Championship\_Predictor

Ryan Abbott

2023-03-07

This project will attempt to predict the winner of the NBA Finals based on several team-based statistics

Sys.setenv("VROOM\_CONNECTION\_SIZE" = 262144)  
library("nbastatR")

## Warning: replacing previous import 'curl::handle\_reset' by 'httr::handle\_reset'  
## when loading 'nbastatR'

## Warning: replacing previous import 'httr::timeout' by 'memoise::timeout' when  
## loading 'nbastatR'

## Warning: replacing previous import 'magrittr::set\_names' by 'purrr::set\_names'  
## when loading 'nbastatR'

## Warning: replacing previous import 'jsonlite::flatten' by 'purrr::flatten' when  
## loading 'nbastatR'

## Warning: replacing previous import 'curl::parse\_date' by 'readr::parse\_date'  
## when loading 'nbastatR'

## Warning: replacing previous import 'purrr::invoke' by 'rlang::invoke' when  
## loading 'nbastatR'

## Warning: replacing previous import 'purrr::flatten\_raw' by 'rlang::flatten\_raw'  
## when loading 'nbastatR'

## Warning: replacing previous import 'purrr::flatten\_dbl' by 'rlang::flatten\_dbl'  
## when loading 'nbastatR'

## Warning: replacing previous import 'jsonlite::unbox' by 'rlang::unbox' when  
## loading 'nbastatR'

## Warning: replacing previous import 'purrr::flatten\_lgl' by 'rlang::flatten\_lgl'  
## when loading 'nbastatR'

## Warning: replacing previous import 'purrr::flatten\_int' by 'rlang::flatten\_int'  
## when loading 'nbastatR'

## Warning: replacing previous import 'purrr::%@%' by 'rlang::%@%' when loading  
## 'nbastatR'

## Warning: replacing previous import 'purrr::flatten\_chr' by 'rlang::flatten\_chr'  
## when loading 'nbastatR'

## Warning: replacing previous import 'purrr::splice' by 'rlang::splice' when  
## loading 'nbastatR'

## Warning: replacing previous import 'purrr::flatten' by 'rlang::flatten' when  
## loading 'nbastatR'

## Warning: replacing previous import 'readr::guess\_encoding' by  
## 'rvest::guess\_encoding' when loading 'nbastatR'

## Warning: replacing previous import 'magrittr::extract' by 'tidyr::extract' when  
## loading 'nbastatR'

## Warning: replacing previous import 'rlang::as\_list' by 'xml2::as\_list' when  
## loading 'nbastatR'

library("dplyr")

## Warning: package 'dplyr' was built under R version 4.1.3

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

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

library("tidyverse")

## Warning: package 'ggplot2' was built under R version 4.1.3

## Warning: package 'tibble' was built under R version 4.1.3

## Warning: package 'tidyr' was built under R version 4.1.3

## Warning: package 'readr' was built under R version 4.1.3

## Warning: package 'purrr' was built under R version 4.1.3

## Warning: package 'stringr' was built under R version 4.1.3

## Warning: package 'forcats' was built under R version 4.1.3

## Warning: package 'lubridate' was built under R version 4.1.3

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --  
## v forcats 1.0.0 v readr 2.1.4  
## v ggplot2 3.4.1 v stringr 1.5.0  
## v lubridate 1.9.2 v tibble 3.1.8  
## v purrr 1.0.1 v tidyr 1.3.0

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()  
## i Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force all conflicts to become errors

