# Independent Capstone Project

Connor Bruce

5/5/2022

## Introduction

This dataset contains the results from roughly 50,000 matches of the game League of Legends as well as some various factors that lead to this result. To provide context, the goal of a game of League of Legends is to destroy the enemy team's towers (must destroy one tower before being able to destroy the next) in order to open up their base for destruction. In order to accomplish this, players gain gold by killing neutral monsters, enemy minions, and enemy characters in order to become stronger by using the gold to buy items. Another way players accomplish this goal is by claiming objectives around the map that grant gold and various unique buffs to the team who claims the objective. These objectives include: Baron which provides gold and empowers your minions, Dragon which provides a random buff depending on the map, and Rift Herald which can be used to deal massive damage to the enemy structures. Destroying an enemy's inhibitor (located inside the enemy base behind a tower), causes your team to spawn super minions in the lane that it was destroyed and open the rest of the base to be destroyed.

The factors in the dataset that will be used are:

- firstBlood: Which team scored the first takedown of the match
- firstTower: Which team took the first tower of the match
- firstInhibitor: Which team destroyed the first inhibitor of the match
- firstBaron: Which team claimed the first Baron objective of the match
- firstDragon: Which team claimed the first Dragon objective of the match
- firstRiftHerald: Which team claimed the first Rift Herald objective of the match

The goal of this project is to use these factors to determine how well the outcome of a match can be predicted based on these six factors. The usefulness behind this falls more in the variable importance which can provide insight as to which objective to focus on during a game. It can also be useful for figuring out how likely a team is to win a game given which objectives are claimed during a game. In order to accomplish this goal, a random forest model - an average of many decision trees - will be created due to the categorical nature of the data. Once is model is created, the variable importance will be calculated in order to determine which objective is the best to focus on during a game.

## Methods and Analysis

### Data Cleaning

When parsing the original dataset, I had to make a decision on which predictors to use. The dataset included columns for the characters on each team, as well as their "loadout". I decided to not use these as my goal was to predict the outcome on the game based on objectives as opposed to the setup of the players. Therefore, the first step is to get the relevant columns. The next step is to change the team assignment to 1 and -1

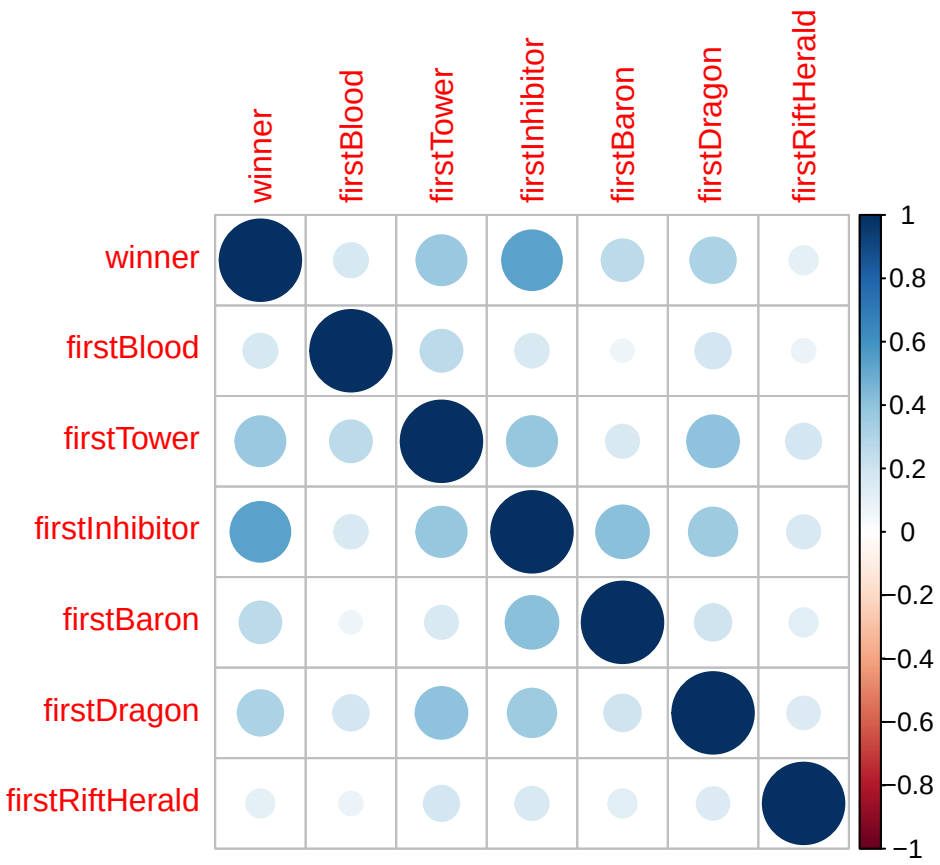
instead of 1 and 2 for clarity. The next step was to change the columns into factors instead of numeric as the variables are categorical. The next step is to take out the games where neither team scored the first takedown (i.e. firstBlood == 0). This is because games where this occurs are games that are either illegitimate or "remakes" where one or more players do no connect to the game.

```
games = read.csv("Data/games.csv")
games_sm = games %>%
  select(winner:firstRiftHerald)
games_sm[games_sm == 2] = -1
games_sm$winner=as.factor(games_sm$winner)
games_sm$firstBlood=as.factor(games_sm$firstBlood)
games_sm$firstTower=as.factor(games_sm$firstTower)
games_sm$firstInhibitor=as.factor(games_sm$firstInhibitor)
games_sm$firstDragon=as.factor(games_sm$firstDragon)
games_sm$firstRiftHerald=as.factor(games_sm$firstRiftHerald)
games_sm$firstBaron=as.factor(games_sm$firstBaron)
games_sm = games_sm %>%
  filter(firstBlood != 0)
```

