BASEBALL-PROJ

Evan Hope

5/23/2022

```
# loading some packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

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

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(tidymodels)
```

```
## -- Attaching packages --------------------------------------- tidymodels 0.2.0 --

## v broom        0.8.0     v rsample      0.1.1
## v dials        0.1.1     v tune         0.2.0
## v infer        1.0.0     v workflows    0.2.6
## v modeldata    0.1.1     v workflowsets 0.2.1
## v parsnip      0.2.1     v yardstick    0.0.9
## v recipes      0.2.0

## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Search for functions across packages at https://www.tidymodels.org/find/
```

```
library(skimr)
library(patchwork)
library(janitor)
```

```
##
## Attaching package: 'janitor'

## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library(discrim)
```

```
##
## Attaching package: 'discrim'

## The following object is masked from 'package:dials':
##
##     smoothness
```

```
library(corrr)
```

```
##
## Attaching package: 'corrr'

## The following object is masked from 'package:skimr':
##
##     focus
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(tinytex)
# setting the seed
set.seed(325)
```

```
tidymodels_prefer()
```

My goal for this project is to import a custom data set and explore lesser known relationships to winning as well as to seek certain stats that are 'commonly' believed to be strongly correlated with winning and possibly finding their 'true' worth. It is important to note that when it comes to stats in baseball, it is useful to have the variable code book handy as well as a source to the more formal definitions of the stats I utilize in my models. To further simplify this, the typical threshold at which a team's winning percentage should be in order to have a strong chance at making playoffs is about 0.525 (85 regular season wins in a 162-game season). This allows to work with a simple 'True or False' response variable. Lets begin by reading in the data set.

```
# reading in the data

baseball_data <- read.csv("C:/Users/Ordai/OneDrive/Desktop/School/Stats/PSTAT 131/baseballproject_final
```

Cleaning the names of the variables and filtering out any missing values. Keep in mind that this data is from the year 2000-2021. With that being said, certain stats in baseball were not yet being recorded in the earlier years. I will ignore these observations.

```
baseball_data <- baseball_data %>%
  clean_names() %>%
  filter(!is.na(frm)) # filtering out missing values
```

I will now create two new variables. playoff_bound tells us whether or not that team met the winning percentage threshold 0f 0.525 . avg_age tells the average age of the whole team.

Additionally, I will be deselecting unimportant variables or variables that are no longer needed now that we have implemented our new variables.

Lastly, I will rename some of the variables so that they are easier to distinguish as an offensive or defensive stat. (NOTE: For coding effeciency purposes, variables with 'b_' in the beginning is offesive/batter stat and '_9' at the end indicates a defensive/pitcher stat. Not all variables will have this prefix/suffix)

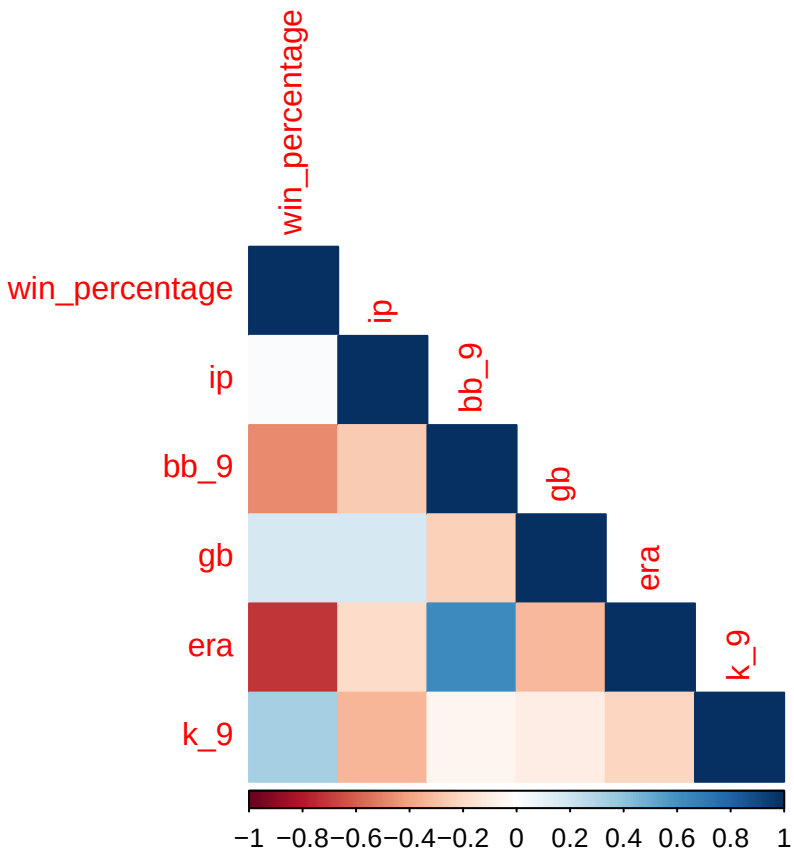
```
baseball_data <- baseball_data %>%
  mutate(playoff_bound = case_when(win_percentage >= 0.525 ~ 'True', win_percentage < 0.525 ~ 'False'),
         avg_age = (p_age + b_age)/2,
         playoff_bound = factor(playoff_bound),
         season = factor(season)) %>%
  select(-p_war, -b_war, -lob, -p_babip, -babip3, -c_str, -csw, -siera) %>%
  rename(b_k_rate = k) %>%
  rename(b_bb_rate = bb) %>%
  rename(b_bb_k = bb_k)
```

EXPLORATORY DATA ANALYSIS (EDA): based on the entire data set.

I will now create a few correlation matrices in order to find variables that have stronger correlations with winning percentages. Since my response variable is non-numeric, I will simply use the win_percentage variable as it closely resembles my response variable.

First a correlation matrix with some pitching stats.

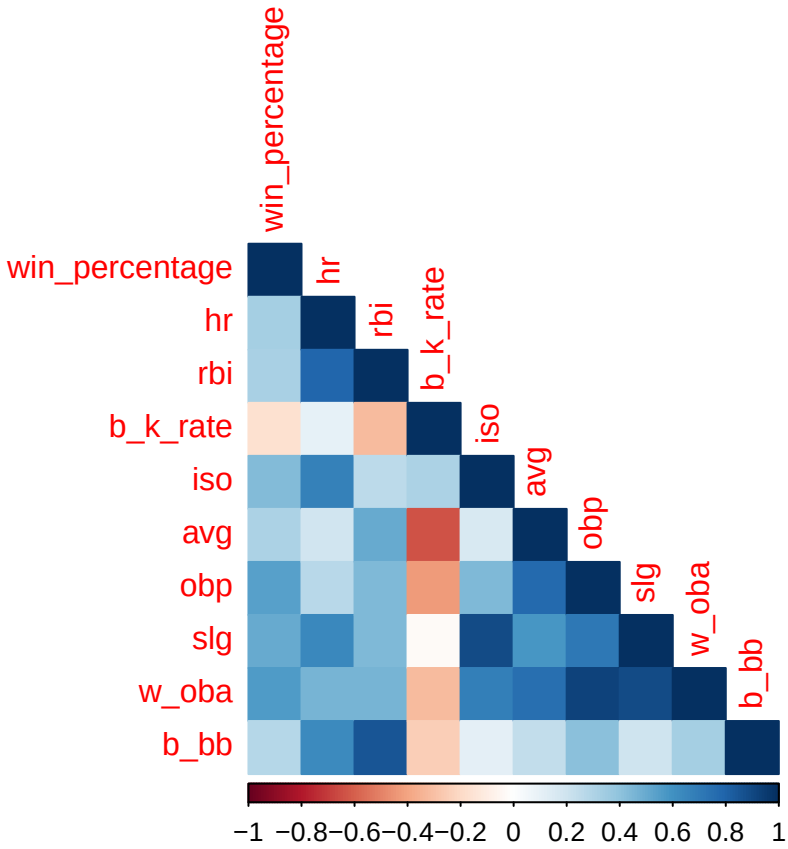
```
baseball_data %>%
  select(win_percentage, ip, bb_9, gb, era, k_9) %>%
  cor(use = "pairwise.complete.obs") %>%
  corrplot(type = 'lower', diag = TRUE,
           method = 'color')
```



I am most interested in the first column for each of these matrices as I begin to narrow down my favorite variables. As for this one, the first thing that catches my eye is the strong negative correleation between team era and winning percentage (as team era decreases, winning increases). The same goes for pitcher walk rates. These correlations are about what I had expected.

Lets move onto a correlation matrix involving some offensive metrics.

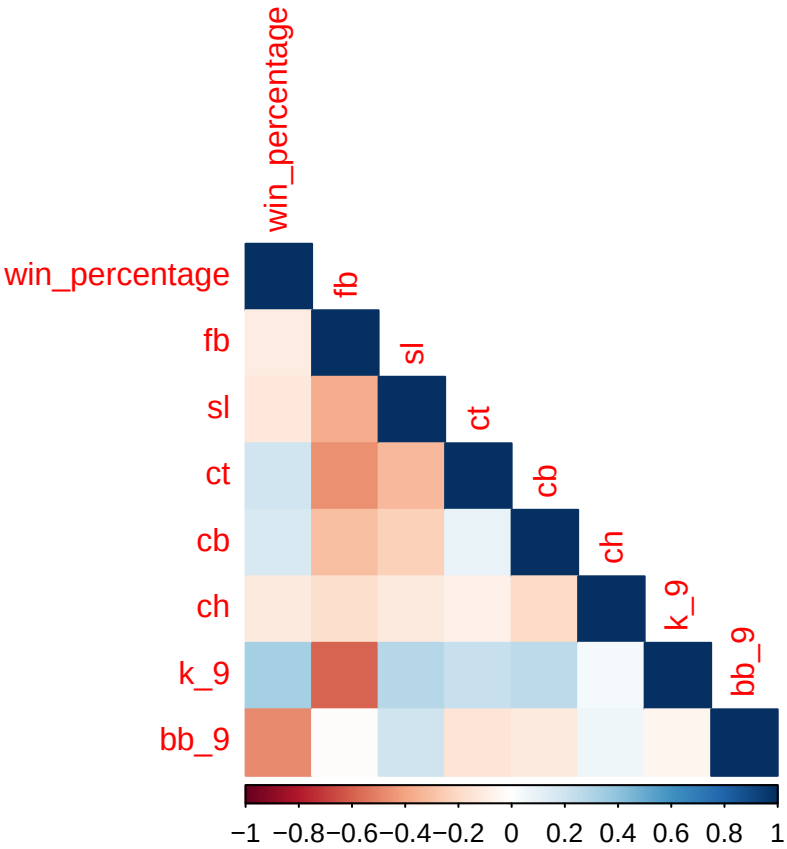
```
baseball_data %>%
  select(win_percentage, hr, rbi, b_k_rate, iso, avg, obp, slg, w_oba, b_bb) %>%
  cor(use = "pairwise.complete.obs") %>%
  corrplot(type = 'lower', diag = TRUE,
           method = 'color')
```



