Building an NFL Betting Database

Stats 405 - Final Project

Guy Dotan

6/24/2018

Contents

1 Introduction

On May 14, 2018, the Supreme Court case *Murphy v. National Collegiate Athletic Association* reached a landmark decision regarding the federal government's right to control a state's ability to sponsor sports betting. With this ruling, every state is now legally allowed to make its own decision to permit in-state sportsbooks. Needless to say, the entire sports landscape will be changed with the legalization of sports betting and the ramifications are yet to be known.

What we do know is that over the upcoming years, new sportsbooks and sports-betting startups will be popping up all over the country. Without a doubt, data analysts will be deeply involved in the development of these businesses. So now is the perfect time to start the journey of digging into this world of sports betting data.

For my project, I would like to build a database that is an archive of sports betting information for as far back as I can find a reliable source. In particular, I want to focus on the NFL. The reason: according to a 2017 ESPN article, "Even amid declining television ratings, concerns over concussions and political controversy over player protests, football remains the most popular sport to bet." From some preliminary research, the following website: http://www.footballlocks.com/ appears to have a lengthy archive of NFL betting odds.

The main challenge of this project will be scraping the relevant websites and then cleaning the data into a usable form. Hopefully upon completion, I will have over a decade's worth of betting data fully aggregated to analyze.

2 Scraping The Betting History Website

The site that we will be scraping is FootballLocks.com. This site has the weekly betting lines for every NFL regular and postseason game from the 2006-07 season to the 2017-18 season. The site splits up this data with each webpage containing one week's data from 2006-2017. For example, the link http://www.footballlocks.com/nfl_lines_week_1.shtml has all the Week 1 lines in that time frame. http://www.footballlocks.com/nfl_lines_week_2.shtml has all the lines from Week 2 in that time frame. As we can see, we can get all the betting data from regular season Weeks 1-17 simply by adjusting the digit in the url. All 12 years of data for each week is contained in a single table on the webpage. So data cleaning is necessary to extract just the relevant data from each page.

```
## Load packages
library("rvest")
library("stringr")
library("XML")
library("data.table")
library("dplyr")
library("ggplot2")
library("sqldf")
library("pander")
```

```
url <- "http://www.footballlocks.com/nfl_lines_"  # base part of url to scrape
reg_lines <- c()  # data frame to hold all regular season lines

## REGULAR SEASON ##
# outer loop for selecting correct url based on week #1-17
for (wk in 1:17) {
  url_reg <- paste0(url,"week_",wk,".shtml")

  # read in HTML table using XML package
  wk_num <- readHTMLTable(url_reg, stringsAsFactors=FALSE)
  #str(wk1, list.len = 2)
  Week_Num <- wk_num[[1]]

  # isolate boundary rows for each season
  cutoffs <- which(Week_Num$V1 == "Date & Time")
  last_cutoff <- max(str_which(Week_Num$V2,"<!--//")) - 1 ## last table cutoff
```

```
  cutoffs <- c(cutoffs, last_cutoff)

  # inner loop to break down tables season by season
  for (i in 1:(length(cutoffs)-1)){
    temp <- Week_Num[ c( (cutoffs[i]-1):(cutoffs[i+1]-2) ) , ]  # select one week of one season
    temp$Year <- str_extract(temp[1,], "\\b[0-9][0-9][0-9][0-9]\\b")[1]  # isolate season name
    temp$Week <- wk  # isolate week number
    temp <- temp[-1,] # remove season row
    temp <- temp[,c(22,23,1,2,3,4,5)]  # eliminate unneeded columns
    names(temp)[3:7] <- temp[1,3:7]  # rename headers
    temp <- temp[-1,] # remove header row
    temp <- temp[grep("[0-9]",temp$`Date & Time`),] # eliminate blank row

    reg_lines <- rbind(reg_lines, temp)  # add to data frame
  }
}
```

We then need to do a similar process with the postseason betting lines. In this case the url isn't just an incremented week number but instead it is the four different playoff round names. We can still loop through the webpages by appending those playoff round names to the url. Then we just combine the regular season and postseason tables into one complete table.

```
# POST SEASON ##
post_lines <- c() # data frame to hold all postseason lines
post_names <- c("wild_card_playoff_games","divisional_playoff_games",
                "conference_championship_playoff_games","super_bowl" )
post_abbrev <- c("WC", "Div", "Conf", "SB")

for (p in 1:length(post_names)) {
  url_post <- paste0(url,post_names[p],".shtml") #

  # read in HTML table using XML package
  p_num <- readHTMLTable(url_post, stringsAsFactors=FALSE)
  #str(wk1, list.len = 2)
  Post_Num <- p_num[[1]]

  # isolate boundary rows for each season
  cutoffs_post <- which(Post_Num$V1 == "Date & Time")
  last_cutoff_post <- max(str_which(Post_Num$V2,"<!--//")) - 1 ## last table cutoff
  cutoffs_post <- c(cutoffs_post, last_cutoff_post)

  # inner loop to break down tables season by season
  for (i in 1:(length(cutoffs_post)-1)){
    temp2 <- Post_Num[ c( (cutoffs_post[i]-1):(cutoffs_post[i+1]-2) ) , ]  # select one week of one season
    temp2$Year <- str_extract(temp2[1,], "\\b[0-9][0-9][0-9][0-9]\\b")[1]  # isolate year name
    temp2$Week <- post_abbrev[p]  # isolate week number
    temp2 <- temp2[-1,] # remove season row
    temp2 <- temp2[,c(22,23,1,2,3,4,5)]  # eliminate unneeded columns
    names(temp2)[3:7] <- temp2[1,3:7]  # rename headers
    temp2 <- temp2[-1,] # remove header row
    temp2 <- temp2[grep("[0-9]",temp2$`Date & Time`),] # eliminate blank row

    post_lines <- rbind(post_lines, temp2)  # add to data frame
  }
}

# 2018 SB listed on site but on a different page for some reason, manually inserted below
```

```
sb2018 <- c("2018", "SB", "2/4 6:30 ET", "New England", "-4.5", "Philadelphia", "49", NA, "post")
post_lines <- rbind(post_lines, sb2018)

reg_lines$Game_Type <- "reg"
post_lines$Game_Type <- "post"

# combine into one full table (reg + post)
nfl_lines <- rbind(reg_lines, post_lines)
```

Here is what this complete table looks like:

Table 1: Top of Scraped Table

Year	Week	Date & Time	Favorite	Line	Underdog	Total	Game_Type
2017	1	9/7 8:30 ET	At New England	-9	Kansas City	42	reg
2017	1	9/10 1:00 ET	At Buffalo	-7.5	NY Jets	42	reg
2017	1	9/10 1:00 ET	Atlanta	-7	At Chicago	48	reg
2017	1	9/10 1:00 ET	At Houston	-5.5	Jacksonville	38	reg
2017	1	9/10 1:00 ET	Philadelphia	-1	At Washington	49.5	reg

Table 2: End of Scraped Table

	Year	Week	Date & Time	Favorite	Line	Underdog	Total	Game_Type
3068	2010	SB	2/7 6:25 ET	Indianapolis	-4.5	New Orleans	57	post
3069	2009	SB	2/1 6:18 ET	Pittsburgh	-6.5	Arizona	46.5	post
3070	2008	SB	2/3 6:18 ET	New England	-12.5	NY Giants	55	post
3071	2007	SB	2/4 6:18 ET	Indianapolis	-6.5	Chicago	47.5	post
3072	2006	SB	2/5 6:18 ET	Pittsburgh	-4	Seattle	47	post
3073	2018	SB	2/4 6:30 ET	New England	-4.5	Philadelphia	49	post

3 Data Cleaning

The next step is to clean up this dataset so we can merge it with other tables and run some analysis. We will start by working on the team name columns `Favorite` and `Underdog`.

```
## Table Cleaning ##
nfl_clean <- nfl_lines
```

