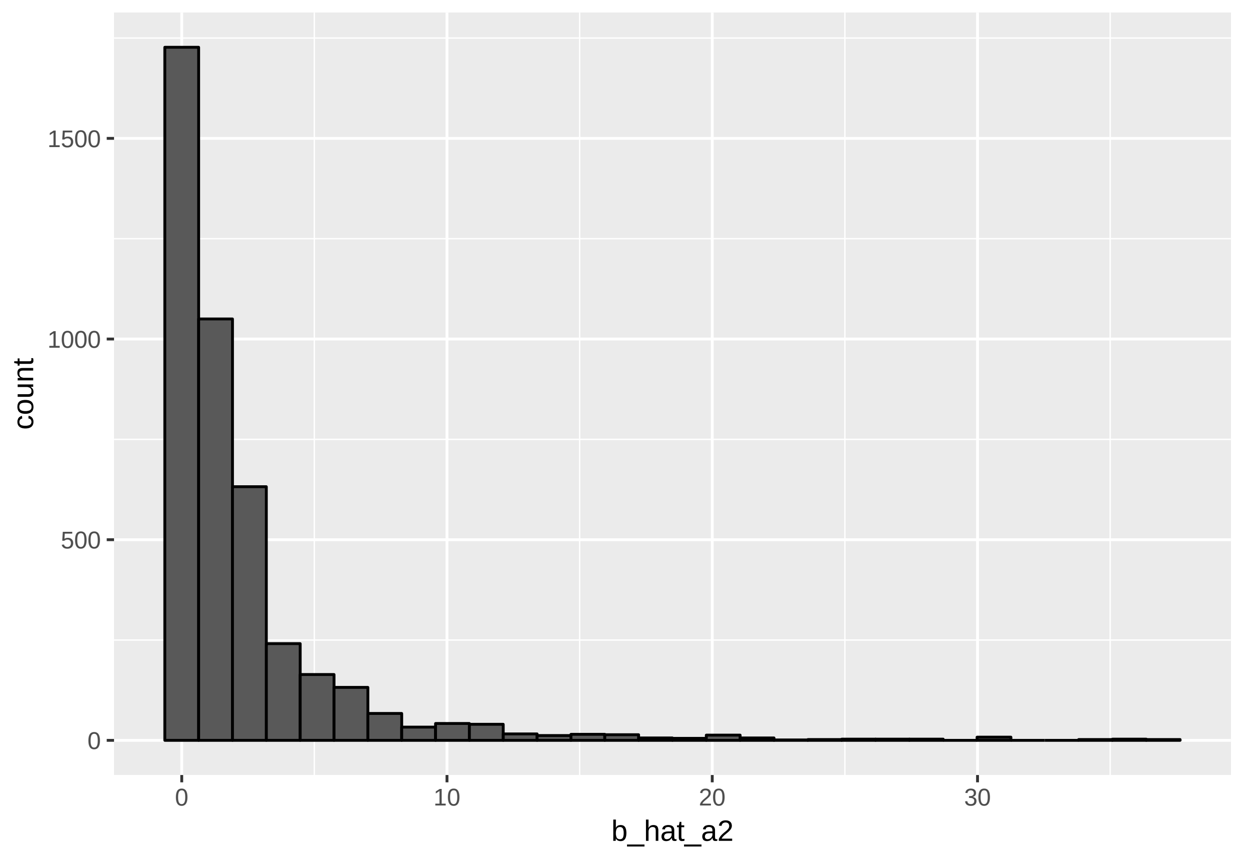
```
## b_hat_l is the average factality rate for each specific location observed in the dataset
train %>%
  group_by(ACTOR1) %>%
  summarize(b_hat_a = mean(FATALITIES)) %>%
  ggplot(aes(b_hat_a)) +
  geom_histogram(bins = 30, color = "black")
```



```r
##Now we use the test data to check the RSME for the model
#first we have to create user averages for the test set so that we can make our predictions
actor_avgs <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(location_avgs, by='LOCATION') %>%
  group_by(ACTOR1) %>%
  summarize(b_hat_a = mean(FATALITIES - mu_hat - b_hat_c- - b_hat_l))

## Then we have to calculate the predicted ratings for the test model
predicted_fatalities <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(location_avgs, by='LOCATION') %>%
  left_join(actor_avgs, by='ACTOR1') %>%
  mutate(pred = mu_hat + b_hat_c +b_hat_l+ b_hat_a) %>%
  .$pred

#next we test to see the RMSE
model_actor_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Regularized Country  Location + Actor 1 Effects Model",
                                     RMSE = model_actor_rmse ))

#View the table
rmse_results %>% knitr::kable()
```

