IFSC Bouldering World Cup Winner Contender

A Classification Report

Tyler Chang

9/27/2022

Contents

1: Overview of the Report

Rock climbing has become an increasingly popular sport over the past few years. With its inclusion in the Summer 2021 Olympic games and the success of films such as *Free Solo*, competitive rock climbing has arguably reached its highest point yet in popularity. As the preeminent organization for competitive rock climbing, the *International Federation for Sport Climbing* (IFSC) hosts a series of competitions, referred to as World Cups, each year in three disciplines: Bouldering, Lead, and Speed. This report is dedicated to the discipline of bouldering[1].

[1]For more information about bouldering, see https://www.climbernews.com/what-is-bouldering/

Given the international nature of IFSC competitions, the selection methods for each nation's climbing team can vary substantially, though many favor a mock-competition format for selection. For potential new members, this mock-competition format works quite well. For those who are already members and are attempting to maintain their place on the team, the current format fails to appreciate their previous performances. While one climber may outperform another at the tryout competition, the pressures of international competition have a well documented impact on many climbers' performances. Climbers who have already achieved success at IFSC events may do better, at least initially, than those without similar experience. As such, considering past IFSC performances may help coaches improve the expected performances of their teams.

Using data taken from 2018 and 2019 IFSC bouldering competitions[2], I aim to predict whether a given climber should be considered as a serious contender for winning a competition. As will be shown, only a very small percentage of the participating climbers actually have a non-miniscule chance of winning an IFSC event.

1.1: The Data Set and Relevant Variables

While data sets for the three aforementioned climbing disciplines as well as a combined format (much akin to that which was used at the 2021 Olympics) are available on the original Kaggle page, I have elected to focus solely on the bouldering data set for one reason, athletes are not required to participate in all of the disciplines. In fact, many are specialists in only a single discipline, making a comparison of performances largely speculative. Relatedly, limiting the data set to only the athletes who compete in multiple disciplines would result in a sample size much too small to be of much use.

With that said, two important questions remain: *What is bouldering and how is it actually scored?*

1.1.1: What is Bouldering?

Bouldering is non-roped, low-altitude climbing (usually no greater than 16 ft. in height). Each climb is referred to as a *problem* and a method used to successfully complete a problem is called the *beta*. When a person completes a problem, they are said to have *topped it* or *gotten a top*. If a problem is topped on a person's first attempt without prior knowledge of the beta, the climber has *onsighted* the problem. If the beta is known beforehand and the top is achieved on the first attempt, the problem has been *flashed*. Any top, regardless of attempts or knowledge of the beta, is called a *send* or *sending the boulder*.

The difficulty of a boulder problem outdoors is typically graded by the consensus of those who have sent the problem. Indoors, however, grading is done by the creators of the boulder problems, the *route setters*. There are also multiple scales for climbing grades: *V-Scale* (sometimes called the *Hueco Scale*), *Font Scale*, and others[3].

1.1.2: How is Bouldering Scored?

IFSC competitions are composed of three rounds-Qualifications, Semifinals, and Finals-and are divided into male and female categories. While the number of competitors in a qualification round across both categories can range from approximately 80 up to 150+, this is shrunk down to 20 men and 20 women for semifinals. From this, the finals are comprised of 6 men and 6 women.

In each round, each climber is given a short window of time to complete (5 minutes in qualifications, 4 minutes in semifinals and finals) a boulder problem. There are 5 problems in qualifications, 4 problems in semifinals, and 4 problems in finals. Points are awarded in two ways:

1. *Tops*: 1 point per problem that is completed within the alloted time.
2. *Zones*: These are given for reaching a predesignated hold that appears between the first and final hold(s) of a climb.

The number of attempts for both zones and tops are also recorded. These do not directly affect tops or zones, but they are used to differentiate climbers with the same number of both tops and zones. The overall scoring goes as follows:

[2]Source: https://www.kaggle.com/datasets/brkurzawa/ifsc-sport-climbing-competition-results
[3]To see what these look like, see https://mojagear.com/rock-climbing-grades-comparison-chart-rating-systems/

1. Tops are the most important. The person with the most tops in the fewest attempts wins the round.
2. If there is a tie in tops and attempts to top, the person who has the most zones in the fewest attempts among those who tied for tops and attempts to top wins the round.
3. If there is a tie in tops, zones, and attempts for both categories, the person who did better in the previous round wins the current round.

The order in which the athletes attempt the problems in qualifications is randomly assigned. For semifinals and finals, the athletes climb in reverse performance order, meaning that the better one performed in the previous round, the later they will climb.

1.1.3: The Relevant Variables

Though there are decently large number of factors that affect a climber's expected performance, the most important are as follows:

1. Tops in each round
2. Zones in each round
3. Attempts to get the tops
4. Attempts to get the zones
5. Starting order

As such, these will be the main predictors used when it comes to making models in Section 2.

1.2: Outline of the Data Pipeline

In the following section, **Methods and Analysis**, I cover importing, cleaning, and exploring the data set, as well as discuss my approach to modeling the data. As the goal of this report is a classification task, I make use of logistic regression, K-Nearest Neighbors, and Random Forest models. Notably, I include two sets of models. In the first, I consider all possible predictors, including data that comes from the final round, to predict whether a given climber is a potential winner. Despite this yielding very highly accurate, sensitive, and specific models, I point out that as predictive models, they have, in some instances, too high a requirement to be of use. Since only a small fraction of climbers ever make a final round, I also redo the model development with the finals-related predictors excluded.

In Section 3, **Results**, I discuss and compare the performances of the individual models. As will be shown, the random forest model outperforms the alternatives in both sets of models.

Finally, in Section 4, **Conclusion**, I provide a brief summary of the report, discuss the limitations of the work done here and potential future work, and include a few final notes.

2: Methods and Analysis

Several libraries are used throughout this report. While not all will be used in this section, I will nonetheless install (if necessary) and call them all now.

```
library(tidyverse)
library(caret)
library(data.table)
library(lubridate)
library(stringr)
library(gridExtra)
library(ggrepel)
library(randomForest)
library(ggridges)
library(knitr)
```

2.1: Importing and Cleaning the Data Set

The data set, *boulder_results.csv*, contains all of the necessary information and is available on both Github (where I will imported it from) and Kaggle (see footnote 2).

```
set.seed(1917, sample.kind = "Rounding")
url_name <- "https://raw.githubusercontent.com/tchang343/IFSC/main/boulder_results.csv"
temp_table <- read.table(file = url_name, header = TRUE, sep = ",")
```

At this point, the table has 5535 rows, 13 columns, and includes NA values, as seen by:

```
dim(temp_table)
```

```
## [1] 5535   13
```

```
any(is.na(temp_table))
```

```
## [1] TRUE
```

Nonetheless, let's take a look at the first six rows of the table.

```
head(temp_table)
```

```
##                                    Competition.Title        Competition.Date
## 1 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 2 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 3 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 4 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 5 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 6 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
##     FIRST       LAST Nation StartNr Rank Qualification Qualification.1
## 1   Naile    MEIGNAN    FRA      15    1        6T7z99
## 2  Miriam       FOGU    ITA      27    2       4T8z814
## 3   Vanda MICHALKOVA    SVK      48    3        6T7z89
## 4    Lola    SAUTIER    FRA      17    4        4T6z69
## 5  Polina   KULAGINA    RUS      39    5       4T6z611
## 6 Kintana      ILTIS    FRA      16    6        6T7z77
##   Qualification.2 Semifinal  Final Category
## 1                  3T4z55 3T4z89  boulder
## 2                  3T4z55 2T4z27  boulder
## 3                  3T3z43 2T3z23  boulder
## 4                  2T3z89 1T3z38  boulder
## 5                  3T4z76 1T3z47  boulder
## 6                  2T3z88 1T2z33  boulder
```

There are several unnecessary columns currently included in the table. Both *Qualification.1* and *Qualification.2* are carry-overs from the data sets for lead climbing, where there are two separate qualification rounds. As such, they are both entirely empty and can be safely removed. Similarly, the *Category* column is meant to differentiate the climbing disciplines in the combined format data set. Since, however, I am only looking at bouldering, this column is populated by a single word being repeated and can also be removed.

```
temp_table_rev <- temp_table %>%
  select(Competition.Title, Competition.Date, FIRST, LAST, Nation, StartNr, Rank,
         Qualification, Semifinal, Final)
```

We should now have only 10 columns but the number of rows should be unchanged.

```
dim(temp_table_rev)
```

```
## [1] 5535   10
```

Before proceeding on, it is important to make it clear how the current table is to be read.

1. *StartNr* is the order in which an athlete climbed in the qualification round.
2. *Rank* is an athletes ranking at a given competition.
3. For each of the three rounds, the format of their score goes as follows:

- The first number (before the T) is the number of tops completed.
- The second number (after the T and before the Z) is the number of zones reached.
- The final number is a combination of the number of attempts for both tops and zones. If there are only two digits, the first digit is the attempts for tops and the second digit is for zones. If there are three digits, the first digit is for tops and the latter two are for zones. If there are four digits, the first two digits are for tops and the latter two are for zones.

4. *FIRST* and *LAST* are the first and last names of the athletes.
5. *Competition.title*, *Competition.date*, and *Nation* are self-explanatory.

Recall that there are NA values somewhere in the table so the next priority is to local which columns have them.

```
na_table <- data.frame(Column_name = "Competition.Title",
                       NAs = any(is.na(temp_table_rev$Competition.Title))) #Setting up a table to show
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "Competition.Date",
                                 NAs = any(is.na(temp_table_rev$Competition.Date))))
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "FIRST",
                                 NAs = any(is.na(temp_table_rev$FIRST))))
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "LAST",
                                 NAs = any(is.na(temp_table_rev$LAST))))
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "Nation",
                                 NAs = any(is.na(temp_table_rev$Nation))))
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "StartNr",
                                 NAs = any(is.na(temp_table_rev$StartNr))))
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "Rank",
                                 NAs = any(is.na(temp_table_rev$Rank))))
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "Qualification",
                                 NAs = any(is.na(temp_table_rev$Qualification))))
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "Semifinal",
                                 NAs = any(is.na(temp_table_rev$Semifinal))))
na_table <- bind_rows(na_table,
                      data.frame(Column_name = "Final",
                                 NAs = any(is.na(temp_table_rev$Final))))
na_table %>% knitr::kable()
```

Column_name	NAs
Competition.Title	FALSE
Competition.Date	FALSE
FIRST	FALSE
LAST	FALSE
Nation	FALSE

Column_name	NAs
StartNr	TRUE
Rank	TRUE
Qualification	FALSE
Semifinal	FALSE
Final	FALSE

We can now see that the NAs are limited to the StartNr and Rank columns. There are 37 NA values in StartNr and 1 NAs in Rank. Let's deal with the Rank column first. To do this, the NA value must be located.

```
which(is.na(temp_table_rev$Rank))
```

```
## [1] 5535
```

The NA value appears in the 5535th row, i.e., the final row. Since the recording of the competition is no longer publicly available and the IFSC website does not include the relevant data in its current iteration, I have no non-speculative means of replacing the NA with a reasonable value. Moreover, the loss of a single row is unlikely to have a significant impact on the overall viability of the data set. Thus, I will simply remove the last row and confirm that there are no more NAs in the Rank column after the removal.

```
temp_table_rev <- temp_table_rev[-5535,]
any(is.na(temp_table_rev$Rank))
```

```
## [1] FALSE
```

Moving on, there were 37 missing values in the StartNr column but unfortunately, a similar issue as was faced with the Rank column appears here as well. Replacing them is not a viable options due to the original data source no longer being publicly available. Still, it is worthwhile to locate where the NA and missing values are to ensure that their removal will not be problematic.

```
which(is.na(temp_table_rev$StartNr))
```

```
##  [1] 1768 1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782
## [16] 1783 1784 1785 4808 4809 4810 4811 4812 4813 4814 4815 4816 4817 4818 4819
## [31] 4820 4821 4822 4823 4824 4825
```

```
#The NA values in the StartNr are held between (inclusively) rows 1768-1785
#and 4808-4825. So, let's take a look at some of those rows.
temp_table_rev[c(1768:1785),]
```

```
##                            Competition.Title    Competition.Date    FIRST
## 1768 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019     Luce
## 1769 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019    Julia
## 1770 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019  Kintana
## 1771 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019  Camille
## 1772 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019     Aida
## 1773 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019    Pleun
## 1774 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019  Pauline
## 1775 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019   Miriam
## 1776 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019 Caterina
## 1777 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019     Vana
## 1778 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019     Elsa
## 1779 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019     Nika
## 1780 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019   Ingrid
## 1781 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019   Hannah
## 1782 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019    Betka
```

```
## 1783 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019 Federica
## 1784 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019      Liv
## 1785 European Youth Cup (B) - Soure (POR) 2019 27  - 28 April 2019   Lucija
##                  LAST Nation StartNr Rank Qualification Semifinal   Final
## 1768          DOUADY    FRA      NA    1      8T8z1310            1T3z16
## 1769            LOTZ    AUT      NA    2      7T8z1111            1T3z16
## 1770           ILTIS    FRA      NA    3       6T6z127            1T3z48
## 1771          POUGET    FRA      NA    4      7T7z1412            1T2z13
## 1772 TORRES ILLAMOLA    ESP      NA    5      5T7z1312            1T1z22
## 1773           FRANK    NED      NA    6       4T6z911            1T1z44
## 1774            OURY    BEL      NA    7      4T7z1014            0T2z04
## 1775            FOGU    ITA      NA    8        6T7z98           0T2z011
## 1776       DAL ZOTTO    ITA      NA    9      4T6z1012            0T1z02
## 1777         PICCINI    CRO      NA   10       4T6z915            0T1z03
## 1778         RAVINET    FRA      NA   11      4T6z1212
## 1779        POTAPOVA    UKR      NA   12      4T5z1012
## 1780     KINDLIHAGEN    NOR      NA   13        4T4z86
## 1781           SMITH    GBR      NA   14       3T7z511
## 1782         DEBEVEC    SLO      NA   15       3T6z612
## 1783         MABBONI    ITA      NA   16       3T6z711
## 1784            EGLI    SUI      NA   17       3T6z714
## 1785          TARKUS    SLO      NA   18       3T5z410
```

In addition to confirming the presence of NA values, we can see that there are blank spaces in the Semifinals and Finals columns. This is to be expected since not all participants climbed in the semifinal or final rounds. Handling these blank spots is important since the non-participation will affect our ability to predict possible winners. For now, however, let us finish handling the NAs by removing the affected rows of StartNr.

```
temp_table_rev <- temp_table_rev[-which(is.na(temp_table_rev$StartNr)),]
```

With that done, I will do a final confirmation that all NA values have been removed from the table and that the dimensions are as expected.

```
dim(temp_table_rev) #Should be 5498 x 10
```

```
## [1] 5498   10
```

```
any(is.na(temp_table_rev))
```

```
## [1] FALSE
```

All of the NA values have been successfully removed! We are now free to deal with changing the qualification, semifinals, and finals columns into a more usable format. As a first step, I am going to replace all of the missing values with the following: 0T0z00. If a person did not qualify for an advanced round or if they did not complete any tops or zones in a given round, their score would be 0T0z00. Once this is converted over to the above mentioned format, this will be recorded as a single 0.

```
temp_table_rev[temp_table_rev == ""] <- "0T0z00"
```

The next step is to convert the data points that are currently formatted as 3T2z89 or similar into several new columns, as described below:

1. *Total_Tops*
2. *Total_Zones*
3. *Total_Top_Attempts*
4. *Total_Zone_Attempts*
5. *Qualification_Tops*
6. *Semifinal_Tops*

7. *Final_Tops*
8. *Qualification_Zones*
9. *Semifinal_Zones*
10. *Final_Zones*
11. *Qualification_Top_Attempts*
12. *Semifinal_Top_Attempts*
13. *Final_Top_Attempts*
14. *Qualification_Zones_Attempts*
15. *Semifinal_Zones_Attempts*
16. *Final_Zones_Attempts*

To create these new columns, I will use the *stringr* library to extract the relevant information from the *Qualification*, *Semifinal*, and *Final* columns. First, since I will be repeatedly referring to these columns, I will make separate variables for them to shorten their names.

```
quali <- temp_table_rev$Qualification
semi <- temp_table_rev$Semifinal
fin <- temp_table_rev$Final
```

Since I have to extract different parts of the strings for each round's associated columns, the regular expressions will vary significantly. As such, I will divide this part into four subsections.