3.1 Team Names

```
## Team Names ##
# Favorite
nfl_clean$FAVORITE <- str_replace_all(nfl_clean$Favorite, 'At ', "")  # only include team (city) name
nfl_clean$FAVORITE <- str_replace_all(nfl_clean$FAVORITE, "\n.*", "") # remove new line text
nfl_clean$FAVORITE <- str_replace_all(nfl_clean$FAVORITE, "\\(.*", "") # remove parenthesis (venue) text
nfl_clean$Fav_Home <- ifelse(grepl('At ', nfl_clean$Favorite),'Y','N')  # identify if favorite is home/away

# Underdog
nfl_clean$UNDERDOG <- str_replace_all(nfl_clean$Underdog, 'At ', "")  # only include team (city) name
nfl_clean$UNDERDOG <- str_replace_all(nfl_clean$UNDERDOG, "\n.*", "") # remove new line text
nfl_clean$UNDERDOG <- str_replace_all(nfl_clean$UNDERDOG, "\\(.*", "") # remove parenthesis (venue) text
```

```r
nfl_clean$Dog_Home <- ifelse(grepl('At ', nfl_clean$Underdog),'Y','N')  # identify if underdog is home/away

# fix two messed up team names
nfl_clean[ nfl_clean$UNDERDOG == "JacksonvilleLondonsize =-1>" , ]$UNDERDOG <- "Jacksonville"
nfl_clean[ nfl_clean$UNDERDOG == "Miami " , ]$UNDERDOG <- "Miami"

# change all 'Los Angeles label' to LA Rams. In 2016 the Rams were the only team
# in LA so 'Los Angeles' was used as team name for that season.
nfl_clean[ nfl_clean$FAVORITE == "Los Angeles" , ]$FAVORITE <- "LA Rams"
nfl_clean[ nfl_clean$UNDERDOG == "Los Angeles" , ]$UNDERDOG <- "LA Rams"
```

Now in order to resolve a couple more inconsistently labeled team names and to handle any potential merges with outside tables, it would be a good idea to assign unique IDs to each franchise. I will get a list of NFL teams and their various team names and locations over the recent NFL history. This list comes from a CSV download from www.pro-football-reference.com/teams/. After some some cleaning and assigning IDs 1 to 32 for each team, we now have a list of all active teams (and their names) since 1960.

```r
# csv from https://www.pro-football-reference.com/teams/
teams <- read.csv("NFL Teams.csv", stringsAsFactors = F, skip = 1)
names(teams)[c(2,3)] <- c("Year_From", "Year_To")

# assign team_ids to all 32 active teams
teams$team_id = 1
for (i in 2:nrow(teams)) {
  teams[i,]$team_id <- ifelse(teams[i,]$AV == "",  teams[i-1,]$team_id, (teams[i-1,]$team_id)+1)
}

# split team name column at the last space (works because all NFL team names are just one word)
teams$LOCATION <- do.call(rbind, strsplit(teams$Tm, ' (?=[^ ]+$)', perl=TRUE))[,1]
teams$NICKNAME <- do.call(rbind, strsplit(teams$Tm, ' (?=[^ ]+$)', perl=TRUE))[,2]

teams <- teams[,c(18,1,19,20,2:17)]

# Since AFL existance in 1960 (AFL-NFL merger in 1970)
recent_teams <- teams[teams$Year_From >= 1960 | teams$Year_To == 2018,]
```

Table 3: Teams List

team_id	Tm	LOCATION	NICKNAME	Year_From	Year_To	W	L	T	W.L.
1	Arizona Cardinals	Arizona	Cardinals	1920	2018	550	740	40	0.427
1	St. Louis Cardinals	St. Louis	Cardinals	1960	1987	186	202	14	0.48
1	Phoenix Cardinals	Phoenix	Cardinals	1988	1993	32	64	0	0.333
1	Arizona Cardinals	Arizona	Cardinals	1994	2018	167	216	1	0.436
2	Atlanta Falcons	Atlanta	Falcons	1966	2018	351	443	6	0.442
3	Baltimore Ravens	Baltimore	Ravens	1996	2018	190	161	1	0.541
4	Buffalo Bills	Buffalo	Bills	1960	2018	409	467	8	0.467
5	Carolina Panthers	Carolina	Panthers	1995	2018	183	184	1	0.499
6	Chicago Bears	Chicago	Bears	1920	2018	749	579	42	0.564
6	Chicago Bears	Chicago	Bears	1922	2018	730	577	39	0.558

Now we want to merge these team IDs into the betting lines table. We want a team_id listed for each favorite and for each underdog. The betting table has teams listed by their location (e.g. San Francisco, New England) so we will merge the team_ids based on the `LOCATION` field in the team table. However, some locations were common for multiple teams, for example: Houston Oilers and Houston Texans or St. Louis Cardinals and St. Louis Rams. This is solved by only taking the most recent team to be in that city. Since our data only dates back to 2006, this strategy works as no location had two different teams in this time frame. In addition, the two NY teams and two LA teams

are listed as `NY Giants`, `NY Jets`, `LA Rams`, and `LA Chargers`. This is resolved by manually giving these cities their appropriate ID.

```
# merge using aggregate to solve cases with multiple matches of same ID
# create aggregate of location with rk = 1 referring to most recent team in that location
city_agg <- recent_teams %>% arrange(LOCATION, Year_To) %>%
  group_by(LOCATION) %>%
  mutate(rank = rank(-Year_To, ties.method = "first"))
city_agg2 <- city_agg[city_agg$rank == 1,]

# LEFT JOIN so all betting lines are kept and unlinked IDs are left as NA
nfl_clean2 <- merge(nfl_clean, city_agg2[,c(1,3)]  , by.x=c("FAVORITE"),
                    by.y=c("LOCATION"), all.x=TRUE)
nfl_clean2 <- merge(nfl_clean2, city_agg2[,c(1,3)] , by.x=c("UNDERDOG"),
                    by.y=c("LOCATION"), all.x=TRUE)

# see which team's had blank IDs
rbind(table(nfl_clean2[is.na(nfl_clean2$team_id.x),]$FAVORITE),
table(nfl_clean2[is.na(nfl_clean2$team_id.y),]$UNDERDOG))
```

```
##      LA Chargers LA Rams NY Giants NY Jets
## [1,]           2       9       110      87
## [2,]           6      15        85     105
```

```
# manually assign IDs because naming convention is off for NY/LA teams.
# recent_teams[recent_teams$NICKNAME %in% c("Chargers", "Rams","Giants","Jets"),c(1:4)]
nfl_clean2[nfl_clean2$UNDERDOG == "LA Chargers",]$team_id.y <- 17
nfl_clean2[nfl_clean2$FAVORITE == "LA Chargers",]$team_id.x <- 17
nfl_clean2[nfl_clean2$FAVORITE == "LA Rams",]$team_id.x <- 18
nfl_clean2[nfl_clean2$UNDERDOG == "LA Rams",]$team_id.y <- 18
nfl_clean2[nfl_clean2$FAVORITE == "NY Giants",]$team_id.x <- 23
nfl_clean2[nfl_clean2$UNDERDOG == "NY Giants",]$team_id.y <- 23
nfl_clean2[nfl_clean2$FAVORITE == "NY Jets",]$team_id.x <- 24
nfl_clean2[nfl_clean2$UNDERDOG == "NY Jets",]$team_id.y <- 24

nfl_clean3 <- nfl_clean2[ , c(3,10,4,5,2,13,
                              1,14,7,9,11,12)]
names(nfl_clean3) <-  c("Year", "Game_Type", "Week", "Date_Time", "Favorite", "Fav_id",
                        "Underdog", "Dog_id", "Line", "Total", "Fav_Home", "Dog_Home")
```

Table 4: Full Table Newly Cleaned (continued below)

