Predicting the 2019 MLB Playoffs

Jayson Faulds

2019-10-30

Introduction

As a baseball-lover since the age of five, the sport has been an integral part of my life. Aside from teaching me many lessons, it has also helped me through grade school as a child. Many baseball essays and projects have been made and completed throughout my primary education. Now, as a graduate student immersed in the study of data analytics and statistical modeling, I cannot help but make baseball the victim of another one of my pet projects.

Here, I will be using 21st century data to construct an ordinal logistic regression model, with the goal of predicting the course of the 2019 MLB postseason.

Data

The data consist of several attributes that I determined to be important indicators of a successful baseball club. These predictors include Total Wins (W), Number of Homeruns (HR), Earned Run Average (ERA), Fielding Percentage (FP), and Batting Average (BA). All data used in this analysis come from the `lahman` package in R, which pulls select data from Sean Lahman's well-known baseball database. But what exactly are we attempting to predict? We want to estimate, for each team that made the playoffs this year, which round they will advance to. In other words, will the Dodgers lose in the NLDS, NLCS, World Series, or win the whole thing? This is a classification problem with an ordinal nature. Thus, we will be using the ordinal extension to the logistic regression model.

As for the observations on which we built the model, we restricted our domain to playoff instances in the 21st century. Therefore, only playoff teams after the year 1999 are considered. We want to have enough data for our model to make accurate predictions (We have 166 observations after filtering), and yet still remain somewhat recent. The following R code shows how I prepared the data for further analysis.

```
# Load relevant packages
suppressPackageStartupMessages(library(Lahman))
suppressPackageStartupMessages(library(dplyr))

# Accesses the data
data(SeriesPost)

# Creates a dataframe of all recent losing teams
losers <- SeriesPost %>%
  filter(yearID >= 2000) %>%
  select(yearID, teamIDloser, round)

# Dataframe of the last 19 WS champs
winners <- SeriesPost %>%
  filter(yearID >= 2000 & round == "WS") %>%
  select(yearID, teamIDwinner, round)

# Adjusting the values in the round column
losers$round[losers$round == "ALDS1" | losers$round == "ALDS2" | losers$round == "NLDS1"
             | losers$round == "NLDS2"
```

```
              | losers$round == "ALWC" | losers$round == "NLWC"] <- "DS"
losers$round[losers$round == "ALCS" | losers$round == "NLCS"] <- "CS"
losers$round[losers$round == "WS"] <- "LWS"

# Renames columns and merges
colnames(losers)[colnames(losers) == "teamIDloser"] <- "teamID"
colnames(winners)[colnames(winners) == "teamIDwinner"] <- "teamID"
teams <- rbind(losers, winners)

# loads the predictors
data("Teams")

# Filters the stats we want, merges with teams dataframe
team_stats <- Teams %>%
  filter(yearID >= 2000) %>%
  filter(DivWin == 'Y' | WCWin == 'Y') %>%
  select(yearID, teamID, W, AB, H, HR, ERA, FP)

teams <- merge(teams, team_stats, by.x = c("yearID", 'teamID'),
               by.y = c("yearID", 'teamID'))

# Makes a variable for batting average
teams$BA <- teams$H/teams$AB
teams <- teams[-c(5, 6)]
```

Exploratory Data Analysis

We begin by looking at the distribution of our target variable as well as the descriptive statistics for our numeric predictors. We follow this by visualizing the histograms for our five numeric predictors, trying to assess the normality. We do not see much skew in the distributions, meaning there is no need for variable transformation (i.e. log transformation). We do see that the distribution for batting average is quite irregular, and for homeruns it is bimodal. We believe these irregular relationships are a result of the change in batting styles over time, as we have seen a recent surge in homerun totals, coupled with higher strikeout rates overall (And maybe some juiced baseballs, wink wink).

```
suppressPackageStartupMessages(library(ggplot2))

table(teams$round)
```

```
##
##  CS  DS LWS  WS
##  38  90  19  19
```

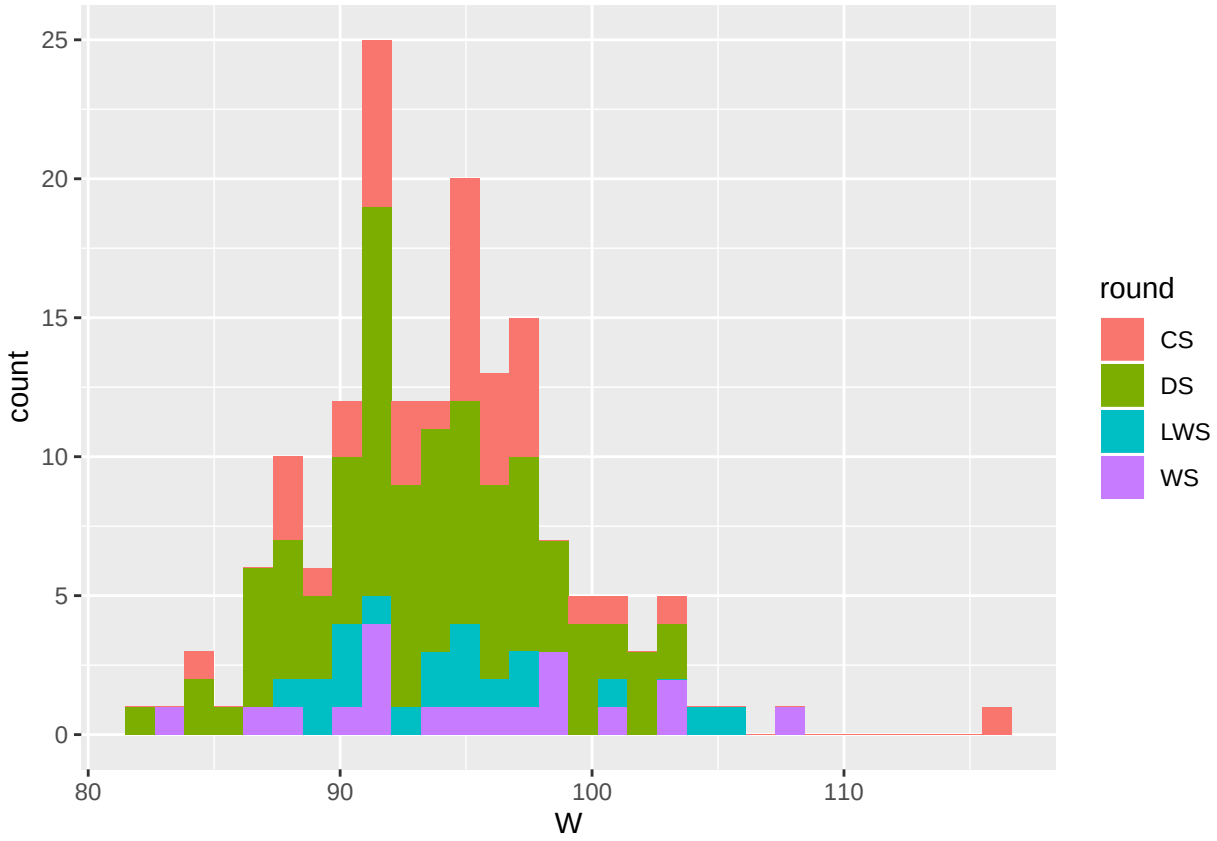
```
numerics <- c("W", "HR", "ERA", "FP", "BA")
lapply(teams[, numerics], summary)
```

```
## $W
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   82.00   91.00   94.00   93.99   97.00  116.00
##
## $HR
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    95.0   162.0   184.0   185.5   210.8   267.0
##
## $ERA
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.940   3.583   3.815   3.852   4.090   4.760
##
## $FP
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9780  0.9830  0.9850  0.9844  0.9860  0.9900
##
## $BA
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2379  0.2561  0.2638  0.2646  0.2718  0.2897
```