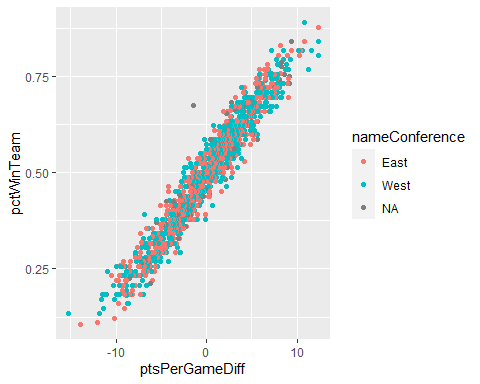
#game\_logs(seasons = 2023)  
#get\_day\_nba\_league\_standings("02/1/2023")  
#get\_team\_season\_stat\_splits(year.season\_start = 2015)  
teamdata <- tibble(read.csv("training\_data\_team\_raw.csv"))  
#teamdata <- na.omit(teamdata$ptsPerGameDiff)  
teamdata

## # A tibble: 1,385 x 199  
## X slugS~1 yearS~2 typeS~3 nameT~4 slugT~5 teamN~6 nameD~7 recor~8 idSea~9  
## <int> <chr> <int> <chr> <chr> <chr> <chr> <chr> <chr> <int>  
## 1 1 1959-60 1960 Regula~ Boston~ BOS Celtics East 0-0 21959  
## 2 2 1959-60 1960 Regula~ Cincin~ <NA> Royals West 0-0 21959  
## 3 3 1959-60 1960 Regula~ Detroi~ DET Pistons West 0-0 21959  
## 4 4 1959-60 1960 Regula~ Minnea~ <NA> Lakers West 0-0 21959  
## 5 5 1959-60 1960 Regula~ New Yo~ NYK Knicks East 0-0 21959  
## 6 6 1959-60 1960 Regula~ Philad~ <NA> Warrio~ East 0-0 21959  
## 7 7 1959-60 1960 Regula~ St. Lo~ <NA> Hawks West 0-0 21959  
## 8 8 1959-60 1960 Regula~ Syracu~ <NA> Nation~ East 0-0 21959  
## 9 9 1960-61 1961 Regula~ Boston~ BOS Celtics East 0-0 21960  
## 10 10 1960-61 1961 Regula~ Cincin~ <NA> Royals West 0-0 21960  
## # ... with 1,375 more rows, 189 more variables: idTeam <int>, pctWinTeam <dbl>,  
## # cityTeam <chr>, TeamSlug <chr>, recordOverall <chr>, recordVsEast <chr>,  
## # recordVsAtlantic <chr>, recordVsCentral <chr>, recordVsWest <chr>,  
## # rankPlayoffs <int>, rankDivision <int>, wins <int>, losses <int>,  
## # gamesBackConference <dbl>, gamesBackDivision <dbl>, ptsPerGameTeam <dbl>,  
## # ptsPerGameOpponent <dbl>, ptsPerGameDiff <dbl>, nameConference <chr>,  
## # recordConference <chr>, RecordDivision <chr>, recordHome <chr>, ...

## Exploratory Data Analysis: Finding variables

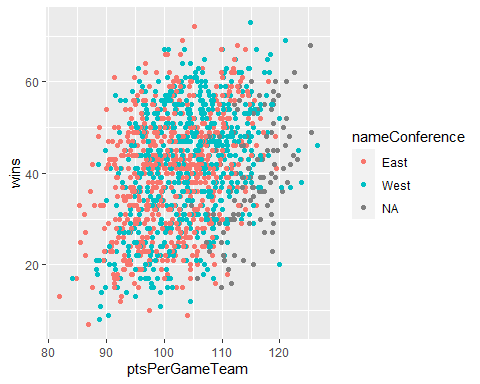
First, I’m thinking that point differentials between a team and their opponents are important to overall success. Lets visualize that.