## Data Exploration and Visualization

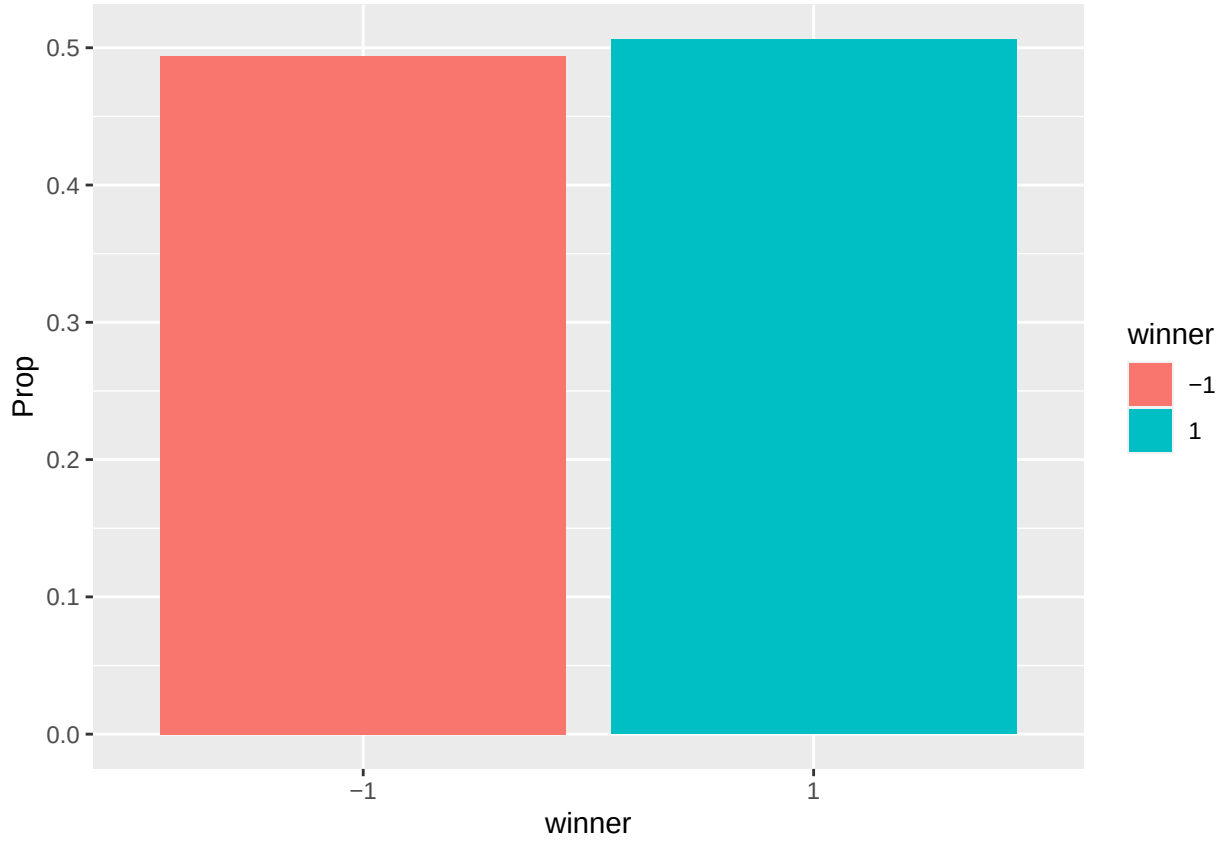
With the data properly processed, my the first direction I went was to create a correlation chart in order to determine if any of the variables were redundant. I expected a small positive correlation as when one team is ahead, they are more likely to be able to claim multiple objectives before the other team. I found that no variables had a very high correlation with other variables so I determined to no omit any variables on these grounds.

```
games_sm_cor = cor(games %>%
                     select(winner:firstRiftHerald))
corrplot(games_sm_cor)
```