Year	Game_Type	Week	Date_Time	Favorite	Fav_id
2018	post	SB	2/4 6:30 ET	New England	21
2018	post	Conf	1/21 3:05 ET	New England	21
2018	post	Conf	1/21 6:40 ET	Minnesota	20
2018	post	Div	1/14 1:05 ET	Pittsburgh	27
2018	post	Div	1/14 4:40 ET	Minnesota	20

Underdog	Dog_id	Line	Total	Fav_Home	Dog_Home
Philadelphia	26	-4.5	49	N	N
Jacksonville	15	-7.5	45.5	Y	N
Philadelphia	26	-3	39	N	Y
Jacksonville	15	-7	41	Y	N
New Orleans	22	-5.5	47	Y	N

3.2 Date and Time

This next cleaning section involves adjusting the date and time columns to create actual date/time variables formatted for R. First we make sure we are only selecting games that were actually played and exclude all postponed ones. Once we have these games selected we can create the crucial `Game_id` variable. This will then be used to merge with another table to get the final results from the game.

```
## Format Date and Time ##
# Label all postponed games
nfl_clean3[grep(c("postponed|ppd|delay"),
                tolower(nfl_clean3$Date_Time)),]$Date_Time <- "Postponed/Delayed"

# fix any incorrectly enterred time (e.g. "1;00" instead of "1:00")
nfl_clean3$Date_Time <- gsub(';',':',nfl_clean3$Date_Time)

# Select only played games
nfl_played <- nfl_clean3[nfl_clean3$Date_Time != 'Postponed/Delayed',]

# create date colum
nfl_played$DATE <- as.Date(paste0(gsub( " .*$", "", nfl_played$Date_Time ),
                                  '/', nfl_played$Year), format = "%m/%d/%Y")

# create time column - select 2nd element of time list and append AM/PM
nfl_played$TIME <- paste(sapply(strsplit(nfl_played$Date_Time," "),"[[", 2),
                   ifelse(sapply(strsplit(nfl_played$Date_Time," "),"[[", 3)=='AM','AM','PM'))

# combine into a date-time format
nfl_played$DATETIME <- strptime(paste(nfl_played$DATE, nfl_played$TIME),
                                "%Y-%m-%d %I:%M %p",  tz = "EST5EDT")

# Create a game_id in order to merge with other data sets.
# Game_id format is yearmonthday_lowerid_higherid
nfl_played$Game_id <-
   ifelse(nfl_played$Dog_id < nfl_played$Fav_id ,
         paste(format(nfl_played$DATE, '%Y%m%d'), nfl_played$Dog_id, nfl_played$Fav_id, sep="_"),
         paste(format(nfl_played$DATE, '%Y%m%d'), nfl_played$Fav_id, nfl_played$Dog_id, sep="_"))
```

The following are the new fields created from the date and team cleaning.

Table 6: Date and Time Fields

Year	Week	Fav_id	Dog_id	DATE	TIME	DATETIME	Game_id
2018	SB	21	26	2018-02-04	6:30 PM	2018-02-04 18:30:00	20180204_21_26
2018	Conf	21	15	2018-01-21	3:05 PM	2018-01-21 15:05:00	20180121_15_21
2018	Conf	20	26	2018-01-21	6:40 PM	2018-01-21 18:40:00	20180121_20_26
2018	Div	27	15	2018-01-14	1:05 PM	2018-01-14 13:05:00	20180114_15_27
2018	Div	20	22	2018-01-14	4:40 PM	2018-01-14 16:40:00	20180114_20_22

3.3 Team Abbreviations

The last stage of cleaning we are doing is to include an abbreviation for each team_id just to make life easier for visualizations and filtering. In this case we are going to use an abbreviation based on each of the current (as of 2018) 32 teams' locations. So for example, we will label St. Louis Rams games as `LAR` even though that wasn't accurate at the time. We can easily scrape a list of team abbreviations from Wikipedia.

```
# Scrape team abbreviations chart from Wikipedia
wiki <- read_html(paste0("https://en.wikipedia.org/wiki/Wikipedia:WikiProject_",
                         "National_Football_League/National_Football_League_team_abbreviations"))
wiki %>%
  html_nodes(xpath = '//*[@id="mw-content-text"]/div/table') %>%
  html_table() -> tm_abv
tm_abv <- tm_abv[[1]]
tm_abv <- tm_abv[-1,]
names(tm_abv) <- c("Abbrev","Name")

# Merge it with a current (2018) teams list
teams_2018 <- recent_teams[recent_teams$Year_To == 2018 & recent_teams$AV != "" ,]
teams_2018 <- merge(teams_2018, tm_abv, by.x = "Tm", by.y = "Name")

nfl_played2 <- merge(nfl_played, teams_2018[,c(2,21)], by.x="Fav_id", by.y = "team_id",sort = F)
nfl_played2 <- merge(nfl_played2, teams_2018[,c(2,21)], by.x="Dog_id", by.y = "team_id",sort = F)

nfl_played2 <- nfl_played2[ , c(3:7,2,17,8,1,18,9:16)]
names(nfl_played2)[c(7,10)] <- c("Fav_abv", "Dog_abv")

nfl_played2 <- nfl_played2[order(nfl_played2$DATETIME, decreasing = T), ]
```

The following is our fully scraped table from our betting website merged with some team names, abbreviations, and IDs using a couple outside NFL team list sources.

Table 7: Complete NFL Betting Table (continued below)

Year	Game_Type	Week	Date_Time	Favorite	Fav_id	Fav_abv	Underdog	Dog_id	Dog_abv
2018	post	SB	2/4 6:30 ET	New England	21	NE	Philadelphia	26	PHI
2018	post	Conf	1/21 6:40 ET	Minnesota	20	MIN	Philadelphia	26	PHI
2018	post	Conf	1/21 3:05 ET	New England	21	NE	Jacksonville	15	JAX
2018	post	Div	1/14 4:40 ET	Minnesota	20	MIN	New Orleans	22	NO
2018	post	Div	1/14 1:05 ET	Pittsburgh	27	PIT	Jacksonville	15	JAX

Line	Total	Fav_Home	Dog_Home	DATE	TIME	DATETIME	Game_id
-4.5	49	N	N	2018-02-04	6:30 PM	2018-02-04 18:30:00	20180204_21_26
-3	39	N	Y	2018-01-21	6:40 PM	2018-01-21 18:40:00	20180121_20_26
-7.5	45.5	Y	N	2018-01-21	3:05 PM	2018-01-21 15:05:00	20180121_15_21
-5.5	47	Y	N	2018-01-14	4:40 PM	2018-01-14 16:40:00	20180114_20_22
-7	41	Y	N	2018-01-14	1:05 PM	2018-01-14 13:05:00	20180114_15_27

4 NFL Schedule and Results

A database of betting lines and totals is great, but this data becomes much more interesting if we can combine it with the final results from these games. We can then combine these actual results with the betting information to run some analysis on how teams performed versus the odds. In order to get the game outcomes we will be scraping from the NFL schedule on this site: https://www.pro-football-reference.com/years/.