2.1.1: Identifying the Number of Tops for Each Round

In order to isolate the number of tops each climber completed during each round of a competition, the digits prior to the *T* from the strings of form *#T#z##* must extracted and converted into a numeric object. Since the maximum number of tops in any given round is 5, there will only be a single digit preceding the *T*. This, alongside there being no letter characters appearing before the relevant digit, makes a single regular expression correctly identify the number of tops for all rounds.

```
#Separating out the number of tops for each climber in the qualification round
quali_tops <- sapply(quali, function(x){
  quali_top <- str_extract(x, pattern = "\\d")
  as.numeric(quali_top)
})

#Same idea but for semifinals
semi_tops <- sapply(semi, function(x){
  semi_top <- str_extract(x, "\\d")
  as.numeric(semi_top)
})

#Same idea but for finals
fin_tops <- sapply(fin, function(x){
  fin_top <- str_extract(x, "\\d")
  as.numeric(fin_top)
})
```

I can now calculate the total number of tops across all rounds for each climber at each competition.

```
total_tops <- quali_tops + semi_tops + fin_tops
```

Now, the *Total_Tops*, *Qualification_Tops*, *Semifinal_Tops*, and *Final_Tops* columns can be made and added to the overall data table.

```
temp_table_rev <- temp_table_rev %>%
  mutate(Total_Tops = total_tops,
         Qualification_Tops = quali_tops,
```

```
         Semifinal_Tops = semi_tops,
         Final_Tops = fin_tops)
```

Let's take a quick look at the table before moving onto making the remaining new columns.

```
head(temp_table_rev)
```

```
##                                    Competition.Title        Competition.Date
## 1 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 2 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 3 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 4 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 5 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 6 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
##     FIRST       LAST Nation StartNr Rank Qualification Semifinal  Final
## 1   Naile    MEIGNAN    FRA      15    1        6T7z99    3T4z55 3T4z89
## 2  Miriam       FOGU    ITA      27    2       4T8z814    3T4z55 2T4z27
## 3   Vanda MICHALKOVA    SVK      48    3        6T7z89    3T3z43 2T3z23
## 4    Lola    SAUTIER    FRA      17    4        4T6z69    2T3z89 1T3z38
## 5  Polina   KULAGINA    RUS      39    5       4T6z611    3T4z76 1T3z47
## 6 Kintana      ILTIS    FRA      16    6        6T7z77    2T3z88 1T2z33
##   Total_Tops Qualification_Tops Semifinal_Tops Final_Tops
## 1         12                  6             3          3
## 2          9                  4             3          2
## 3         11                  6             3          2
## 4          7                  4             2          1
## 5          8                  4             3          1
## 6          9                  6             2          1
```

2.1.2: Identifying the Number of Zones in Each Round

The process for identifying the number of zones in each round is very similar to the one used to identify the tops, with the only substantial difference being in the regular expression used with the *str_extract()* function. As in the previous section, I will first extract the number of zones from the strings of form *#T#z##*, compute the total number of zones across all rounds for each climber, and create three new columns: *Qualification_Zones*, *Semifinal_Zones*, and *Final_Zones*.

```
#Getting the number of zones in qualifications for each climber at each competition.
quali_zones <- sapply(quali, function(q){
  temp <- str_extract(q, "T.")
  quali_zone <- str_sub(temp, 2)
  as.numeric(quali_zone)
})

#Number of zones in semifinals for each climber at each competition.
semi_zones <- sapply(semi, function(q){
  temp <- str_extract(q, "T.")
  semi_zone <- str_sub(temp, 2)
  as.numeric(semi_zone)
})

#Number of zones in finals for each climber at each competition.
fin_zones <- sapply(fin, function(q){
  temp <- str_extract(q, "T.")
  fin_zone <- str_sub(temp, 2)
```

```
  as.numeric(fin_zone)
})

#Finding the total number of zones per competition for each climber
total_zones <- quali_zones + semi_zones + fin_zones

#Adding total_zones, quali_zones, semi_zones, and fin_zones as new columns in the table.
temp_table_rev <- temp_table_rev %>%
  mutate(Total_Zones = total_zones,
         Qualification_Zones = quali_zones,
         Semifinal_Zones = semi_zones,
         Final_Zones = fin_zones)
```

2.1.3: Identifying the Number of Attempts in Each Round

Separating the numbers of attempts to get tops and the numbers of attempts to get zones from the original strings is more complicated than the previous two steps. Unlike the number of tops or zones, the number of digits is less consistent across zones. Since the number of attempts for both tops and zones are presented formatted as a single string of two to four digits, the first step is to be able to determine which digits refer to attempts to top and which refer to zone attempts.

To do this, note that it is impossible to receive credit for a top without also getting credit for a zone. This ensures that the number of zones will never be smaller than the number of tops, meaning that if there are three digits following the z in the string, the first digit is the number of attempts to top and the latter two are for attempts at zones. If there are two or four digits after the z, each type of attempt is represented by one or two digits, respectively.

To handle this, I will define two functions which, once the final two to four digits are separated off from the original $\#T\#z\#\#$ string, further split the substring based on its length. The first of these functions will address attempts to top and the second will be for attempts at zones.

```
#This version of the function is for attempts required to get the tops.
top_att_splitr <- function(v){
  if(nchar(v) == 2){
    top_att <- str_extract(v, "\\d")
  }
  else if(nchar(v) == 3){
    top_att <- str_extract(v, "\\d")
  }
  else if(nchar(v) == 4){
    top_att <- str_sub(v, 1, 2)
  }
  return(top_att)
}

#Same idea as before but it now addresses the zone attempts.
zone_att_splitr <- function(u){
  if(nchar(u) == 2){
    zone_att <- str_sub(u, 2)
  }
  else if(nchar(u) == 3){
    zone_att <- str_sub(u, 2)
  }
  else if(nchar(u) == 4){
    zone_att <- str_sub(u, 3)
```

```
  }
  return(zone_att)
}
```

With these function defined, the extraction process can proceed in much the same way as the previous steps.

```
#Number of attempts for tops in the qualification round for each climber at each
#competition.
quali_top_attempts <- sapply(quali, function(q){
  temp <- str_sub(q, 5)
  quali_top_attempt <- top_att_splitr(temp)
  as.numeric(quali_top_attempt)
})

#Number of attempts for tops in the semifinal round for each climber at each competition.
semi_top_attempts <- sapply(semi, function(q){
  temp <- str_sub(q, 5)
  semi_top_attempt <- top_att_splitr(temp)
  as.numeric(semi_top_attempt)
})

#Number of attempts for tops in the final round for each climber at each competition.
final_top_attempts <- sapply(fin, function(q){
  temp <- str_sub(q, 5)
  fin_top_attempt <- top_att_splitr(temp)
  as.numeric(fin_top_attempt)
})

#Determining the total number of attempts required for the tops for each climber at each
#competition.
total_top_attempts <- quali_top_attempts + semi_top_attempts + final_top_attempts

#Adding Total_Attempts_to_Top, Quali_Top_Attempts, Semifinal_Top_Attempts, and
#Final_Top_Attempts as new columns.
temp_table_rev <- temp_table_rev %>%
  mutate(Total_Attempts_to_Top = total_top_attempts,
         Qualification_Top_Attempts = quali_top_attempts,
         Semifinal_Top_Attempts = semi_top_attempts,
         Final_Top_Attempts = final_top_attempts)

#Number of attempts for zones in the qualification round for each climber at each
#competition.
quali_zone_attempts <- sapply(quali, function(q){
  temp <- str_sub(q, 5)
  quali_zone_attempt <- zone_att_splitr(temp)
  as.numeric(quali_zone_attempt)
})

#Number of attempts for zones in the semifinal round for each climber at each competition.
semi_zone_attempts <- sapply(semi, function(q){
  temp <- str_sub(q, 5)
  semi_zone_attempt <- zone_att_splitr(temp)
  as.numeric(semi_zone_attempt)
})
```

```
#Number of attempts for zones in the final round for each climber at each competition.
final_zone_attempts <- sapply(fin, function(q){
  temp <- str_sub(q, 5)
  final_zone_attempt <- zone_att_splitr(temp)
  as.numeric(final_zone_attempt)
})

#Determining the total number of attempts required for the zones for each climber at each
#competition.
total_zone_attempts <- quali_zone_attempts + semi_zone_attempts + final_zone_attempts

#Adding Total_Attempts_to_Zone, Quali_Zones_Attempts, Semi_Zones_Attempts, and
#Final_Zones_Attempts to the table as new columns.
temp_table_rev <- temp_table_rev %>%
  mutate(Total_Attempts_to_Zone = total_zone_attempts,
         Qualification_Zones_Attempts = quali_zone_attempts,
         Semifinal_Zones_Attempts = semi_zone_attempts,
         Final_Zones_Attempts = final_zone_attempts)
```

2.1.4: Final Adjustments to the Table

At the moment, the names of the athletes are split into two columns: *FIRST* and *LAST*. I will combine them into a single column and adjust the cases such that only the first letter of each part of the names is capitalized.

```
temp_table_rev <- temp_table_rev %>%
  mutate(Last2 = str_to_title(LAST)) %>%
  mutate(Name = str_c(FIRST, Last2, sep = " "))
```

Since all of the information about tops, zones, and attempts has been extracted and recorded in new columns, we no longer need the original *Qualification*, *Semifinal*, and *Final* columns. Thus, I will remove them as part of making the final version of the table.

```
bouldering <- temp_table_rev %>%
  select(Competition.Title, Competition.Date, Name, Nation, StartNr, Rank,
         Total_Tops, Total_Zones, Total_Attempts_to_Top, Total_Attempts_to_Zone,
         Qualification_Tops, Qualification_Zones, Qualification_Top_Attempts,
         Qualification_Zones_Attempts, Semifinal_Tops, Semifinal_Zones,
         Semifinal_Top_Attempts, Semifinal_Zones_Attempts, Final_Tops,
         Final_Zones, Final_Top_Attempts, Final_Zones_Attempts) %>%
  rename(Competition = Competition.Title, Date = Competition.Date)
```

The final step is to add one last column: *Winner_Contender*. This column will denote whether a given climber should be considered a contender for winning an IFSC bouldering world cup. Each climber will receive a 1 or 0. If a climber completes at least one zone in a final round of any competition, they are assigned a 1. If not, they receive a 0. This designation is handled by the following function.

```
win_con <- sapply(fin_zones, function(q){
  if(q >= 1){
    win_con = 1
  }
  else{
    win_con = 0
  }
  return(win_con)
```

```
})
```

Now, we can make the *Winner_Contender* column in the following way.

```
bouldering <- bouldering %>%
  mutate(Winner_Contender = win_con)
```

Finally, I will do a final quality check of the table (confirming dimensions, no NA values, and looking at the first six rows of the table).

```
dim(bouldering) #This should be 5498 x 23
```

```
## [1] 5498   23
```

```
any(is.na(bouldering))
```

```
## [1] FALSE
```

```
head(bouldering)
```

```
##                                          Competition                      Date
## 1 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 2 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 3 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 4 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 5 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
## 6 European Youth Championships (B) - Brixen (ITA) 2019 20  - 22 September 2019
##               Name Nation StartNr Rank Total_Tops Total_Zones
## 1    Naile Meignan    FRA      15    1         12          15
## 2      Miriam Fogu    ITA      27    2          9          16
## 3 Vanda Michalkova    SVK      48    3         11          13
## 4     Lola Sautier    FRA      17    4          7          12
## 5  Polina Kulagina    RUS      39    5          8          13
## 6    Kintana Iltis    FRA      16    6          9          12
##   Total_Attempts_to_Top Total_Attempts_to_Zone Qualification_Tops
## 1                    22                    23                  6
## 2                    15                    26                  4
## 3                    14                    15                  6
## 4                    17                    26                  4
## 5                    17                    24                  4
## 6                    18                    18                  6
##   Qualification_Zones Qualification_Top_Attempts Qualification_Zones_Attempts
## 1                   7                         9                           9
## 2                   8                         8                          14
## 3                   7                         8                           9
## 4                   6                         6                           9
## 5                   6                         6                          11
## 6                   7                         7                           7
##   Semifinal_Tops Semifinal_Zones Semifinal_Top_Attempts
## 1              3              4                     5
## 2              3              4                     5
## 3              3              3                     4
## 4              2              3                     8
## 5              3              4                     7
## 6              2              3                     8
##   Semifinal_Zones_Attempts Final_Tops Final_Zones Final_Top_Attempts
## 1                        5          3           4                  8
```

```
## 2                       5          2           4                   2
## 3                       3          2           3                   2
## 4                       9          1           3                   3
## 5                       6          1           3                   4
## 6                       8          1           2                   3
##   Final_Zones_Attempts Winner_Contender
## 1                    9                1
## 2                    7                1
## 3                    3                1
## 4                    8                1
## 5                    7                1
## 6                    3                1
```

2.2: Exploring the Bouldering Data Table

This section is divided into two parts: *Numeric Insights* and *Data Visualizations*. In the first part, I offer a brief overview of the *bouldering* data table and discuss efficiency rates for both tops and zones. In the latter section, I show and discuss a variety of plots displaying the relationships between particular climbers, nations, rounds, starting number, and climbing performance.

