cfbModel

Matthew Kirslis

2022-10-27

setGlobalChunkOptions

```
# set R markdown global chunk options
  # echo = FALSE prevents code, but not results from appearing
  # include = FALSE prevents code & results from appearing
  # message = FALSE prevents messages generated by code from appearing
  # warning = FALSE prevents warnings generated by code from appearing
  # eval = FALSE prevents code from running
knitr::opts_chunk$set(echo=TRUE)
knitr::opts_chunk$set(include=TRUE)
knitr::opts_chunk$set(message=FALSE)
knitr::opts_chunk$set(warning=FALSE)
knitr::opts_chunk$set(eval=TRUE)
```

setWorkingDirectory

```
# set working directory
  # this is the default location where R will load & save files
setwd("C:/Users/MattDesktop/OneDrive/cfbModel")
###setwd("C:/Users/mkirs/OneDrive/cfbModel")
```

installPackages

loadPackages

```
# load relevant packages
  # we can access the "Help" tab w/ the console command "?package"
  # loading packages is only needed once per R session
library(ranger)
library(tidyverse)
library(grid)
library(gridExtra)
library(rmarkdown)
library(readr)
library(lubridate)
library(moments)
library(zoo)
library(knitr)
library(ggcorrplot)
library(ggthemes)
library(stringr)
library(scales)
```

loadData

```
# load yearly dates tables
  # these .csv files contain "game_id" & "start_date"
X2019gameDates <- read_csv("2019gameDates.csv")
X2020gameDates <- read_csv("2020gameDates.csv")
X2021gameDates <- read_csv("2021gameDates.csv")
X2022gameDates <- read_csv("2022gameDates.csv")

# load 2019-2022 weekly game data
  # these .csv files contain various game stats, ex. passing attempts, tackles
  # one entry for the home team, one entry for the away team for each "game_id"
# load weekly 2019 games
X2019week1 <- read_csv("2019week1.csv")
X2019week2 <- read_csv("2019week2.csv")
X2019week3 <- read_csv("2019week3.csv")
X2019week4 <- read_csv("2019week4.csv")
X2019week5 <- read_csv("2019week5.csv")
X2019week6 <- read_csv("2019week6.csv")
X2019week7 <- read_csv("2019week7.csv")
X2019week8 <- read_csv("2019week8.csv")
X2019week9 <- read_csv("2019week9.csv")
X2019week10 <- read_csv("2019week10.csv")
X2019week11 <- read_csv("2019week11.csv")
X2019week12 <- read_csv("2019week12.csv")
X2019week13 <- read_csv("2019week13.csv")
X2019week14 <- read_csv("2019week14.csv")
X2019week15 <- read_csv("2019week15.csv")

# load weekly 2020 games
X2020week1 <- read_csv("2020week1.csv")
X2020week2 <- read_csv("2020week2.csv")
X2020week3 <- read_csv("2020week3.csv")
X2020week4 <- read_csv("2020week4.csv")
X2020week5 <- read_csv("2020week5.csv")
X2020week6 <- read_csv("2020week6.csv")
X2020week7 <- read_csv("2020week7.csv")
X2020week8 <- read_csv("2020week8.csv")
X2020week9 <- read_csv("2020week9.csv")
X2020week10 <- read_csv("2020week10.csv")
X2020week11 <- read_csv("2020week11.csv")
X2020week12 <- read_csv("2020week12.csv")
X2020week13 <- read_csv("2020week13.csv")
X2020week14 <- read_csv("2020week14.csv")
X2020week15 <- read_csv("2020week15.csv")
X2020week16 <- read_csv("2020week16.csv")

# load weekly 2021 games
X2021week1 <- read_csv("2021week1.csv")
X2021week2 <- read_csv("2021week2.csv")
X2021week3 <- read_csv("2021week3.csv")
X2021week4 <- read_csv("2021week4.csv")
X2021week5 <- read_csv("2021week5.csv")
```

```
X2021week6 <- read_csv("2021week6.csv")
X2021week7 <- read_csv("2021week7.csv")
X2021week8 <- read_csv("2021week8.csv")
X2021week9 <- read_csv("2021week9.csv")
X2021week10 <- read_csv("2021week10.csv")
X2021week11 <- read_csv("2021week11.csv")
X2021week12 <- read_csv("2021week12.csv")
X2021week13 <- read_csv("2021week13.csv")
X2021week14 <- read_csv("2021week14.csv")
X2021week15 <- read_csv("2021week15.csv")

# load weekly 2022 games
X2022week1 <- read_csv("2022week1.csv")
X2022week2 <- read_csv("2022week2.csv")
X2022week3 <- read_csv("2022week3.csv")
X2022week4 <- read_csv("2022week4.csv")
X2022week5 <- read_csv("2022week5.csv")
X2022week6 <- read_csv("2022week6.csv")
X2022week7 <- read_csv("2022week7.csv")
X2022week8 <- read_csv("2022week8.csv")
```

appendData

```
# append yearly dates data frames
XallGameDates <- rbind(X2019gameDates, X2020gameDates,
                       X2021gameDates, X2022gameDates)

# remove yearly dates data frames
remove(X2019gameDates, X2020gameDates, X2021gameDates, X2022gameDates)

# append weekly games data frames
XallWeeks <- rbind(X2019week1,
                   X2019week2,
                   X2019week3,
                   X2019week4,
                   X2019week5,
                   X2019week6,
                   X2019week7,
                   X2019week8,
                   X2019week9,
                   X2019week10,
                   X2019week11,
                   X2019week12,
                   X2019week13,