| method | RMSE |
|---|---:|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |
| Country + Event Type Effects Model | 4.576674 |
| Country + Event + Location Effects Model | 3.559413 |
| Country + Event + Location + Actor 1 Effects Model | 3.390773 |
| Country + Event + Location + Actor 1 & 2Effects Model | 3.309326 |
| Regularized Country Effect Model | 4.630509 |
| Regularized Country + Location Effects Model | 3.592064 |
| Regularized Country Location + Actor 1 Effects Model | 4.158283 |

```r
## b_hat_l is the average factality rate for each specific location observed in the dataset
train %>%
  group_by(ACTOR2) %>%
  summarize(b_hat_a2 = mean(FATALITIES)) %>%
  ggplot(aes(b_hat_a2)) +
  geom_histogram(bins = 30, color = "black")
```



```r
##Now we use the test data to check the RSME for the model
#first we have to create user averages for the test set so that we can make our predictions
actor2_avgs <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(location_avgs, by='LOCATION') %>%
  left_join(actor_avgs, by='ACTOR1') %>%
  group_by(ACTOR2) %>%
  summarize(b_hat_a2 = mean(FATALITIES - mu_hat - b_hat_c- b_hat_l-b_hat_a))

## Then we have to calculate the predicted ratings for the test model
predicted_fatalities <- test %>%
  left_join(country_avgs, by='COUNTRY') %>%
  left_join(location_avgs, by='LOCATION') %>%
  left_join(actor_avgs, by='ACTOR1') %>%
  left_join(actor2_avgs, by='ACTOR2') %>%
  mutate(pred = mu_hat + b_hat_c + b_hat_l+ b_hat_a+ b_hat_a2) %>%
  .$pred

#next we test to see the RMSE
model_actor_rmse <- RMSE(predicted_fatalities, test$FATALITIES)

#finally we add this model to our RSME table
rmse_results <- bind_rows(rmse_results,
                          data_frame(method="Regularized Country + Location + Actor 1 & 2 Effects Model
                                     RMSE = model_actor_rmse ))
```

| method | RMSE |
|---|---|
| Overall average | 4.814038 |
| Country Effect Model | 4.630574 |
| Country + Event Type Effects Model | 4.576674 |
| Country + Event + Location Effects Model | 3.559413 |
| Country + Event + Location + Actor 1 Effects Model | 3.390773 |
| Country + Event + Location + Actor 1 & 2Effects Model | 3.309326 |
| Regularized Country Effect Model | 4.630509 |
| Regularized Country + Location Effects Model | 3.592064 |
| Regularized Country Location + Actor 1 Effects Model | 4.158283 |
| Regularized Country + Location + Actor 1 & 2 Effects Model | 3.804482 |

With these new models, we see that using data regularization for country averages does nothing to help model reliability

## Conclusion

In attempting to create a high quality fatality prediction model for conflict in Asia and Africa, using the ACLED data covering conflict events over 21 years and 74 countries, I was unable to use machine learning to produce a reliable model for prdiction. Every iteration of the model that I attempted was well above 1. It appears that this data is particularly ill-suited for this type of prediction model. Knowing the type of event, the location, the actors invloved did not provide enough information to accuarately pinpoint the likely count of fatalities. Additional data may help to create a better understanding of these conflict predictions as this was a relatively small sample size to conduct machine learning with. Knowing more about the roots of the conflict or the occurance of conflict in a time-series test may help to shed light on this topic and give a more accurate prediction that would be reliable.