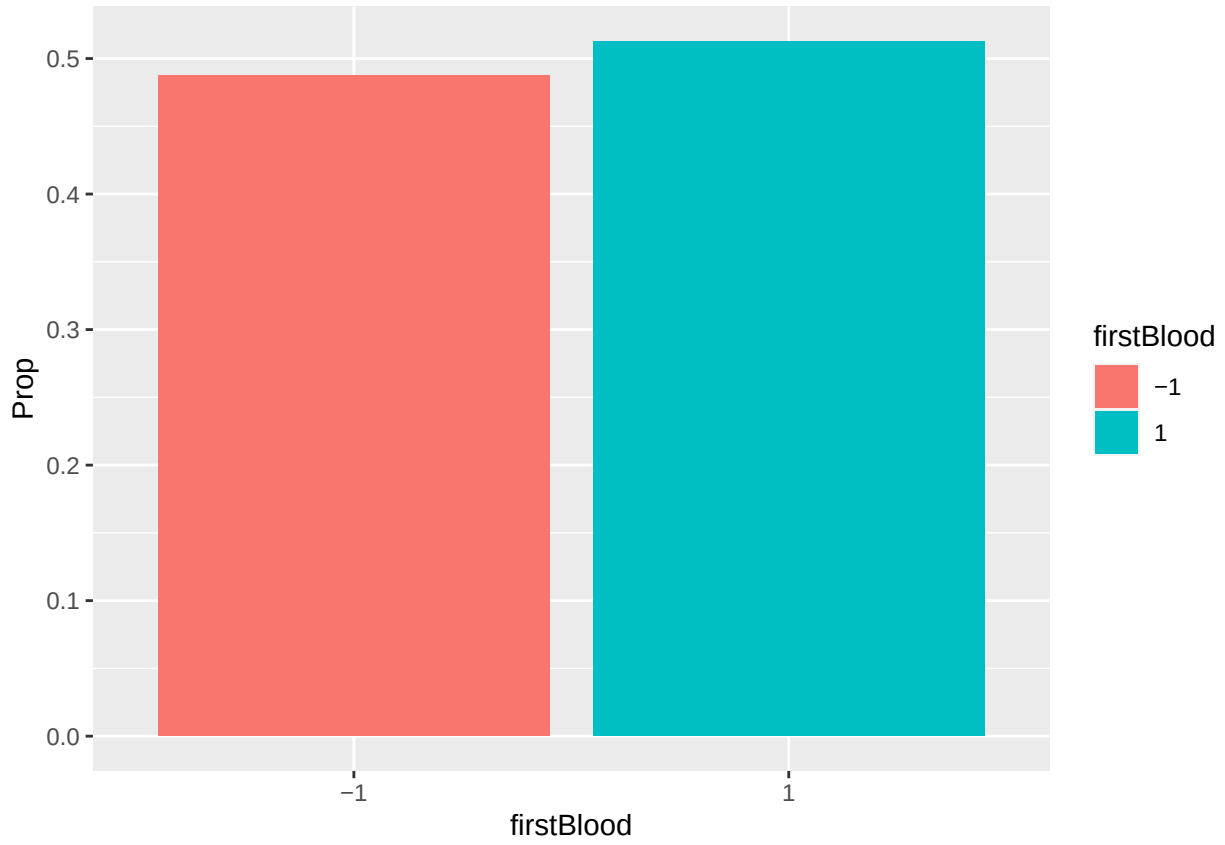



My next step in exploration was to take a look at the distributions of all of the variables to get an idea if any of the data is skewed towards on team. In theory, the proportion of team -1 and team 1 for each of the variable should be equal as no one side is given any strong advantages. When looking at each of the charts, the trend of equal proportion held for all of them so I, once again, did not omit any variables.