                   X2019week14,
                   X2019week15,
                   X2020week1,
                   X2020week2,
                   X2020week3,
                   X2020week4,
                   X2020week5,
                   X2020week6,
                   X2020week7,
                   X2020week8,
                   X2020week9,
                   X2020week10,
                   X2020week11,
                   X2020week12,
                   X2020week13,
                   X2020week14,
                   X2020week15,
                   X2020week16,
                   X2021week1,
                   X2021week2,
                   X2021week3,
                   X2021week4,
                   X2021week5,
                   X2021week6,
                   X2021week7,
                   X2021week8,
                   X2021week9,
                   X2021week10,
                   X2021week11,
                   X2021week12,
```

```
                  X2021week13,
                  X2021week14,
                  X2021week15,
                  X2022week1,
                  X2022week2,
                  X2022week3,
                  X2022week4,
                  X2022week5,
                  X2022week6,
                  X2022week7,
                  X2022week8)

# remove weekly games data frames
remove(X2019week1,
       X2019week2,
       X2019week3,
       X2019week4,
       X2019week5,
       X2019week6,
       X2019week7,
       X2019week8,
       X2019week9,
       X2019week10,
       X2019week11,
       X2019week12,
       X2019week13,
       X2019week14,
       X2019week15,
       X2020week1,
       X2020week2,
       X2020week3,
       X2020week4,
       X2020week5,
       X2020week6,
       X2020week7,
       X2020week8,
       X2020week9,
       X2020week10,
       X2020week11,
       X2020week12,
       X2020week13,
       X2020week14,
       X2020week15,
       X2020week16,
       X2021week1,
       X2021week2,
       X2021week3,
       X2021week4,
       X2021week5,
       X2021week6,
       X2021week7,
       X2021week8,
       X2021week9,
```

```
       X2021week10,
       X2021week11,
       X2021week12,
       X2021week13,
       X2021week14,
       X2021week15,
       X2022week1,
       X2022week2,
       X2022week3,
       X2022week4,
       X2022week5,
       X2022week6,
       X2022week7,
       X2022week8)

# XallWeeks is our untidy games data frame
# XallGameDates is our dates data frame

# data frame vs. data table?
# tables have less syntax -> table code runs quicker
```

wrangleData

```
# this chunk converts XallWeeks into a tidy data frame we can analyze

# pivot "stat_category" w/ values from "stat"
  # a common symptom of columns needing a pivot are repeated entries
  # ex. entries in "stat_category": "passingTDs", "kickingPoints", "fumbles"
XallWeeks <- pivot_wider(XallWeeks, names_from = stat_category,
                         values_from = stat)

# rename games "id" column to "game_id"
  # we need matching column names to merge the games & dates data frames
XallGameDates <- rename(XallGameDates, game_id = id)

# create home & away helper tables
  # we want "game_id" to be our unique identifier
XallWeeksHome <- XallWeeks %>% filter(homeAway == "home")
XallWeeksAway <- XallWeeks %>% filter(homeAway == "away")

# merge home & away helper tables by "game_id"
  # upon merging, columns ending in ".x" = home, columns ending in ".y" = away
XallWeeks <- left_join(XallWeeksHome, XallWeeksAway, by = "game_id")

# remove home & away helper tables
remove(XallWeeksHome)
remove(XallWeeksAway)

# remove 'homeAway' columns
XallWeeks <- select(XallWeeks, -homeAway.x, -homeAway.y)

# change "possessionTime" to %mm format
  # losing the seconds level data is evaluated as insignificant
XallWeeks <- XallWeeks %>%
  mutate(possessionTime.x = substr(XallWeeks$possessionTime.x,
                                   start = 0,
                                   stop = 2))
XallWeeks <- XallWeeks %>%
  mutate(possessionTime.y = substr(XallWeeks$possessionTime.y,
                                   start = 0,
                                   stop = 2))

# split "totalPenaltiesYards" into "penalties" & "penYards"
  # ex. "5-60" -> "5", "60"
XallWeeks <- XallWeeks %>%
  mutate(penalties.x = sub("-.*", "", totalPenaltiesYards.x))
XallWeeks <- XallWeeks %>%
  mutate(penYards.x = sub(".*-", "", totalPenaltiesYards.x))
XallWeeks <- XallWeeks %>%
  mutate(penalties.y = sub("-.*", "", totalPenaltiesYards.y))
XallWeeks <- XallWeeks %>%
  mutate(penYards.y = sub(".*-", "", totalPenaltiesYards.y))

# remove "totalPenaltiesYards"
```

```r
XallWeeks <- select(XallWeeks, -totalPenaltiesYards.x, -totalPenaltiesYards.y)