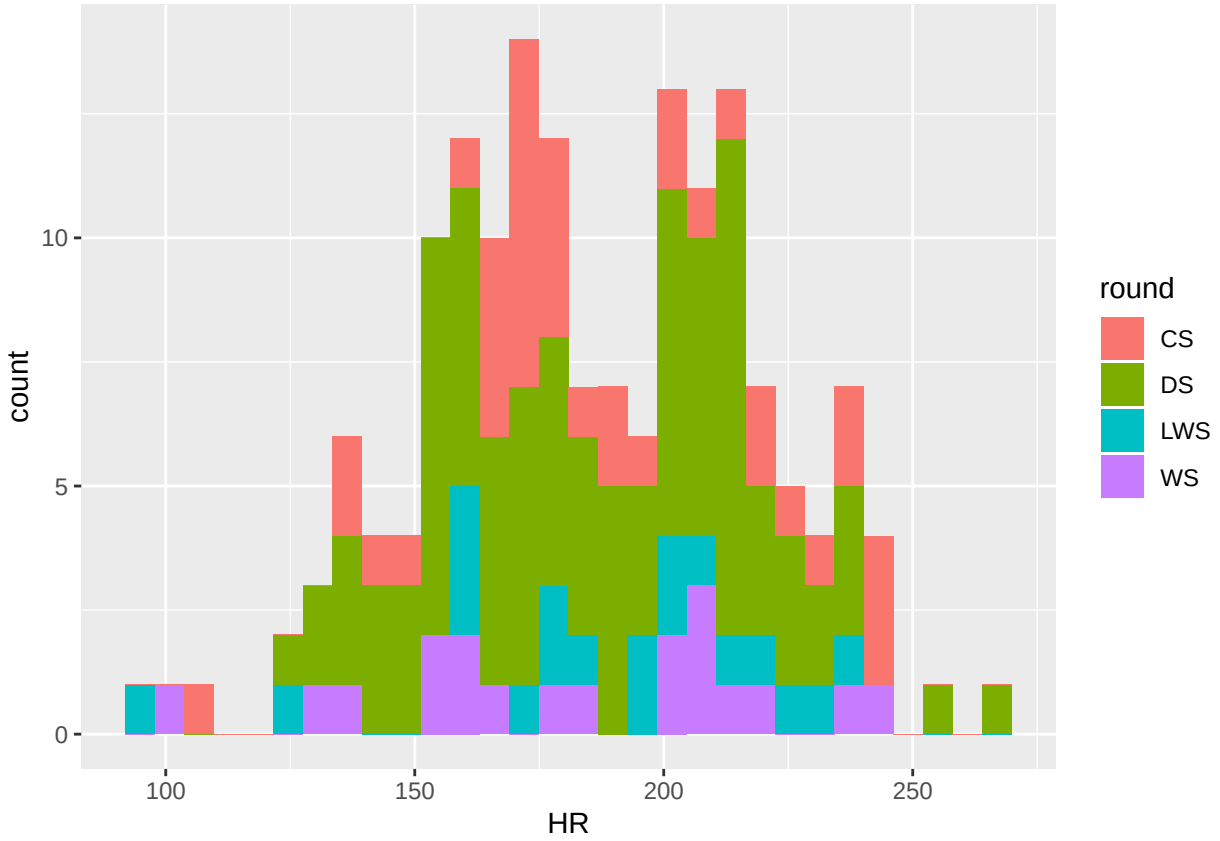
```
# Histograms of each numeric feature
ggplot(teams, aes(x = W, fill = round)) +
  geom_histogram() # relatively normal, one potential outlier, wins don't seem to matter much
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



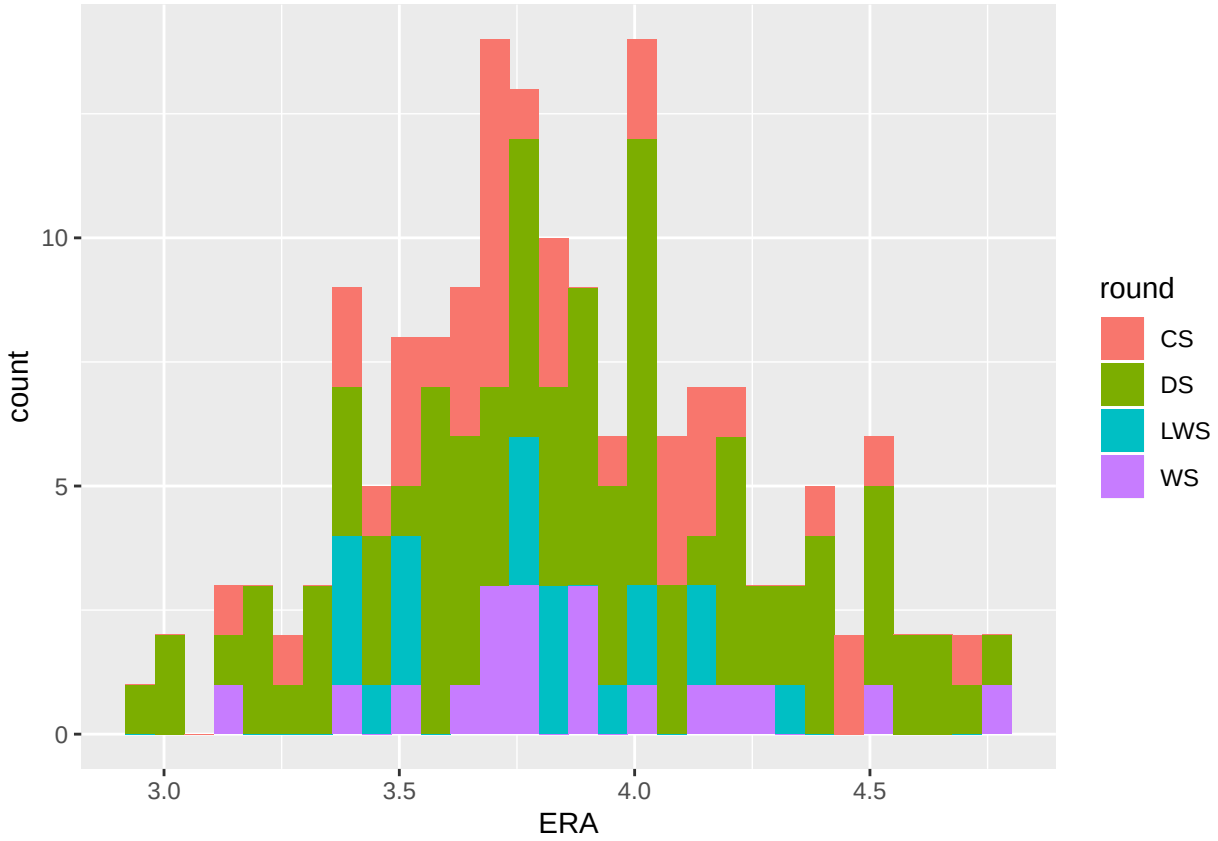

```
ggplot(teams, aes(x = HR, fill = round)) +
  geom_histogram() # bimodal, maybe a few outliers, don't seem to matter much either
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