Pro-Football-Reference splits up the schedule by adjusting the year in the URL. All we have to do is loop through the desired years, appending that to the URL, and extract the relevant table. Finally, just a bit of cleaning to put the table into a readable form.

```
pfr_sched <- c()
url_pfr <- "https://www.pro-football-reference.com/years/"

# loop through seasons in URL to extract schedule table
for (yr_pfr in 2005:2017) {
pfr <- read_html(paste0(url_pfr,yr_pfr,"/games.htm"))
pfr %>%
  html_nodes(xpath = '///*[@id="games"]') %>%
  html_table() -> pfr_temp
pfr_temp <- pfr_temp[[1]]
pfr_temp$Season_id <- yr_pfr
names(pfr_temp) <- c("Week", "Day", "Date", "Time", "Winner.tie", "at", "Loser.tie", "box",
                     "PtsW", "PtsL", "YdsW", "TOW", "YdsL", "TOL", "Season_id")
pfr_temp <- pfr_temp[pfr_temp$box != "",]
pfr_sched <- rbind(pfr_sched, pfr_temp)
}

pfr_sched$DATE <- NA
# add a DATE field to match betting table date
for (i in 1:nrow(pfr_sched)) {
   if (format(as.Date(pfr_sched$Date[i],'%b %d'), '%m') <= '02') {
      pfr_sched$DATE[i] <-
        as.Date(paste0(pfr_sched$Date[i], ' ', pfr_sched$Season_id[i]+1), format = "%b %d %Y")
    } else {
      pfr_sched$DATE[i] <-
        as.Date(paste0(pfr_sched$Date[i], ' ', pfr_sched$Season_id[i]), format = "%b %d %Y") } }
pfr_sched$DATE <- as.Date(pfr_sched$DATE, origin = "1970-01-01")
```

Table 9: Fully Scraped Schedule Table (continued below)

Week	Day	Date	Time	Winner.tie	at	Loser.tie
1	Thu	September 8	9:07PM	New England Patriots		Oakland Raiders
1	Sun	September 11	1:00PM	Kansas City Chiefs		New York Jets
1	Sun	September 11	1:05PM	Buffalo Bills		Houston Texans
1	Sun	September 11	1:05PM	Pittsburgh Steelers		Tennessee Titans
1	Sun	September 11	1:05PM	Tampa Bay Buccaneers	@	Minnesota Vikings

box	PtsW	PtsL	YdsW	TOW	YdsL	TOL	Season_id	DATE
boxscore	30	20	379	0	338	1	2005	2005-09-08
boxscore	27	7	389	1	390	3	2005	2005-09-11
boxscore	22	7	316	0	120	5	2005	2005-09-11
boxscore	34	7	424	0	303	4	2005	2005-09-11
boxscore	24	13	345	2	248	5	2005	2005-09-11

We want to include the `team_id` variable from before into the schedule table. Once we have those included we can recreate the `Game_id` variable and then merge it with our betting data. The schedule table uses the full name which we have included in the team list. However, like before, we will use the most recent instance of a team name to resolve any issues of multiple entries in our team list with the same name.

```
# merge using aggregate to solve cases with multiple matches of same ID
# create aggregate of city with rk = 1 referring to most recent team in that city
team_agg <- teams %>% arrange(Tm, Year_To) %>%
  group_by(Tm) %>%
  mutate(rank = rank(-Year_To, ties.method = "first"))
team_agg2 <- team_agg[team_agg$rank == 1,]

# LEFT JOIN schedule table with team list and IDs
pfr_sched2 <- merge(pfr_sched, team_agg2[,c(1,2)], by.x="Winner.tie", by.y = "Tm", all.x= T, sort = F)
pfr_sched2 <- merge(pfr_sched2, team_agg2[,c(1,2)], by.x="Loser.tie", by.y = "Tm", all.x = T, sort = F)

# Create Game_id field
pfr_sched2$DATE <- as.Date(pfr_sched2$DATE, origin = "1970-01-01")
pfr_sched2$Game_id <-
  ifelse(pfr_sched2$team_id.x < pfr_sched2$team_id.y,
  paste(format(pfr_sched2$DATE, '%Y%m%d'), pfr_sched2$team_id.x, pfr_sched2$team_id.y, sep="_"),
  paste(format(pfr_sched2$DATE, '%Y%m%d'), pfr_sched2$team_id.y, pfr_sched2$team_id.x, sep="_"))

pfr_sched2 <- pfr_sched2[, c(15,3:6,2,17,7,1,18,8:14,16,19)]
names(pfr_sched2)[c(7,10)] <- c("W_id", "L_id")

# LEFT JOIN betting data with schedule table
nfl_full <- merge(nfl_played2, pfr_sched2[,c(1,7,10,19,12,13)], by = "Game_id", all.x = T)
```

We now have all the NFL betting data from before linked with the schedule table to create a combined database of all the information we want. With all this merging it's important to see if there was any strange behavior that might have occurred. We see that one of the games did not join correctly with the schedule table and has no results for the score.

```
# mislabeled date in betting table
nfl_full[is.na(nfl_full$PtsW),]
```

```
##             Game_id Year Game_Type Week    Date_Time    Favorite Fav_id
## 1095 20100919_22_28 2010       reg    2 9/19 8:20 ET New Orleans     22
##      Fav_abv      Underdog Dog_id Dog_abv Line Total Fav_Home Dog_Home
## 1095      NO San Francisco     28      SF -4.5    44        N        Y
##            DATE    TIME            DATETIME Season_id W_id L_id PtsW PtsL
## 1095 2010-09-19 8:20 PM 2010-09-19 20:20:00        NA   NA   NA <NA> <NA>
```

```
pfr_sched2[(pfr_sched2$Week == 2 & pfr_sched2$Season == 2010 &
           (pfr_sched2$W_id == 28 | pfr_sched2$L_id == 28)) ,]
```

```
##      Season_id Week Day         Date   Time         Winner.tie W_id at
## 1586      2010    2 Mon September 20 8:36PM New Orleans Saints   22  @
##                Loser.tie L_id      box PtsW PtsL YdsW TOW YdsL TOL
## 1586 San Francisco 49ers   28 boxscore   25   22  287   0  417   4
##            DATE        Game_id
## 1586 2010-09-20 20100920_22_28
```

There was an error entering the Week 2 - 2010 - SF vs NO game on the betting website. This game actually occurred on September 20, not September 19. We will manually adjust it below.

```
# manually adjust 9/20/2010 game between SF & NO
row_to_adj = which(nfl_full$Game_id == '20100919_22_28')
nfl_full[row_to_adj,18] <- nfl_full[1095,18]+60*60*24
nfl_full[row_to_adj,c(1,5,16,19:23)] <- c("20100920_22_28", "9/20 8:20 ET", "2010-09-19",
                                          2010, 22, 28, 25, 22)
```

We continue to check if there is anything potentially off with our merged tables. Each week in the regular season should have at most 16 games. Some regular season weeks have fewer due to bye weeks and postponed games. The number of postseason games in each round should be 4-4-2-1. Something is happening in 2016 - Week 15.

```
# table is off in 2016 - Week 15 (can't have 17 games in a week)
table(nfl_full$Week, nfl_full$Season_id)
```

```
##
##        2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017
##   1       0   16   16   16   16   16   16   16   16   16   16   16   15
##   2       0   16   16   15   16   16   16   16   16   16   16   16   16
##   3       0   14   16   16   16   16   16   16   16   16   16   16   16
##   4       0   14   14   13   14   14   16   15   15   13   15   15   16
##   5       0   14   14   14   14   13   13   14   14   15   14   14   14
##   6       0   13   13   14   14   14   13   14   15   15   14   15   14
##   7       0   13   14   14   13   14   13   13   15   15   14   15   15
##   8       0   14   13   14   13   13   13   14   13   15   14   13   13
##   9       0   14   14   14   13   13   14   14   12   13   13   13    0
##   10      0   16   14   14   15   14   16   14   14   13   14   14    0
##   11      0   16   16   16   16   16   14   14   15   14   14   14    0
##   12      0   16   16   16   16   16   16   16   14   15   16   16    0
##   13      0   16   16   16   16   16   16   16   16   16   16   15    0
##   14      0   16   16   16   16   16   16   16   16   16   16   16    0
##   15      0   16   16   16   16   16   16   16   16   16   16   17    0
##   16      0   16   16   16   16   16   16   16   16   16   16   16    0
##   17      0   16   16   16   16   16   16   16   16   16   16   16    0
##   WC      0    4    4    4    4    4    4    4    4    4    4    4    4
##   Div     0    4    4    4    4    4    4    4    4    4    4    4    4
##   Conf    0    2    2    2    2    2    2    2    2    2    2    2    2
##   SB      1    1    1    1    1    1    1    1    1    1    1    1    1
```

```
head(nfl_full[nfl_full$Week == '15' & nfl_full$Season_id == 2016,1:6])
```

```
##             Game_id Year Game_Type Week     Date_Time    Favorite
## 2878  20161212_3_21 2016       reg   15 12/12 8:30 ET New England
## 2879 20161215_18_29 2016       reg   15 12/15 8:25 ET     Seattle
## 2880 20161217_19_24 2016       reg   15 12/17 8:25 ET       Miami
## 2881  20161218_1_22 2016       reg   15 12/18 4:05 ET     Arizona
## 2882 20161218_10_21 2016       reg   15 12/18 4:25 ET New England
## 2883 20161218_11_23 2016       reg   15 12/18 1:00 ET   NY Giants
```