# split "completionAttempts" into "completions" & "passAtts"
  # ex. "26-32" -> "26", "32"
XallWeeks <- XallWeeks %>%
  mutate(completions.x = sub("-.*", "", completionAttempts.x))
XallWeeks <- XallWeeks %>%
  mutate(passAtts.x = sub(".*-", "", completionAttempts.x))
XallWeeks <- XallWeeks %>%
  mutate(completions.y = sub("-.*", "", completionAttempts.y))
XallWeeks <- XallWeeks %>%
  mutate(passAtts.y = sub(".*-", "", completionAttempts.y))

# remove "completionAttempts"
XallWeeks <- select(XallWeeks, -completionAttempts.x, -completionAttempts.y)

# split "thirdDownEff" into "thirdDownConv" & "thirdDownAtt"
  # ex. "5-15" -> "5", "15"
XallWeeks <- XallWeeks %>%
  mutate(thirdDownConv.x = sub("-.*", "", thirdDownEff.x))
XallWeeks <- XallWeeks %>%
  mutate(thirdDownAtt.x = sub(".*-", "", thirdDownEff.x))
XallWeeks <- XallWeeks %>%
  mutate(thirdDownConv.y = sub("-.*", "", thirdDownEff.y))
XallWeeks <- XallWeeks %>%
  mutate(thirdDownAtt.y = sub(".*-", "", thirdDownEff.y))

# remove "thirdDownEff" columns
XallWeeks <- select(XallWeeks, -thirdDownEff.x, -thirdDownEff.y)

# split "fourthDownEff" into "fourthDownConv" & "fourthDownAtt"
  # ex. "1-3" -> "1", "3"
XallWeeks <- XallWeeks %>%
  mutate(fourthDownConv.x = sub("-.*", "", fourthDownEff.x))
XallWeeks <- XallWeeks %>%
  mutate(fourthDownAtt.x = sub(".*-", "", fourthDownEff.x))
XallWeeks <- XallWeeks %>%
  mutate(fourthDownConv.y = sub("-.*", "", fourthDownEff.y))
XallWeeks <- XallWeeks %>%
  mutate(fourthDownAtt.y = sub(".*-", "", fourthDownEff.y))

# remove "fourthDownEff"
XallWeeks <- select(XallWeeks, -fourthDownEff.x, -fourthDownEff.y)

# remove redundant "passesIntercepted"
  # we have "interceptions"
XallWeeks <- select(XallWeeks, -passesIntercepted.x, -passesIntercepted.y)

# remove redundant "turnovers"
  # we have the sum of "fumblesLost" & "interceptions"
XallWeeks <- select(XallWeeks, -turnovers.x, -turnovers.y)

# arrange data by sequential/ascending "game_id" values
```

```r
XallWeeks <- arrange(XallWeeks, XallWeeks$game_id)

# create helper string list for character variables
XallWeeksCols <- colnames(select(XallWeeks, -school.x, -school.y,
                                 -conference.x, -conference.y))

# format data as numerical
XallWeeks <-  XallWeeks %>% mutate_at(vars(all_of(XallWeeksCols)), as.numeric)

# remove helper string list chr variables
remove(XallWeeksCols)

# replace "NA" values in "conference.y" with "None"
XallWeeks$conference.y <- XallWeeks$conference.y %>% replace_na("None")

# replace all "NA" values with 0
XallWeeks[is.na(XallWeeks)] <- 0

# arrange data by sequential/ascending "game_id" values
XallWeeks <- arrange(XallWeeks, XallWeeks$game_id)

# merge games & dates data frames to introduce "start_date"
XallWeeks <- left_join(XallWeeks, XallGameDates, by = "game_id")

# remove dates data frame
remove(XallGameDates)

# format "start_date" as R date (so R can understand)
XallWeeks <- XallWeeks %>%
  mutate(start_date = as.Date(start_date, tz = "EST", "%YYYY-%mm-%dd"))

# format "game_id" column as character
XallWeeks <-  XallWeeks %>% mutate_at(vars(game_id), as.character)

# create target variables "HMOV" & "TOTPTS" columns
  # HMOV = home margin of victory = (home points - away points)
  # TOTPTS = total points = (home points + away points)
XallWeeks <- XallWeeks %>% mutate(HMOV = points.x - points.y)
XallWeeks <- XallWeeks %>% mutate(TOTPTS = points.x + points.y)

# arrange data by sequential/ascending "start_date" column values
XallWeeks <- arrange(XallWeeks, XallWeeks$start_date)

# XallWeeks is now tidy
```

transposeData

```
# create data frames preDash & XallWeeks1
  # XallWeeks will become rolling avgs w/ a 1 game lag
  # preDash will become current rolling avgs
  # XallWeeks1 will store our raw data in its current form
preDash <- XallWeeks
XallWeeks1 <- XallWeeks
```

transformData

```
# create "i" = (# of previous games considered in rolling avgs)
i <- 7

# group by "school.x"
XallWeeks <- XallWeeks %>% group_by(school.x)

# arrange data by sequential/ascending "start_date" values
XallWeeks <- arrange(XallWeeks, XallWeeks$start_date)

# create .x lagged rolling avgs for home team
XallWeeks <- XallWeeks %>%