df <- teamdata %>% select(ptsPerGameDiff, slugSeason, nameTeam, recordOverall, pctWinTeam, nameConference, wins) %>% arrange(desc(ptsPerGameDiff))  
  
plot <- ggplot(df, aes(x=ptsPerGameDiff,y=pctWinTeam,col=nameConference))  
  
plot <- plot + geom\_point()  
  
plot

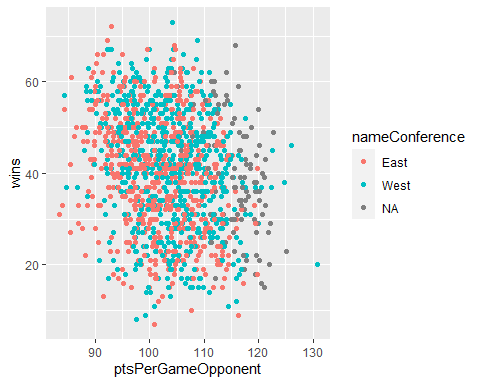


The correlation between these two variables is astoundingly close. The model can be better optimized though.

ppg <- teamdata %>% select(ptsPerGameTeam, ptsPerGameOpponent, nameTeam, slugSeason, pctWinTeam, wins, nameConference, recordOpponentOver500WinPct, ptsPerGameDiff, recordAheadAtThirdWinPct, recordAwayWinPct, recordAheadAtHalfWinPct, recordHomeWinPct) %>% arrange(desc(ptsPerGameTeam))  
  
plot <- ggplot(ppg, aes(x=ptsPerGameTeam, y=wins, col=nameConference)) + geom\_point()  
plot2 <- ggplot(ppg, aes(x=ptsPerGameOpponent, y=wins, col=nameConference)) + geom\_point()  
plot3 <- ggplot(ppg, aes(x=recordOpponentOver500WinPct, y=wins, col=nameConference)) + geom\_point()  
plot4 <- ggplot(ppg, aes(x=recordAheadAtHalfWinPct, y=wins, col=nameConference)) + geom\_point()  
plot5 <- ggplot(ppg, aes(x=recordAwayWinPct, y=wins, col=nameConference)) + geom\_point()  
plot6 <- ggplot(ppg, aes(x=recordHomeWinPct, y=wins, col=nameConference)) + geom\_point()  
  