2.2.1: Some Numeric Insights

As a first step, let's look at some of the summary statistics for the columns of the bouldering data table.

```
bouldering %>% summary()
```

```
##  Competition            Date               Name              Nation
##  Length:5498        Length:5498        Length:5498        Length:5498
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##     StartNr          Rank          Total_Tops     Total_Zones
##  Min.   :  1   Min.   :  1.0   Min.   : 0.00   Min.   : 0.000
##  1st Qu.: 65   1st Qu.: 14.0   1st Qu.: 0.00   1st Qu.: 0.000
##  Median :142   Median : 28.0   Median : 1.00   Median : 3.000
##  Mean   :193   Mean   : 33.3   Mean   : 2.34   Mean   : 3.495
##  3rd Qu.:277   3rd Qu.: 47.0   3rd Qu.: 4.00   3rd Qu.: 6.000
##  Max.   :799   Max.   :125.0   Max.   :12.00   Max.   :16.000
##  Total_Attempts_to_Top Total_Attempts_to_Zone Qualification_Tops
##  Min.   : 0.000        Min.   : 0.000         Min.   :0.000
##  1st Qu.: 0.000        1st Qu.: 0.000         1st Qu.:0.000
##  Median : 2.000        Median : 5.000         Median :0.000
##  Mean   : 4.264        Mean   : 7.138         Mean   :1.895
##  3rd Qu.: 7.000        3rd Qu.:11.000         3rd Qu.:4.000
##  Max.   :43.000        Max.   :97.000         Max.   :8.000
##  Qualification_Zones Qualification_Top_Attempts Qualification_Zones_Attempts
##  Min.   :0.000       Min.   : 0.000             Min.   : 0.000
##  1st Qu.:0.000       1st Qu.: 0.000             1st Qu.: 0.000
##  Median :1.000       Median : 0.000             Median : 2.500
##  Mean   :2.635       Mean   : 3.244             Mean   : 5.077
##  3rd Qu.:5.000       3rd Qu.: 6.000             3rd Qu.: 9.000
##  Max.   :8.000       Max.   :22.000             Max.   :89.000
##  Semifinal_Tops   Semifinal_Zones  Semifinal_Top_Attempts
```

```
##  Min.   :0.0000   Min.   :0.0000   Min.   : 0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 0.0000
##  Median :0.0000   Median :0.0000   Median : 0.0000
##  Mean   :0.1835   Mean   :0.3992   Mean   : 0.4292
##  3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.: 0.0000
##  Max.   :4.0000   Max.   :4.0000   Max.   :18.0000
##  Semifinal_Zones_Attempts   Final_Tops      Final_Zones     Final_Top_Attempts
##  Min.   : 0.000           Min.   :0.0000   Min.   :0.0000   Min.   : 0.0000
##  1st Qu.: 0.000           1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 0.0000
##  Median : 0.000           Median :0.0000   Median :0.0000   Median : 0.0000
##  Mean   : 1.015           Mean   :0.2608   Mean   :0.4611   Mean   : 0.5899
##  3rd Qu.: 0.000           3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.: 0.0000
##  Max.   :67.000           Max.   :4.0000   Max.   :4.0000   Max.   :21.0000
##  Final_Zones_Attempts Winner_Contender
##  Min.   : 0.000       Min.   :0.0000
##  1st Qu.: 0.000       1st Qu.:0.0000
##  Median : 0.000       Median :0.0000
##  Mean   : 1.047       Mean   :0.1424
##  3rd Qu.: 0.000       3rd Qu.:0.0000
##  Max.   :27.000       Max.   :1.0000
```

Given that the mean number of tops is 2.34 and the median is 1 out of a possible 13, we can see that the vast majority of climbers do not top most of the problems. Similarly, though non-trivially higher, the typical climber does not reach the majority of zones either, withe mean and median being 3.495 and 3, respectively.

With the measures of center being quite low for both tops and zones, it follows that a good deal of climbers are separated in the ranks by their number of attempts. So, it may prove helpful to instead consider the ratio of average attempts to average successes for both tops and zones.

```
top_att_avg <- mean(bouldering$Total_Attempts_to_Top) / mean(bouldering$Total_Tops)
zone_att_avg <- mean(bouldering$Total_Attempts_to_Zone) / mean(bouldering$Total_Zones)
quali_top_att_avg <- mean(bouldering$Qualification_Top_Attempts) /
                          mean(bouldering$Qualification_Tops)
quali_zone_att_avg <- mean(bouldering$Qualification_Zones_Attempts) /
                           mean(bouldering$Qualification_Zones)
semi_top_att_avg <- mean(bouldering$Semifinal_Top_Attempts) /
                         mean(bouldering$Semifinal_Tops)
semi_zone_att_avg <- mean(bouldering$Semifinal_Zones_Attempts) /
                          mean(bouldering$Semifinal_Zones)
fin_top_att_avg <- mean(bouldering$Final_Top_Attempts) / mean(bouldering$Final_Tops)
fin_zone_att_avg <- mean(bouldering$Final_Zones_Attempts) / mean(bouldering$Final_Zones)

#Making a table to show the comparison
att_avg_table <- data.frame(Category = "Overall Attempts to Top",
                            Ratio = top_att_avg)
att_avg_table <- bind_rows(att_avg_table,
                           data.frame(Category = "Overall Attempts to Zone",
                                      Ratio = zone_att_avg))
att_avg_table <- bind_rows(att_avg_table,
                           data.frame(Category = "Qualification Attempts to Top",
                                      Ratio = quali_top_att_avg))
att_avg_table <- bind_rows(att_avg_table,
                           data.frame(Category = "Qualification Attempts to Zone",
                                      Ratio = quali_zone_att_avg))
att_avg_table <- bind_rows(att_avg_table,
```

```
                          data.frame(Category = "Semifinal Attempts to Top",
                                     Ratio = semi_top_att_avg))
att_avg_table <- bind_rows(att_avg_table,
                          data.frame(Category = "Semifinal Attempts to Zone",
                                     Ratio = semi_zone_att_avg))
att_avg_table <- bind_rows(att_avg_table,
                          data.frame(Category = "Final Attempts to Top",
                                     Ratio = fin_top_att_avg))
att_avg_table <- bind_rows(att_avg_table,
                          data.frame(Category = "Final Attempts to Zone",
                                     Ratio = fin_zone_att_avg))
att_avg_table %>% knitr::kable()
```

Category	Ratio
Overall Attempts to Top	1.822217
Overall Attempts to Zone	2.042306
Qualification Attempts to Top	1.711736
Qualification Attempts to Zone	1.926624
Semifinal Attempts to Top	2.338950
Semifinal Attempts to Zone	2.541230
Final Attempts to Top	2.261506
Final Attempts to Zone	2.271400

Reading the table, we can see that the qualification round is typically the easiest round and the semifinals are usually the hardest round. Interestingly, in the final round, the number of attempts needed to reach zone is very close (within 0.012) to the number of attempts needed to top a problem. This suggests that the impact of zones on placement in the final round is likely less than in previous rounds.

To get a better grasp on what sort of representation is covered by the bouldering table, let's look at how many distinct nations, athletes, and competitions are included in the table.

```
bouldering %>% summarize(Number_of_Countries = n_distinct(Nation),
                         Number_of_Athletes = n_distinct(Name),
                         Number_of_Competitions = n_distinct(Competition))
```

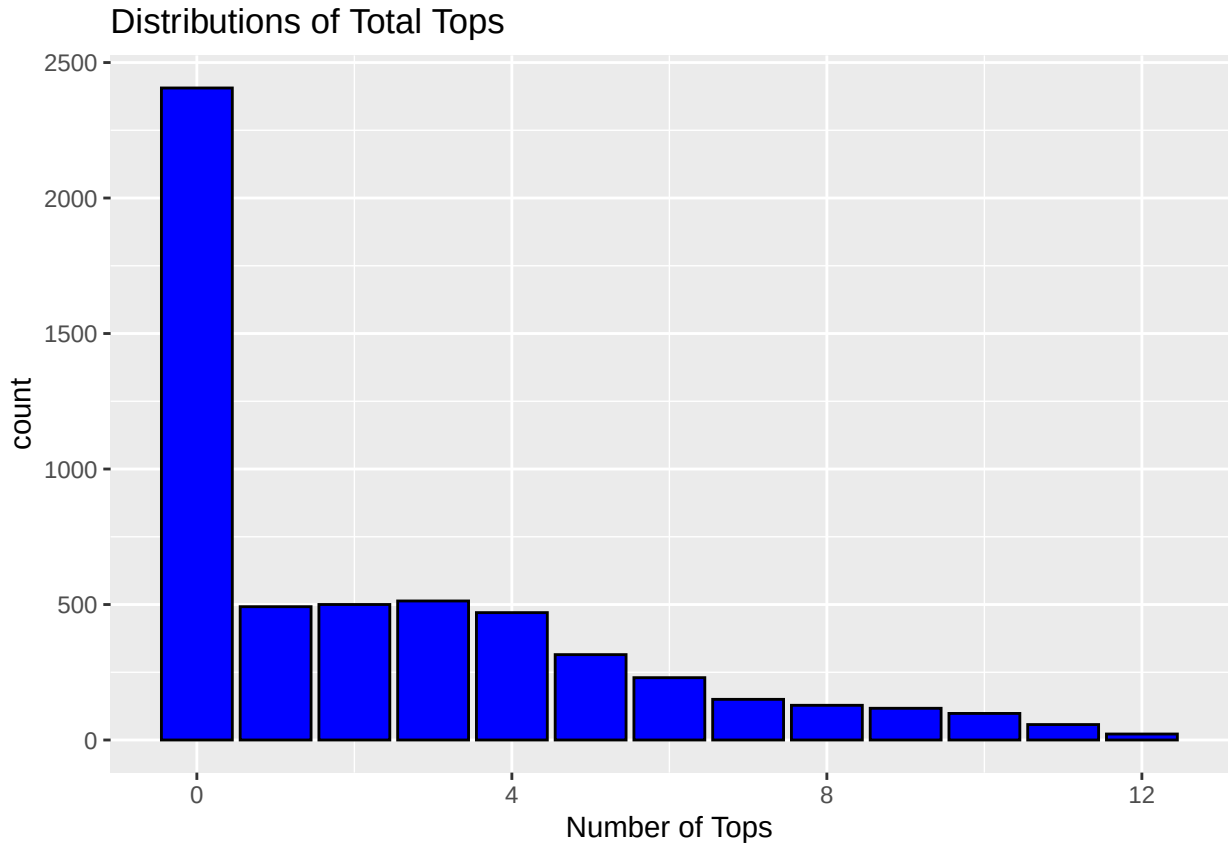
```
##   Number_of_Countries Number_of_Athletes Number_of_Competitions
## 1                  64                1518                     22
```

With only 1518 athletes and 64 countries competitions, it follows that, as one might expect, several athletes competed at multiple tournaments. We cannot yet say much about what affect nation-based or competition-specific biases may be at play. That said, much more can be learned by visualizing the data.

2.2.2: Data Visualizations

This first plot depicts the overall distribution of all tops achieved by each climber across all rounds at each competition.

```
plot1 <- bouldering %>%
  ggplot(aes(Total_Tops)) +
  geom_bar(color = "black", fill = "blue") +
  ggtitle("Distributions of Total Tops") +
  xlab("Number of Tops")

plot1
```



We can see that the vast majority of climbers did not top even a single problem and almost none topped all 12 problems. Still, this only gives a broad overview of the distribution. To gain greater specificity, consider:

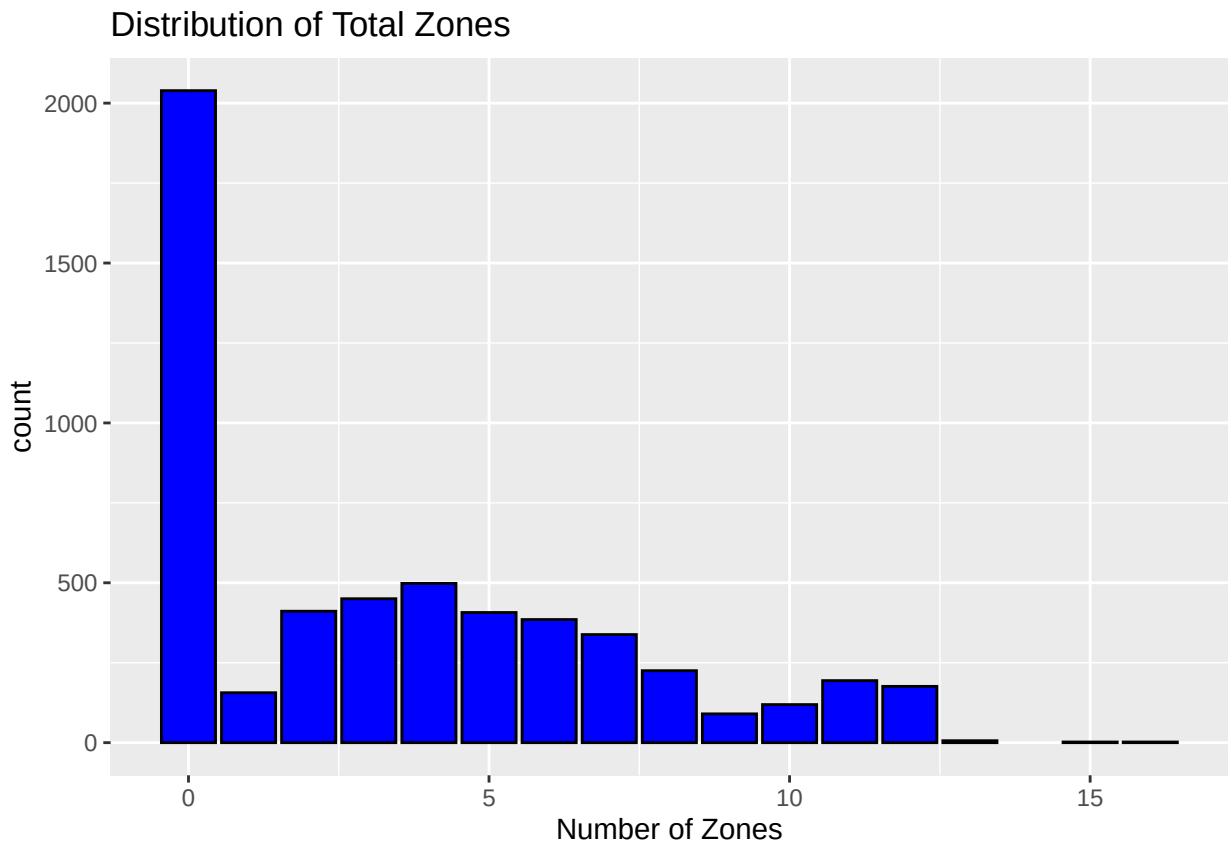
```
bouldering %>%
  group_by(Total_Tops) %>%
  summarize(prop_top = n()/nrow(bouldering)) %>%
  arrange(desc(prop_top))
```

```
## # A tibble: 13 x 2
##    Total_Tops prop_top
##         <dbl>    <dbl>
##  1          0  0.438
##  2          3  0.0933
##  3          2  0.0909
##  4          1  0.0895
##  5          4  0.0855
##  6          5  0.0573
##  7          6  0.0418
##  8          7  0.0273
##  9          8  0.0233
## 10          9  0.0213
## 11         10  0.0178
## 12         11  0.0104
## 13         12  0.00400
```

43.8% of climbers never topped a problem and only 0.4% topped all problems. We also observe clusters around 1 to 4 and 7 to 9 tops. This is to be expected since those completing 1 to 4 problems are likely those who made it to semifinals but not finals, and those scoring 7 to 9 tops being those who qualified for finals.