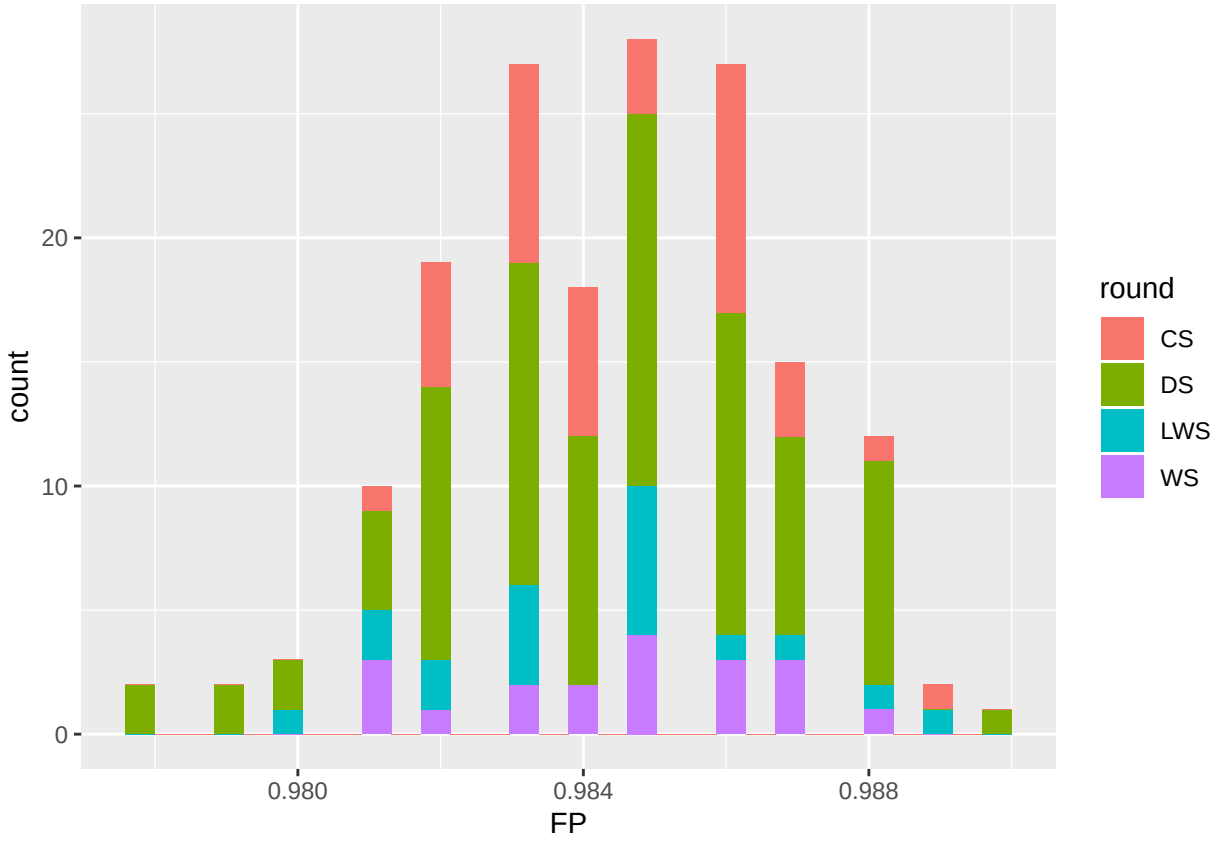

```
ggplot(teams, aes(x = ERA, fill = round)) +
  geom_histogram() # relatively normal,
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



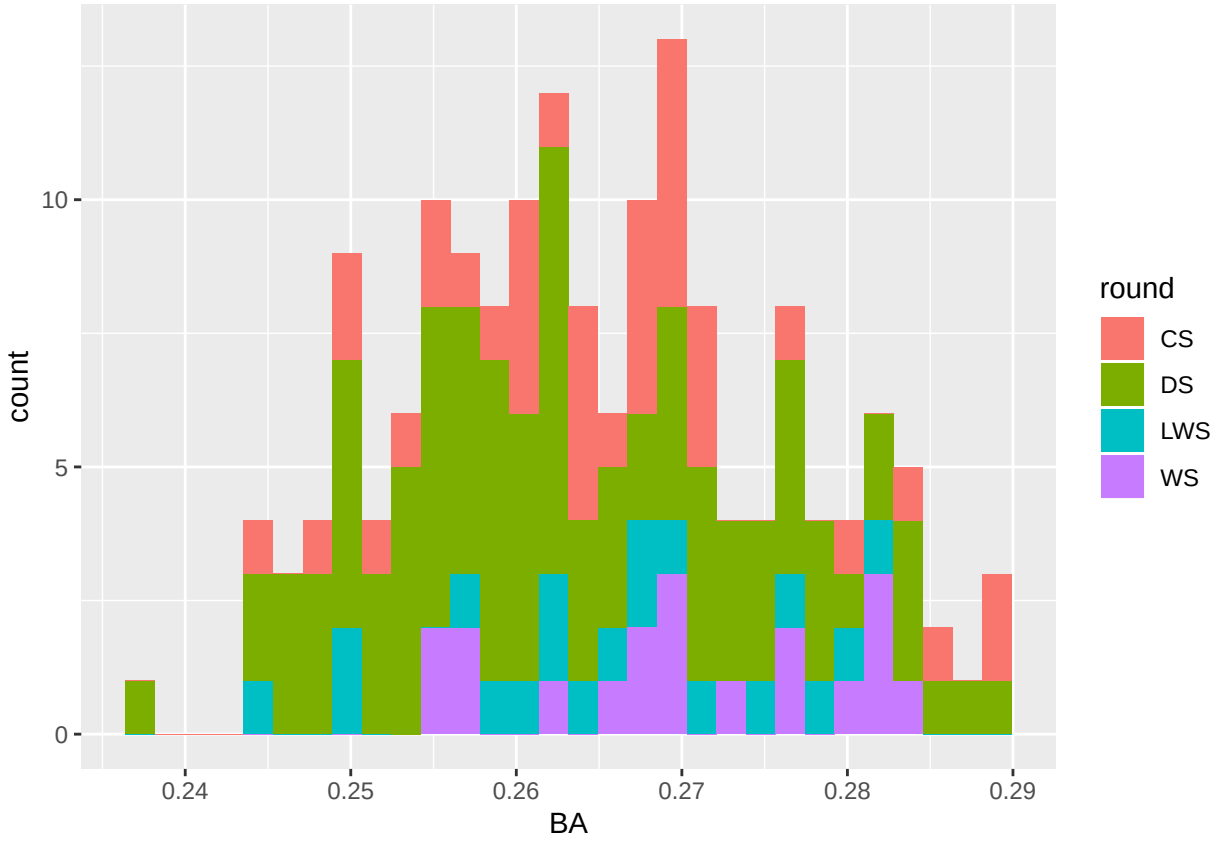

```
ggplot(teams, aes(x = FP, fill = round)) +
  geom_histogram() # fine
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