Looking into that week we see that NE played twice in that week, which is not possible. Let's see what's going on in Weeks 14 and 15 for New England.

```
# Two entries on 12/12/16. One listed as Week 14, one as Week 15.
nfl_full[(nfl_full$Fav_id == 21 | nfl_full$Dog_id == 21) &
             (nfl_full$Week == '14' | nfl_full$Week == '15') &
              nfl_full$Season_id == 2016,1:6]
```

```
##             Game_id Year Game_Type Week     Date_Time    Favorite
## 2877  20161212_3_21 2016       reg   14 12/12 8:30 ET New England
## 2878  20161212_3_21 2016       reg   15 12/12 8:30 ET New England
## 2882 20161218_10_21 2016       reg   15 12/18 4:25 ET New England
```

```
# Accurate schedule table lists Week 14 game on 12/12 and Week 15 game on 12/18
pfr_sched2[(pfr_sched2$W_id == 21 | pfr_sched2$L_id == 21) &
           (pfr_sched2$Week == '14' | pfr_sched2$Week == '15') &
            pfr_sched2$Season_id == 2016,1:6]
```

```
##     Season_id Week Day        Date   Time           Winner.tie
## 149      2016   15 Sun December 18 4:25PM New England Patriots
## 517      2016   14 Mon December 12 8:30PM New England Patriots
```

```
# Incorrect game is duplicated 12/12 game which was listed as Week 15
row_to_delete = which(nfl_full$Game_id == '20161212_3_21' & nfl_full$Week == '15')
nfl_full <- nfl_full[-row_to_delete,]

# correct 2016 table
table(nfl_full[nfl_full$Season_id == 2016,]$Week)
```

```
##
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
##   16   16   16   15   14   15   15   13   13   14   14   16   15   16   16
##   16   17   WC  Div Conf   SB
##   16   16    4    4    2    1
```

We now have exactly what we're looking for. Betting data from 2006-2017 merged with the actual results from those games. We can now run some analysis with this data.

Table 11: Complete Betting Table (continued below)

Game_id	Year	Game_Type	Week	Date_Time	Favorite	Fav_id	Fav_abv
20180204_21_26	2018	post	SB	2/4 6:30 ET	New England	21	NE
20180121_20_26	2018	post	Conf	1/21 6:40 ET	Minnesota	20	MIN
20180121_15_21	2018	post	Conf	1/21 3:05 ET	New England	21	NE
20180114_20_22	2018	post	Div	1/14 4:40 ET	Minnesota	20	MIN
20180114_15_27	2018	post	Div	1/14 1:05 ET	Pittsburgh	27	PIT

Table 12: Table continues below

Underdog	Dog_id	Dog_abv	Line	Total	Fav_Home	Dog_Home	DATE	TIME
Philadelphia	26	PHI	-4.5	49	N	N	2018-02-04	6:30 PM
Philadelphia	26	PHI	-3	39	N	Y	2018-01-21	6:40 PM
Jacksonville	15	JAX	-7.5	45.5	Y	N	2018-01-21	3:05 PM
New Orleans	22	NO	-5.5	47	Y	N	2018-01-14	4:40 PM
Jacksonville	15	JAX	-7	41	Y	N	2018-01-14	1:05 PM

DATETIME	Season_id	W_id	L_id	PtsW	PtsL
2018-02-04 18:30:00	2017	26	21	41	33
2018-01-21 18:40:00	2017	26	20	38	7
2018-01-21 15:05:00	2017	21	15	24	20
2018-01-14 16:40:00	2017	20	22	29	24
2018-01-14 13:05:00	2017	15	27	45	42

5 Analysis

With our data cleaned and ready to work with we can adjust the data to examine how a team performs compared to the betting line or spread. If a line is listed as -7.5, for example, that means the favorite needs to win by at least 8 points or more in order to "cover the spread." This is the mechanism used to handicap the superior teams in order to encourage bettors to place bets for both teams and ensure there is equal number of bets for both teams.

Thus, in order to conduct some analysis of these betting spreads it will be helpful to convert the lines into numeric form. Then we can compare the betting lines and totals to the actual results from the game. Some betting lines are listed as `PK` which in betting terms means "pick-em" game. A pick-em game means the spread of the game is 0, essentially the teams are evenly matched. We convert those games to a line of 0 and convert the totals to numeric as well.

```
nfl_full$PtsW <- as.numeric(nfl_full$PtsW)
nfl_full$PtsL <- as.numeric(nfl_full$PtsL)

# Convert to numeric columns
nfl_played[nfl_played$Line == 'PK',]$Line <- 0
nfl_played$Line <- as.numeric(nfl_played$Line)
nfl_played$Total <- as.numeric(nfl_played$Total)
```

5.1 Performance vs. Spread

As of now, our betting table lists each game once and the line is relative to the favorite. If we want to see how a team performed vs. the spread we will need each game listed twice: with a negative spread (e.g. -7.5) for the favorite and a positive line (e.g. +7.5) for the underdog. Unless, of course, the game is a pick-em with a spread of 0.

I actually found SQL to be the most intuitive way to adjust the betting table to be based on a team's performance. `SQLDF` is an R package that allows for SQL Lite code to be written directly into R. We will now have the score and result of every game as well as the team's performance against the spread (ATS).

```
nfl <- nfl_full[, c(19,4,3,1,16,17,7,6,8,10,9,11,12,14,15,13,22,23,20,21)]
team_perf<-sqldf("
       SELECT a.*,
             CASE WHEN a.score_diff > 0 THEN 'Win'
                  WHEN a.score_diff < 0 THEN 'Loss'
                  ELSE 'Tie' END as rec,
              CASE WHEN a.tm_line + a.score_diff > 0 THEN 'Cover'
                   WHEN a.tm_line + a.score_diff < 0 THEN 'Loss'
                   ELSE 'Push' END as vs_spread,
              CASE WHEN a.score_comb > a.total  THEN 'Over'
                   WHEN a.score_comb < a.total THEN 'Under'
                   ELSE 'Push' END as o_u
       FROM (SELECT n.season_id, n.week, t.team_id, t.abbrev, n.date, n.total,
       CASE t.team_id WHEN n.fav_id THEN n.dog_abv
                      ELSE n.fav_abv END as opp,
       CASE t.team_id WHEN n.fav_id THEN n.fav_home
                      ELSE n.dog_home END as at_home,
       CASE t.team_id WHEN n.fav_id then n.line
                     ELSE n.line*-1 END as tm_line,
       CASE t.team_id WHEN n.w_id then floor(ptsw)||'-'||floor(ptsl)
                      ELSE floor(ptsl)||'-'||floor(ptsw) END as score,
       CASE t.team_id WHEN n.w_id then ptsw-ptsl
                      ELSE ptsl-ptsw END as score_diff,
       ptsl+ptsw as score_comb
      FROM  nfl n, teams_2018 t
      WHERE t.team_id IN (n.fav_id, n.dog_id)) a
```

```
      ORDER BY a.date
      ")
team_perf$date <- as.Date(team_perf$date, origin = "1970-01-01")
```