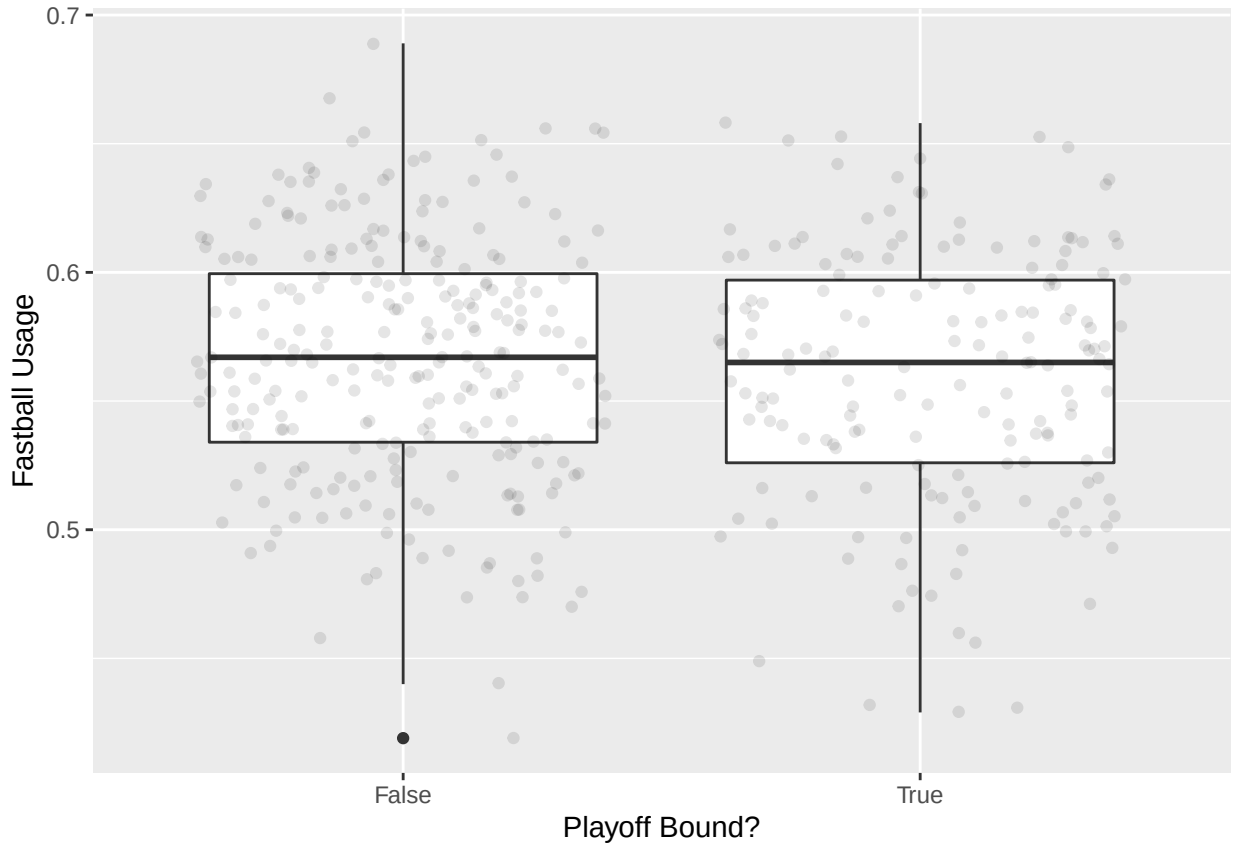
It is important to note that some of the variables in this matrix are used to calculate other variables in the same matrix. Hence why there is a very strong positive correlations between some of the variables. The point of this matrix is that I wanted to see which of the hitting stats had STRONGER correlations to winning. For example, homeruns appear to have a similar or perhaps lesser correlation to winning percentage than the walk rate (b_bb). This might come as a surprise. You would think that hitting a homerun would have a stronger relationship with winning than just simply getting on base because hitting a homerun automatically clears the bases for runs (points) while getting on base only provides a NON-GUARANTEED chance to score a run. This is likely because homeruns are a lot harder to come by than walks and the amount of walks a typical player draws during a season is almost always higher than the amount of homeruns hit.

Third, a correlation matrix with the types of pitches being thrown.

```
baseball_data %>%
  select(win_percentage, fb, sl, ct, cb, ch, k_9, bb_9) %>%
  cor(use = "pairwise.complete.obs") %>%
  corrplot(type = 'lower', diag = TRUE,
           method = 'color')
```