  mutate(points.x = rollmean(lag(points.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(rushingTDs.x = rollmean(lag(rushingTDs.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(puntReturnYards.x = rollmean(lag(puntReturnYards.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(puntReturnTDs.x = rollmean(lag(puntReturnTDs.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(puntReturns.x = rollmean(lag(puntReturns.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(passingTDs.x = rollmean(lag(passingTDs.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(kickReturnYards.x = rollmean(lag(kickReturnYards.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(kickReturnTDs.x = rollmean(lag(kickReturnTDs.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(kickReturns.x = rollmean(lag(kickReturns.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(kickingPoints.x = rollmean(lag(kickingPoints.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(interceptionYards.x = rollmean(lag(interceptionYards.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(interceptionTDs.x = rollmean(lag(interceptionTDs.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(fumblesRecovered.x = rollmean(lag(fumblesRecovered.x, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(totalFumbles.x = rollmean(lag(totalFumbles.x, n = 1),
```

```
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(sacks.x = rollmean(lag(sacks.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(possessionTime.x = rollmean(lag(possessionTime.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(interceptions.x = rollmean(lag(interceptions.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(fumblesLost.x = rollmean(lag(fumblesLost.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(yardsPerRushAttempt.x = rollmean(lag(yardsPerRushAttempt.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(rushingAttempts.x = rollmean(lag(rushingAttempts.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(rushingYards.x = rollmean(lag(rushingYards.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(yardsPerPass.x = rollmean(lag(yardsPerPass.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(netPassingYards.x = rollmean(lag(netPassingYards.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(totalYards.x = rollmean(lag(totalYards.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(firstDowns.x = rollmean(lag(firstDowns.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(tacklesForLoss.x = rollmean(lag(tacklesForLoss.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(defensiveTDs.x = rollmean(lag(defensiveTDs.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(tackles.x = rollmean(lag(tackles.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(qbHurries.x = rollmean(lag(qbHurries.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(passesDeflected.x = rollmean(lag(passesDeflected.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(completions.x = rollmean(lag(completions.x, n = 1),
                              k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
```

```
  mutate(passAtts.x = rollmean(lag(passAtts.x, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(thirdDownConv.x = rollmean(lag(thirdDownConv.x, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(thirdDownAtt.x = rollmean(lag(thirdDownAtt.x, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(fourthDownConv.x = rollmean(lag(fourthDownConv.x, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(fourthDownAtt.x = rollmean(lag(fourthDownAtt.x, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(penYards.x = rollmean(lag(penYards.x, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(penalties.x = rollmean(lag(penalties.x, n = 1),
                               k = i, fill = NA, align = "right"))

# ungroup by "school.x"
XallWeeks <- XallWeeks %>% ungroup(school.x)

# group by "school.y"
XallWeeks <- XallWeeks %>% group_by(school.y)

# arrange data by sequential/ascending "start_date" column values
XallWeeks <- arrange(XallWeeks, XallWeeks$start_date)

# create .y lagged rolling averages for away team
XallWeeks <- XallWeeks %>%
  mutate(points.y = rollmean(lag(points.y, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(rushingTDs.y = rollmean(lag(rushingTDs.y, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(puntReturnYards.y = rollmean(lag(puntReturnYards.y, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(puntReturnTDs.y = rollmean(lag(puntReturnTDs.y, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(puntReturns.y = rollmean(lag(puntReturns.y, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(passingTDs.y = rollmean(lag(passingTDs.y, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(kickReturnYards.y = rollmean(lag(kickReturnYards.y, n = 1),
                               k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
```

```
  mutate(kickReturnTDs.y = rollmean(lag(kickReturnTDs.y, n = 1),
                                    k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(kickReturns.y = rollmean(lag(kickReturns.y, n = 1),
                                  k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(kickingPoints.y = rollmean(lag(kickingPoints.y, n = 1),