Now, let us see if a similar trend applies to the total zones.

```
plot2 <- bouldering %>%
  ggplot(aes(Total_Zones)) +
  geom_bar(color = "black", fill = "blue") +
  ggtitle("Distribution of Total Zones") +
  xlab("Number of Zones")

plot2
```




```
bouldering %>%
  group_by(Total_Zones) %>%
  summarize(prop_zone = n()/nrow(bouldering)) %>%
  arrange(desc(prop_zone))
```

```
## # A tibble: 16 x 2
##    Total_Zones prop_zone
##          <dbl>     <dbl>
##  1           0  0.371
##  2           4  0.0906
##  3           3  0.0818
##  4           2  0.0748
##  5           5  0.0740
##  6           6  0.0700
##  7           7  0.0615
##  8           8  0.0409
##  9          11  0.0353
## 10          12  0.0320
```

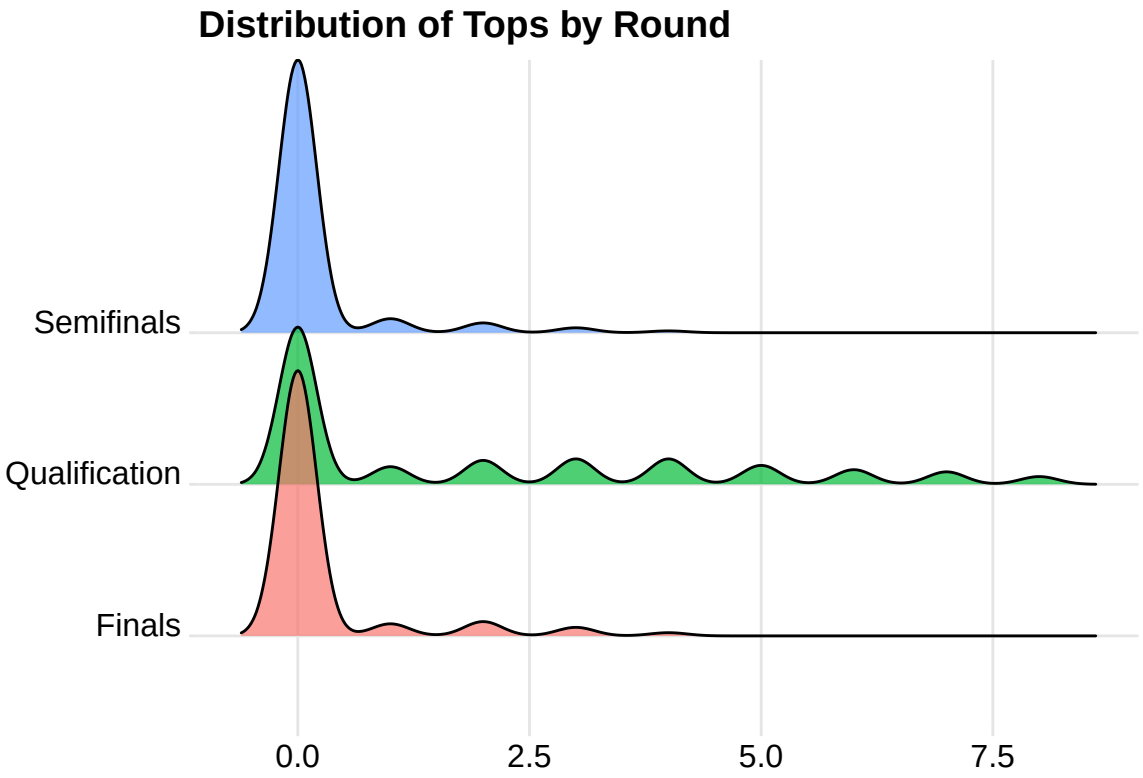
```
## 11           1  0.0284
## 12          10  0.0216
## 13           9  0.0164
## 14          13  0.00109
## 15          15  0.000364
## 16          16  0.000364
```

As with the distribution of total tops, 0 is the most common number of zones at 37.1% and 16 (the maximum number) being the least common at 0.0364%. This implies that the probability that a given climber has a perfect performance, meaning that all tops and zones are completed, are exceedingly low (approx. 0.0146%). Interestingly, 4 is the second most common number of zones at 9.06% and those with only a single top are less common than those with 0, 4, 3, 2, 5, 6, 7, 8, 11, or 12 zones. This suggests that reaching a high number of zones does not necessarily translate into a high number of tops. Furthermore, the second plot having a less consistently decreasing trend than the tops plot gives reason to think that the number of total tops and the number of total zones may have distinct effects on predictions made about the overall data set.

Now, instead of looking at overall performance, let us consider how the distribution of tops and zones change between rounds. To do this, I will begin by creating a new table that includes indicators for which round a given climber's performance belongs. Following that, I will display the comparative distributions using a ridge plot.

```
tops_by_round <- data.frame(round = c(rep("Qualification", 5498),
                                      rep("Semifinals", 5498),
                                      rep("Finals", 5498)),
                            values = c(quali_tops, semi_tops, fin_tops))

r_top_comp_plot <- tops_by_round %>%
  ggplot(aes(x = values, y = round, fill = round)) +
  geom_density_ridges(alpha = 0.7) +
  theme_ridges() +
  theme(legend.position = "none") +
  ylab("") +
  xlab("") +
  ggtitle("Distribution of Tops by Round")

r_top_comp_plot
```

```
## Picking joint bandwidth of 0.203
```

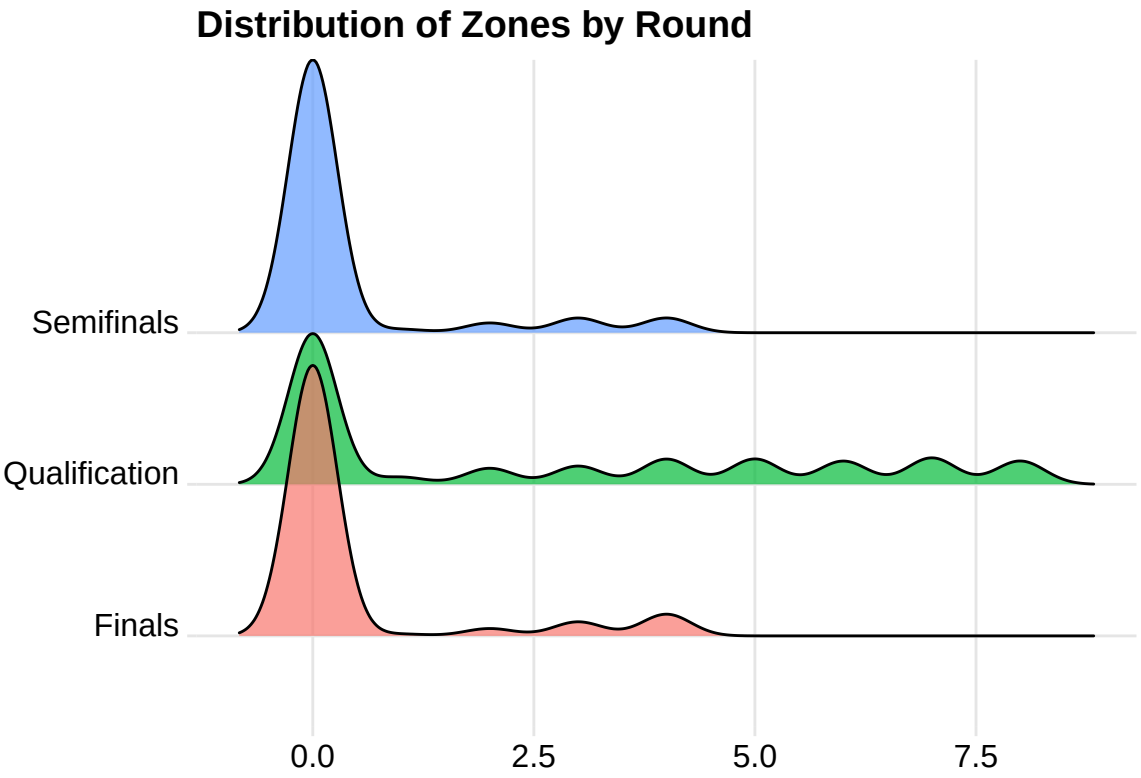


Though the trends for each round appear similar, there are some noteworthy differences. The Qualification round has the greatest number of climbers completing the majority of the problems and the semifinals appears to have the least. Moreover, the semifinals, in comparison to finals, has a greater height at 0 tops and more shallow peaks for all other values, thereby strengthening the earlier suggestion that the semifinals is usually the most difficult round to earn tops in.

It is important to see if a similar comparison holds for zones as well. To check this, consider:

```
zones_by_round <- data.frame(round = c(rep("Qualification", 5498),
                                       rep("Semifinals", 5498),
                                       rep("Finals", 5498)),
                             values = c(quali_zones, semi_zones, fin_zones))
r_zone_comp_plot <- zones_by_round %>%
  ggplot(aes(x = values, y = round, fill = round)) +
  geom_density_ridges(alpha = 0.7) +
  theme_ridges() +
  theme(legend.position = "none") +
  ylab("") +
  xlab("") +
  ggtitle("Distribution of Zones by Round")

r_zone_comp_plot
```

```
## Picking joint bandwidth of 0.277
```



The trends for zones per round do appear quite similar, though not identical, to those observed about tops. Given this, little else is likely to be learned from further examination of trends regarding average zones and tops in isolation of other factors.

Not all countries have equal access to climbing facilities, natural formations, or financing for national-level teams. As such, there may well be nation-specific biases. To see this, we can look ath the average number of tops and zones are for each country.

```
#Average Tops
bouldering %>%
  group_by(Nation) %>%
  summarize(avg_tops = mean(Total_Tops)) %>%
  arrange(desc(avg_tops))
```

```
## # A tibble: 64 x 2
##    Nation avg_tops
##    <chr>     <dbl>
##  1 FRA        4.89
##  2 SLO        4.06
##  3 AUT        4.02
##  4 ITA        3.65
##  5 BUL        3.56
##  6 LUX        3.38
##  7 GER        3.12
##  8 NED        3.01
##  9 ESP        2.83
## 10 BEL        2.76
## # ... with 54 more rows
```

```
#Average Zones
bouldering %>%
```

```
  group_by(Nation) %>%
  summarize(avg_zones = mean(Total_Zones)) %>%
  arrange(desc(avg_zones))
```

```
## # A tibble: 64 x 2
##    Nation avg_zones
##    <chr>      <dbl>
##  1 FRA         6.47
##  2 SLO         5.53
##  3 AUT         5.52
##  4 BUL         5.12
##  5 ITA         5.09
##  6 MKD         5
##  7 NED         4.79
##  8 GER         4.45
##  9 LUX         4.23
## 10 PHI         4.19
## # ... with 54 more rows
```

We see that French, Slovenian, and Australian athletes are the only groups who averaged 4 or more tops per competition. Notably, France, the nation withe highest average tops, has a significant lead of approximately 0.83 over Slovenia, the second highest. This same trend and ordering holds for zones, where France again displays a significant advantage over even their closest rival.

It is also worth noting that the number of climbers from each country is not equal, as seen by...

```
bouldering %>%
  group_by(Nation) %>%
  summarize(number_of_athletes = n()) %>%
  arrange(desc(number_of_athletes))
```

```
## # A tibble: 64 x 2
##    Nation number_of_athletes
##    <chr>               <int>
##  1 FRA                   323
##  2 ITA                   306
##  3 AUT                   294
##  4 GER                   275
##  5 GBR                   238
##  6 BEL                   237
##  7 CZE                   221
##  8 JPN                   200
##  9 RUS                   200
## 10 SLO                   199
## # ... with 54 more rows
```

The average may not be the most appropriate measure of center for this data set. Perhaps the median is better suited since it is less affected by outliers, such as prodigies or filler members. To see how the two measures of center compare, consider:

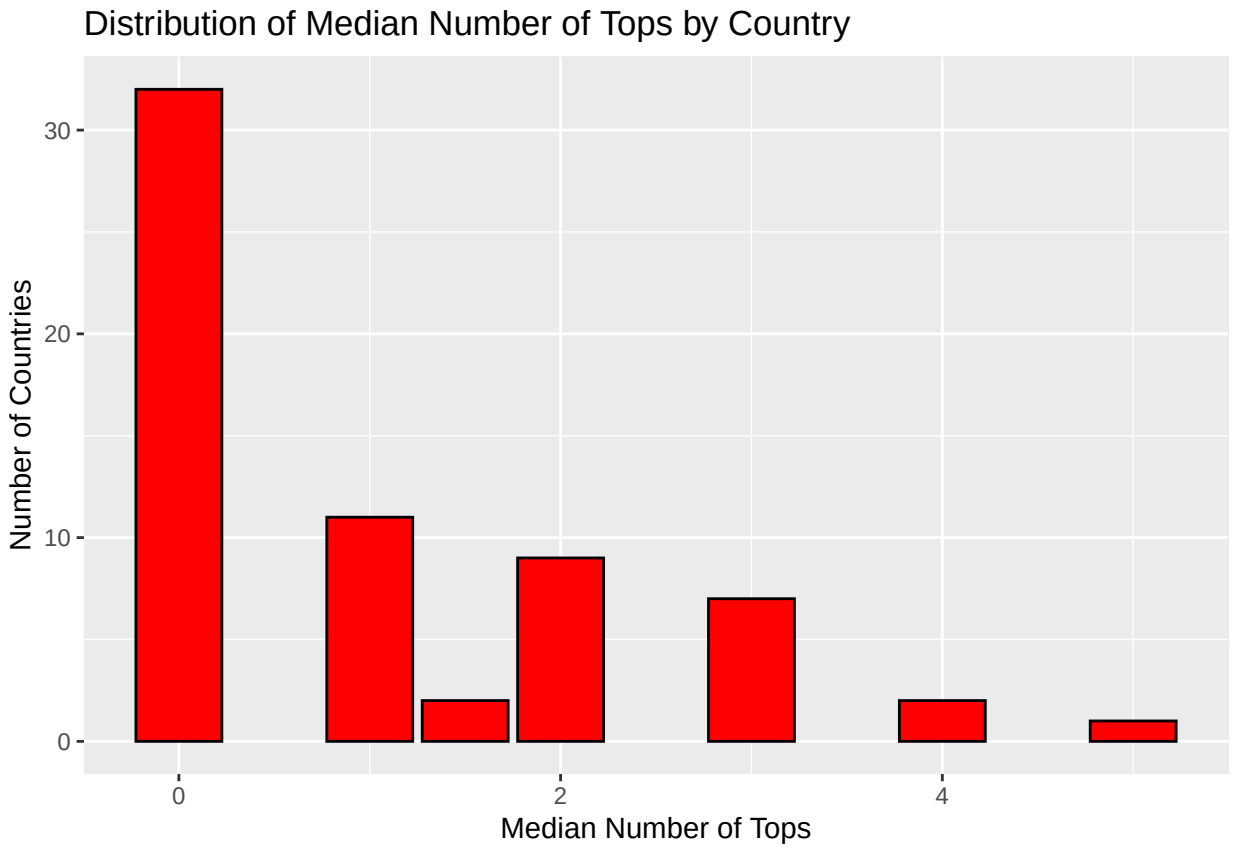
```
mean_med_comp_table <- bouldering %>%
  group_by(Nation) %>%
  summarize(avg_tops = mean(Total_Tops),
            avg_zones = mean(Total_Zones),
            med_tops = median(Total_Tops),
            med_zones = median(Total_Zones)) %>%
```

```
  arrange(desc(avg_tops))

head(mean_med_comp_table)
```