  
plot

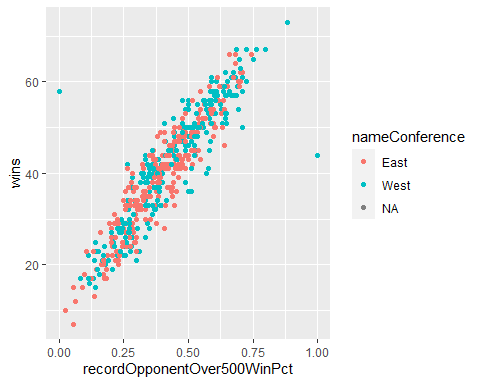


plot2



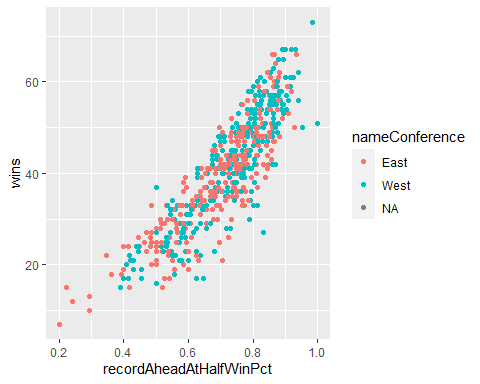
plot3

## Warning: Removed 847 rows containing missing values (`geom\_point()`).



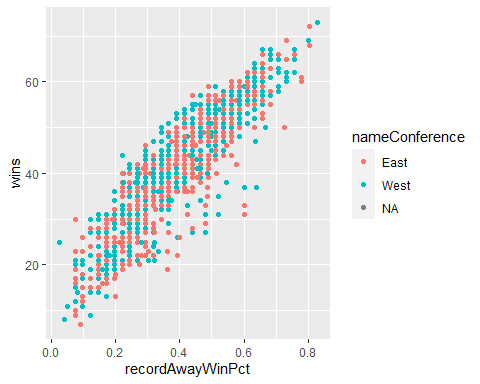
plot4

## Warning: Removed 819 rows containing missing values (`geom\_point()`).



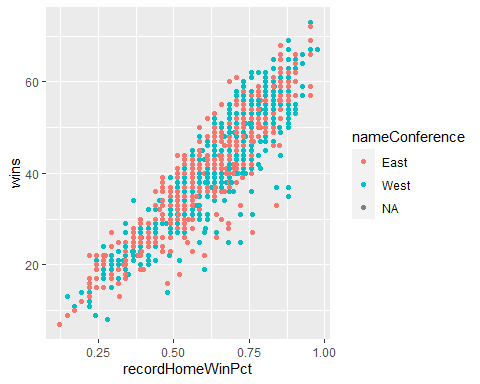
plot5

## Warning: Removed 111 rows containing missing values (`geom\_point()`).



plot6

## Warning: Removed 111 rows containing missing values (`geom\_point()`).



Most of these results do not surprise me. Points scored per game is a very relative statistic, as a team with slower pacing may have dozens of possessions less than a team with faster pacing. The same applies for points allowed per game.

A team’s record against other teams with a winning record strongly correlates to a better win percentage as well.

One statistic that I am a bit surprised correlates so strongly with total wins is a team’s record when they lead the game at halftime. I expected a relatively strong positive correlation, but with half of a game still remaining to play, it is easy to envision that the opposing team can mount a comeback.

## Model Testing

champion.lm = lm(wins~ptsPerGameDiff+recordOpponentOver500WinPct+recordAheadAtThirdWinPct+recordAheadAtHalfWinPct+recordAwayWinPct, ppg)

summary(champion.lm)

##   
## Call:  
## lm(formula = wins ~ ptsPerGameDiff + recordOpponentOver500WinPct +   
## recordAheadAtThirdWinPct + recordAheadAtHalfWinPct + recordAwayWinPct,   
## data = ppg)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -12.9884 -0.9527 0.3280 1.6768 7.0055   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 15.75098 1.49063 10.567 < 2e-16 \*\*\*  
## ptsPerGameDiff 1.31566 0.07664 17.167 < 2e-16 \*\*\*  
## recordOpponentOver500WinPct 12.63903 1.76985 7.141 3.05e-12 \*\*\*  
## recordAheadAtThirdWinPct 5.89503 2.10869 2.796 0.00537 \*\*   
## recordAheadAtHalfWinPct 11.07972 2.01754 5.492 6.18e-08 \*\*\*  
## recordAwayWinPct 17.41142 1.91125 9.110 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.723 on 532 degrees of freedom  
## (847 observations deleted due to missingness)  
## Multiple R-squared: 0.9518, Adjusted R-squared: 0.9514   
## F-statistic: 2102 on 5 and 532 DF, p-value: < 2.2e-16

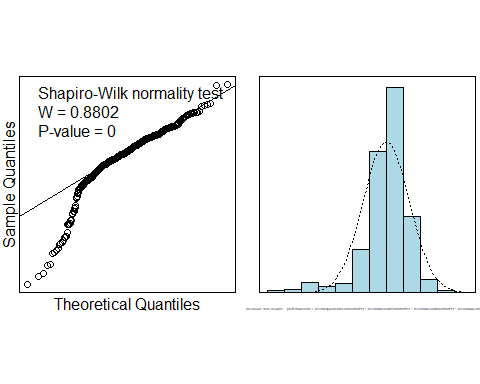
The following code chunk is just to check the normality of the distribution. I may perform some probability analysis at a later time, but for now, this is all I have done.

Evidently, the distribution is not normal when looking at the lesser teams.