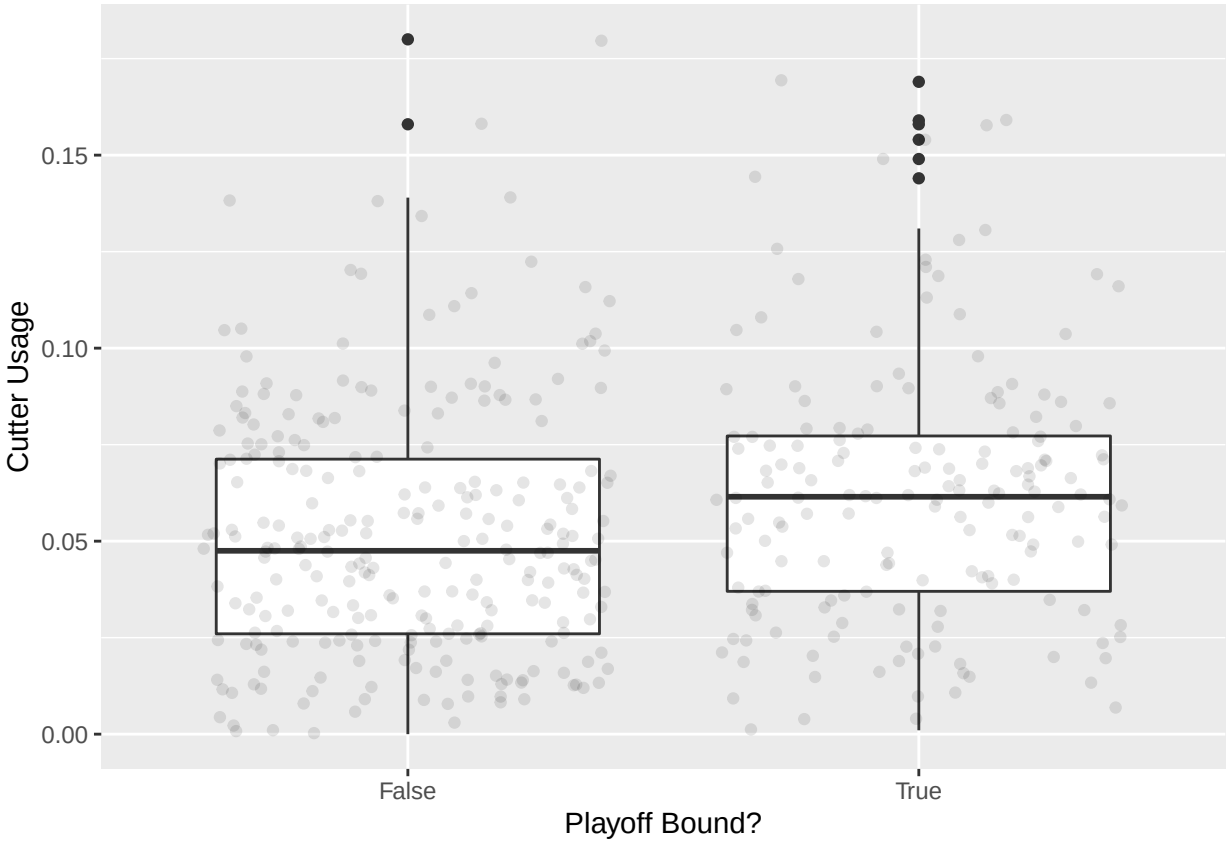


This is where things get a bit more interesting. This matrix tells me that the rate at which a team throws fastballs (fb) is slightly negatively correlated with winning while the cut-fastball (ct) has a slight positive correlation with winning percentage. Lets investigate these pitches a tad further.




```
## Warning: Removed 4 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```

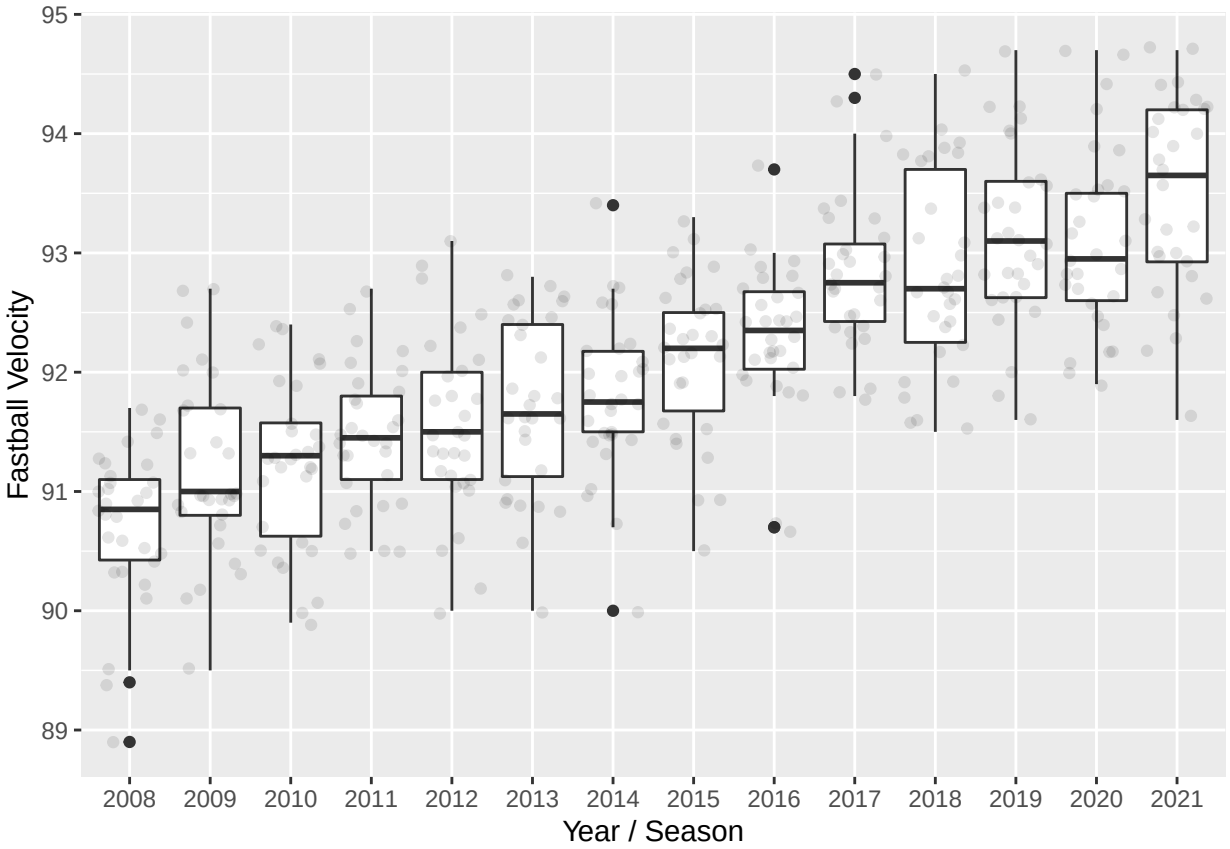


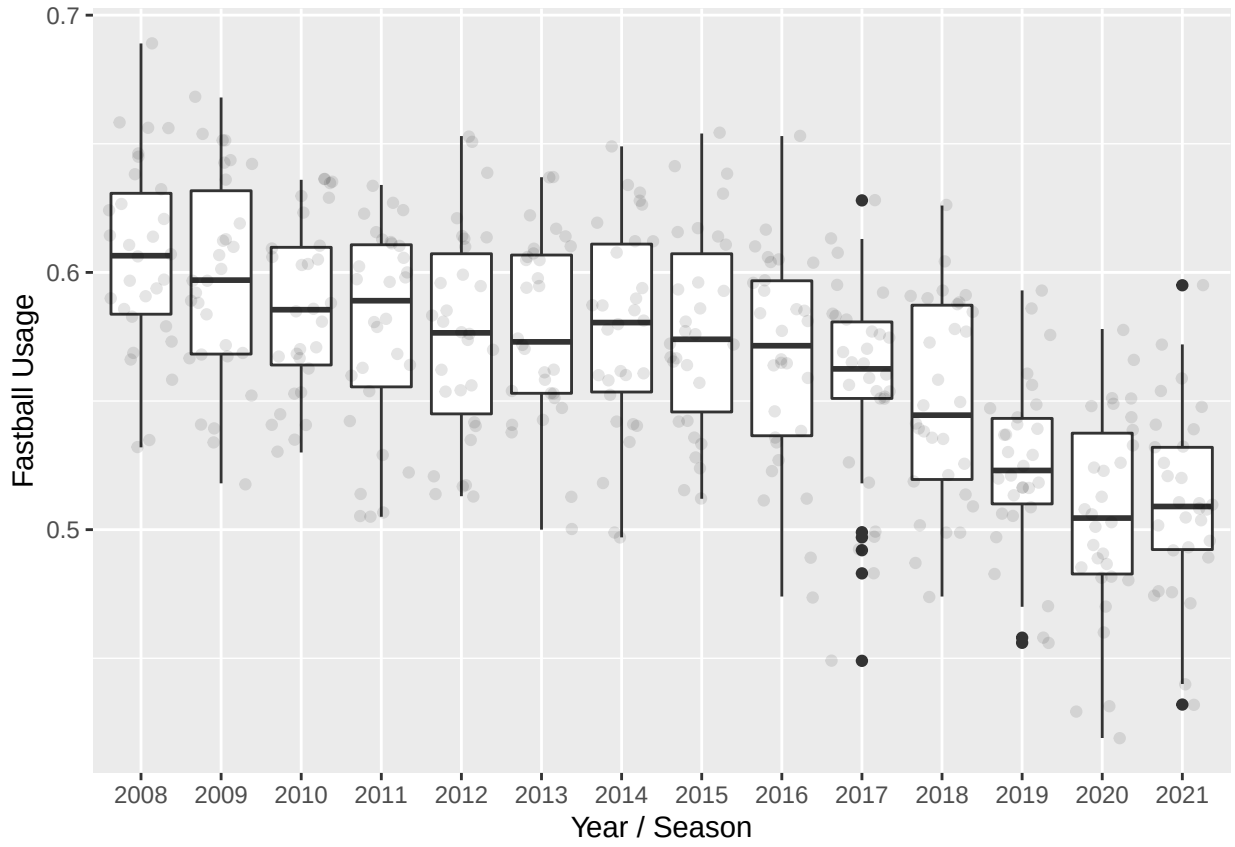
As we can see, there IS NOT a very noticeable difference in whether or not a team is playoff bound depending on how many fastballs are thrown throughout the season. However, there is an apparent relationship between throwing more cutters and being playoff bound.

This could indicate that a fastball with 'cutting' movement (cutter) is more effective than a 'straight' fastball. Another interesting note to make here is the relationship between the fastball usage rate and k_9 (strikeouts per 9 innings). There is a very strong negative correlation between the two while slider, cutter, and curveball rates all have positive correlations. On the other hand, there is little to no correlation between fastball usage and walks while cutters and curveballs tend to lead to more walks. This may hint that the fastball is a more controllable pitch to get ahead in the count with to avoid falling behind and walking batters while the remaining pitches are less controllable but provide more strikeouts. And as we can see above in the first matrix, strikeouts have a positive relationship with being playoff bound.

Before moving on, lets look at two more box plots involving fastballs and cutters over time.

```
ggplot(baseball_data, aes(factor(season), f_bv)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1) +
  xlab("Year / Season") +
  ylab("Fastball Velocity")
```

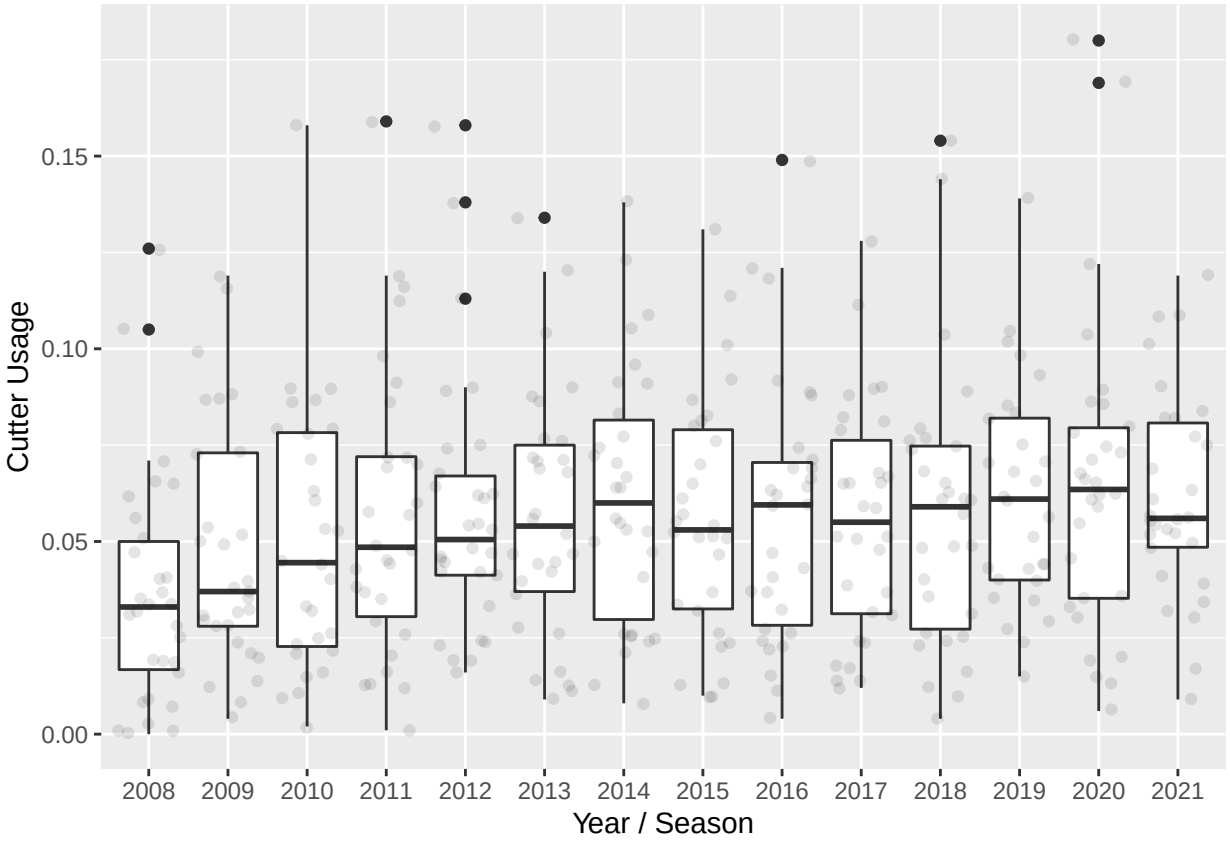

Fastball Velocity
95
94
93
92
91
90
89
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
Year / Season




```
ggplot(baseball_data, aes(factor(season), ct)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1) +
  xlab("Year / Season") +
  ylab("Cutter Usage")
```