Table 14: Team Record and ATS

season_id	week	team_id	abbrev	date	total	opp	at_home	tm_line	score	score_diff	score_comb	rec	vs_spread	o_u
2005	SB	27	PIT	2006-02-05	47	SEA	N	-4	21-10	11	31	Win	Cover	Under
2005	SB	29	SEA	2006-02-05	47	PIT	N	4	10-21	-11	31	Loss	Loss	Under
2006	1	19	MIA	2006-09-07	34	PIT	N	1.5	17-28	-11	45	Loss	Loss	Over
2006	1	27	PIT	2006-09-07	34	MIA	Y	-1.5	28-17	11	45	Win	Cover	Over
2006	1	14	IND	2006-09-10	48	NYG	N	-3	26-21	5	47	Win	Cover	Under

With the breakdown of each game determined we can now group the results based on each team in each season. Once again SQL was easier to use in order to perform this aggregation. The resulting table is shown below.

```
tm_by_yr <- sqldf("
       SELECT season_id, team_id, max(abbrev) as abbrev,
              sum(case when rec = 'Win' then 1 else 0 end) as rec_win,
              sum(case when rec = 'Loss' then 1 else 0 end) as rec_loss,
              sum(case when rec = 'Tie' then 1 else 0 end) as rec_tie,
              sum(case when vs_spread = 'Cover' then 1 else 0 end) as ats_cover,
              sum(case when vs_spread = 'Loss' then 1 else 0 end) as ats_loss,
              sum(case when vs_spread = 'Push' then 1 else 0 end) as ats_push
       FROM team_perf
       GROUP BY team_id, season_id
       ORDER BY season_id, team_id
      ")
tm_by_yr$rec_wpct <- format(
                     round((tm_by_yr$rec_win + 0.5*tm_by_yr$rec_tie)/
                     (tm_by_yr$rec_win + tm_by_yr$rec_loss + tm_by_yr$rec_tie),3),
                     nsmall = 3)
tm_by_yr$ats_wpct <- format(
                     round((tm_by_yr$ats_cover + 0.5*tm_by_yr$ats_push)/
                     (tm_by_yr$ats_cover + tm_by_yr$ats_loss + tm_by_yr$ats_push),3),
                     nsmall = 3)
tm_by_yr <- tm_by_yr[,c(1:6,11,7:9,10)]
```

Table 15: Team Rec vs. Spread

season_id	team_id	abbrev	rec_win	rec_loss	rec_tie	ats_wpct	ats_cover	ats_loss	ats_push	rec_wpct
2005	27	PIT	1	0	0	1.000	1	0	0	1.000
2005	29	SEA	0	1	0	0.000	0	1	0	0.000
2006	1	ARI	5	11	0	0.500	8	8	0	0.312
2006	2	ATL	7	9	0	0.438	7	9	0	0.438
2006	3	BAL	13	4	0	0.588	10	7	0	0.765

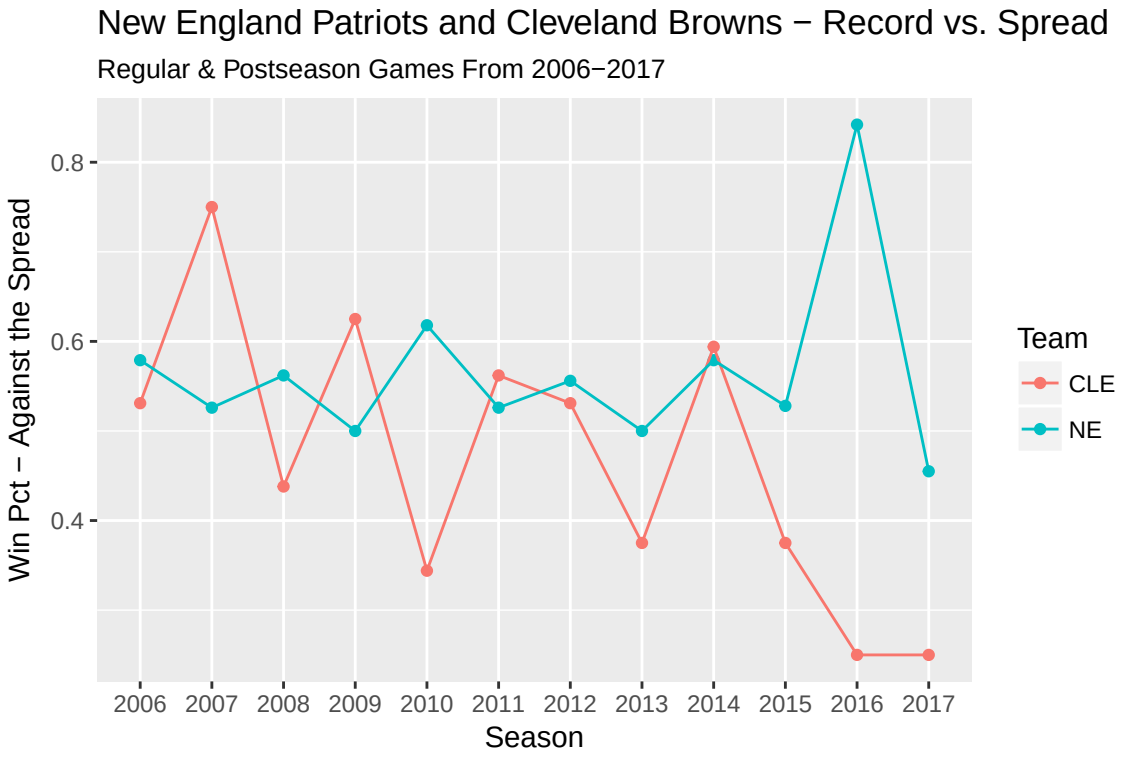
I decided to group this once more to find the total team record across the entire dataset. This gives us a sense of which team had the best record over the entire time frame (and best record against the spread).

```
tm_total <- sqldf("
       SELECT team_id, max(abbrev) as abbrev,
              sum(rec_win) as rec_win,
              sum(rec_loss) as rec_loss,
              sum(rec_tie) as rec_tie,
              round((sum(rec_win) + sum(rec_tie)*0.5) /
              (sum(rec_win)+sum(rec_loss)+sum(rec_tie)),3) as rec_wpct,
              sum(ats_cover) as ats_cover,
              sum(ats_loss) as ats_loss,
              sum(ats_push) as ats_push,
              round((sum(ats_cover) + sum(ats_push)*0.5) /
              (sum(ats_cover)+sum(ats_loss)+sum(ats_push)),3) as ats_wpct
       FROM tm_by_yr
       group by team_id
      ")
tm_total$rec_wpct <- format(tm_total$rec_wpct, nsmall = 3)
tm_total$ats_wpct <- format(tm_total$ats_wpct, nsmall = 3)
```