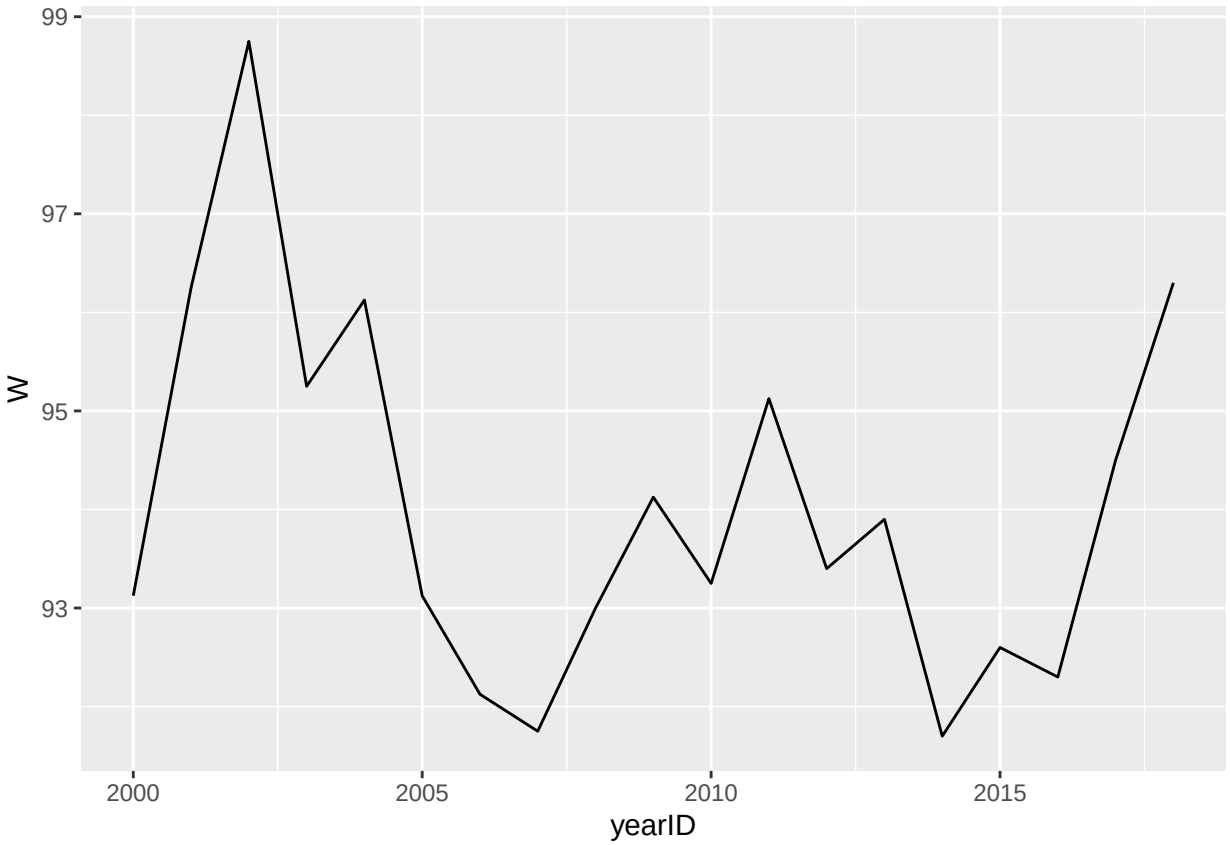

```
ggplot(teams, aes(x = BA, fill = round)) +
  geom_histogram() # pretty noisy, one outlier, seems to matter the most
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

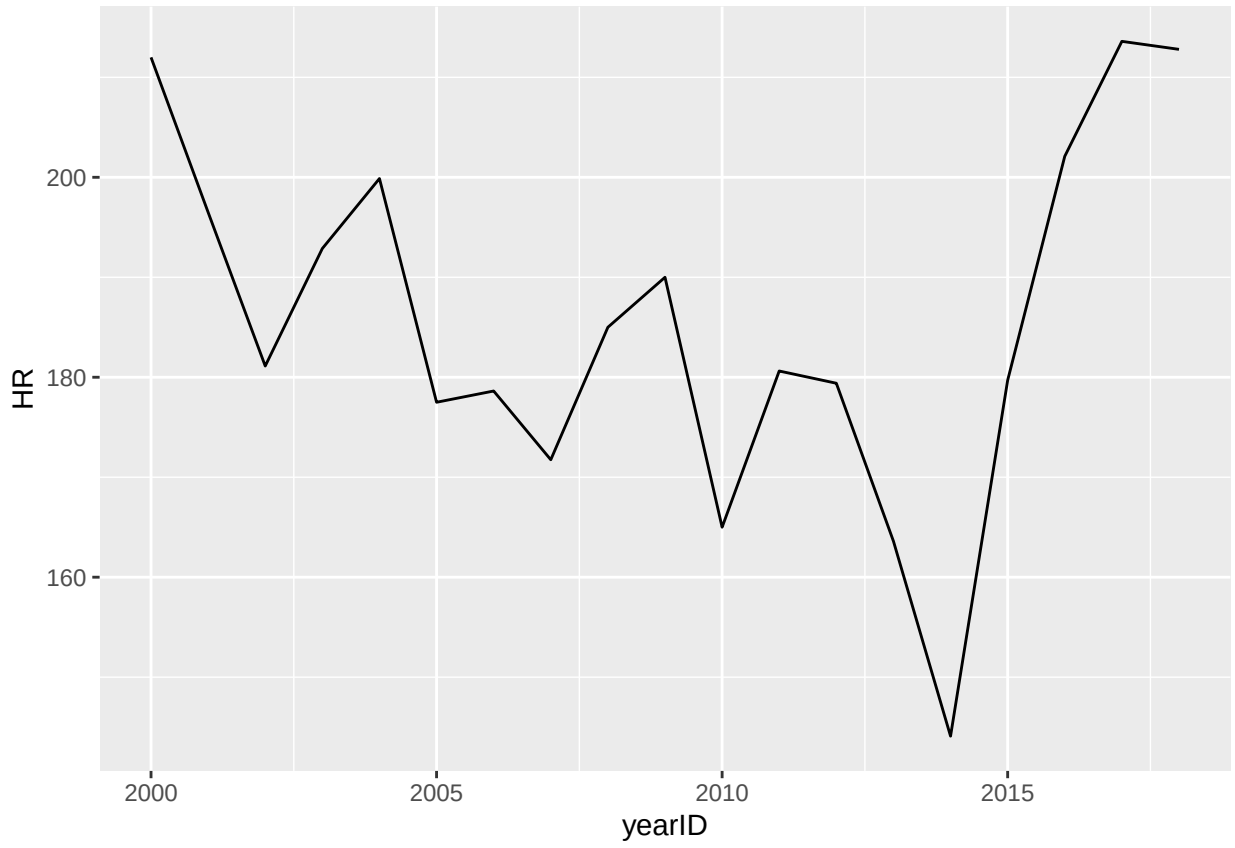


One phenomena of particular importance to explore is the shift of our numeric features over time. We are not incorporating time into our model, meaning the mold of a "World Series caliber" ballclub in the year 2019 may be quite different from that of the year 2000. We do not have a way around this, as we suffer from a lack of data in general for this type of analysis (We do not have enough data to start in the year 2010, for example). As mentioned earlier, the amount of homeruns that major league ballclubs hit has gone up over time. Additionally, we see that ERA has steadily decreased over time, perhaps an indication of the rising prevalence of lockdown bullpen pitching. The biggest concern we have after viewing these time series representations is the overinflation of homeruns. Lots of teams hit lots of homeruns in today's game. This might lead to multiple teams getting high predictions (Multiple teams being projected to go far in the postseason) due to the sheer amount of homeruns being hit, dwarfing what we see in the early 2000s. We can continue with the model-building but must be aware of this potential issue.

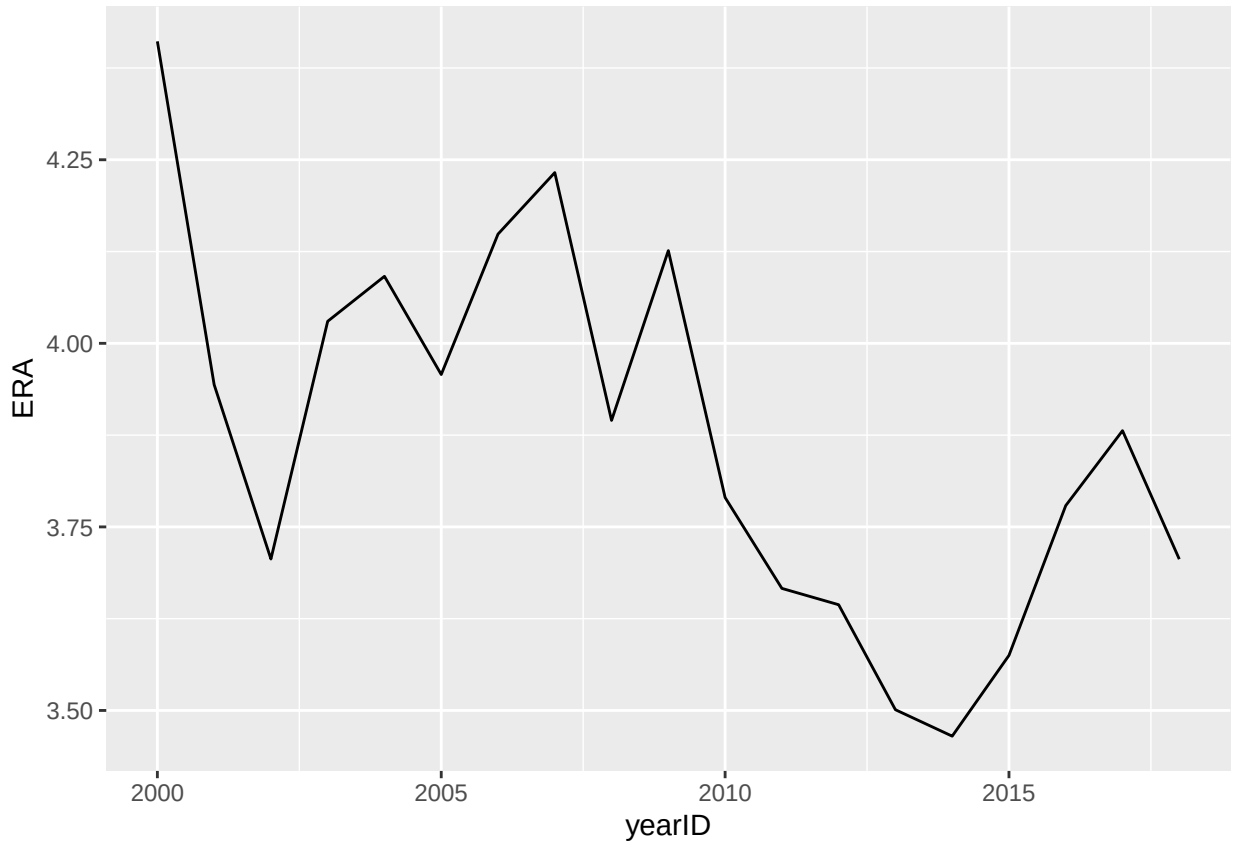
```
# Plotting time series for each numeric feature
teams %>%
  group_by(yearID) %>%
  summarise(W = mean(W)) %>%
  ggplot(aes(x = yearID, y = W)) + # pretty random
  geom_line()
```



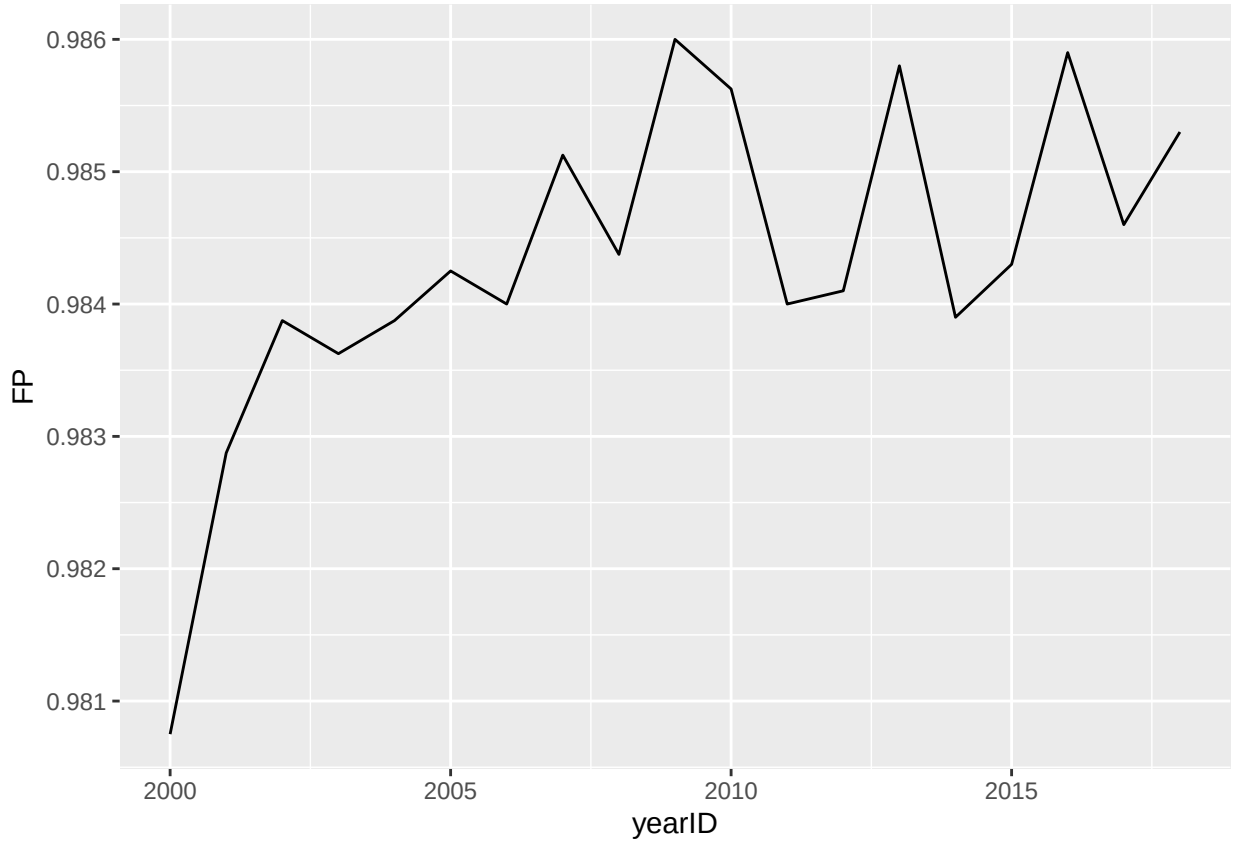