```
## Warning: Removed 4 rows containing non-finite values (stat_boxplot).
```

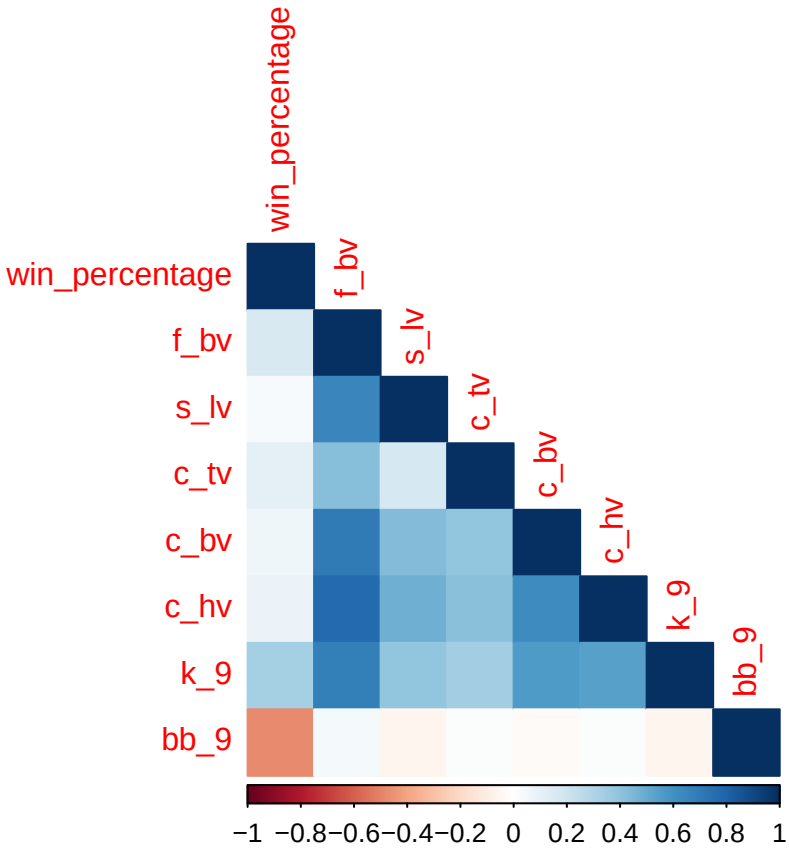
```
## Warning: Removed 4 rows containing missing values (geom_point).
```



Sure enough it looks like teams have already picked up on this trend between fastballs, cutters and winning. From the year 2000, the cutter is being thrown more and the normal fastball is being thrown less despite the average velocity of the fastball increasing year by year! This could be due to the amount of wear and tear throwing harder does to the human body leading to more injuries and ultimately more losing.

Now a matrix with the average velocities of these pitches.

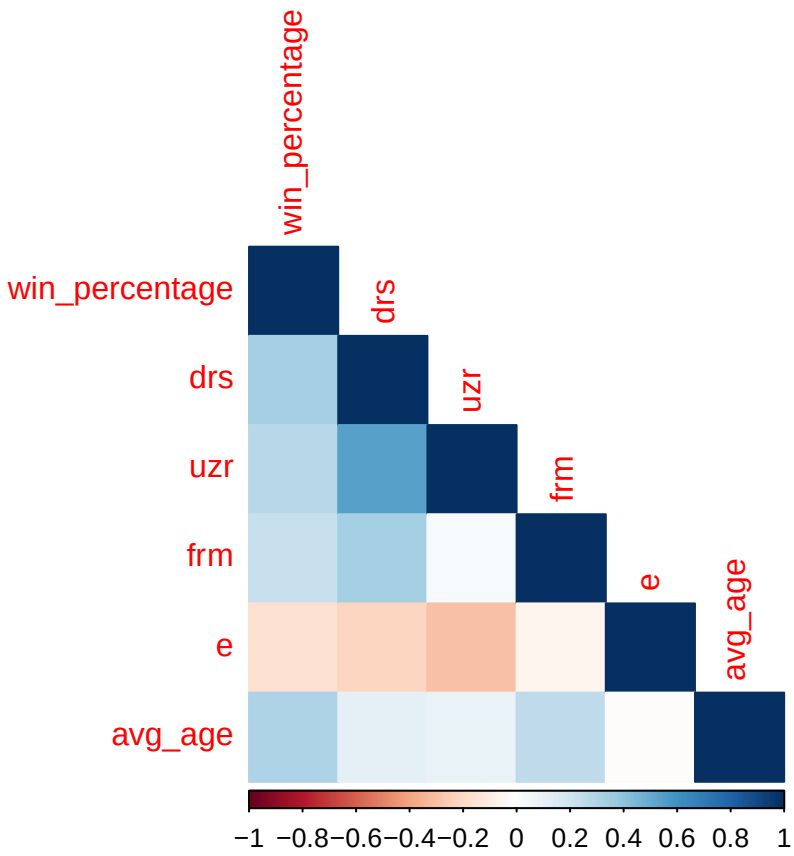
```
baseball_data %>%
  select(win_percentage, f_bv, s_lv, c_tv, c_bv, c_hv, k_9, bb_9) %>%
  cor(use = "pairwise.complete.obs") %>%
  corrplot(type = 'lower', diag = TRUE,
           method = 'color')
```



It is a bit surprising to see that the average velocities of each pitch has little to no correlation at all to winning even though there are strong correlations between each of the velocities and the strikeout / walk rates! This may hint at the 'execution' of a pitch located in the right spot being more important than the differentiating speeds. Both rates, however, are correlated to the winning percentage to a higher degree. For this reason, I will leave out the average velocity for each pitch type for the remainder of the project. Including them may also lead to overfitting.

Lastly, I will now look at some defensive metrics as well the average age of the team.

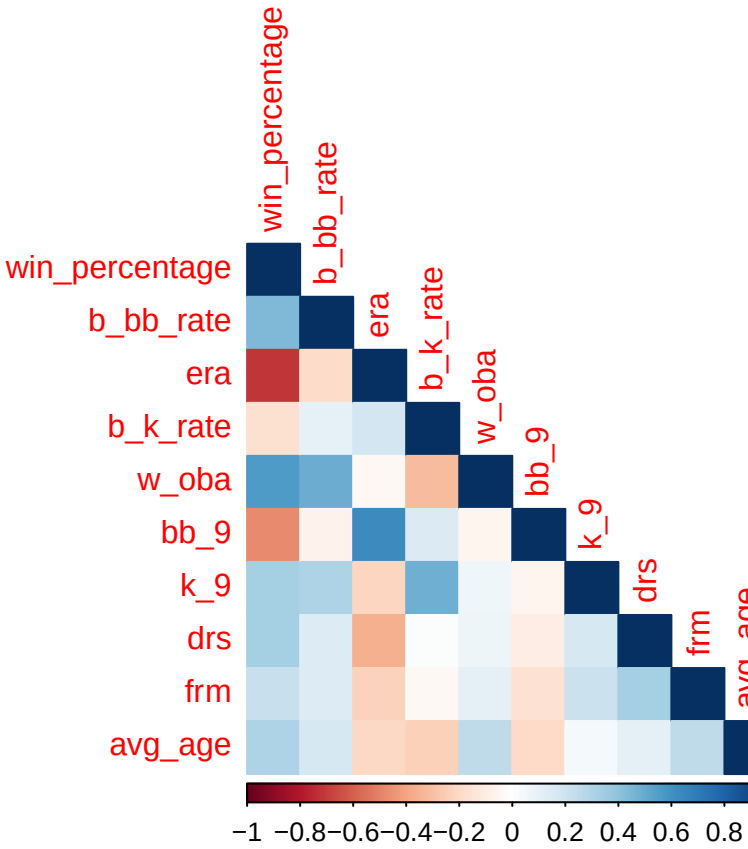
```
baseball_data %>%
  select(win_percentage, drs, uzr, frm, e, avg_age) %>%
  cor(use = "pairwise.complete.obs") %>%
  corrplot(type = 'lower', diag = TRUE,
           method = 'color')
```



The defense metrics correlate with winning the way I had expected. Better defensive ratings and less errors leads to more winning. However, there exists a correlation I was not expecting. The average age of a team is POSITIVELY correlated with winning! Typically I wouldn't expect any correlation, but if I was told there was a correlation, I would have expected teams with lower average ages to be positively correlated with winning as younger players tend to be less injury prone and in a prime athletic state. This correlation matrix says otherwise. Perhaps it could be team leadership and a stronger veteran presence among the 'older' teams. Or it could be the additional years of experience endured by the older players that provided them with more time to fine tune their skills at the professional level.

Now that I've looked at some correlations between a lot of the variables I was initially interested in, I want to narrow it down to my favorite variables. I will then create one last correlation matrix to find any other correlations between my final selection of variables that havent yet been grouped together.

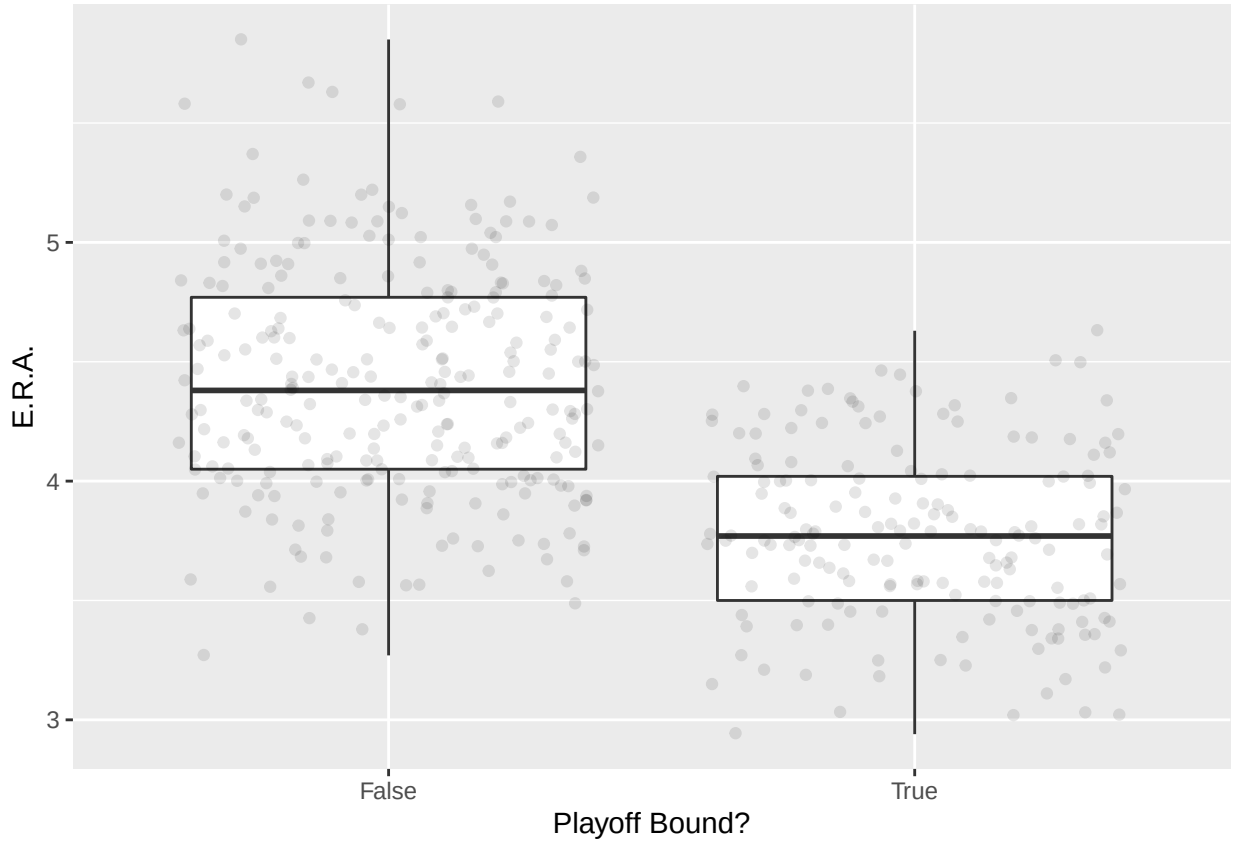
Below is final correlation matrix with the variables that peaked my most interest along with my final data set. Please refer to my codebook txt file to get a slightly better understanding of the variables that will be used in



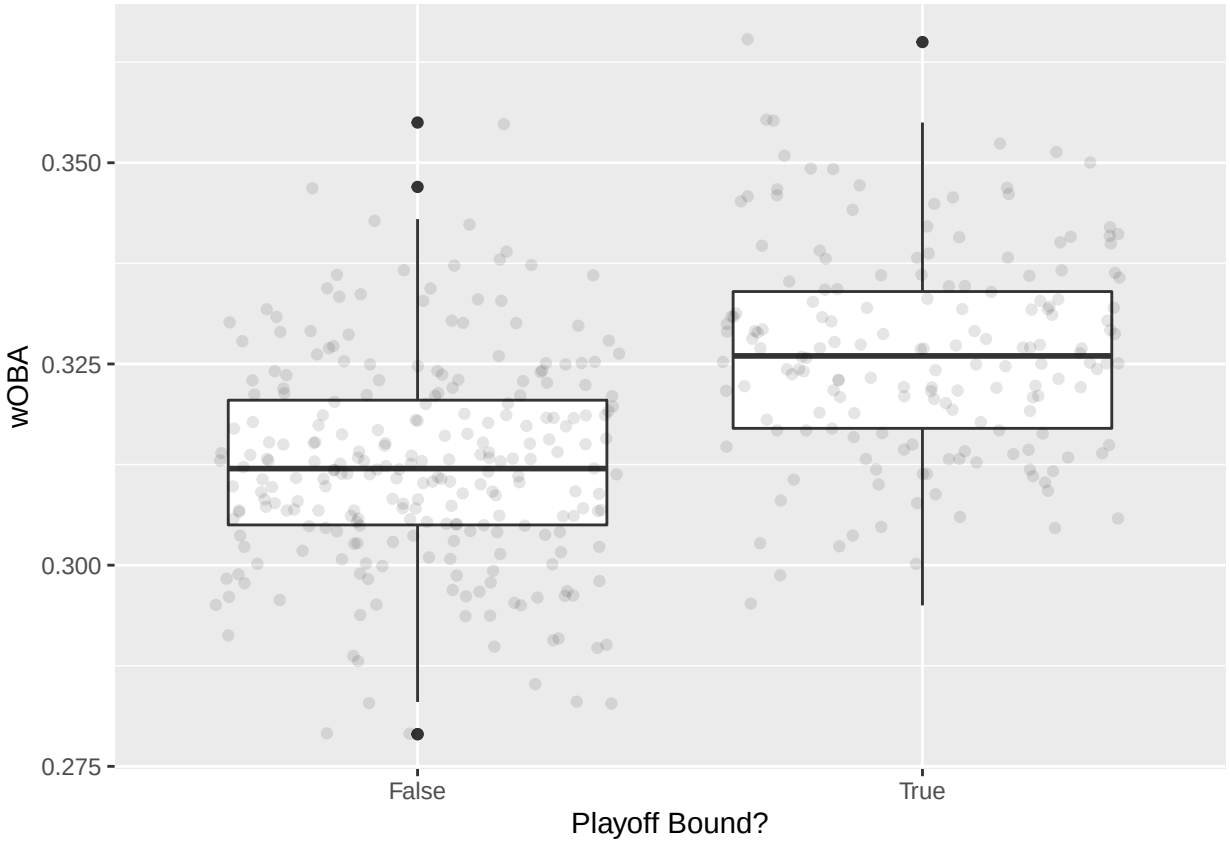
my recipe and for the rest of the project.

The most noticeable relationships are the negative correlation between era and winning_percentage as well as the positive correlation between w_oba and winning_percentage. Lets investigate these correlations further.