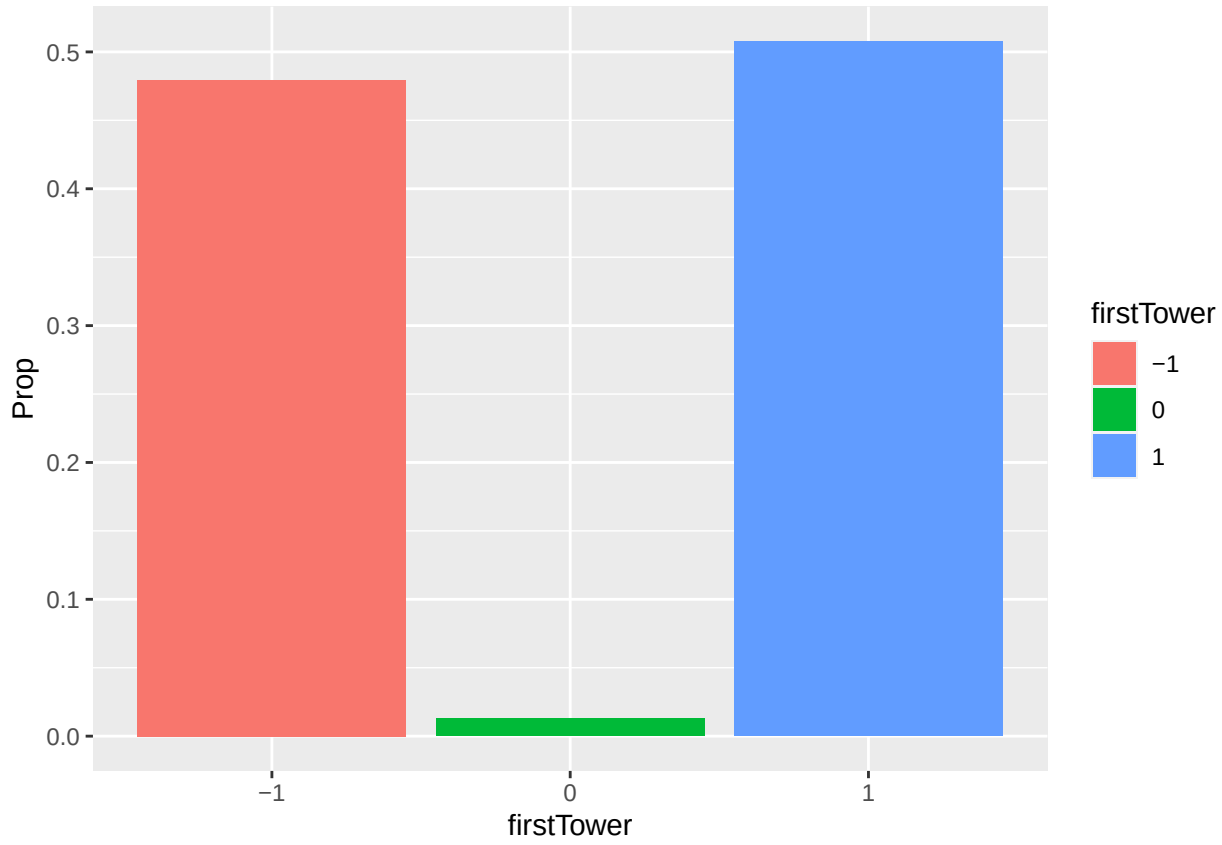
```
##Distribution of wins
games_sm %>%
  count(winner) %>%
  mutate(Prop = n/sum(n)) %>%
  ggplot(aes(x = winner, y = Prop, fill = winner)) +
  geom_bar(stat = "identity")
```