```
teams %>%
  group_by(yearID) %>%
  summarise(HR = mean(HR)) %>%
  ggplot(aes(x = yearID, y = HR)) + # up and down but trends downward, and then back up post-2014
  geom_line()
```



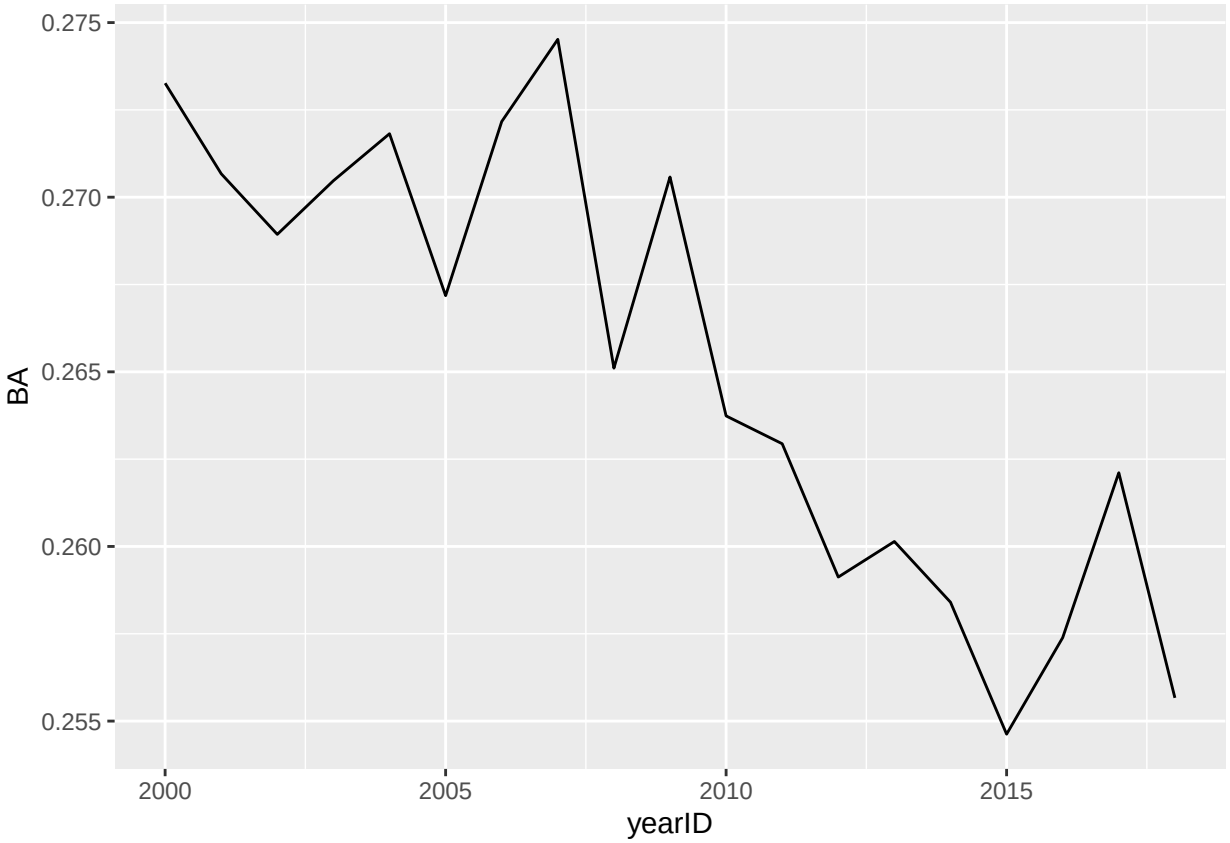