```
ggplot(baseball_data, aes(factor(playoff_bound), era)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1) +
  xlab("Playoff Bound?") +
  ylab("E.R.A.")
```



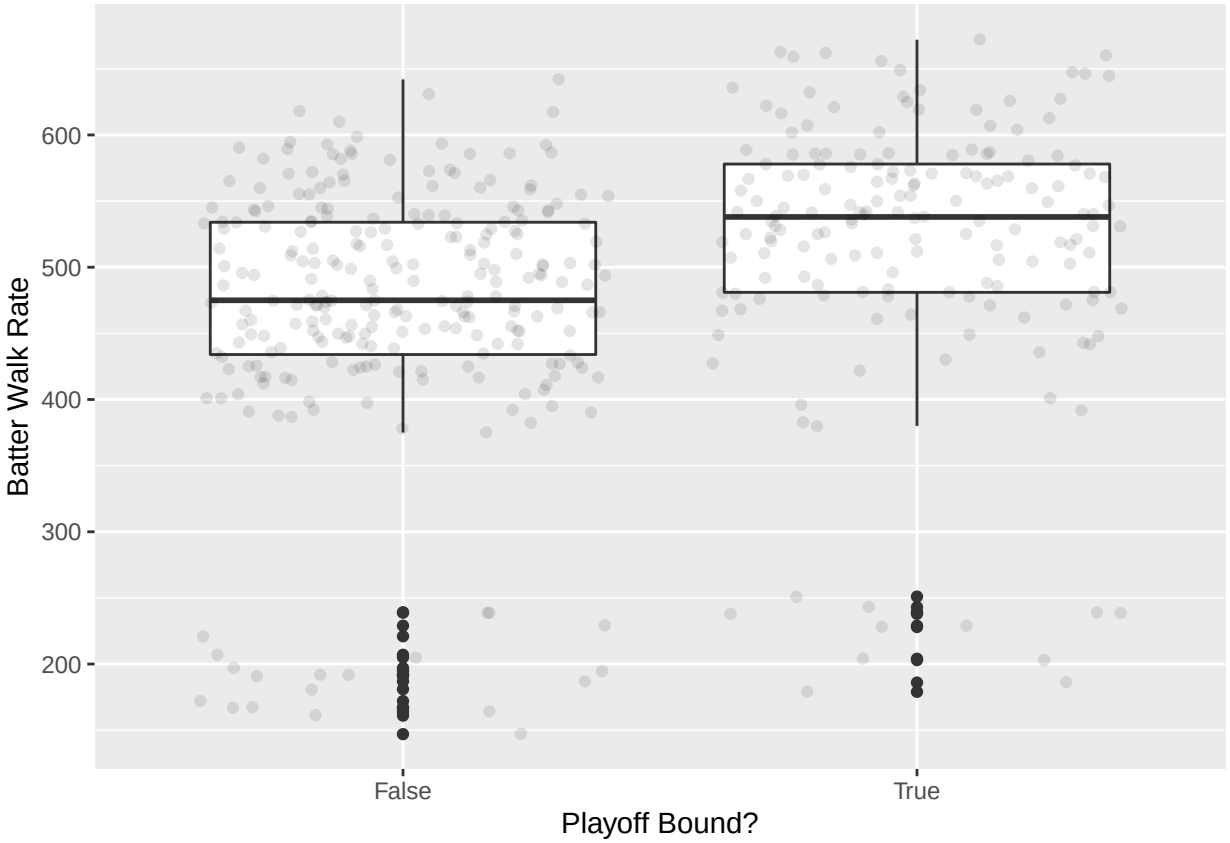

```
ggplot(baseball_data, aes(factor(playoff_bound), w_oba)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1) +
  xlab("Playoff Bound?") +
  ylab("wOBA")
```



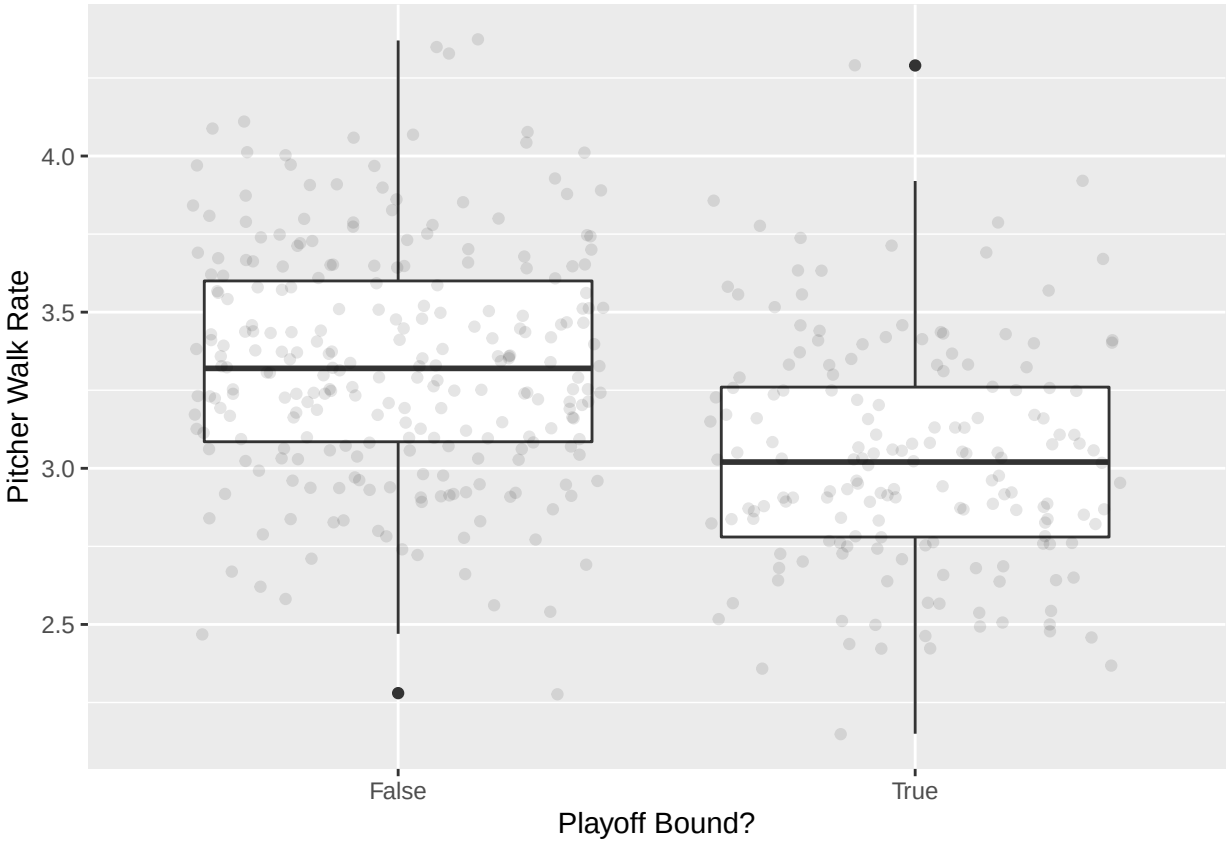
Simply put, having a pitching staff capable preventing runs and a batting lineup capable of getting on base and hitting for power. (power as an umbrella term for: double, triple, and homerun). This comes to no surprise and makes sense. Now is the time I look further. What other predictors have correlations with era and w_oba?

For w_oba it is easy to see that the batter walk rate is positively related. As for era, it is a similar case in terms of being influenced heavily by walk rates. When on defense, a walk may seem to be not much of a problem. However, the last correlation matrix tells that there is a strong positive correlation between walk rates and era. Lets visualize how both the batter's and pitcher's walk rate relates with winning in another box plot.