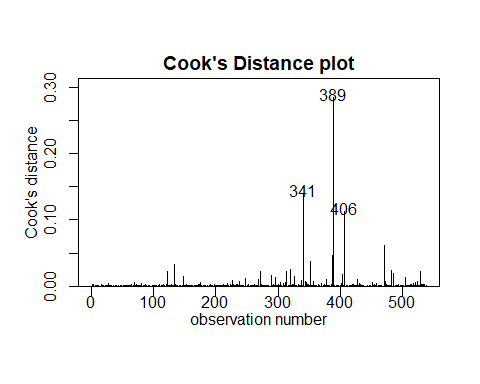
library(s20x)

## Warning: package 's20x' was built under R version 4.1.3

normcheck(champion.lm,shapiro.wilk = TRUE)

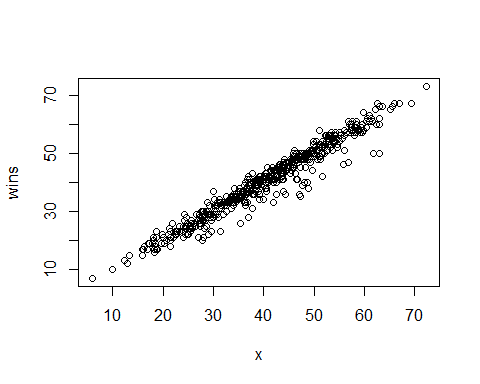


cooks20x(champion.lm)



Here is the scatter plot for the final linear model. It is remarkably accurate, with some teams performing slightly worse than predicted but virtually zero performing better than expected.

pred <- function(x1, x2, x3, x4, x5) {  
 champion.lm$coefficients[1] + champion.lm$coefficients[2]\*x1 + champion.lm$coefficients[3]\*x2 + champion.lm$coefficients[4]\*x3 + champion.lm$coefficients[5]\*x4 + champion.lm$coefficients[6]\*x5  
}  
  
x=pred(ppg$ptsPerGameDiff,ppg$recordOpponentOver500WinPct,ppg$recordAheadAtThirdWinPct,ppg$recordAheadAtHalfWinPct,ppg$recordAwayWinPct)  
  
plot(wins~x, ppg)



## Model Testing: 2022 season

team\_data\_test <- standings(seasons = 2022)

## Getting 2021-22 Regular Season NBA standings data  
## Missing TeamSlug in dictionary  
## Missing ClinchedPlayIn in dictionary  
## Missing Score\_80\_Plus in dictionary  
## Missing Opp\_Score\_80\_Plus in dictionary  
## Missing Score\_Below\_80 in dictionary  
## Missing Opp\_Score\_Below\_80 in dictionary  
## Missing TotalPoints in dictionary  
## Missing OppTotalPoints in dictionary  
## Missing DiffTotalPoints in dictionary

team\_data\_test

## # A tibble: 30 x 194  
## slugSeason yearSeason typeS~1 nameT~2 slugT~3 teamN~4 nameC~5 slugP~6 nameD~7  
## <chr> <dbl> <chr> <chr> <chr> <chr> <chr> <chr> <chr>   
## 1 2021-22 2022 Regula~ Atlant~ ATL Hawks East - pi Southe~  
## 2 2021-22 2022 Regula~ Boston~ BOS Celtics East - a Atlant~  
## 3 2021-22 2022 Regula~ Brookl~ BKN Nets East - pi Atlant~  
## 4 2021-22 2022 Regula~ Charlo~ CHA Hornets East - pi Southe~  
## 5 2021-22 2022 Regula~ Chicag~ CHI Bulls East - x Central  
## 6 2021-22 2022 Regula~ Clevel~ CLE Cavali~ East - pi Central  
## 7 2021-22 2022 Regula~ Dallas~ DAL Maveri~ West - x Southw~  
## 8 2021-22 2022 Regula~ Denver~ DN Nuggets West - x Northw~  
## 9 2021-22 2022 Regula~ Detroi~ DET Pistons East - o Central  
## 10 2021-22 2022 Regula~ Golden~ GSW Warrio~ West - x Pacific  
## # ... with 20 more rows, 185 more variables: slugStreakLongHomeStreak <chr>,  
## # slugStreakLongAway <chr>, slugStreakHomeCurrent <chr>,  
## # slugStreakAwayCurrent <chr>, slugStreakCurrent <chr>,  
## # recordLeadInFGPCT <chr>, idSeason <dbl>, idTeam <dbl>, pctWinTeam <dbl>,  
## # cityTeam <chr>, TeamSlug <chr>, recordConference <chr>,  
## # RecordDivision <chr>, recordOverall <chr>, recordHome <chr>,  
## # recordAway <chr>, recordLast10 <chr>, recordLast10Home <chr>, ...