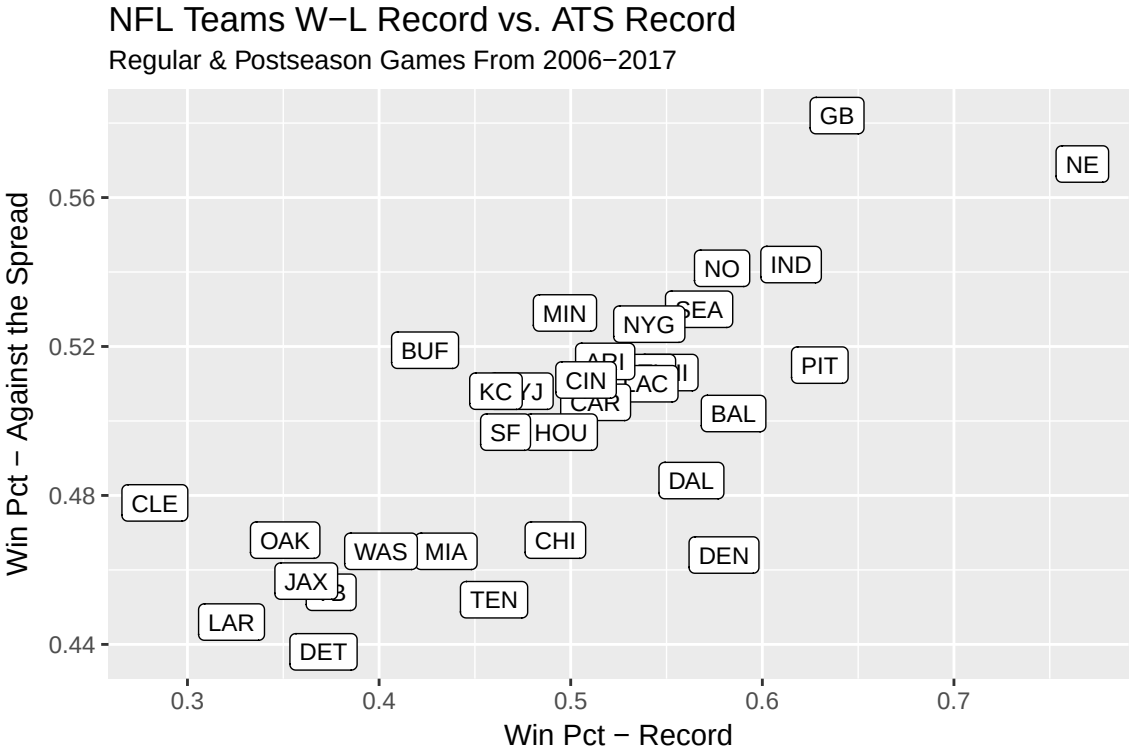
Table 16: Best/Worst Records since 2006

	team_id	abbrev	rec_win	rec_loss	rec_tie	rec_wpct	ats_cover	ats_loss	ats_push	ats_wpct
1	21	NE	161	49	0	0.767	117	88	5	0.569
2	12	GB	128	72	1	0.639	114	81	6	0.582
3	27	PIT	126	74	0	0.63	100	94	6	0.515
30	25	OAK	65	120	0	0.351	86	98	1	0.468
31	18	LAR	59	124	1	0.323	80	100	4	0.446
32	8	CLE	52	132	0	0.283	85	93	6	0.478

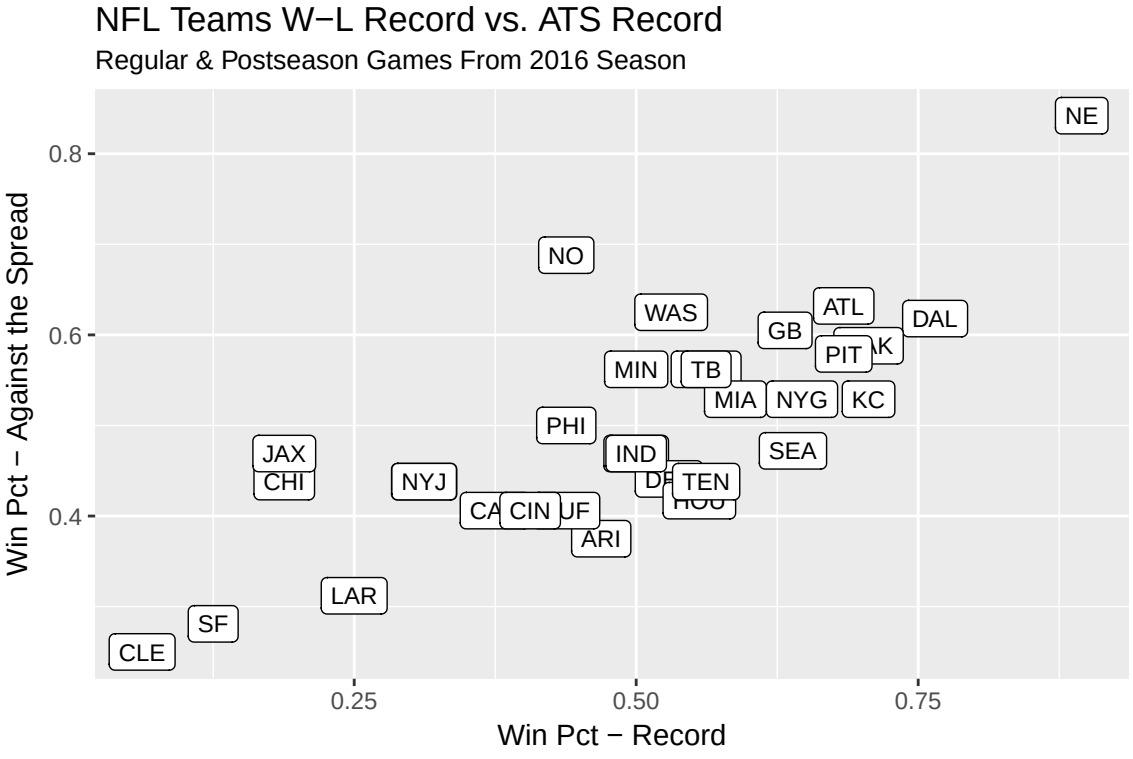
If someone has paid any attention to the NFL over the past decade this table is not surprising. We see the team with the best record since 2006 is the New England Patriots while the team with the worst record is the Cleveland Browns. But how do those overall records compare to the overall performance against the Vegas spreads?



Interestingly enough, the Browns actually outperformed the Patriots several different times compared to the spread over this time frame. But if we look at the overall trend, good teams do seem to have a better record versus the Vegas spread than the bad teams. If all game lines were perfect, that would mean on the long run teams have a .500 record versus the spread despite their actual W-L record. We see that although the winning percentage against the spread for these better teams (NE, NYG, PIT, etc) is closer to .500 than their actual record, a general trend between W-L record and ATS record does exist.



We observe a similar relationship when taking a look at a single season's worth of data. The following is the results from just the 2016 season.



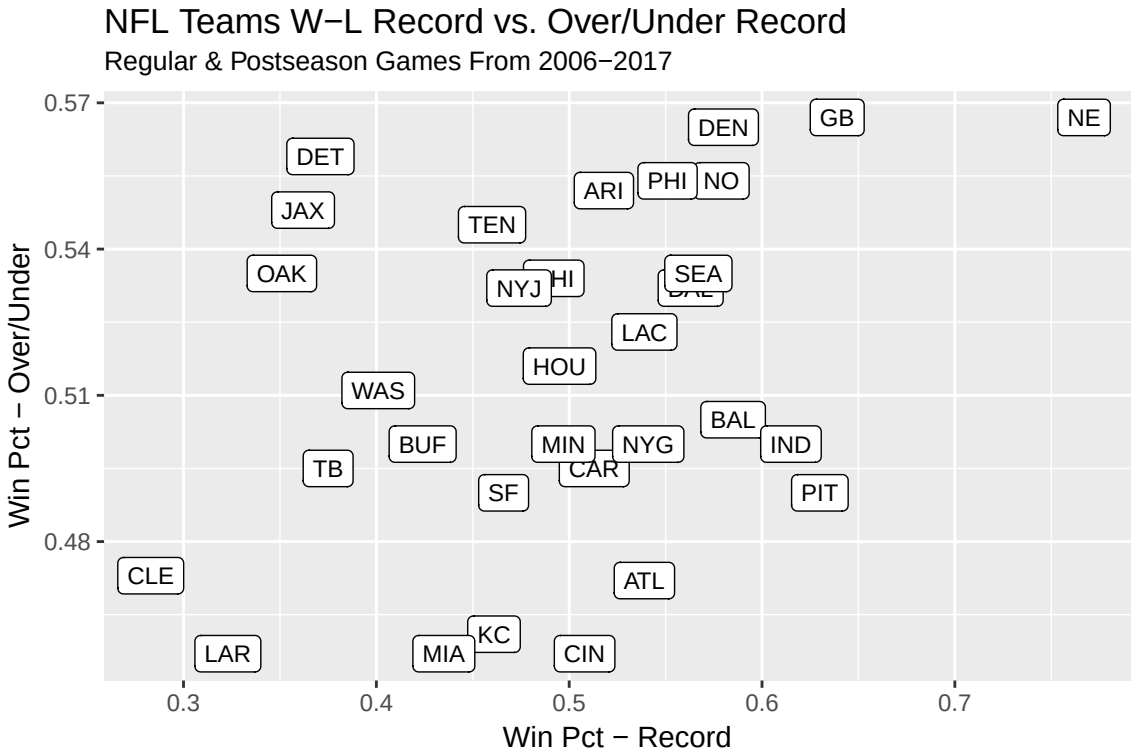
5.2 Performance vs. Total

The `Total` is a betting figure where a person can attempt to predict what the combined score of the game will be. If someone does not have a guess who will win, but has a sense if the game will be high scoring or low scoring, they can choose to bet the Over/Under on the total. We can take a look at if teams that win more games, more commonly hit the over.