                                    k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(interceptionYards.y = rollmean(lag(interceptionYards.y, n = 1),
                                        k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(interceptionTDs.y = rollmean(lag(interceptionTDs.y, n = 1),
                                      k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(fumblesRecovered.y = rollmean(lag(fumblesRecovered.y, n = 1),
                                       k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(totalFumbles.y = rollmean(lag(totalFumbles.y, n = 1),
                                   k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(sacks.y = rollmean(lag(sacks.y, n = 1),
                            k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(possessionTime.y = rollmean(lag(possessionTime.y, n = 1),
                                     k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(interceptions.y = rollmean(lag(interceptions.y, n = 1),
                                    k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(fumblesLost.y = rollmean(lag(fumblesLost.y, n = 1),
                                  k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(yardsPerRushAttempt.y = rollmean(lag(yardsPerRushAttempt.y, n = 1),
                                          k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(rushingAttempts.y = rollmean(lag(rushingAttempts.y, n = 1),
                                      k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(rushingYards.y = rollmean(lag(rushingYards.y, n = 1),
                                   k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(yardsPerPass.y = rollmean(lag(yardsPerPass.y, n = 1),
                                   k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(netPassingYards.y = rollmean(lag(netPassingYards.y, n = 1),
                                      k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(totalYards.y = rollmean(lag(totalYards.y, n = 1),
                                 k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(firstDowns.y = rollmean(lag(firstDowns.y, n = 1),
                                 k = i, fill = NA, align = "right"))
```

```
XallWeeks <- XallWeeks %>%
  mutate(tacklesForLoss.y = rollmean(lag(tacklesForLoss.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(defensiveTDs.y = rollmean(lag(defensiveTDs.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(tackles.y = rollmean(lag(tackles.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(qbHurries.y = rollmean(lag(qbHurries.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(passesDeflected.y = rollmean(lag(passesDeflected.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(completions.y = rollmean(lag(completions.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(passAtts.y = rollmean(lag(passAtts.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(thirdDownConv.y = rollmean(lag(thirdDownConv.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(thirdDownAtt.y = rollmean(lag(thirdDownAtt.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(fourthDownConv.y = rollmean(lag(fourthDownConv.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(fourthDownAtt.y = rollmean(lag(fourthDownAtt.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(penYards.y = rollmean(lag(penYards.y, n = 1),
                             k = i, fill = NA, align = "right"))
XallWeeks <- XallWeeks %>%
  mutate(penalties.y = rollmean(lag(penalties.y, n = 1),
                             k = i, fill = NA, align = "right"))

# ungroup by "school.y"
XallWeeks <- XallWeeks %>% ungroup(school.y)

# arrange data by sequential/ascending "start_date" column values
XallWeeks <- arrange(XallWeeks, XallWeeks$start_date)

# remove games with no valid rolling averages (first i games for each team)
  # home and away games are considered separately
XallWeeks <- XallWeeks %>% na.omit()

# create "completionPct"
XallWeeks <- XallWeeks %>% mutate(completionPct.x = completions.x/passAtts.x)
XallWeeks <- XallWeeks %>% mutate(completionPct.y = completions.y/passAtts.y)
```

```r
# create "thirdDownEff"
XallWeeks <- XallWeeks %>%
  mutate(thirdDownEff.x = thirdDownConv.x/thirdDownAtt.x)
XallWeeks <- XallWeeks %>%
  mutate(thirdDownEff.y = thirdDownConv.y/thirdDownAtt.y)

# create "fourthDownEff"
XallWeeks <- XallWeeks %>%
  mutate(fourthDownEff.x = fourthDownConv.x/fourthDownAtt.x)
XallWeeks <- XallWeeks %>%
  mutate(fourthDownEff.y = fourthDownConv.y/fourthDownAtt.y)

# create "yardsPerInt"
XallWeeks <- XallWeeks %>%
  mutate(yardsPerInt.x = interceptionYards.x/interceptions.x)
XallWeeks <- XallWeeks %>%
  mutate(yardsPerInt.y = interceptionYards.y/interceptions.y)