```
ggplot(baseball_data, aes(factor(playoff_bound), b_bb)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1) +
  xlab("Playoff Bound?") +
  ylab("Batter Walk Rate")
```



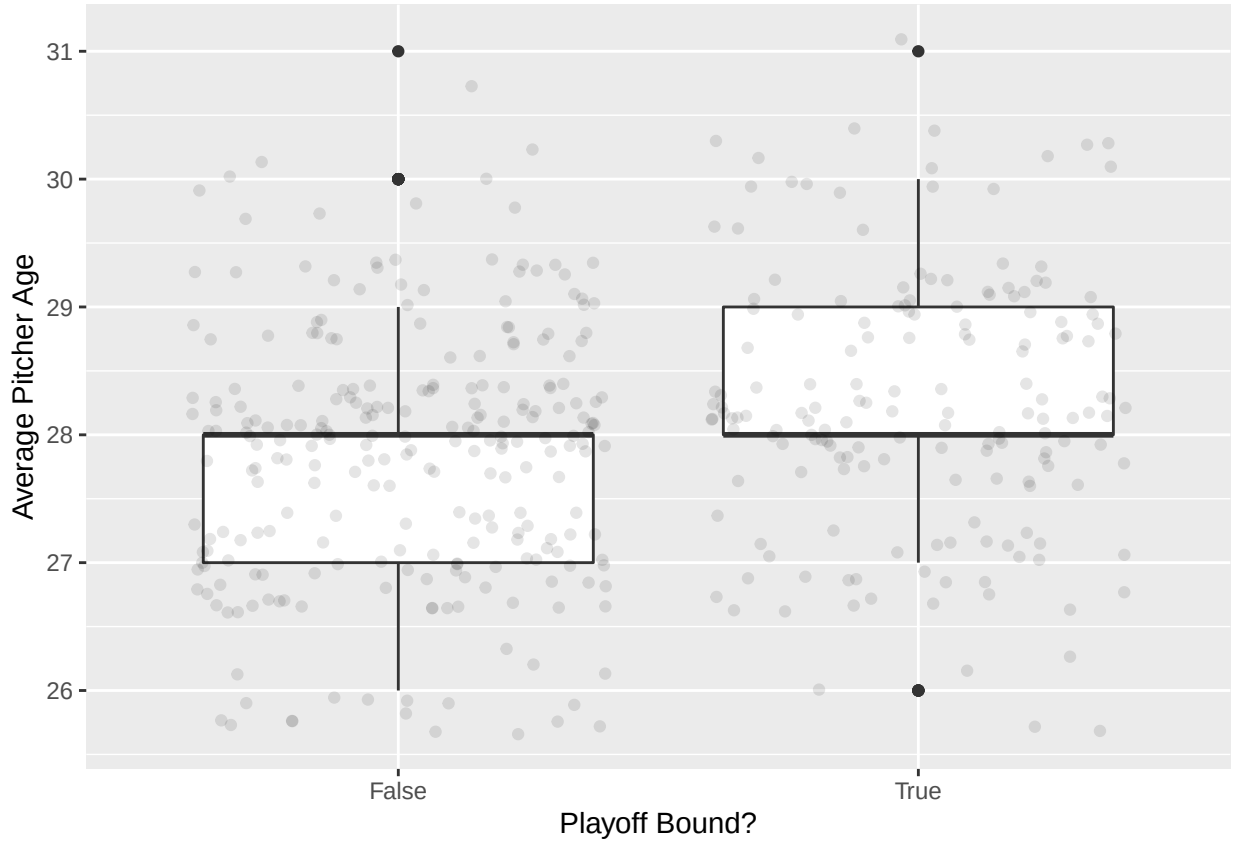

```
ggplot(baseball_data, aes(factor(playoff_bound), bb_9)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1) +
  xlab("Playoff Bound?") +
  ylab("Pitcher Walk Rate")
```



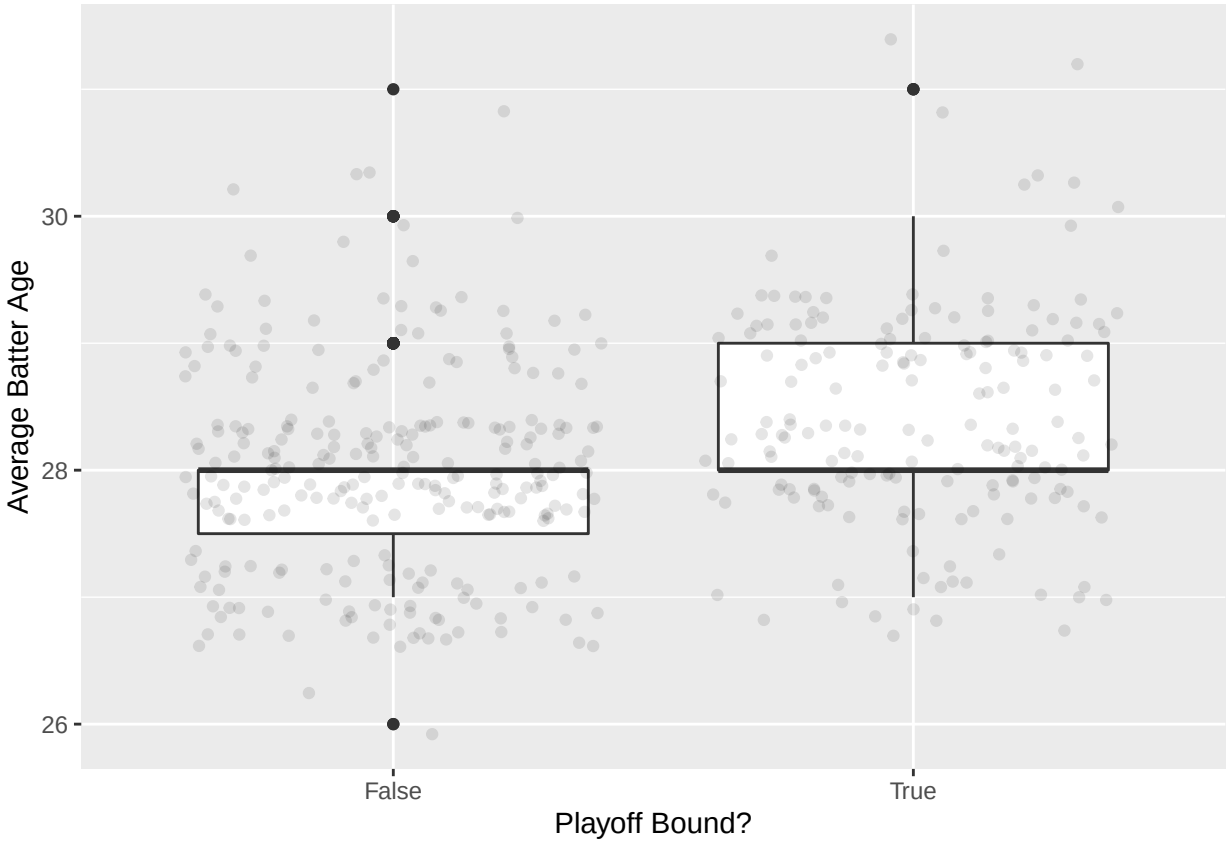
Here we can see that having higher batter walk rates and lower pitcher walk rates tend to lead to more winning by a noticeable margin. Why is this important? This puts a strong emphasis on drawing walks! And this is despite the number of walks holding 'less weight' in calculating w_oba while other hit outcomes (single, double, etc) hold 'greater weight'. This can be understood more by observing the wOBA formula on fangraphs.com that I have linked in my files. The point here is to avoid overlooking the true value of walks given their surprisingly large contribution to being playoff bound and wOBA.

Another interesting note is the average age having a positive relationship with w_oba. There can be many reasons for this. My theory is that older hitters have more experience and therefore are more likely to provide on offense. This is just a guess as 'experience' isnt necessarily measurable. However, the amount of years a player has been playing at the major league level is and one cannot gain major league experience without having major league playing time. Lets investigate this relationship by splitting the average age into average pitcher age and average batter age.