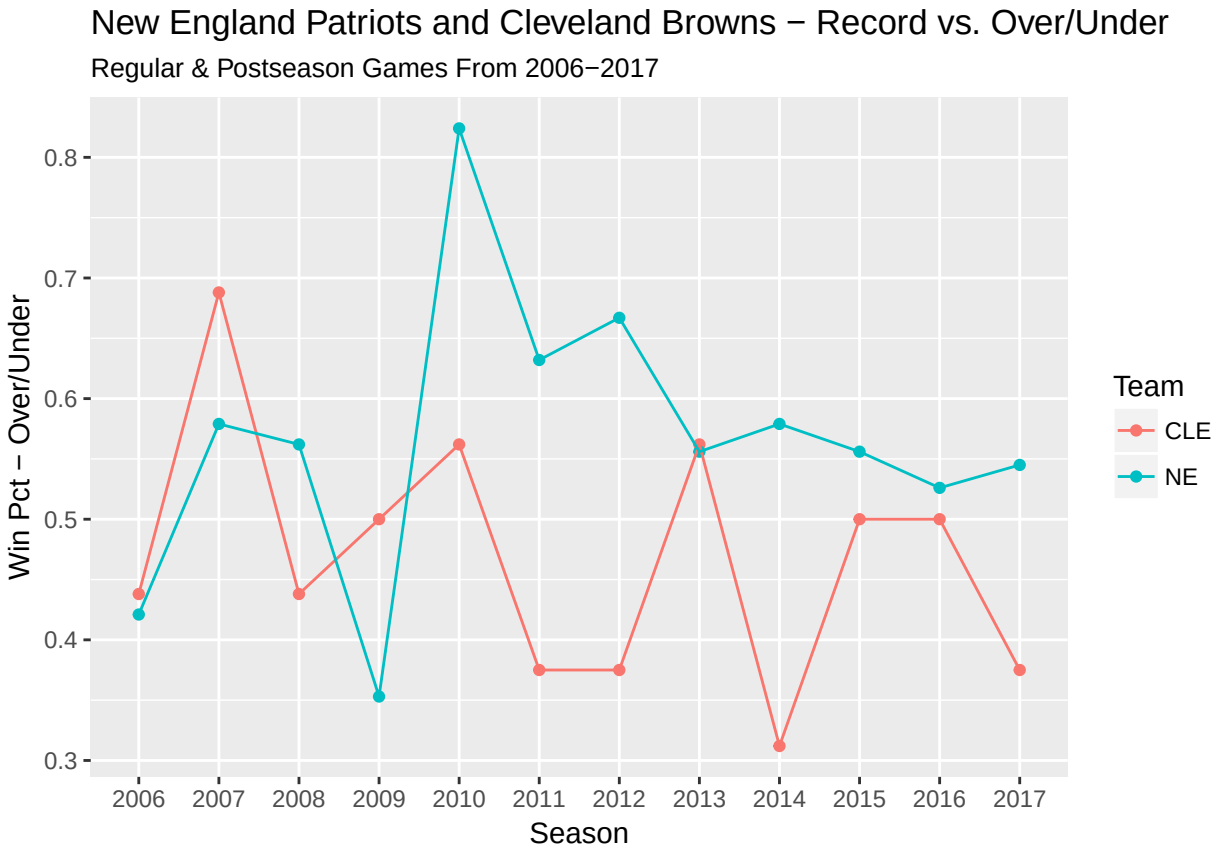
```
ou_by_yr <- sqldf("
       SELECT season_id, team_id, max(abbrev) as abbrev,
              sum(case when rec = 'Win' then 1 else 0 end) as rec_win,
              sum(case when rec = 'Loss' then 1 else 0 end) as rec_loss,
              sum(case when rec = 'Tie' then 1 else 0 end) as rec_tie,
              sum(case when o_u = 'Over' then 1 else 0 end) as ou_over,
              sum(case when o_u = 'Under' then 1 else 0 end) as ou_under,
              sum(case when o_u = 'Push' then 1 else 0 end) as ou_push
       FROM team_perf
       GROUP BY team_id, season_id
       ORDER BY season_id, team_id
      ")
ou_by_yr$rec_wpct <- format(
                     round((ou_by_yr$rec_win + 0.5*ou_by_yr$rec_tie)/
                     (ou_by_yr$rec_win + ou_by_yr$rec_loss + ou_by_yr$rec_tie),3),
                     nsmall = 3)
ou_by_yr$ou_wpct <- format(
                     round((ou_by_yr$ou_over + 0.5*ou_by_yr$ou_push)/
                     (ou_by_yr$ou_over + ou_by_yr$ou_under + ou_by_yr$ou_push),3),
                     nsmall = 3)
ou_by_yr <- ou_by_yr[,c(1:6,10,7:9,11)]
```

```
ou_total <- sqldf("
       SELECT team_id, max(abbrev) as abbrev,
              sum(rec_win) as rec_win,
              sum(rec_loss) as rec_loss,
              sum(rec_tie) as rec_tie,
              round((sum(rec_win) + sum(rec_tie)*0.5) /
              (sum(rec_win)+sum(rec_loss)+sum(rec_tie)),3) as rec_wpct,
              sum(ou_over) as ats_cover,
              sum(ou_under) as ats_loss,
              sum(ou_push) as ats_push,
              round((sum(ou_over) + sum(ou_push)*0.5) /
              (sum(ou_over)+sum(ou_under)+sum(ou_push)),3) as ou_wpct
       FROM ou_by_yr
       group by team_id
      ")
ou_total$rec_wpct <- format(ou_total$rec_wpct, nsmall = 3)
ou_total$ou_wpct <- format(ou_total$ou_wpct, nsmall = 3)
```

The trend does seem to exist for better teams covering the total more often. However, the evidence is far weaker than it is for covering the spread. One simple reason for this would be that defense is an important component of how good a team is. So it's quite possible that the better teams hold their opponent to lower scoring amounts and thus the total is not reached despite how many points the better team scored themself.



Once again in a comparison of the Patriots compared to the Browns, we do see that the Patriots go over the total more often than Cleveland does. Although the discrepancy is not significant.



6 Conclusion

There is a plethora of more analysis we could take a look at with our betting data. We did not even delve into home versus away data. In general Las Vegas gives a baseline of about 3 points to the spread to the home team just to offset the home-field advantage. We could take a look at whether this home-field advantage truly exists or if Vegas should even increase that 3-point bonus to an even higher number.

We could also take a look at playoff versus regular season performance. Do teams get more conservative in the postseason and thus don't go over the total more often? Do better teams cover the spreads more often in the postseason or does the importance of the playoffs make the lesser teams step up to the occasion?

All in all, the exercise here was to start building an important dataset to the emerging sports betting landscape. The proliferation of sports gambling is on the horizon and it won't be long before access to this information will be crucial.