ppg\_test <- team\_data\_test %>% select(slugSeason, nameTeam, wins) #%>% arrange(desc(ptsPerGameDiff))  
  
test <- team\_data\_test %>% select(recordOpponentOver500WinPct,ptsPerGameDiff,recordAheadAtThirdWinPct,recordAheadAtHalfWinPct,recordAwayWinPct)  
  
ppg\_test <- ppg\_test %>% mutate(projected\_wins = c(pred(test$ptsPerGameDiff, test$recordOpponentOver500WinPct, test$recordAheadAtThirdWinPct, test$recordAheadAtHalfWinPct, test$recordAwayWinPct)))  
  
arrange(ppg\_test, desc(projected\_wins))

## # A tibble: 30 x 4  
## slugSeason nameTeam wins projected\_wins  
## <chr> <chr> <dbl> <dbl>  
## 1 2021-22 Phoenix Suns 64 64.6  
## 2 2021-22 Memphis Grizzlies 56 57.2  
## 3 2021-22 Boston Celtics 51 56.4  
## 4 2021-22 Golden State Warriors 53 54.3  
## 5 2021-22 Miami Heat 53 53.3  
## 6 2021-22 Utah Jazz 49 51.9  
## 7 2021-22 Milwaukee Bucks 51 51.9  
## 8 2021-22 Philadelphia 76ers 51 51.1  
## 9 2021-22 Dallas Mavericks 52 50.7  
## 10 2021-22 Toronto Raptors 48 50.4  
## # ... with 20 more rows

## NBA Teams’ Projected Wins, 2023

team\_data\_2023 <- standings(seasons = 2023)

## Getting 2022-23 Regular Season NBA standings data  
## Missing TeamSlug in dictionary  
## Missing ClinchedPlayIn in dictionary  
## Missing Score\_80\_Plus in dictionary  
## Missing Opp\_Score\_80\_Plus in dictionary  
## Missing Score\_Below\_80 in dictionary  
## Missing Opp\_Score\_Below\_80 in dictionary  
## Missing TotalPoints in dictionary  
## Missing OppTotalPoints in dictionary  
## Missing DiffTotalPoints in dictionary

ppg\_test <- team\_data\_2023 %>% select(slugSeason, nameTeam, wins)  
  
  
ppg\_test <- ppg\_test %>% mutate(projected\_wins = c(pred(team\_data\_2023$ptsPerGameDiff, team\_data\_2023$recordOpponentOver500WinPct, team\_data\_2023$recordAheadAtThirdWinPct, team\_data\_2023$recordAheadAtHalfWinPct, team\_data\_2023$recordAwayWinPct)))  
  
arrange(ppg\_test, desc(projected\_wins))

## # A tibble: 30 x 4  
## slugSeason nameTeam wins projected\_wins  
## <chr> <chr> <dbl> <dbl>  
## 1 2022-23 Boston Celtics 45 56.0  
## 2 2022-23 Milwaukee Bucks 46 55.7  
## 3 2022-23 Denver Nuggets 46 54.7  
## 4 2022-23 Philadelphia 76ers 42 54.4  
## 5 2022-23 New York Knicks 39 51.3  
## 6 2022-23 Cleveland Cavaliers 41 51.3  
## 7 2022-23 Sacramento Kings 38 48.5  
## 8 2022-23 Phoenix Suns 36 47.1  
## 9 2022-23 Memphis Grizzlies 38 47.0  
## 10 2022-23 Brooklyn Nets 36 45.6  
## # ... with 20 more rows