```
ggplot(baseball_data, aes(factor(playoff_bound), p_age)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1) +
  xlab("Playoff Bound?") +
  ylab("Average Pitcher Age")
```



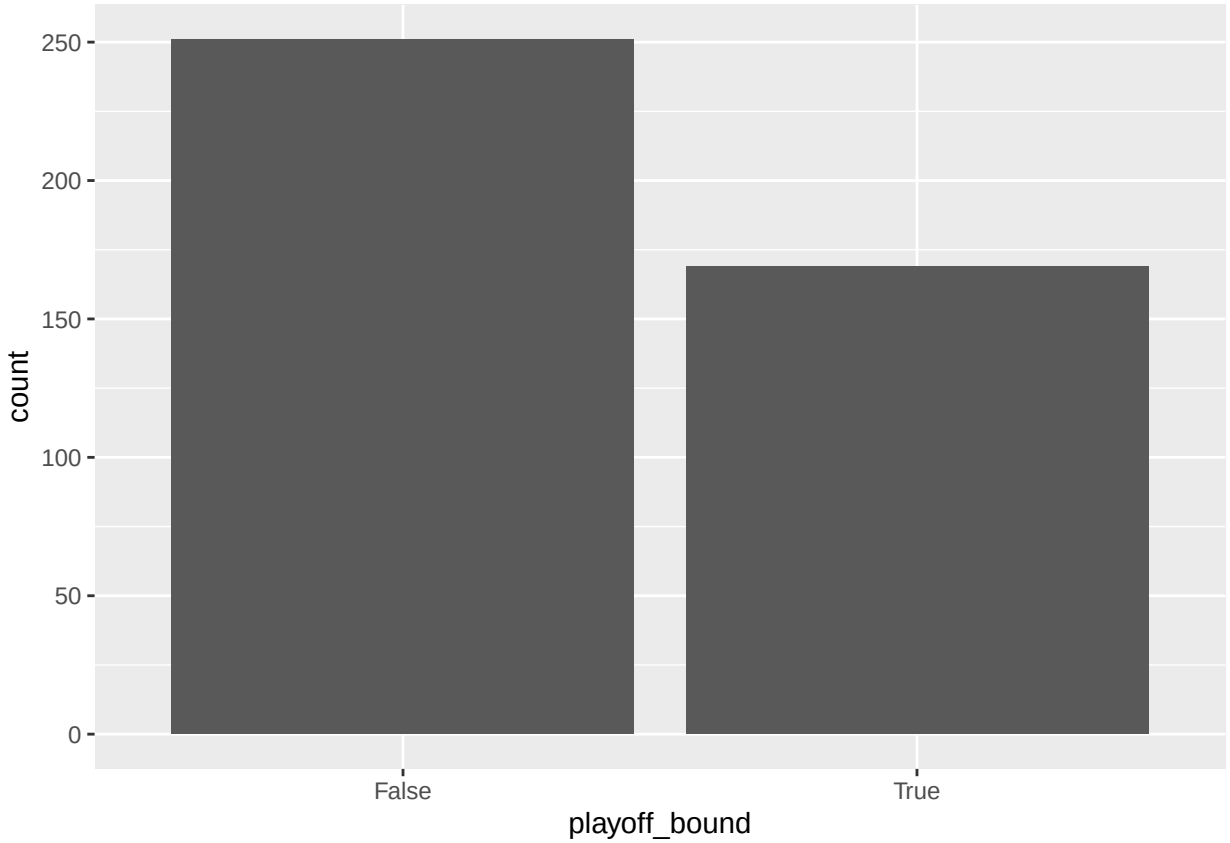

```
ggplot(baseball_data, aes(factor(playoff_bound), b_age)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1) +
  xlab("Playoff Bound?") +
  ylab("Average Batter Age")
```



The average age is negatively correlated with era. Younger pitchers tend to be better at run prevention. Could this be that their lack of time in the league means that there is less data gathered on that young pitcher making him less predictable? This would be my assumption for this relationship since I have not done enough research in this area.

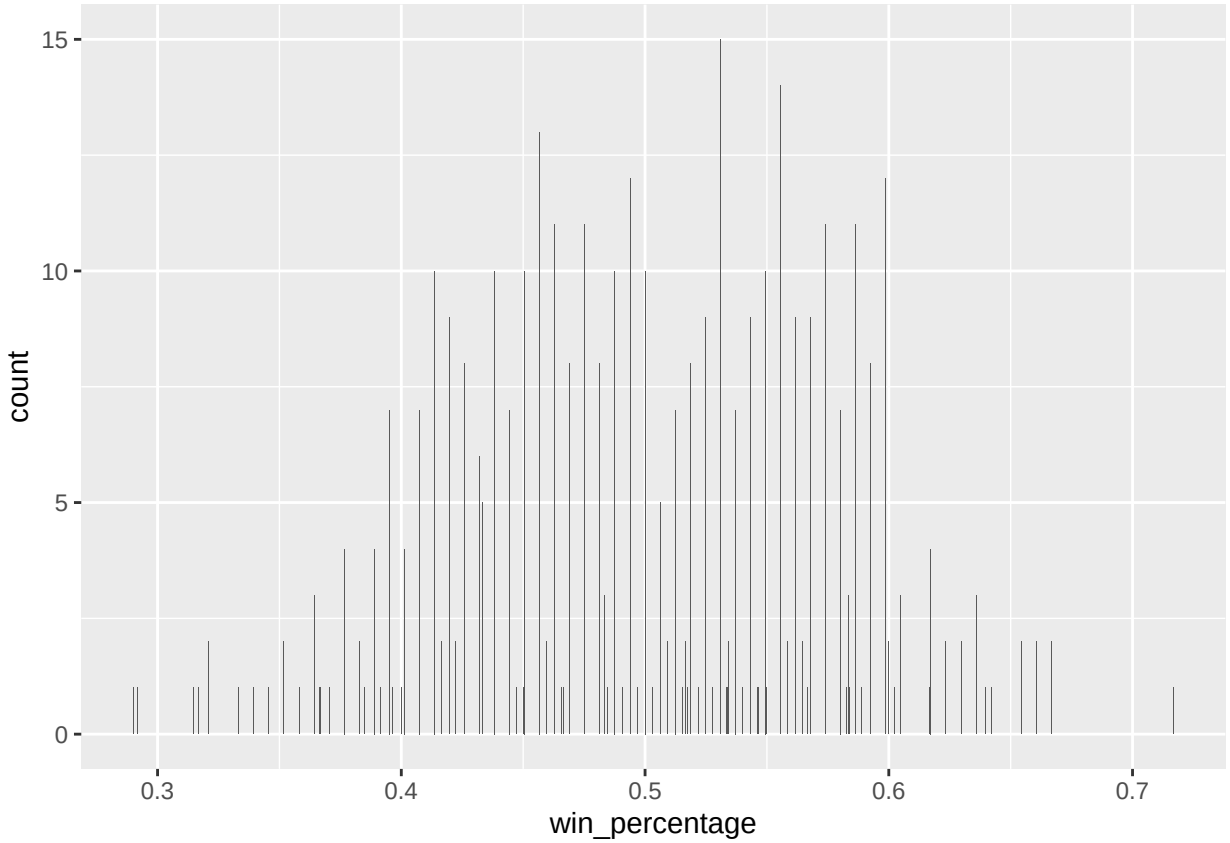
I will now look at the distribution of my chosen response variable, playoff_bound.

```
baseball_data %>%
  ggplot(aes(x = playoff_bound)) +
  geom_bar()
```



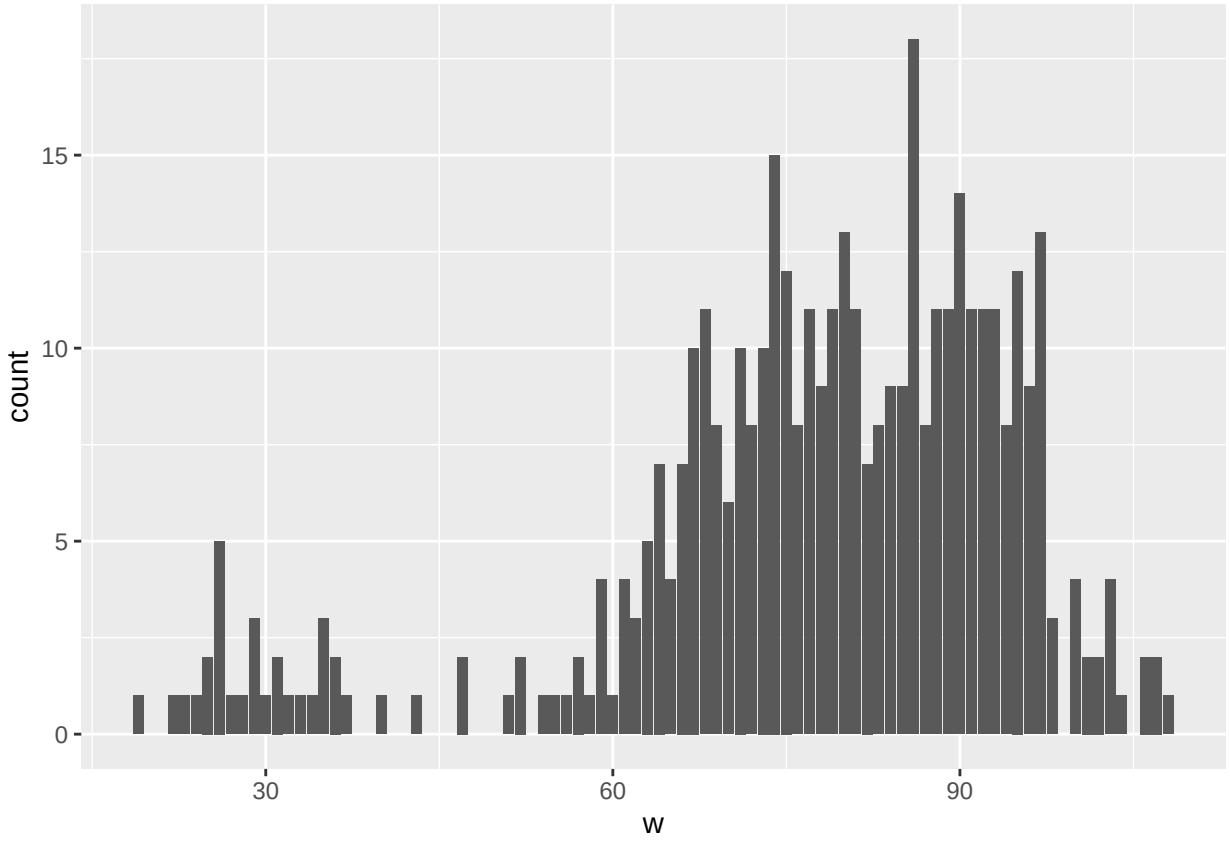
This plot doesn't tell us anything new as we know it is more rare for a team to make the playoffs than not. I will now also look at the distribution for all winning percentage values since it is closely related to our response variable.

```
baseball_data %>%
  ggplot(aes(x = win_percentage)) +
  geom_bar()
```



And another histogram of the amount of wins in a season distribution (from orginal dataset) for an even better visual.