```
teams %>%
  group_by(yearID) %>%
  summarise(ERA = mean(ERA)) %>%
  ggplot(aes(x = yearID, y = ERA)) + # steady decline over time (better bullpen pitching?)
  geom_line()
```




```
teams %>%
  group_by(yearID) %>%
  summarise(FP = mean(FP)) %>%
  ggplot(aes(x = yearID, y = FP)) + # pretty stable as expected
  geom_line()
```




```
teams %>%
  group_by(yearID) %>%
  summarise(BA = mean(BA)) %>%
  ggplot(aes(x = yearID, y = BA)) + # steady decline over time
  geom_line()
```



Model Building

We now enter the modeling stage and build our ordinal logistic regression model. As a light refresher, the regular logit function shows the log odds of being in a particular class as a function of a linear combination of our predictors. The ordinal logit function is quite similar, with only a slight difference. Rather than the log odds of being in a particular class, we are predicting the log odds of being less than or equal to a particular class. This can be written mathematically as follows:

$logit(P(Y <= j)) = \beta_0 - \beta_1 X_1 - ... - \beta_p X_p$

In the following code block, we build the model and obtain parameter estimates as well as an AIC value for model comparison purposes. Taking a look at the parameter estimates at the bottom, we must recall these are coefficients for the odds. Thus, a one unit increase in ERA would yield a 0.295 increase in odds of getting to a higher round. This makes sense, as higher ERA should lessen chances of advancing. Initially, the massive coefficient for batting average is a little jarring, but this is because the interpretation is quite strange: for a one unit increase in batting average, we expect a 3.73e22 increase in odds of advancing to a further round. But a one unit increase in batting average would be over 1.000, which does not make any practical sense in the first place. As for our five variables, three of them appear to be statistically significant: ERA, fielding percentage, and batting average.

```
suppressPackageStartupMessages(library(MASS))
#suppressPackageStartupMessages(library(AICcmodavg))

# makes our target variable an ordered factor
teams$round <- factor(teams$round, order = TRUE, levels = c("DS", "CS", "LWS", "WS"))
```

```r
# Builds the model and prints the summary
mod <- polr(round ~ W + HR + ERA + FP + BA, data = teams, Hess = TRUE)
summary(mod) # AIC = 392.4725
```

```
## Call:
## polr(formula = round ~ W + HR + ERA + FP + BA, data = teams,
##     Hess = TRUE)
##
## Coefficients:
##         Value Std. Error  t value
## W   -0.004759   0.033181  -0.1434
## HR   0.004473   0.005243   0.8531
## ERA -1.220853   0.445356  -2.7413
## FP  -3.851338   1.833354  -2.1007
## BA  51.973190   0.198359 262.0153
##
## Intercepts:
##        Value    Std. Error t value
## DS|CS    5.8144   1.8528     3.1382
## CS|LWS   6.9125   1.8627     3.7110
## LWS|WS   7.7728   1.8751     4.1452
##
## Residual Deviance: 376.4725
## AIC: 392.4725
```

```r
#AICc(mod)  AICc = 392.6924

# 95% confidence interval for our parameter estimates (Exponentiated to undo the log scale)
interval <- exp(confint.default(mod))
print(interval)
```

```
##            2.5 %       97.5 %
## W   9.325871e-01 1.062129e+00
## HR  9.942132e-01 1.014859e+00
## ERA 1.232261e-01 7.061193e-01
## FP  5.845634e-04 7.725723e-01
## BA  2.528285e+22 5.501911e+22
```

```r
exp(coef(mod)) # wins and HR seem to be insignificant estimators
```

```
##            W           HR          ERA           FP           BA
## 9.952526e-01 1.004483e+00 2.949785e-01 2.125129e-02 3.729665e+22
```

Because two of our predictors were not statistically significant, we build our model three more times, dropping wins, homeruns, and both, to see how this impacts model performance. After building the models and comparing AIC scores, all four models are quite close, with the best model having dropped both predictors. However, all models are close enough that we do not lose much prdictive power by including all five variables, so we end up continuing on with our original, full model. Additionally, acknowledging the lack of data we have, we also compared the models using the AICc metric, which is more robust to small datasets. This evaluation metric yielded the same results, and they are commented out below in the code block.

```r
mod2 <- polr(round ~ HR + ERA + FP + BA, data = teams, Hess = TRUE) # Removes wins
summary(mod2) # AIC = 390.4935
```

```
## Call:
## polr(formula = round ~ HR + ERA + FP + BA, data = teams, Hess = TRUE)
##
## Coefficients:
##         Value Std. Error  t value
## HR   0.004178   0.004991   0.8371
## ERA -1.184451   0.423389  -2.7975
## FP  -4.251371   0.761962  -5.5795
## BA  50.904941   0.154549 329.3783
##
## Intercepts:
##        Value    Std. Error t value
## DS|CS    5.6705   0.7709     7.3560
## CS|LWS   6.7684   0.7777     8.7034
## LWS|WS   7.6285   0.7944     9.6034
##
## Residual Deviance: 376.4935
## AIC: 390.4935
```

```r
#AICc(mod2) AICc = 390.6642

mod3 <- polr(round ~ W + ERA + FP + BA, data = teams, Hess = TRUE) # Removes HRs
summary(mod3) # AIC = 391.1524
```

```
## Call:
## polr(formula = round ~ W + ERA + FP + BA, data = teams, Hess = TRUE)
##
## Coefficients:
##         Value Std. Error  t value
## W    0.008108    0.03171   0.2557
## ERA -0.964634    0.40298  -2.3937
## FP  -7.390003    1.80104  -4.1032
## BA  46.547964    0.22681 205.2294
##
## Intercepts:
##        Value    Std. Error t value
## DS|CS    2.2599   1.8197     1.2419
## CS|LWS   3.3531   1.8296     1.8327
## LWS|WS   4.2114   1.8416     2.2868
##
## Residual Deviance: 377.1524
## AIC: 391.1524
```

```r
#AICc(mod3) AICc = 391.3231

mod4 <- polr(round ~ ERA + FP + BA, data = teams, Hess = TRUE) # Removes wins and HRs
summary(mod4) # AIC = 389.1799
```

```
## Call:
```

```
## polr(formula = round ~ ERA + FP + BA, data = teams, Hess = TRUE)
##
## Coefficients:
##      Value Std. Error  t value
## ERA -1.042     0.4725 -2.20520
## FP  -2.909    66.5477 -0.04371
## BA  50.399    16.2303  3.10525
##
## Intercepts:
##        Value   Std. Error t value
## DS|CS   6.6311 65.9797     0.1005
## CS|LWS  7.7269 65.9810     0.1171
## LWS|WS  8.5867 65.9812     0.1301
##
## Residual Deviance: 377.1799
## AIC: 389.1799
```