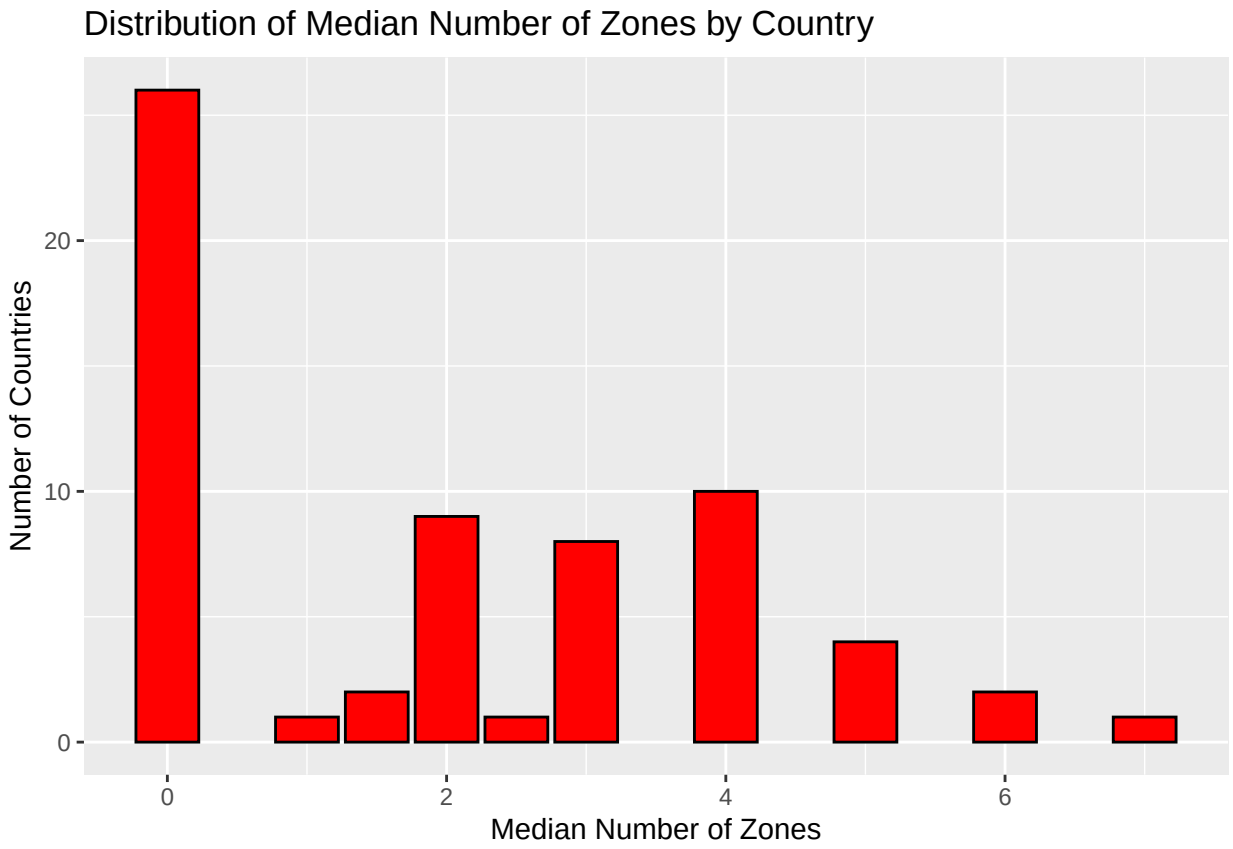
```
## # A tibble: 6 x 5
##   Nation avg_tops avg_zones med_tops med_zones
##   <chr>     <dbl>     <dbl>    <dbl>     <dbl>
## 1 FRA        4.89      6.47        5         7
## 2 SLO        4.06      5.53        4         6
## 3 AUT        4.02      5.52        4         6
## 4 ITA        3.65      5.09        3         5
## 5 BUL        3.56      5.12        3         4
## 6 LUX        3.38      4.23        2         5
```

The means and medians do seem to tell somewhat varying stories, as neither provides a consistently higher or lower metric than the other. This might be explained by the presence of climbers who perform markedly better than their compatriots in some nations. If this is the case, the median would be expected to fall below the mean, as the exceptionally well-performing climber would be given equal weight to their worst performing teammate. Still, we can gain better insight into the distributions of the medians through the following plots.

```
#Median Tops
plot3 <- mean_med_comp_table %>%
  ggplot(aes(med_tops)) +
  geom_bar(color = "black", fill = "red") +
  ggtitle("Distribution of Median Number of Tops by Country") +
  labs(x = "Median Number of Tops", y = "Number of Countries")

plot3
```



```
#Median Zones
plot4 <- mean_med_comp_table %>%
  ggplot(aes(med_zones)) +
  geom_bar(color = "black", fill = "red") +
  ggtitle("Distribution of Median Number of Zones by Country") +
  labs(x = "Median Number of Zones", y = "Number of Countries")

plot4
```



One final way of measuring each country's typical performance is efficiency, defined as *mean(total_tops)/ mean(total_attempts_to_top)* and *mean(total_zones)/mean(total_attempts_to_zone)*. This will give some insight into how often each nation's athletes succeeded in getting a zone or top.

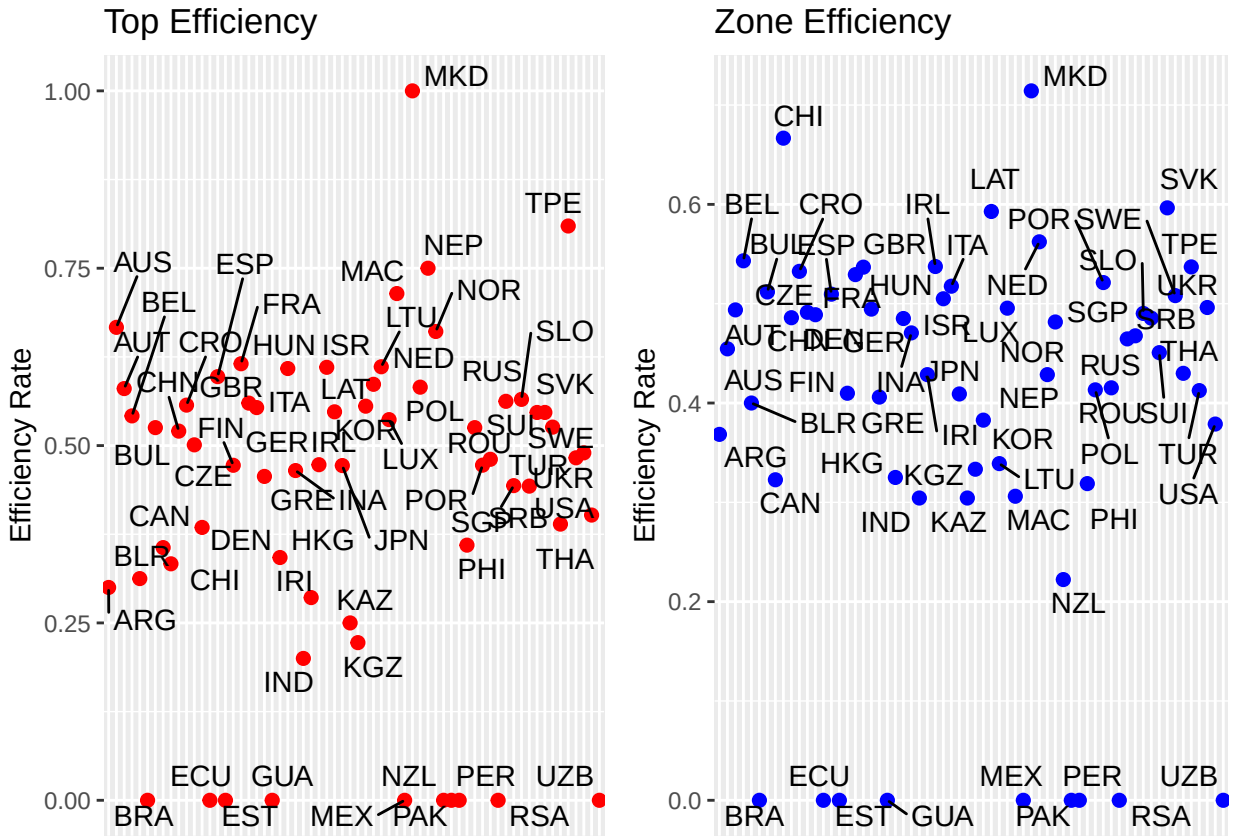
```
#Setting up a new table for efficiency rates
efficiency_table <- bouldering %>%
  group_by(Nation) %>%
  summarize(top_eff = mean(Total_Tops) / mean(Total_Attempts_to_Top),
            zone_eff = mean(Total_Zones) / mean(Total_Attempts_to_Zone))

#There are NaN where no one topped or got a zone since that causes a 0/0 issue.
#So, I'll replace the NaN values with a 0.
efficiency_table$top_eff[which(is.nan(efficiency_table$top_eff))] <- 0
efficiency_table$zone_eff[which(is.nan(efficiency_table$zone_eff))] <- 0

#Sorting the efficiency table in terms of highest to lowest efficiency at getting tops.
efficiency_table <- efficiency_table %>%
  arrange(desc(top_eff))
```

```
head(efficiency_table)
```

```
## # A tibble: 6 x 3
##   Nation top_eff zone_eff
##   <chr>    <dbl>    <dbl>
## 1 MKD      1        0.714
## 2 TPE      0.810    0.537
## 3 NEP      0.75     0.429
## 4 MAC      0.714    0.306
## 5 AUS      0.667    0.455
## 6 NOR      0.661    0.482
```

This shows that North Macedonia (MKD), Taiwan (TPE), Nepal (NEP), and Macao (MAC) all are more efficient than France, Norway, Austria, and other countries that placed much higher in terms of mean and median numbers of tops and zones. This is likely due to the current efficiency_table not considering how many tops or zones a country typically receives. To see the whole distributions, consider the following plots:

```
plot5 <- efficiency_table %>%
  ggplot(aes(Nation, top_eff, label = Nation)) +
  geom_point(size = 2, color = "red") +
  ggtitle("Top Efficiency") +
  ylab("Efficiency Rate") +
  geom_text_repel(max.overlaps = 20) +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank())

plot6 <- efficiency_table %>%
  ggplot(aes(Nation, zone_eff, label = Nation)) +
  geom_point(size = 2, color = "blue") +
  ggtitle("Zone Efficiency") +
  ylab("Efficiency Rate") +
  geom_text_repel(max.overlaps = 20) +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank())

grid.arrange(plot5, plot6, ncol = 2, nrow = 1)
```

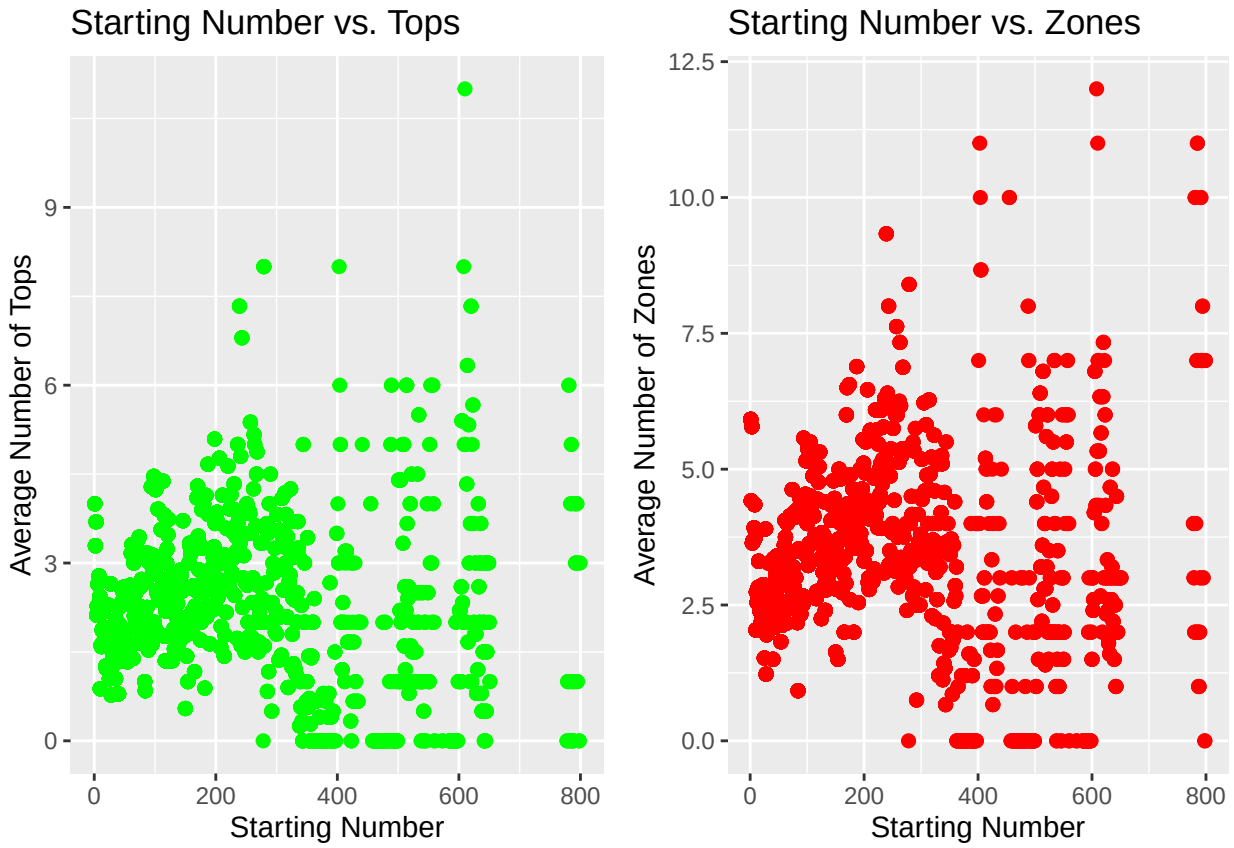


With a small number of exceptions (Brazil, Ecuador, Guatemala, Pakistan, Uzbekistan, Mexico, Peru, South Africa, Estonia, and North Macedonia), the efficiency rates of most countries for both tops and zones is between 0.3 and 0.6. Notably, the countries that had the highet mean and median tops and zones were not the most efficient countries. In fact, France, Austria, and Slovenia all landed around the middle for both efficiency plots. This does not mean that efficiency cannot be used in predictions but it does seem less helpful in terms of predicting whether a given climber actually has a reasonable chance of winning a competition.