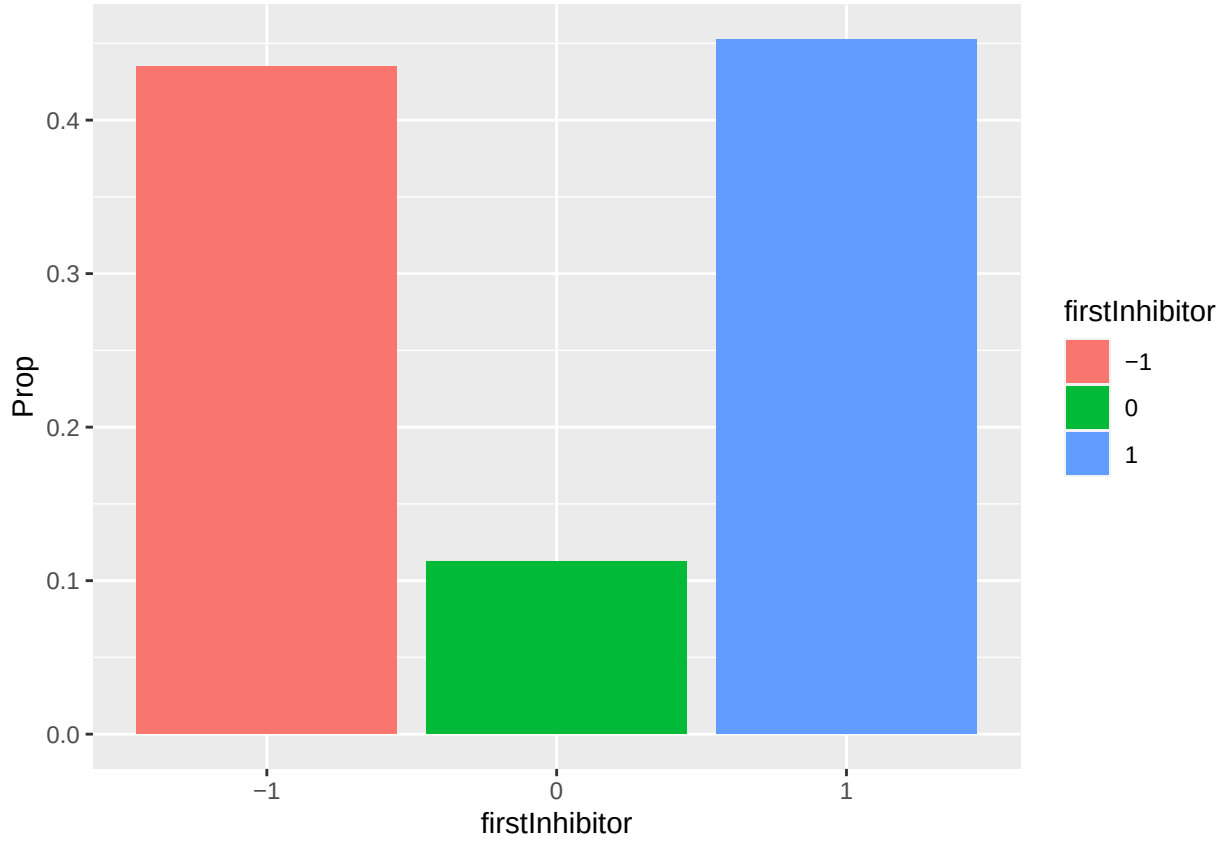

```
##Distribution of firstBlood
games_sm %>%
  count(firstBlood) %>%
  mutate(Prop = n/sum(n)) %>%
  ggplot(aes(x = firstBlood, y = Prop, fill = firstBlood)) +
  geom_bar(stat = "identity")
```