```
#AICc(mod4) AICc = 389.3077
```

Predictions

With our model built, we can now use it to make predictions about the 2019 postseason. We manually impute the statistics for the ten teams that made the playoffs this year, and use our model to predict which class they each belong to. One thing to note here is that every team will get projected to lose in the first round. This is because there is an inherent imbalance in the data. There are far more teams in our dataset that lose in the first round, then win the world series (Remember, only one team can win it each year). And there are not enough distinctive features in the 19 teams that have won since 2000 for our model to adequately detect a champion from our ten teams.

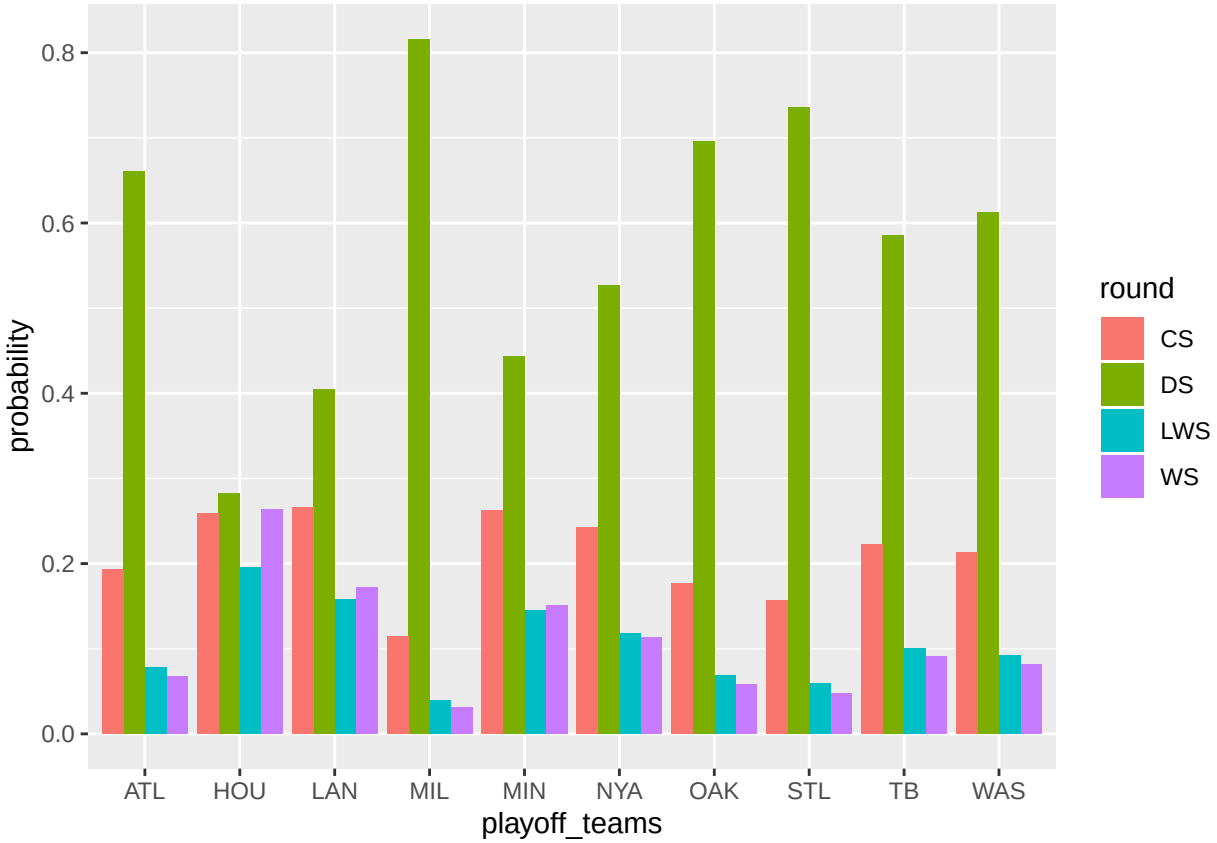
After predicting using our model, we obtain four probabilities for each team. These are the probabilities a team loses in the division series, champion series, world series, or winning it all (Wildcard games are coded as division series). Thus, we have a few different ways to compare our ten teams. We can compare their division series probabilities, and the team with a lower probability will advance. We can compare their champion series probabilities, and see which team is higher, and that team will advance. We can also compare world series chances, and advance teams based on that. The general rule we establish here is that when we are in the division series, we compare teams based on their probability to advance to the champion series. When we are in the champion series, we compare teams based on their probability to advance to the world series. When we are in the world series, we compare teams based on their probability to win the world series. This seems vexing but makes more sense when you view the bar chart below, allowing you to visually compare the ten teams.

```
suppressPackageStartupMessages(library(tidyr))

# Dataframe for 2019 playoff teams
teams_19 <- c("WAS", "LAN", "ATL", "STL", "MIL", "NYA", "HOU", "TB", "OAK", "MIN")
new_teams <- data.frame(W = c(93, 106, 97, 91, 89, 103, 107, 96, 97, 101),
                        HR = c(231, 279, 249, 210, 250, 306, 288, 217, 257, 307),
                        ERA = c(4.27, 3.37, 4.19, 3.80, 4.40,
                                4.31, 3.66, 3.65, 3.97, 4.18),
                        FP = c(0.985, 0.982, 0.987, 0.989, 0.983,
                               0.983, 0.988, 0.985, 0.986, 0.981),
                        BA = c(0.265, 0.257, 0.258, 0.245, 0.246,
                               0.267, 0.274, 0.254, 0.249, 0.270))
```

```r
# Makes predictions with the full model
predictions <- predict(mod, new_teams, type = "probs")
team_predictions <- cbind(teams_19, predictions)

colnames(team_predictions)[1] <- "playoff_teams"

team_predictions <- as.data.frame(team_predictions)
team_predictions_gathered <- gather(team_predictions, key = "round",
                                    value = "probability", -playoff_teams)
```

```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```

```r
team_predictions_gathered$probability <- as.numeric(team_predictions_gathered$probability)
team_predictions_gathered$probability <- round(team_predictions_gathered$probability, 3)

# Plots the probabilities for each team
ggplot(team_predictions_gathered, aes(x = playoff_teams, y = probability, fill = round)) +
  geom_bar(stat = 'identity', position = position_dodge())
```



To further describe the bar chart above, we see four bars for each team. Each bar represent the probability estimated by our model for a different round of the playoffs. As expected, the green bar, corresponding to the division series, is quite high for each team. As discussed earlier, this is due to our model predicting every team to lose in the first round, because of the imbalance of our data. If you look at the purple bar,

representing the world series chances, we see that our model gives the Houston Astros the greatest chance at winning the world series, supposedly facing off against the Los Angeles Dodgers from the National League.

Now, we can finally predict the plight of the 2019 postseason:

Wildcard matchups:

WAS VS. MIL –winner: WAS

OAK VS. TB –winner: TB

NLDS:

LAD VS. WAS –winner: LAD

ATL VS. STL –winner: ATL

ALDS:

HOU VS. TB –winner: HOU

MIN VS. NYY –winner: MIN (well that didn't age well)

NLCS:

LAD VS. ATL –winner: LAD

ALCS:

HOU VS. MIN –winner: HOU

WS:

HOU VS. LAD –winner: HOU

Ding ding ding! We have a theoretical winner and they are the Houston Astros. We should not be surprised at all by this prediction, as the Astros dominate almost every single cateogory we have used in our model. But if there's one thing we can all be sure about, its that we cannot quantify the factor of unpredictability and craziness that we see each October. Also, maybe one day, against the grain of every single predictive model out there, my Miami Marlins can win the world series once again. Thank you for reading!