Given the number of climbers at any given competition far exceeds the number of problems in any round, there is (up to) a several hour gap between when the first athletes starts climbing and the final athlete begins. This, alongside potential fatigue/stress from waiting, makes an athlete's starting number a potential source of bias. To see whether this is the case, consider the following two plots:

```
plot7 <- bouldering %>%
  group_by(StartNr) %>%
  mutate(avg_tops = mean(Total_Tops)) %>%
  ggplot(aes(StartNr, avg_tops)) +
  geom_point(size = 2, color = "green") +
  ggtitle("Starting Number vs. Tops") +
  labs(x = "Starting Number", y ="Average Number of Tops")

#Let's now see if the same holds true for zones.
plot8 <- bouldering %>%
  group_by(StartNr) %>%
  mutate(avg_zones = mean(Total_Zones)) %>%
  ggplot(aes(StartNr, avg_zones)) +
  geom_point(size = 2, color = "red") +
  ggtitle("Starting Number vs. Zones") +
  labs(x = "Starting Number", y = "Average Number of Zones")
```

```
#Showing plots 7 and 8 side by side
grid.arrange(plot7, plot8, ncol = 2, nrow = 1)
```



The shapes of the plots are similar but not identical. In both, the best performing athlete had a starting number of approximately 600. For tops, this best performer is fairly isolated from all others. For zones, there are several other athletes who performed somewhat comparably who started anywhere from approximately 400th to 800th. Overall, however, it does seem that those who climbed at the very beginning did worse than those who climbed later on.

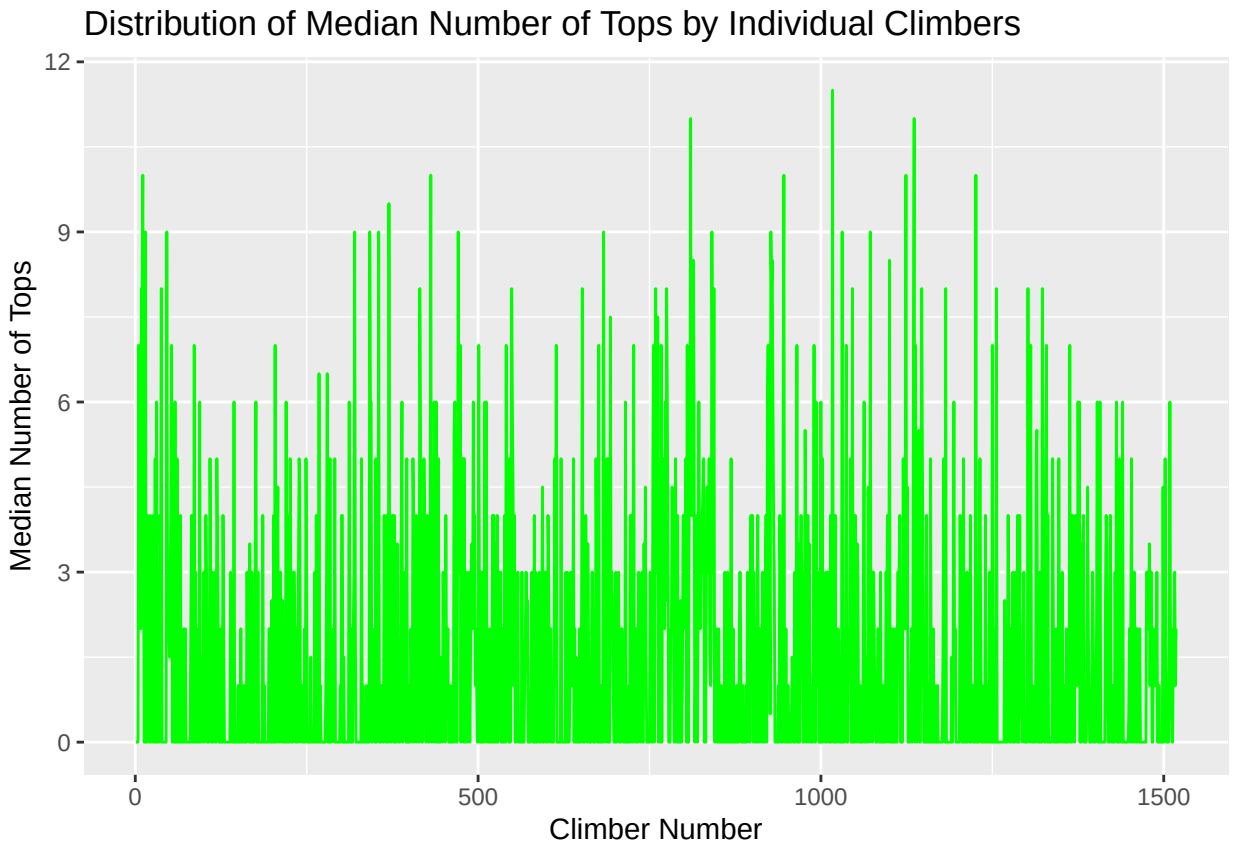
Since some climbers competed at multiple events and are generally more skilled than others, it is worthwhile to see how much of an effect individual climbers had on the distribution of tops and zones. Since the number of tops is discrete and there are outliers, I have chosen to use the median instead of the mean for the following plots.

```
#Individual Climbers vs. median Tops and Zones
indiv_meds <- bouldering %>%
  group_by(Name) %>%
  summarize(med_tops = median(Total_Tops),
            med_zones = median(Total_Zones)) %>%
  arrange(Name)

#Computing summary statistics about the above table.
summary(indiv_meds)
```

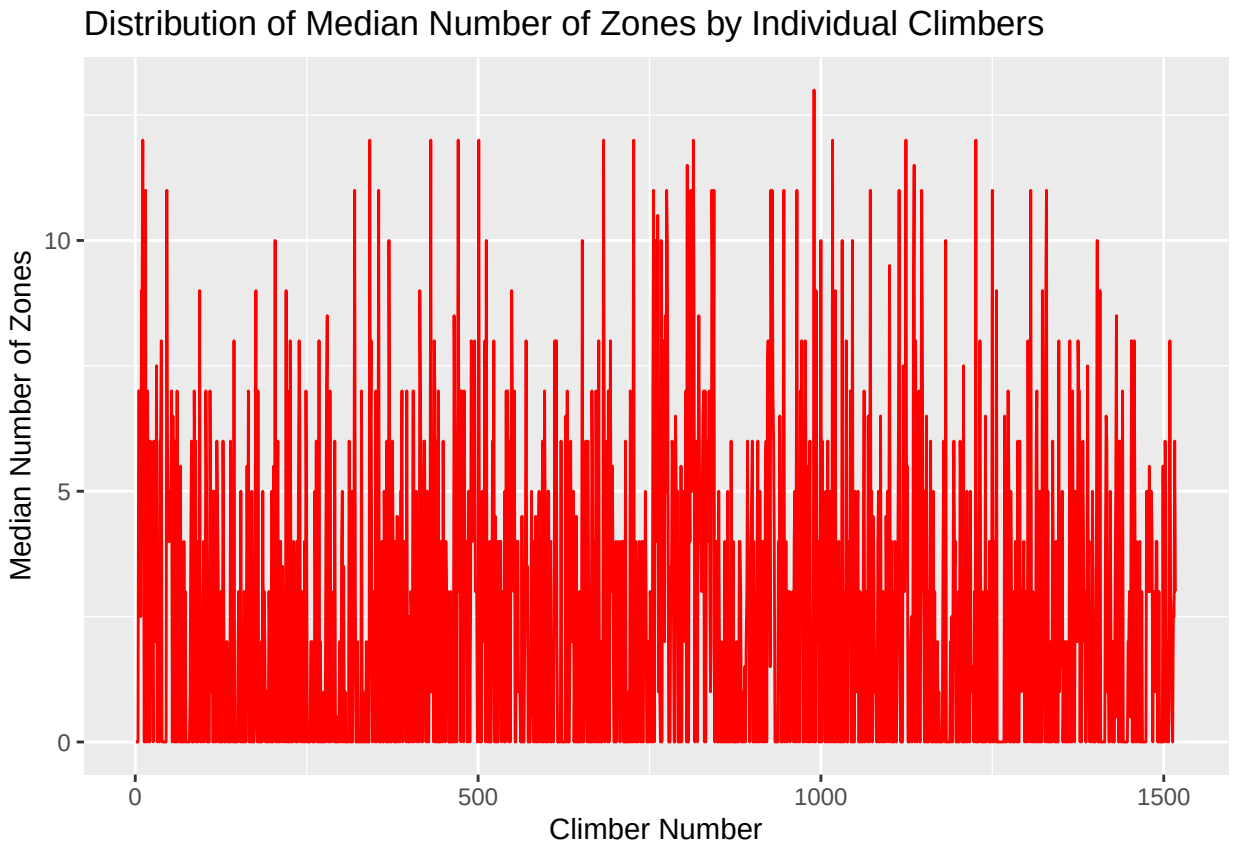
```
##      Name              med_tops        med_zones
##  Length:1518        Min.   : 0.000   Min.   : 0.000
##  Class :character   1st Qu.: 0.000   1st Qu.: 0.000
##  Mode  :character   Median : 0.000   Median : 2.000
```

```
##                     Mean   : 1.585   Mean   : 2.538
##                     3rd Qu.: 3.000   3rd Qu.: 4.000
##                     Max.   :11.500   Max.   :13.000
```

```r
#Adding a numerical ordering to the climbers in the indiv_avgs table
indiv_meds <- indiv_meds %>%
  mutate(number = c(1:nrow(indiv_meds)))

#Plotting the two tables' data side by side (tops first, zones second).
plot9 <- indiv_meds %>%
  arrange(desc(med_tops)) %>%
  ggplot(aes(number, med_tops)) +
  geom_line(color = "green") +
  ggtitle("Distribution of Median Number of Tops by Individual Climbers") +
  labs(x = "Climber Number", y = "Median Number of Tops")

plot9
```




```r
plot10 <- indiv_meds %>%
  arrange(desc(med_zones)) %>%
  ggplot(aes(number, med_zones)) +
  geom_line(color = "red") +
  ggtitle("Distribution of Median Number of Zones by Individual Climbers") +
  labs(x = "Climber Number", y = "Median Number of Zones")

plot10
```

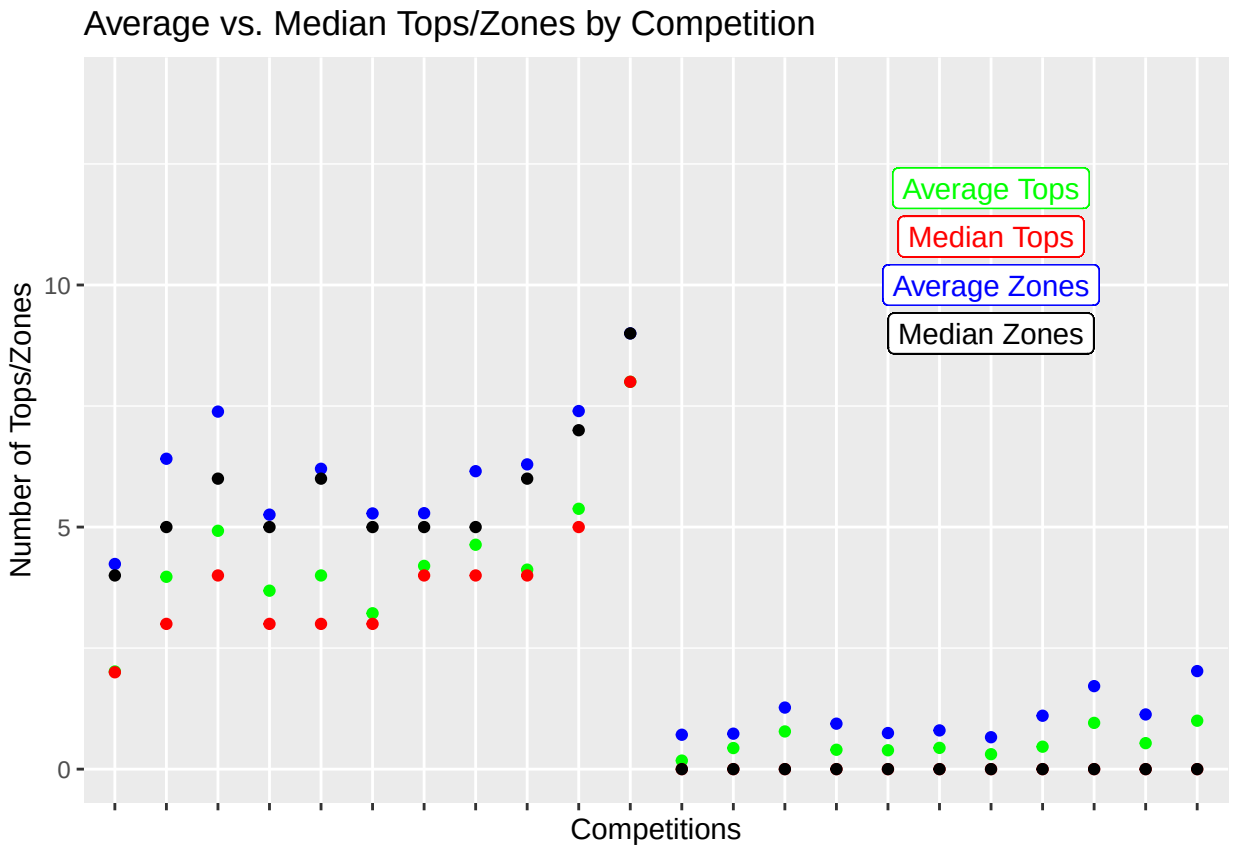


Though the actual median was nearly uniformly higher for zones than tops, the actual shapes of the plots are very similar to one another. Those who completed more zones also tended to complete more tops, though the discrepancy in the heights of the plots does imply that not all zones, even amongst the better performing athletes, were successfully converted into tops.