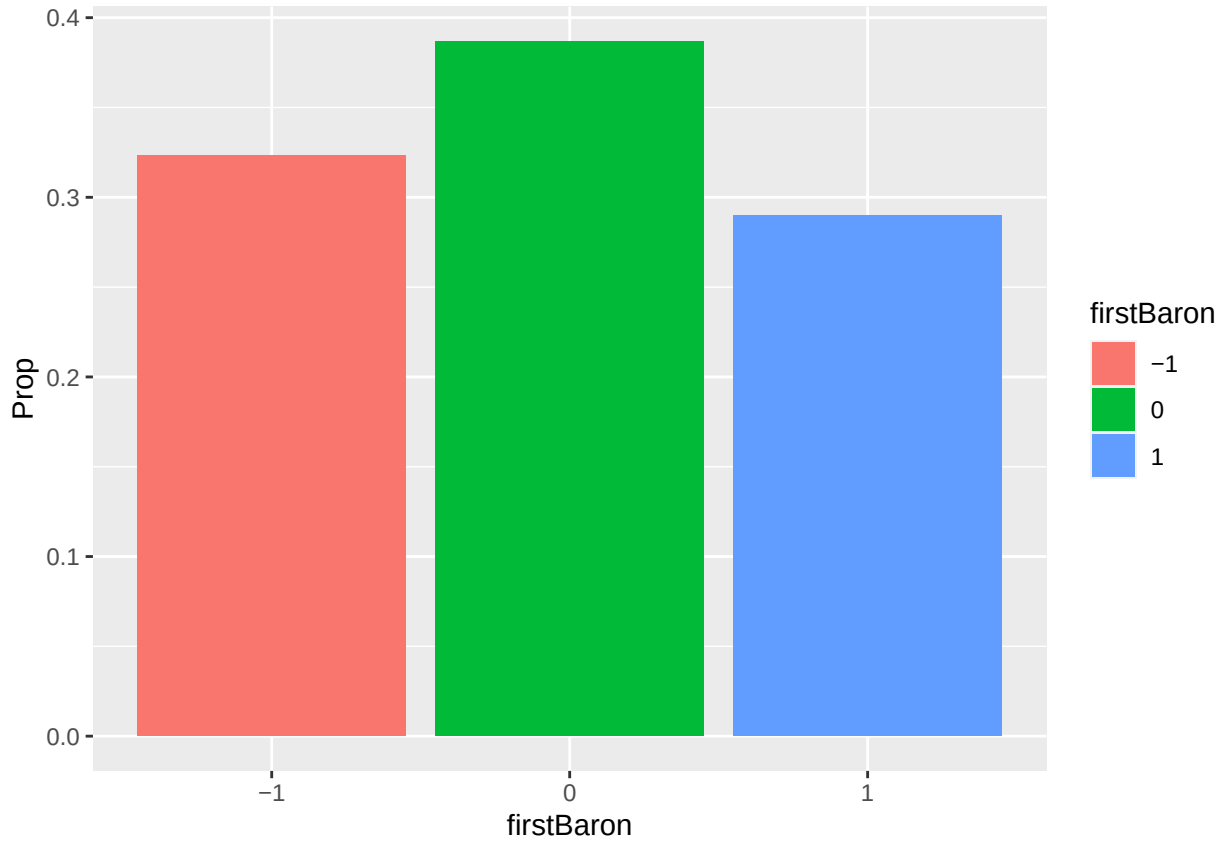

```
##Distribution of firstTower
games_sm %>%
  count(firstTower) %>%
  mutate(Prop = n/sum(n)) %>%
  ggplot(aes(x = firstTower, y = Prop, fill = firstTower)) +
  geom_bar(stat = "identity")
```



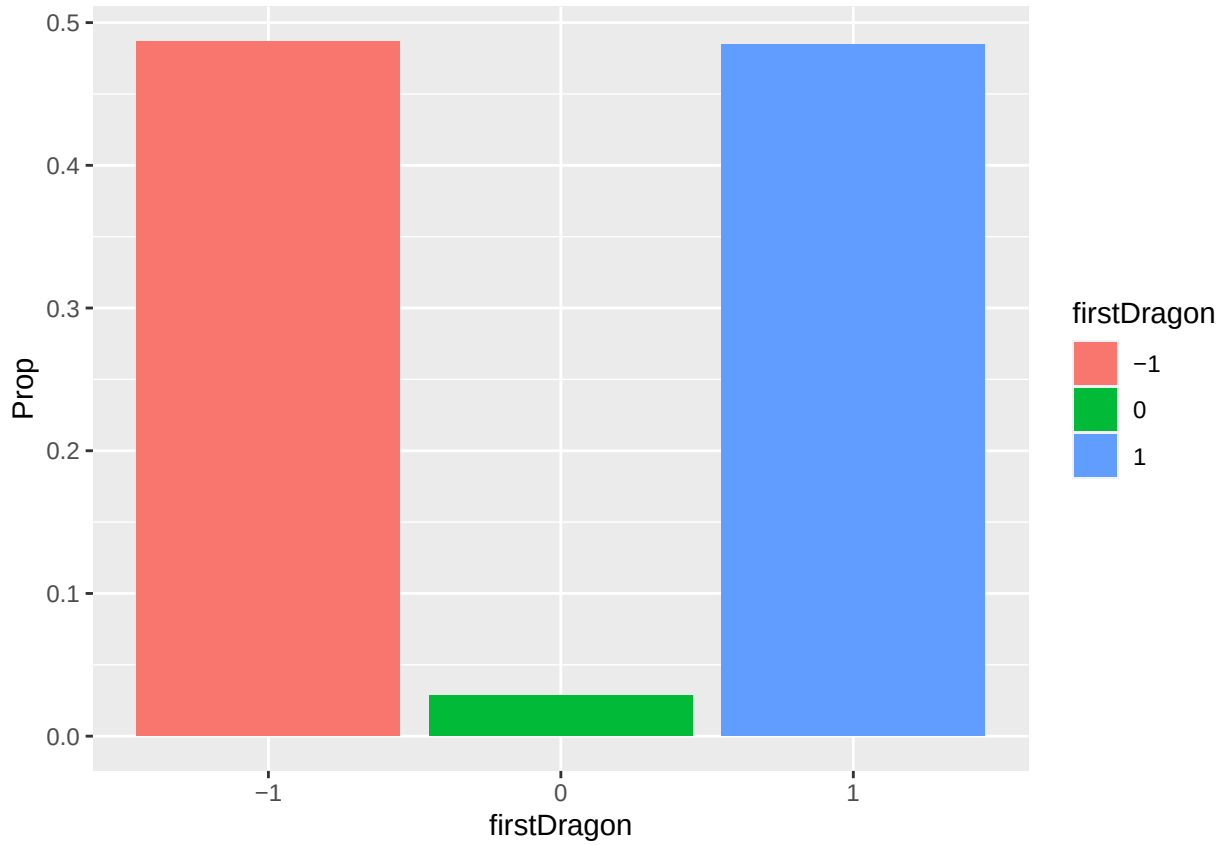

```
##Distribution of firstInhibitor
games_sm %>%
  count(firstInhibitor) %>%
  mutate(Prop = n/sum(n)) %>%
  ggplot(aes(x = firstInhibitor, y = Prop, fill = firstInhibitor)) +
  geom_bar(stat = "identity")
```