# remove invalid values
  # rare stats like "fourthDownConv" often do not occur over a 7 game span
XallWeeks[XallWeeks == Inf] <- 0
XallWeeks[XallWeeks == -Inf] <- 0
XallWeeks <- na.omit(XallWeeks)

# create categorical variables for non-numerical data
XallWeeks$conference.x<-as.factor(XallWeeks$conference.x)
XallWeeks$conference.y<-as.factor(XallWeeks$conference.y)
XallWeeks$school.x<-as.factor(XallWeeks$school.x)
XallWeeks$school.y<-as.factor(XallWeeks$school.y)

# remove third & fourth down conversions & attempts in favor of eff
XallWeeks <- select(XallWeeks, -thirdDownAtt.x, -thirdDownConv.x,
                    -fourthDownConv.x, -fourthDownAtt.x,
                    -thirdDownAtt.y, -thirdDownConv.y,
                    -fourthDownAtt.y, -fourthDownConv.y)

# remove "netPassingYards" in favor of "yardsPerPass"
XallWeeks <- select(XallWeeks, -netPassingYards.x, -netPassingYards.y)

# remove "rushingYards" in favor of "yardsPerRushAttempt"
XallWeeks <- select(XallWeeks, -rushingYards.x, -rushingYards.y)

# remove "completions" in favor of "completionPct"
XallWeeks <- select(XallWeeks, -completions.x, -completions.y)

# remove "interceptionYards" in favor of "yardsPerInt"
XallWeeks <- select(XallWeeks, -interceptionYards.x, -interceptionYards.y)

# remove "totalYards" in favor of rush/pass yards stats
XallWeeks <- select(XallWeeks, -totalYards.x, -totalYards.y)

# remove "interceptionTDs" in favor of "defensiveTDs"
XallWeeks <- select(XallWeeks, -interceptionTDs.x, -interceptionTDs.y)
```

```
# remove "points" in favor of rushing, passing, ..., etc. TDs
XallWeeks <- select(XallWeeks, -points.x, -points.y)

# remove all kick data
XallWeeks <- select(XallWeeks, -kickReturnYards.x, -kickReturnTDs.x,
                    -kickReturns.x, -kickReturnYards.y,
                    -kickReturnTDs.y, -kickReturns.y,)

# remove all punt data
XallWeeks <- select(XallWeeks, -puntReturnYards.x, -puntReturnTDs.x,
                    -puntReturns.x, -puntReturnYards.y,
                    -puntReturnTDs.y, -puntReturns.y,)

# remove "fumblesRecovered"
XallWeeks <- select(XallWeeks, -fumblesRecovered.x, -fumblesRecovered.y)

# remove "penalties" due to correlation w/ "penYards"
XallWeeks <- select(XallWeeks, -penalties.x, -penalties.y)

# duplicate steps without rolling avg lag for preDash data frame

# group by "school.x"
preDash <- preDash %>% group_by(school.x)

# arrange data by sequential/ascending "start_date" column values
preDash <- arrange(preDash, preDash$start_date)

# create .x rolling averages for home team
preDash <- preDash %>%
  mutate(points.x = rollmean(points.x,
                             k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(rushingTDs.x = rollmean(rushingTDs.x,
                             k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(puntReturnYards.x = rollmean(puntReturnYards.x,
                             k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(puntReturnTDs.x = rollmean(puntReturnTDs.x,
                             k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(puntReturns.x = rollmean(puntReturns.x,
                             k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(passingTDs.x = rollmean(passingTDs.x,
                             k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(kickReturnYards.x = rollmean(kickReturnYards.x,
                             k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(kickReturnTDs.x = rollmean(kickReturnTDs.x,
                             k = i, fill = NA, align = "right"))
preDash <- preDash %>%
```

```
  mutate(kickReturns.x = rollmean(kickReturns.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(kickingPoints.x = rollmean(kickingPoints.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(interceptionYards.x = rollmean(interceptionYards.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(interceptionTDs.x = rollmean(interceptionTDs.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(fumblesRecovered.x = rollmean(fumblesRecovered.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(totalFumbles.x = rollmean(totalFumbles.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(sacks.x = rollmean(sacks.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(possessionTime.x = rollmean(possessionTime.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(interceptions.x = rollmean(interceptions.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(fumblesLost.x = rollmean(fumblesLost.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(yardsPerRushAttempt.x = rollmean(yardsPerRushAttempt.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(rushingAttempts.x = rollmean(rushingAttempts.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(rushingYards.x = rollmean(rushingYards.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(yardsPerPass.x = rollmean(yardsPerPass.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(netPassingYards.x = rollmean(netPassingYards.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(totalYards.x = rollmean(totalYards.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(firstDowns.x = rollmean(firstDowns.x,