Before discussing my approach to modeling the data, I will examine one final potential source of bias: the timing of the competitions. Since no two competitions have overlapping dates, this is, in this case, the same as examining whether particular events (i.e. locations) had an effect on climbers' performances.

```
comp_spec_table <- bouldering %>%
  group_by(Competition) %>%
  summarize(avg_tops = mean(Total_Tops),
            avg_zones = mean(Total_Zones),
            med_tops = median(Total_Tops),
            med_zones = median(Total_Zones))

#Visually comparing average tops, median tops, average zones, and median zones by competition
plot11 <- comp_spec_table %>%
  ggplot(aes(group = Competition)) +
  geom_point(aes(Competition, avg_tops), color = "green") +
  geom_point(aes(Competition, med_tops), color = "red") +
  geom_point(aes(Competition, avg_zones), color = "blue") +
  geom_point(aes(Competition, med_zones), color = "black") +
  ylim(0, 14) +
  labs(x = "Competitions", y = "Number of Tops/Zones") +
  ggtitle("Average vs. Median Tops/Zones by Competition") +
  geom_label(aes(x = 18, y = 12, label = "Average Tops"), color = "green") +
  geom_label(aes(x = 18, y = 11, label = "Median Tops"), color = "red") +
```

```
  geom_label(aes(x = 18, y = 10, label = "Average Zones"), color = "blue") +
  geom_label(aes(x = 18, y = 9, label = "Median Zones"), color = "black") +
  theme(axis.text.x=element_blank())

plot11
```



This plot[4] clearly shows that the typical performance of the athletes, regardless of whether one considers mean or median as the measure of center, is not even close to being uniform across competitions. The first half of the tournaments have notably higher mean and median numbers of tops and zones than any of those in the latter half, though this might be explained by some of the competitions being at the junior level (under 19 years of age) and others being at the senior level (17 years or older)[5].

2.3: Modeling Approach

This section will outline my general approach to modeling the data as well as show the code involved in making the discussed models. While I display the results of each model here, I will reserve the discussion of said results until the next section.

2.3.1: Overview of the Modeling Process

Given that the primary goal of these models is to predict whether a climber is an actual contender to win an IFSC competition, all of the models used here will be classification models based around the *Winner_Contender* column in the *bouldering* table.

[4]For some of the competitions, the median tops looks like they are missing. They are not; since the median was 0, they are overlapping the median number of zones and are therefore obscured.

[5]Transitioning to senior level world cups becomes an option when someone turns 17 but is not mandatory until one turns 19.

To that end, a training set, to which 80% of the data will be allocated, and a testing set, to which the remaining 20% will be assigned, will be defined. I have selected this split in the data for two reasons:

1. While assigning more data to the training set would likely improve the performance of the models, doing so would leave too little data for the testing set. Since the whole data set is only approximately 5500 rows and given the prevalence of athletes with 0 tops and 0 zones, I had concerns that a testing set composed of only approximately 550 data points might lack non-zero values.
2. Given this concern, it was also tempting to assign more data to the testing set, i.e., a 70/30 split instead. As was seen in Section 2.2, many of the predicting factors have relatively small differences between many of the data points. Were too little of the bouldering data assigned to the training set, these more subtle differnces might get disregarded, thereby weaking the predictions.

I have chosen a 80/20 split as a compromise between these two concerns. Relatedly, while I am ultimately using *accuracy* as the metric of success for a model, I am also taking account of *specificity* and *sensitivity* in order to differentiate models that yield highly similar degrees of accuracy.

Importantly, at no point should any model attempt to predict zone-performance based on tops. While zones are a requirement for tops, the reverse is not true. As such, in any practical context, no model could predict information about zones based on tops prior to the athletes attempting the zones that are meant to be predicted. Relatedly, I am not going to include all of the possible predictors from the bouldering table in the models. In particular, the following have be excluded:

1. *Name*: Given the similar trends between zones and tops, little additional information is gained by its inclusion.
2. *Rank*: This refers to the climbers' ranks post-competition and so, does not inform on how a climber will perform at said competition.
3. *Date*: This is functionally identify to *Competition* and is therefore redundant.
4. *Nation*: While there was a difference in the performances of nations, the data gleaned from this column is too susceptible to being influenced by outliers[6]
5. *Competition*: This is being removed for two reasons: (1) Only 22 competitions are included in the data table and given the 1500+ competitors, this might lead to over-grouping of the data, and (2) I have insufficient information to determine whether the variation observed in the relevant plot is due to climate, location, or divisional (junior vs. senior competition) reasons.

As such, the bouldering has been modified in the following way:

```
bouldering <- bouldering %>%
  select(Winner_Contender, StartNr, Total_Tops, Total_Zones, Total_Attempts_to_Top,
         Total_Attempts_to_Zone, Qualification_Tops, Qualification_Zones,
         Qualification_Top_Attempts,Qualification_Zones_Attempts, Semifinal_Tops,
         Semifinal_Zones, Semifinal_Top_Attempts, Semifinal_Zones_Attempts, Final_Tops,
         Final_Zones, Final_Top_Attempts, Final_Zones_Attempts)

#Before Making any actual models, I am going to change the Winner_Contender column into factors.
bouldering$Winner_Contender <- as.factor(bouldering$Winner_Contender)
```

I employ here *K-Nearest Neighbors (KNN)*, *Random Forest (rf)*, and *Logistic Regression (glm)* models. I have included two sets of models. The first is based on the bouldering table as described above. As will be shown, this yields multiple models which appear to make perfectly accurate predictions. This is explained by information that is unrealistically informative being included in the present version of the *bouldering* table. Though these models are therefore unrealistic, I have chosen to include them here in order to help illustrate the development process for the final model.

In Section 2.3.4, I will create KNN, rf, and glm models based on a version of the *bouldering* table that excludes all of the columns that involve information from the final round. By refining the bouldering table in

[6]There are outlier climbers in IFSC competitions. For example, Janja Garnbret, a Slovenian climber, wins over 80% of the competitions she participates in and there are some who have competed over 100 times without ever qualifying for semifinals.

this way, we create a more realistic data set for predicting whether a climber is a real contender for winning prior to the final round actually happening.

2.3.2: Making the Testing and Training Sets

```
index <- createDataPartition(bouldering$Winner_Contender, times = 1, p = 0.8, list = FALSE)
boulder_train <- bouldering %>% slice(index)
boulder_test <- bouldering %>% slice(-index)
```

I will just perform a quick quality check before moving on to create the first model.

```
dim(boulder_train) #should be 4399 x 20
```

```
## [1] 4399   18
```

```
dim(boulder_test) #should be 1099 x 20
```

```
## [1] 1099   18
```

```
any(is.na(boulder_train))
```

```
## [1] FALSE
```

```
any(is.na(boulder_test))
```

```
## [1] FALSE
```

The training and testing sets have been successfully created. Note that there are presently 19 potential predictors for the *Winner_Contender* column.

2.3.3: Unrealistically Accurate Models

Included in this section are three models that rely on the present version of the *bouldering* table, which, crucially, includes the following columns:

1. *Total_Tops*
2. *Total_Zones*
3. *Total_Attempts_to_Top*
4. *Total_Attempts_to_Zone*
5. *Final_Tops*
6. *Final_Zones*
7. *Final_Top_Attempts*
8. *Final_Zones_Attempts*

As will become apparent, their inclusion makes the models predict with extremely high and sometime perfect accuracy. This, however, is due to the above mentioned columns being based on information that could not be known until after a competition is finished, i.e., not until the predicted outcome would already be known definitively. Nonetheless, it is helpful to see how their inclusion affects the models, hence their inclusion here.

I begin with the KNN model.

```
#Model 1
#Making the model itself
knn_fit <- train(Winner_Contender ~ ., method = "knn", data = boulder_train)

#Making the predictions
y_hat_knn <- predict(knn_fit, boulder_test, type = "raw")

#Making the confusion matrix.
knn_cf_mat <- confusionMatrix(y_hat_knn, boulder_test$Winner_Contender)
```

While I will not discuss the results of the models until Section 3, I will now create a table that tracks the results of the models.

```
model_comparison <- data.frame(Model = "K Nearest Neighbors (KNN)",
                               Accuracy = knn_cf_mat$overall["Accuracy"],
                               Sensitivity = knn_cf_mat$byClass["Sensitivity"],
                               Specificity = knn_cf_mat$byClass["Specificity"])
```

Since the processes for making the models in this section are quite similar to one another, I will minimize the annotation within the code, noting only significant deviations. With that said, here is the random forest model.

```
#Model 2
rf_fit <- train(Winner_Contender ~ ., method = "rf", data = boulder_train)
y_hat_rf <- predict(rf_fit, boulder_test, type = "raw")
rf_cf_mat <- confusionMatrix(y_hat_rf, boulder_test$Winner_Contender)

#Adding model 2 to the table.
model_comparison <- bind_rows(model_comparison,
                              data.frame(Model = "Random Forest (rf)",
                                         Accuracy = rf_cf_mat$overall["Accuracy"],
                                         Sensitivity = rf_cf_mat$byClass["Sensitivity"],
                                         Specificity = rf_cf_mat$byClass["Specificity"]))
```

Finally, I will create a logistic regression model, though due to a large volume of warnings associated with using said model, will ultimately favor the random forest model over it.

```
#Model 3
glm_fit <- train(Winner_Contender ~ ., method = "glm", data = boulder_train)
y_hat_glm <- predict(glm_fit, boulder_test, type = "raw")
glm_cf_mat <- confusionMatrix(y_hat_glm, boulder_test$Winner_Contender)

#Adding it to the table
model_comparison <- bind_rows(model_comparison,
                              data.frame(Model = "Logistic Regression (glm)",
                                         Accuracy = glm_cf_mat$overall["Accuracy"],
                                         Sensitivity = glm_cf_mat$byClass["Sensitivity"],
                                         Specificity = glm_cf_mat$byClass["Specificity"]))
```

2.3.4: Realistic Models

Given the minimal practical use for a prediction about who might win when the winner is already known, the models in the prior section are in an important sense unrealistic. To make the models and as a result, their predictive power, more realistic, I will first restrict the *bouldering* table by removing all of the columns listed in the previous section.

```
bouldering2 <- bouldering %>%
  select(Winner_Contender, StartNr, Qualification_Tops, Qualification_Zones,
         Qualification_Top_Attempts, Qualification_Zones_Attempts, Semifinal_Tops,
         Semifinal_Zones, Semifinal_Top_Attempts, Semifinal_Zones_Attempts)
```

Since I have restricted the bouldering table and redefined it as *bouldering2*, I also have to recreate the training and testing sets. As before, I will use a 80/20 split.

```
index <- createDataPartition(bouldering2$Winner_Contender, times = 1, p = 0.8, list = FALSE)
boulder_train2 <- bouldering2 %>% slice(index)
boulder_test2 <- bouldering2 %>% slice(-index)
```

Now, the models can be created in a very similar manner to those made in the previous section.

```
#KNN
knn_fit2 <- train(Winner_Contender ~ ., method = "knn", data = boulder_train2)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
y_hat_knn2 <- predict(knn_fit2, boulder_test2, type = "raw")
knn_cf_mat2 <- confusionMatrix(y_hat_knn2, boulder_test2$Winner_Contender)

#Random Forests
rf_fit2 <- train(Winner_Contender ~ ., method = "rf", data = boulder_train2)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
y_hat_rf2 <- predict(rf_fit2, boulder_test2, type = "raw")
rf_cf_mat2 <- confusionMatrix(y_hat_rf2, boulder_test2$Winner_Contender)

#Logistic Regression
glm_fit2 <- train(Winner_Contender ~ ., method = "glm", data = boulder_train2)
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
y_hat_glm2 <- predict(glm_fit2, boulder_test2, type = "raw")
glm_cf_mat2 <- confusionMatrix(y_hat_glm2, boulder_test2$Winner_Contender)

#Making a table to compare these three new models
model_comparison2 <- data.frame(Model = "K Nearest Neighbors (KNN)",
                                Accuracy = knn_cf_mat2$overall["Accuracy"],
                                Sensitivity = knn_cf_mat2$byClass["Sensitivity"],
                                Specificity = knn_cf_mat2$byClass["Specificity"])
model_comparison2 <- bind_rows(model_comparison2,
                               data.frame(Model = "Random Forest (rf)",
                                          Accuracy = rf_cf_mat2$overall["Accuracy"],
                                          Sensitivity = rf_cf_mat2$byClass["Sensitivity"],
                                          Specificity = rf_cf_mat2$byClass["Specificity"]))
model_comparison2 <- bind_rows(model_comparison2,
                               data.frame(Model = "Logistic Regression (glm)",
                                          Accuracy = glm_cf_mat2$overall["Accuracy"],
                                          Sensitivity = glm_cf_mat2$byClass["Sensitivity"],
                                          Specificity = glm_cf_mat2$byClass["Specificity"]))
```

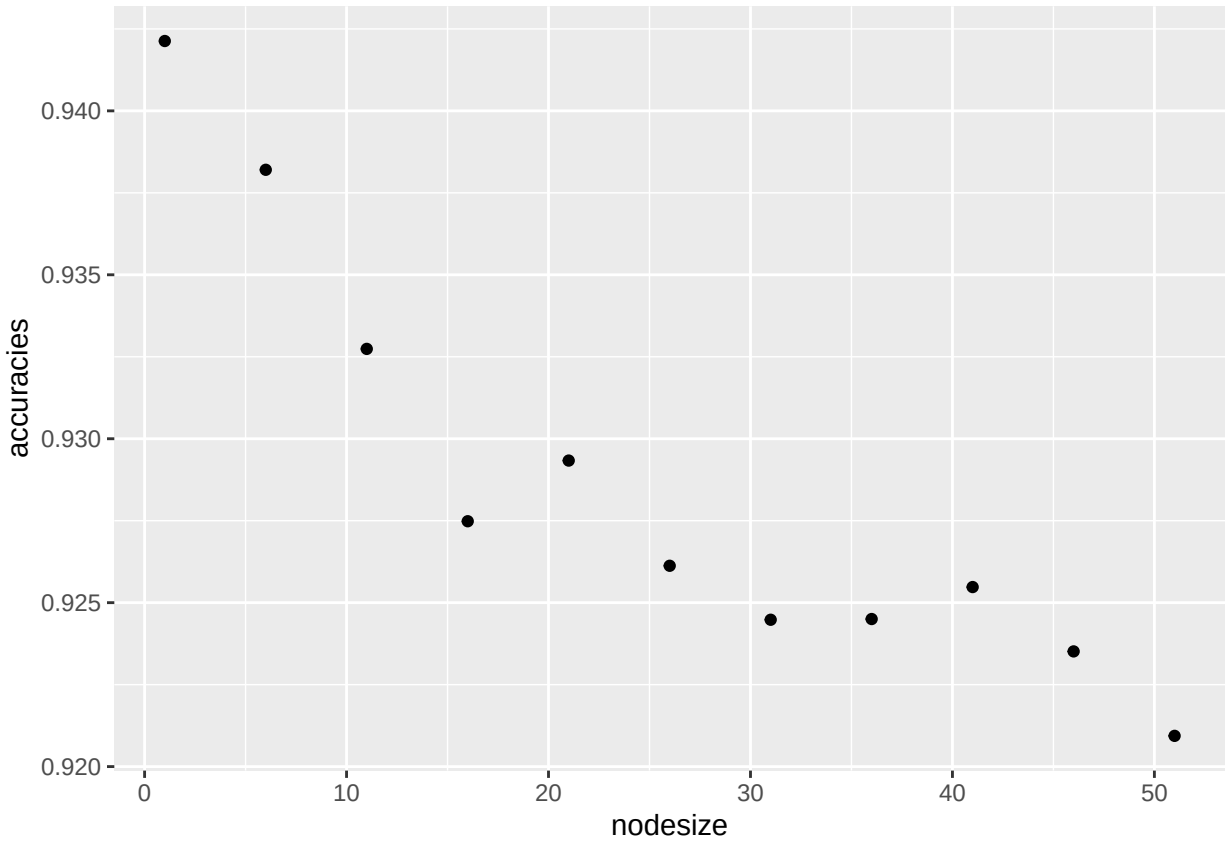
As will become apparent in the next section, all of these models leave room for possible improvement in accuracy, though the random forest performs the best. Thus, I will also create a new random forest model with custom parameters to try and raise the model's performance.