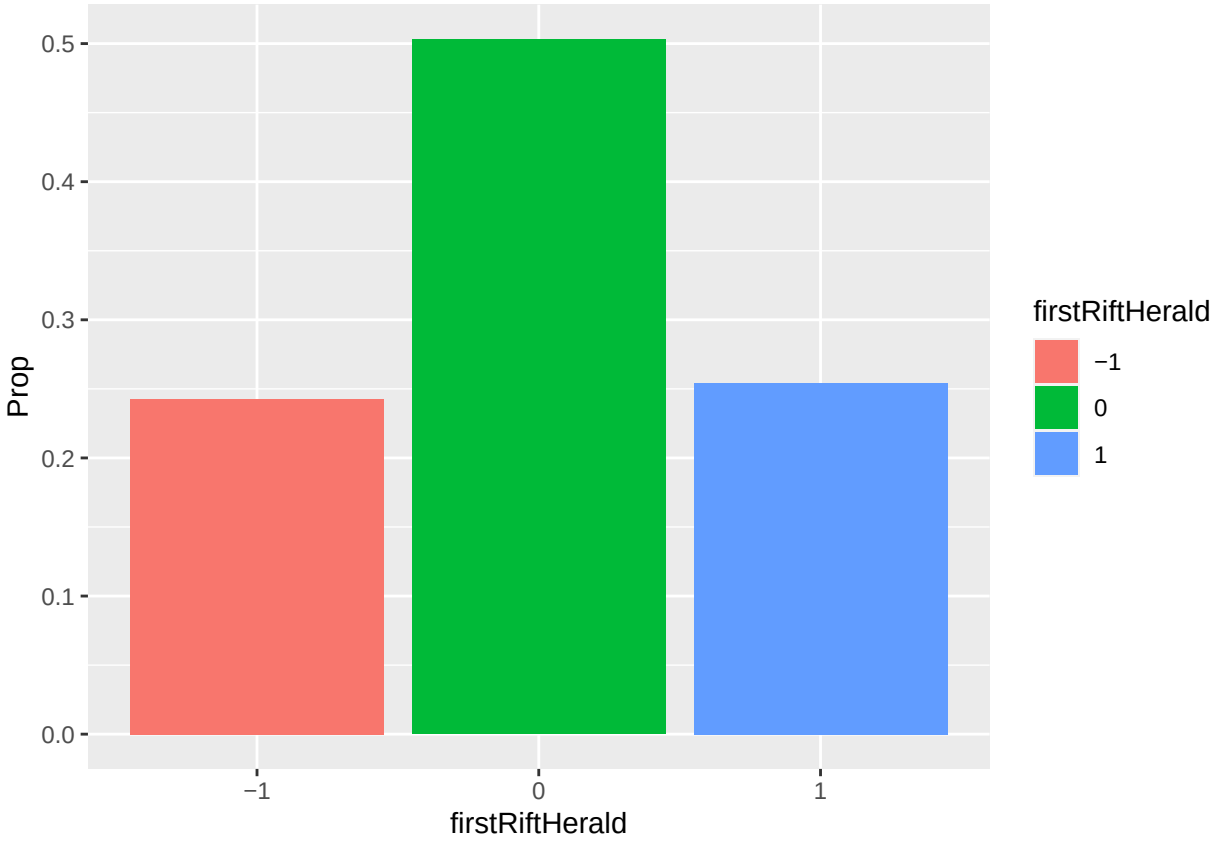

```
##Distribution of firstBaron
games_sm %>%
  count(firstBaron) %>%
  mutate(Prop = n/sum(n)) %>%
  ggplot(aes(x = firstBaron, y = Prop, fill = firstBaron)) +
  geom_bar(stat = "identity")
```




```
##Distribution of firstDragon
games_sm %>%
  count(firstDragon) %>%
  mutate(Prop = n/sum(n)) %>%
  ggplot(aes(x = firstDragon, y = Prop, fill = firstDragon)) +
  geom_bar(stat = "identity")
```




```
##Distribution of firstRiftHerald
games_sm %>%
  count(firstRiftHerald) %>%
  mutate(Prop = n/sum(n)) %>%
  ggplot(aes(x = firstRiftHerald, y = Prop, fill = firstRiftHerald)) +
  geom_bar(stat = "identity")
```