                                  k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(tacklesForLoss.x = rollmean(tacklesForLoss.x,
                                  k = i, fill = NA, align = "right"))
```

```
preDash <- preDash %>%
  mutate(defensiveTDs.x = rollmean(defensiveTDs.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(tackles.x = rollmean(tackles.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(qbHurries.x = rollmean(qbHurries.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(passesDeflected.x = rollmean(passesDeflected.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(completions.x = rollmean(completions.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(passAtts.x = rollmean(passAtts.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(thirdDownConv.x = rollmean(thirdDownConv.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(thirdDownAtt.x = rollmean(thirdDownAtt.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(fourthDownConv.x = rollmean(fourthDownConv.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(fourthDownAtt.x = rollmean(fourthDownAtt.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(penYards.x = rollmean(penYards.x,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(penalties.x = rollmean(penalties.x,
                                   k = i, fill = NA, align = "right"))

# ungroup by "school.x"
preDash <- preDash %>% ungroup(school.x)

# group by "school.y"
preDash <- preDash %>% group_by(school.y)

# arrange data by sequential/ascending "start_date" column values
preDash <- arrange(preDash, preDash$start_date)

# create .y rolling avgs for away team
preDash <- preDash %>%
  mutate(points.y = rollmean(points.y,
                                   k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(rushingTDs.y = rollmean(rushingTDs.y,
                                   k = i, fill = NA, align = "right"))
```

```
preDash <- preDash %>%
  mutate(puntReturnYards.y = rollmean(puntReturnYards.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(puntReturnTDs.y = rollmean(puntReturnTDs.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(puntReturns.y = rollmean(puntReturns.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(passingTDs.y = rollmean(passingTDs.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(kickReturnYards.y = rollmean(kickReturnYards.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(kickReturnTDs.y = rollmean(kickReturnTDs.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(kickReturns.y = rollmean(kickReturns.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(kickingPoints.y = rollmean(kickingPoints.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(interceptionYards.y = rollmean(interceptionYards.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(interceptionTDs.y = rollmean(interceptionTDs.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(fumblesRecovered.y = rollmean(fumblesRecovered.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(totalFumbles.y = rollmean(totalFumbles.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(sacks.y = rollmean(sacks.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(possessionTime.y = rollmean(possessionTime.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(interceptions.y = rollmean(interceptions.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(fumblesLost.y = rollmean(fumblesLost.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(yardsPerRushAttempt.y = rollmean(yardsPerRushAttempt.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(rushingAttempts.y = rollmean(rushingAttempts.y,
```

```