```
baseball_data %>%
  ggplot(aes(x = w)) +
  geom_bar()
```



DATA SPLITTING

Since removing the missing values decreased the original amount of total observations, I will split the data at 60% training and 40% testing. Stratified sampling was used on our response variable (playoff_bound) since the distribution of the wins and winning percentage was a bit skewed.

```
set.seed(325)
baseball_data_split <- baseball_data %>%
  initial_split(prop = 0.6, strata = "playoff_bound")

baseball_train <- training(baseball_data_split)
baseball_test <- testing(baseball_data_split)
```

FOLDING THE DATA: With 10 folds and 5 repeats.

```
set.seed(325)
train_folds <- vfold_cv(baseball_train, v = 10, repeats = 5)
```

Creating my recipe with my variables of interest. Reminder: some predictors are closely related to one another to I will create some interactions within my recipe and normalize all predictors.

```
playoff_recipe <- recipe(playoff_bound ~ b_bb_rate + era + b_k_rate + w_oba + bb_9 + k_9 + drs + frm + a
  step_normalize(all_predictors()) %>%
  step_interact(terms = ~ bb_9:era) %>%
  step_interact(terms = ~ w_oba:b_bb_rate) %>%
  step_interact(terms = ~ k_9:era) %>%
  step_interact(terms = ~ drs:era)
```

MODEL FITTING WITH 10-FOLD CROSS VALIDATION

Now that we have a recipe ready and to undergo cross validation, we will fit 4 models. Logistic, LDA, QDA, and Naive-Bayes. I will set up their workflows and engines as well as calculate their metrics. I will summarize my findings below.

I will also tune the parameters for each model by using 'fit_resamples'.

LOGISTIC REGRESSION

```
log_reg <- logistic_reg() %>%
  set_engine("glm") %>%
  set_mode("classification")
```

```
log_wkflow <- workflow() %>%
  add_model(log_reg) %>%
  add_recipe(playoff_recipe)

logistic_fit <- fit_resamples(log_wkflow, train_folds)
```

```
log_reg_metrics = collect_metrics(logistic_fit)

logreg_mean_vector = log_reg_metrics %>%
  pull(mean)

logreg_stderr_vector = log_reg_metrics %>%
  pull(std_err)
```

Linear Discriminant Analysis (LDA)

```
lda <- discrim_linear() %>%
  set_mode("classification") %>%
  set_engine("MASS")

lda_wkflow <- workflow() %>%
  add_model(lda) %>%
  add_recipe(playoff_recipe)

lda_fit <- fit_resamples(lda_wkflow, train_folds)
```

```
lda_metrics = collect_metrics(lda_fit)

lda_mean_vector = lda_metrics %>%
  pull(mean)

lda_stderr_vector = lda_metrics %>%
  pull(std_err)
```

```
qda <- discrim_quad() %>%
  set_mode("classification") %>%
  set_engine("MASS")

qda_wkflow <- workflow() %>%
```

```
  add_model(qda) %>%
  add_recipe(playoff_recipe)

qda_fit <- fit_resamples(qda_wkflow, train_folds)
```

```
qda_metrics = collect_metrics(qda_fit)

qda_mean_vector = qda_metrics %>%
  pull(mean)

qda_stderr_vector = qda_metrics %>%
  pull(std_err)
```

```
naive_bayes_model <- naive_Bayes() %>%
  set_mode("classification") %>%
  set_engine("klaR") %>%
  set_args(usekernel = FALSE)

nb_wkflow <- workflow() %>%
  add_model(naive_bayes_model) %>%
  add_recipe(playoff_recipe)

nb_fit <- fit_resamples(nb_wkflow, train_folds)
```

```
## ! Fold01, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold02, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold03, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold04, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold05, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold06, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold07, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold08, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold09, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold10, Repeat1: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold01, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold02, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold03, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
```

```
## ! Fold04, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold05, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold06, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold07, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold08, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold09, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold10, Repeat2: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold01, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold02, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold03, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold04, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold05, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold06, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold07, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold08, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold09, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold10, Repeat3: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold01, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold02, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold03, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold04, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold05, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold06, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
## ! Fold07, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
```

```
## ! Fold08, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold09, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold10, Repeat4: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold01, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold02, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold03, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold04, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold05, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold06, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold07, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold08, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold09, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...

## ! Fold10, Repeat5: preprocessor 1/1, model 1/1 (predictions): Numerical 0 probability for a...
```

```
nb_metrics = collect_metrics(qda_fit)

nb_mean_vector = nb_metrics %>%
  pull(mean)

nb_stderr_vector = nb_metrics %>%
  pull(std_err)
```

Below are my 4 models along with their 1.) accuracy and 2.) standard error.

```
## [1] "Logisitc Regression Model"

## [1] 0.8630462

## [1] 0.01052548

## [1] "LDA"

## [1] 0.8518154

## [1] 0.00916243

## [1] "QDA"
```

```
## [1] 0.8397538
```

```
## [1] 0.009187587
```

```
## [1] "Naive Bayes"
```

```
## [1] 0.8397538
```

```
## [1] 0.009187587
```

From this we can clearly see that the logistic regression model performed the best as it at the highest average accuracy value and the lowest standard error of the four models. We will now fit our logistic model to the entire training set. This will also be the fit we apply to the test set.

```
logreg_train_fit <- fit(log_wkflow, baseball_train)
```

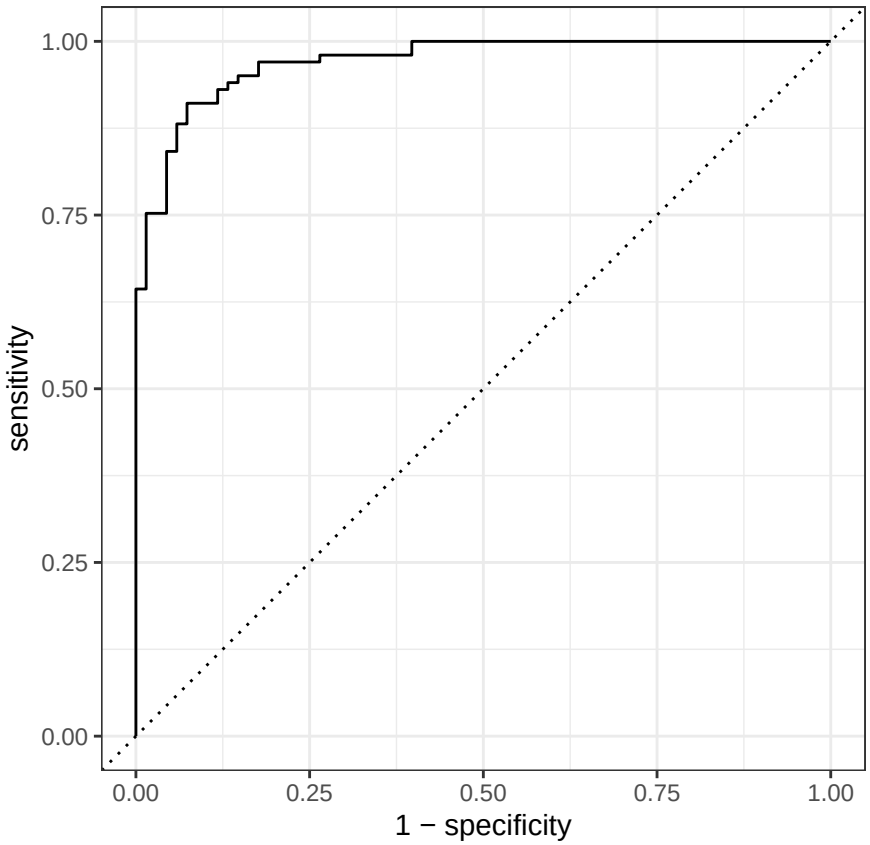
```
logreg_acc <- augment(logreg_train_fit, new_data = baseball_test) %>%
  accuracy(truth = playoff_bound, estimate = .pred_class)

logreg_acc
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.899
```

Training set accuracy: 0.8630462 Testing set accuracy: 0.8994083

Our testing accuracy was slightly higher than our training accuracy. This is likely due to the missing values discussed earlier in the project.

```
augment(logreg_train_fit, new_data = baseball_test) %>%
  roc_curve(playoff_bound, .pred_False) %>%
  autoplot()
```




```
augment(logreg_train_fit, new_data = baseball_test) %>%
  roc_auc(playoff_bound, .pred_False)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.971
```

Large roc_auc value is good to see for our model.

```
test1 <- data.frame(b_bb_rate = 0.106,
                    era = 3.29,
                    b_k_rate = 0.230,
                    w_oba = 0.303,
                    bb_9 = 2.64,
                    k_9 = 10.43,
                    drs = 16,
                    frm = -2.3,
                    avg_age = 27.0)

predict(logreg_train_fit, test1)
```

```
## # A tibble: 1 x 1
##   .pred_class
```

```
##   <fct>
## 1 True
```

Our model's prediction turned out well considering the predictor variables matched Cleveland Indian's 2020 season and that team in 2020 did, in fact, make the playoffs!

CONCLUSION: I hypothesized that the less significant in game events hold important value to a winning season while other stats would be 'overvalued'. My research tells me that I am partly right.

The correlations at the beginning caught my attention right off the bat, no pun intended. Walks are usually seen as one of the more boring outcomes in a game as opposed to getting a hit or homerun. However, walks are a very important aspect of the game both offensively and defensively. This showed in the last correlation matrix as both bb_9 and b_bb_rate had strong correlations with winning percentage. Meanwhile, w_oba and era are heavily correlated with winning as one would normally expect. I hypothesized that it would hold less of a relationship with winning than it showed. A suprising relationship I was not expecting at all was the correlation between the average ages of a team and winning. Younger pitchers and older hitters appear to be the optimal 'bare bones' approach to constructing a team.

After testing 4 different models, I found that my logistic model worked the best out of the 4. This makes sense as our response variable held only 2 possible values and was classified as a factor variable. I eventually made a prediction with my model and it correctly predicted that a team would make the playoffs given certain predictive stat values matching the criteria!

There are many sub-routes anyone can take on this research. For example, I had to refrain from exploring one of these routes once I began investigating the relationships over time with fastball and cutter usage rates. As more and more data is gathered and as the game continues to evolve, I believe studying these trends are crucial not only for team development and construction, but also for individual player development. Though not mentioned in this project, 'newer' information like spin rates and exit velocities can help propel these studies.