### Spliting the Data into Training and Testing

```
y = games_sm$winner
test_index = createDataPartition(y, times = 1, p = .2, list = FALSE)
train_game = games_sm[-test_index,]
test_game = games_sm[test_index,]
```

### Logistic Regression

The first model I decided to use was a logistic regression due to the categorical nature of the data.

```
fit_glm = train(winner ~ ., data = train_game, method = "glm")
preds_glm = predict(fit_glm, test_game)
```

### Random Forest Model

The second model I decided to use is a random forest model. This is because, from experience, I believe that the outcome of a game, when based on these variables, can boil down to a decision-style progression. This is because the objective in question tend to follow an order in game (some don't appear until later in the game or some MUST come before others). I went with 100 decision trees in the forest as it was a large enough number to capture any outlier games but not too large as to cause very long run times.

```
fit_rf = randomForest(train_game$winner ~ .,
                      data = train_game,
                      ntree = 100)
preds_rf = predict(fit_rf, test_game)
preds_rf = as.factor(preds_rf)
```

## Results

I was pleased with the results. Both models boast an accuracy of ~90%. The most useful outcome was the variable importance. It can be seen that the objective that holds the most importance is the first Inhibitor, with a distant second going to first Baron, then first Tower. This provides useful information on how to go through decision-making during games. There are often points when playing where there are multiple options on what to do with limited time. An common example is after wiping the opposing team, the triumphant team has limited time to take objectives uncontested. Often a decision later in the game is either destroying the first inhibitor or defeating Baron in order to get its buff. Based on the results from this model, most of the time, it is more advantageous to take the inhibitor over the Baron. Another useful takeaway is the lack of importance assigned to first blood. Some games, a player will become discouraged when the opposing team scores first blood. However, when looking at the data, while it does have a positive correlation with winning, it does not hold much weight in securing a win.

```
glm_acc = confusionMatrix(preds_glm, as.factor(test_game$winner))$overall["Accuracy"]
rf_acc = confusionMatrix(preds_rf, as.factor(test_game$winner))$overall["Accuracy"]
knitr::kable(data.frame(Model = c("Logisitic","Random Forest"),
                        Accuracy = c(glm_acc,rf_acc)))
```

| Model | Accuracy |
|---|---|
| Logisitic | 0.9020416 |
| Random Forest | 0.9048881 |

```
var_imp = as.data.frame(importance(fit_rf))
var_imp %>% rename(Importance = MeanDecreaseGini) %>%
  arrange(desc(Importance))
```

```
##                 Importance
## firstInhibitor  8900.89969
## firstBaron      2088.70147
## firstTower      1471.69544
## firstDragon      883.11284
## firstRiftHerald  170.48613
## firstBlood        84.94169
```

## Conclusion

To conclude, the two models were chosen were logistic regression and a random forest model due to the categorical nature of the data. Both of these models had an accuracy of roughly 90%. The primary usefulness came from the variable importance which can help with determining decision making in game with regards to prioritizing objectives. The limitations of this model is that it only takes into account who took the first

of each objective. In game, the objectives respawn or there are multiple of each so while one team may take the first, the other team could have taken more of the same objective afterward. As for future work, the data that was originally removed from the dataset (such as the characters picked by the players) could be used as new variables. This would give an idea to how strong a team composition is as some character synergize better with others and some characters "counter" characters on the opposing team giving one team an advantage over the other.