Since I do not yet know which nodesize will maximize my accuracy, I will test out every fifth value between 1 and 51. Following that, I visualize the various accuracies plotted against their associated nodesize.

```
nodesize <- seq(1, 51, 5)
accuracies <- sapply(nodesize, function(q){
  train(Winner_Contender ~ ., method = "rf", data = boulder_train2,
        tuneGrid = data.frame(mtry = 9),
        nodesize = q)$results$Accuracy
})
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
qplot(nodesize, accuracies)
```



We can now apply this accuracy maximizing nodesize to construct one last model before adding it to *model_comparison2*.

```
rf_fit3 <- randomForest(Winner_Contender ~ ., data = boulder_train2,
                        nodesize = nodesize[which.max(accuracies)])
y_hat_rf3 <- predict(rf_fit3, boulder_test2)
rf_cf_mat3 <- confusionMatrix(y_hat_rf3, boulder_test2$Winner_Contender)

final_model_comparison <- bind_rows(model_comparison2,
                                    data.frame(Model = "Random Forest [manually adjusted]",
                                               Accuracy = rf_cf_mat3$overall["Accuracy"],
                                               Sensitivity = rf_cf_mat3$byClass["Sensitivity"],
                                               Specificity = rf_cf_mat3$byClass["Specificity"]))
```

3: Results

With all of the models and the comparison tables now made, we are well positioned to determine which model performs best. To begin, let's look at the *model_comparison* table.

```
model_comparison %>% knitr::kable(caption = "Model Comparison Table")
```

Table 3: Model Comparison Table

	Model	Accuracy	Sensitivity	Specificity
Accuracy...1	K Nearest Neighbors (KNN)	0.9763421	0.9946978	0.8653846
Accuracy...2	Random Forest (rf)	1.0000000	1.0000000	1.0000000

	Model	Accuracy	Sensitivity	Specificity
Accuracy...3	Logistic Regression (glm)	1.0000000	1.0000000	1.0000000

While the KNN model might appear to just be a really well fitted model, the perfect accuracy, sensitivity, and specificity of the random forest and logistic regression models should raise a bit of suspicion. Bear in mind that I called these models unrealistic earlier due to their predicting on information which would be unknown at any time where such predictions would be useful. They rely heavily on information coming from the final rounds, as seen by...

```
varImp(knn_fit)
```

```
## ROC curve variable importance
##
##                               Importance
## Final_Zones                       100.00
## Final_Zones_Attempts              100.00
## Total_Zones                        95.19
## Total_Tops                         90.52
## Total_Attempts_to_Zone             90.37
## Final_Top_Attempts                 87.84
## Final_Tops                         87.84
## Total_Attempts_to_Top              82.06
## Qualification_Tops                 58.99
## Qualification_Zones                55.81
## Qualification_Zones_Attempts       45.46
## Qualification_Top_Attempts         44.07
## Semifinal_Tops                     20.34
## Semifinal_Top_Attempts             19.57
## Semifinal_Zones                    17.95
## Semifinal_Zones_Attempts           16.74
## StartNr                             0.00
```

```
varImp(rf_fit)
```

```
## rf variable importance
##
##                                Overall
## Final_Zones_Attempts          100.0000
## Final_Zones                    79.1762
## Final_Top_Attempts             57.2996
## Final_Tops                     55.2436
## Total_Zones                    50.4806
## Total_Tops                     23.9980
## Total_Attempts_to_Zone         21.7313
## Total_Attempts_to_Top          12.0565
## Qualification_Tops             11.1089
## Qualification_Zones             6.2883
## Semifinal_Tops                  5.1027
## Qualification_Top_Attempts      2.9917
## Qualification_Zones_Attempts    2.7475
## Semifinal_Zones                 1.9396
## Semifinal_Top_Attempts          1.4426
## Semifinal_Zones_Attempts        0.7571
## StartNr                         0.0000
```

```
varImp(glm_fit)
```

```
## glm variable importance
##
##                                Overall
## Total_Zones                   100.0000
## Qualification_Zones            89.7429
## Semifinal_Zones                80.1006
## Qualification_Top_Attempts     25.4647
## Total_Attempts_to_Top          25.3690
## Total_Tops                     25.0614
## Qualification_Tops             24.3506
## Semifinal_Tops                 20.7233
## Semifinal_Top_Attempts         17.9775
## StartNr                         0.3758
## Qualification_Zones_Attempts    0.2074
## Total_Attempts_to_Zone          0.1508
## Semifinal_Zones_Attempts        0.0000
```

For both the KNN and Random Forest models, the 8 most important variables depend on information coming from the final rounds. For the logistic regression model, 4 of the variables with non-zero importance require knowledge about the final round. This makes the models unrealistically accurate, sensitive, and specific, thereby rendering their usefulness essentially trivial.

Fortunately, we can and did craft more realistic models, as represented by the *model_comparison2* table.

```
final_model_comparison %>% knitr::kable(caption = "Final Model Comparison",
                                        row_spec = 2, color = "black", background = "yellow")
```

Table 4: Final Model Comparison

	Model	Accuracy	Sensitivity	Specificity
Accuracy...1	K Nearest Neighbors (KNN)	0.8844404	0.9490986	0.4935897
Accuracy...2	Random Forest (rf)	0.9490446	0.9766702	0.7820513
Accuracy...3	Logistic Regression (glm)	0.9162875	0.9607635	0.6474359
Accuracy...4	Random Forest [manually adjusted]	0.9481347	0.9713680	0.8076923

We observe a significant drop in accuracy and specificity in these more realistic models over their unrealistic counterparts. Notably, the sensitivity of the models remains quite high, with the lowest rating still being above 95%, but the specificity ratings never exceeding 90%. This is an acceptable trade-off, since were those who are actual contenders to be deemed not to be, their nation's team might exclude them and thereby lessen their chances of winning. If, however, someone who is not a contender were to be selected for a team, it is less likely but not impossible that they could still win. So, while you would want to minimize the number of non-contenders on a team, their inclusion would not necessarily entail a poorer result.

Among the realistic models, both the random forest and the logistic regression models are somewhat comparably accurate and sensitive, but the KNN model is notably less accurate. This could be due the highly similar nature of many of the athletes and as a result, their similarity to their neighbors. As will be shown below, the number of tops in qualifications is the most important variable for all of the models. Thus, it is hardly surprising that, given the large number of athletes with 0 tops, the KNN model would struggle to make highly accurate predictions. It is also notable that the KNN model is the only one that has a specificity rating around 50%. So, for these reasons, the KNN model should not be considered the best model.

The goal of these predictions is to tell whether an athlete is a contender to win a competition. The value of such predictions is best judged on accuracy and sensitivity. Accuracy works well as a baseline metric

since relying on an accurate selection mechanic for team building will allow for the makeup of said team to be mostly, if not wholly, composed of qualified persons. Sensitivity is also important here since a person who is incorrectly classified as a non-contender despite their being a contender could deprive teams of their optimal members. Low specificity, on the other hand, could increase the likelihood of including non-contender on a nation's team but as noted before, being a non-contender does not mean that said person has a zero probability of winning; it just means that it is unlikely. That said, higher specificity can only serve to benefit a team-selection process. Since the random forest models are both more accurate than the logistic regression model, the latter is not the best model. Noting that the specificity ratings for both random forest models are (approximately) equal and their respective accuracies are separated by less than 0.2%, the slightly improved sensitivity of the second model gives significant support for it as the best model.

It is also worthwhile to note that the most important variables are not consistent across the models, as seen by...

```
varImp(knn_fit2) #Qualification_Tops and Qualification_Zones are now the most important features.
```

```
## ROC curve variable importance
##
##                              Importance
## Qualification_Tops               100.00
## Qualification_Zones               94.54
## Qualification_Zones_Attempts      77.46
## Qualification_Top_Attempts        73.96
## Semifinal_Tops                    33.90
## Semifinal_Top_Attempts            32.80
## Semifinal_Zones                   29.36
## Semifinal_Zones_Attempts          27.16
## StartNr                            0.00
```

```
varImp(rf_fit2) #Qualification_Tops and StartNr are now the most important features.
```

```
## rf variable importance
##
##                              Overall
## Qualification_Tops           100.000
## StartNr                       60.689
## Qualification_Zones           44.519
## Semifinal_Tops                41.295
## Qualification_Zones_Attempts  21.641
## Qualification_Top_Attempts    13.637
## Semifinal_Top_Attempts         2.787
## Semifinal_Zones                0.657
## Semifinal_Zones_Attempts       0.000
```

```
varImp(glm_fit2) #Qualification_Tops and Semifinal_Tops are now the most important features.
```

```
## glm variable importance
##
##                              Overall
## Qualification_Tops           100.000
## Semifinal_Tops                77.944
## Qualification_Top_Attempts    24.490
## Semifinal_Zones_Attempts      20.591
## Qualification_Zones           12.420
## Semifinal_Zones                7.962
## Qualification_Zones_Attempts   6.875
## Semifinal_Top_Attempts         2.395
```

```
## StartNr                          0.000
```

While all of the models are most affected by the number of tops in the qualification round, the KNN model is next-most affected by Qualification zones, the random forest model by starting number, and the logistic regression model by tops in semifinals. Interestingly, while winning a competition involves making it to semifinals, how an athlete performs in said round is not the most important factor for any of the models.

The three most important variables of the second model, *Random Forest (rf)*, do not involve the semifinal round at all. For the logistic regression model, the second most important variable requires knowing the end result of the semifinal round. In a practical context, being able to make predictions without a need for information from the semifinals round is quite useful, especially since most climbers do not make it to the semifinals often, if ever.

For these reasons, the second model, *Random Forest (rf)*, is the best model.

4: Conclusion

4.1: Summary of the Report and Potential Impact

IFSC bouldering world cups are extremely competitive and each nation that participates goes to great lengths to select only the best climbers for their team. While there are secondary benefits to including a variety of climbers, the primary selection criteria is whether a given climber could win a competition. This report has now outlined several classification models that predict exactly this. Through the data exploration and visualizations performed in Sections 2.2, we gained insight into the improbability of any given climber ever winning a competition, learned that many competitors are, performance-wise, highly similar to one another, and found that there were demonstrable patterns of difficulty throughout the competitions' rounds. Bearing this in mind, a not small number of predictors were included in each of the models before this group of predictors was substantively reduced to better align the models with their practical purposes. At the end of this process, it was concluded that a random forest model provided the most accurate and sensitive predictions, making it the preferred model for predicting contenders to win an IFSC bouldering world cup.

4.2: Limitations and Future Work

Perhaps the two greatest limitations to the work done here are the relatively small scope of the original data set and the manner in which being a contender to win was defined. To the former limitation, the original data set included just above 5500 rows of data spanning approximately 2 years of competitions (2018 and 2019). The IFSC has been operating bouldering world cups for upwards of 20 years, meaning that it is likely possible to compile a much larger data set. On a related note, the small size of the data set used here forced me to treat junior and senior world cups as equally informative events (dividing them would have made each new set too small). This may have skewed the predictions; as seen in the final plot of Section 2.2.2, the typical performance across competitions is nowhere near uniform. There very likely is competition specific bias and this bias may originate from the senior/junior level distinction.

In regard to the latter limitation, I chose here to categorize each climber in a binary fashion; you either were a contender or you were not. Though a binary categorization such as the one used here can deliver a reasonably strong selection criteria for national climbing teams, a few improvements should be considered for future work. One improvement would be to have multiple categories, i.e., something along the lines of strong contender, contender, weak contender, and not a contender as categories. This would permit for the criteria used to determine the 'chance' of a climber winning to be more diverse and robust. Alternatively, the logical statement(s) used to decide who is or is not a contender could be made multivariate, meaning that instead of only a certain number of zones needing to be completed to qualify as a contender, some number of tops or attempts would also have to be reached or not exceeded.

That said, the work done here represents an attempt to predict the outcome of climbing competitions, events which are infamously unpredictable and prone to having dark horse winners. How best to accommodate this trend, however, is a task for another time.

4.3: Final Notes

This report is presented as the final project for the Harvardx Professional Certificate in Data Science.

The original data set, available at https://www.kaggle.com/datasets/brkurzawa/ifsc-sport-climbing-competition-results, was created by Brett Kurzawa and uses data scraped from ifsc-climbing.org.

Some webpages and books were consulted in the making of this report. They include:

1. ifsc-climbing.org
2. https://www.kaggle.com/datasets/brkurzawa/ifsc-sport-climbing-competition-results
3. R for Everyone: Advanced Analytics and Graphics, Second Edition (Jared P. Lander)
4. https://rafalab.github.io/dsbook/introduction-to-machine-learning.html#evaluation-metrics