Testing Assumptions

At the end of the analysis, here we test our assumptions that we need to make for our model to provide valid results. There are two major assumptions that we are checking here, the first of which being the proportional odds assumption. This assumption essentially states that our the difference between rounds is equivalent. In other words, the difference between losing in the division round vs losting in the champion round is the same as the difference between losing in the world series and losing in the champion round. We check this assumption in the following code. We are looking at the assortment of numbers in the output, and trying to identify any major discrepancies that may undermine our assumption. We see that a few of the numbers are unstable in each of the predictor variables we have used, which may suggest that our assumption of proportional odds ratios is not completely met.

```
suppressPackageStartupMessages(library(Hmisc))

# Tests the proportional odds assumption
sf <- function(y) {
  c('Y>=1' = qlogis(mean(y >= 1)),
    'Y>=2' = qlogis(mean(y >= 2)),
    'Y>=3' = qlogis(mean(y >= 3)),
    'Y>=4' = qlogis(mean(y >= 4)))
}
```

```
(s <- with(teams, summary(as.numeric(round) ~ W + HR + ERA + FP + BA, fun=sf)))
```

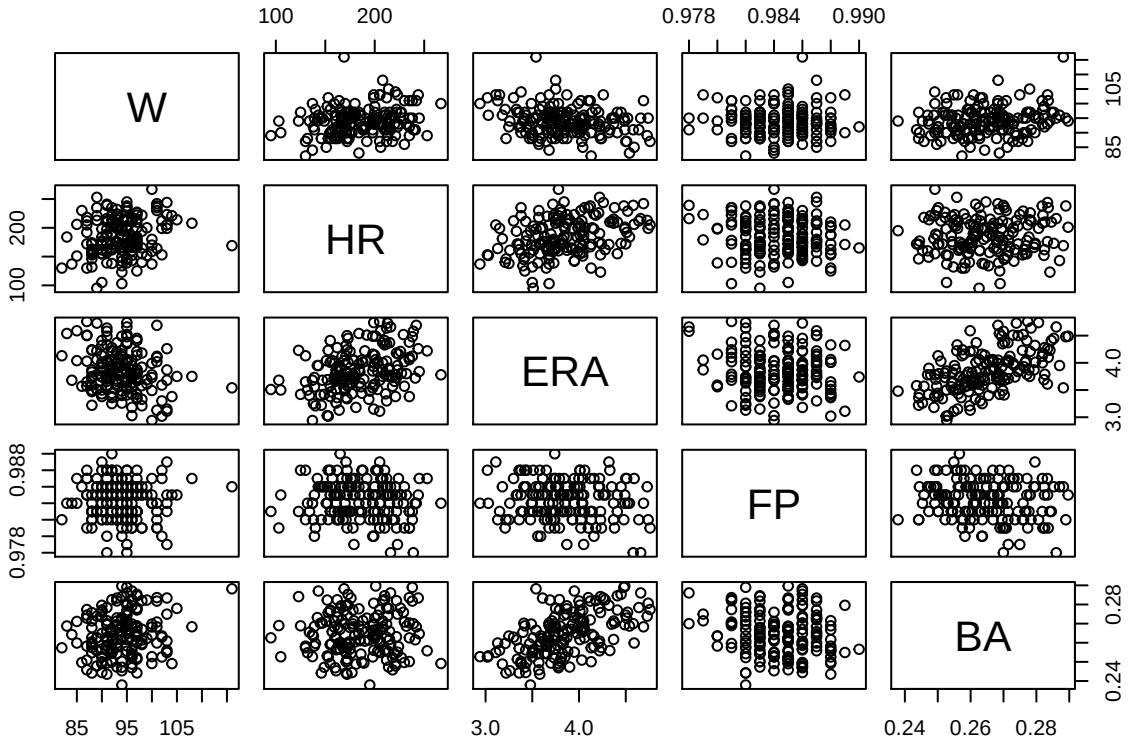
```
## as.numeric(round)     N= 166
##
## +-------+-------------+---+----+-----------+----------+---------+
## |       |             |N  |Y>=1|Y>=2       |Y>=3      |Y>=4     |
## +-------+-------------+---+----+-----------+----------+---------+
## |W      |[82, 92)     | 52|Inf |-0.38946477|-1.3156768|-2.240710|
## |       |[92, 95)     | 37|Inf |-0.38299225|-1.2878543|-2.110213|
## |       |[95, 98)     | 48|Inf | 0.16705408|-1.4663371|-2.708050|
## |       |[98,116]     | 29|Inf |-0.06899287|-0.6418539|-1.145132|
## +-------+-------------+---+----+-----------+----------+---------+
## |HR     |[ 95,163)    | 43|Inf |-0.42488319|-1.0678406|-1.637609|
## |       |[163,185)    | 41|Inf | 0.14660347|-1.5804504|-2.538974|
## |       |[185,212)    | 41|Inf |-0.14660347|-1.0033021|-1.974081|
## |       |[212,267]    | 41|Inf |-0.24512246|-1.2685113|-2.224624|
## +-------+-------------+---+----+-----------+----------+---------+
## |ERA    |[2.94,3.59)  | 42|Inf |-0.28768207|-1.1631508|-2.564949|
## |       |[3.59,3.82)  | 41|Inf | 0.24512246|-1.1314021|-1.580450|
## |       |[3.82,4.12)  | 42|Inf |-0.28768207|-1.1631508|-2.251292|
## |       |[4.12,4.76]  | 41|Inf |-0.34484049|-1.4170660|-1.974081|
## +-------+-------------+---+----+-----------+----------+---------+
## |FP     |[0.978,0.984)| 63|Inf |-0.15906469|-1.1631508|-2.251292|
## |       |[0.984,0.986)| 46|Inf |-0.17435339|-1.0414539|-1.897120|
## |       |0.986        | 27|Inf | 0.07410797|-1.7491999|-2.079442|
## |       |[0.987,0.990]| 30|Inf |-0.40546511|-1.1895841|-1.871802|
## +-------+-------------+---+----+-----------+----------+---------+
## |BA     |[0.238,0.256)| 42|Inf |-0.69314718|-1.7917595|-2.564949|
## |       |[0.256,0.264)| 41|Inf |-0.44628710|-1.4170660|-2.970414|
## |       |[0.264,0.273)| 42|Inf | 0.58778666|-1.0360919|-1.791759|
## |       |[0.273,0.290]| 41|Inf |-0.14660347|-0.7672552|-1.417066|
## +-------+-------------+---+----+-----------+----------+---------+
## |Overall|             |166|Inf |-0.16907633|-1.2144441|-2.045994|
## +-------+-------------+---+----+-----------+----------+---------+
```

Lastly, we test the assumption of multicollinearity of our variables. We do not want to have high correlation amongst our variables, which could bias the model. We do not see very strong correlations between our variables, meaning we have met this assumption of non-multicollinearity.

```
# Scatterplot matrix to check for multicollinearity
cor(teams[4:8])
```

```
##                W          HR        ERA          FP          BA
## W    1.00000000  0.21438122 -0.2509385 -0.01620415  0.19309374
## HR   0.21438122  1.00000000  0.3645350 -0.07820773  0.05367286
## ERA -0.25093849  0.36453502  1.0000000 -0.13714980  0.55833828
## FP  -0.01620415 -0.07820773 -0.1371498  1.00000000 -0.14171694
## BA   0.19309374  0.05367286  0.5583383 -0.14171694  1.00000000
```

```
pairs(~ W + HR + ERA + FP + BA, data = teams)
```



References

Much of the statistical process in this report is modeled off of the UCLA Institution for Digital Research & Education's example online. They provide a simple and concise explanation of all the methods used, and their template allowed a beginner like me to utilize a new type of statistical model: the ordinal logistic regression model.

Ordinal Logistic Regression. UCLA: Statistical Consulting Group. from (https://stats.idre.ucla.edu/r/dae/ordinal-logistic-regression/) (accessed October 13, 2019).

The data comes from the Lahman database, more spefically from the lahman package in R. The data can be obtained through the R package, as I have done in this report. It is also available through the following link, bringing you to the seanlahman.com download page:

(http://www.seanlahman.com/baseball-archive/statistics/)