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(rushingYards.y = rollmean(rushingYards.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(yardsPerPass.y = rollmean(yardsPerPass.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(netPassingYards.y = rollmean(netPassingYards.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(totalYards.y = rollmean(totalYards.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(firstDowns.y = rollmean(firstDowns.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(tacklesForLoss.y = rollmean(tacklesForLoss.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(defensiveTDs.y = rollmean(defensiveTDs.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(tackles.y = rollmean(tackles.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(qbHurries.y = rollmean(qbHurries.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(passesDeflected.y = rollmean(passesDeflected.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(completions.y = rollmean(completions.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(passAtts.y = rollmean(passAtts.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(thirdDownConv.y = rollmean(thirdDownConv.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(thirdDownAtt.y = rollmean(thirdDownAtt.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(fourthDownConv.y = rollmean(fourthDownConv.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(fourthDownAtt.y = rollmean(fourthDownAtt.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
  mutate(penYards.y = rollmean(penYards.y,
                              k = i, fill = NA, align = "right"))
preDash <- preDash %>%
```

```
  mutate(penalties.y = rollmean(penalties.y,
                                k = i, fill = NA, align = "right"))

# ungroup by "school.y"
preDash <- preDash %>% ungroup(school.y)

# arrange data by sequential/ascending "start_date" column values
preDash <- arrange(preDash, preDash$start_date)

# remove games with no valid rolling averages (first i games for each team)
  # home and away games are considered separately
preDash <- preDash %>% na.omit()

# create "completionPct"
preDash <- preDash %>% mutate(completionPct.x = completions.x/passAtts.x)
preDash <- preDash %>% mutate(completionPct.y = completions.y/passAtts.y)

# create "thirdDownEff"
preDash <- preDash %>% mutate(thirdDownEff.x = thirdDownConv.x/thirdDownAtt.x)
preDash <- preDash %>% mutate(thirdDownEff.y = thirdDownConv.y/thirdDownAtt.y)

# create "fourthDownEff"
preDash <- preDash %>%
  mutate(fourthDownEff.x = fourthDownConv.x/fourthDownAtt.x)
preDash <- preDash %>%
  mutate(fourthDownEff.y = fourthDownConv.y/fourthDownAtt.y)

# create "yardsPerInt"
preDash <- preDash %>%
  mutate(yardsPerInt.x = interceptionYards.x/interceptions.x)
preDash <- preDash %>%
  mutate(yardsPerInt.y = interceptionYards.y/interceptions.y)

# remove invalid values
  # rare stats like "fourthDownConv" often do not occur over a 7 game span
preDash[preDash == Inf] <- 0
preDash[preDash == -Inf] <- 0
preDash <- na.omit(preDash)

# create categorical variables
preDash$conference.x<-as.factor(preDash$conference.x)
preDash$conference.y<-as.factor(preDash$conference.y)
preDash$school.x<-as.factor(preDash$school.x)
preDash$school.y<-as.factor(preDash$school.y)

# remove third & fourth down conversions & attempts in favor of eff
preDash <- select(preDash, -thirdDownAtt.x, -thirdDownConv.x,
                  -fourthDownConv.x, -fourthDownAtt.x,
                  -thirdDownAtt.y, -thirdDownConv.y,
                  -fourthDownAtt.y, -fourthDownConv.y)

# remove "netPassingYards" in favor of "yardsPerPass"
preDash <- select(preDash, -netPassingYards.x, -netPassingYards.y)
```

```r
# remove "rushingYards" in favor of "yardsPerRushAttempt"
preDash <- select(preDash, -rushingYards.x, -rushingYards.y)

# remove "completions" in favor of "completionPct"
preDash <- select(preDash, -completions.x, -completions.y)

# remove "interceptionYards" in favor of "yardsPerInt"
preDash <- select(preDash, -interceptionYards.x, -interceptionYards.y)

# remove "totalYards" in favor of rush/pass yards stats
preDash <- select(preDash, -totalYards.x, -totalYards.y)

# remove "interceptionTDs" in favor of "defensiveTDs"
preDash <- select(preDash, -interceptionTDs.x, -interceptionTDs.y)

# remove "points" in favor of rushing, passing, ..., etc. TDs
preDash <- select(preDash, -points.x, -points.y)

# remove all kick data
preDash <- select(preDash, -kickReturnYards.x, -kickReturnTDs.x,
                  -kickReturns.x, -kickReturnYards.y,
                  -kickReturnTDs.y, -kickReturns.y,)

# remove all punt data
preDash <- select(preDash, -puntReturnYards.x, -puntReturnTDs.x,
                  -puntReturns.x, -puntReturnYards.y,
                  -puntReturnTDs.y, -puntReturns.y,)

# remove "fumblesRecovered"
preDash <- select(preDash, -fumblesRecovered.x, -fumblesRecovered.y)

# remove "penalties" due to correlation w/ "penYards"
preDash <- select(preDash, -penalties.x, -penalties.y)

# our data frames:
  # XallWeeks1: raw game data
  # preDash: rolling avg data
  # XallWeeks: 1 game lagged rolling avg data
```

targetVariableAnalysis

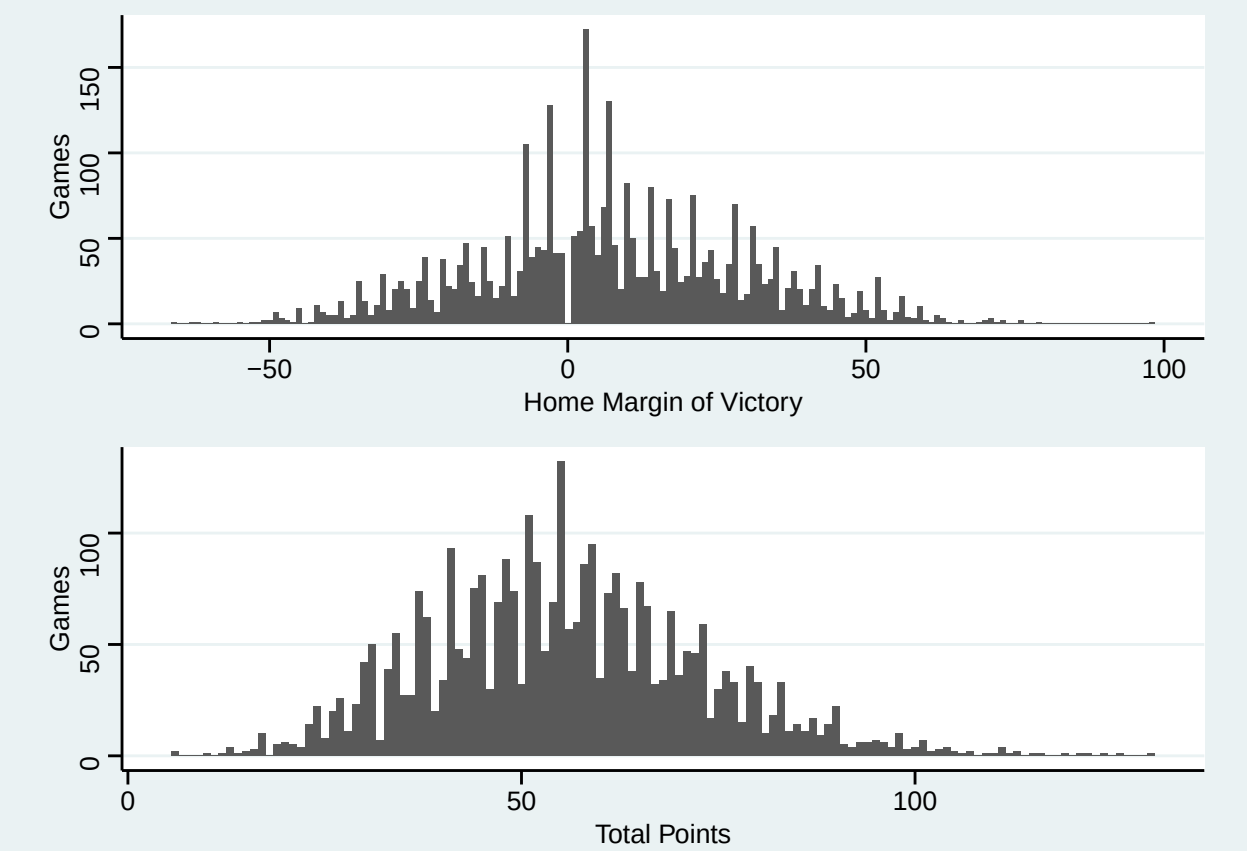
```
# find summary stats for the target variables "HMOV" & "TOTPTS"
summary(XallWeeks1$HMOV)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -66.000  -7.000   6.000   7.125  22.000  98.000
```

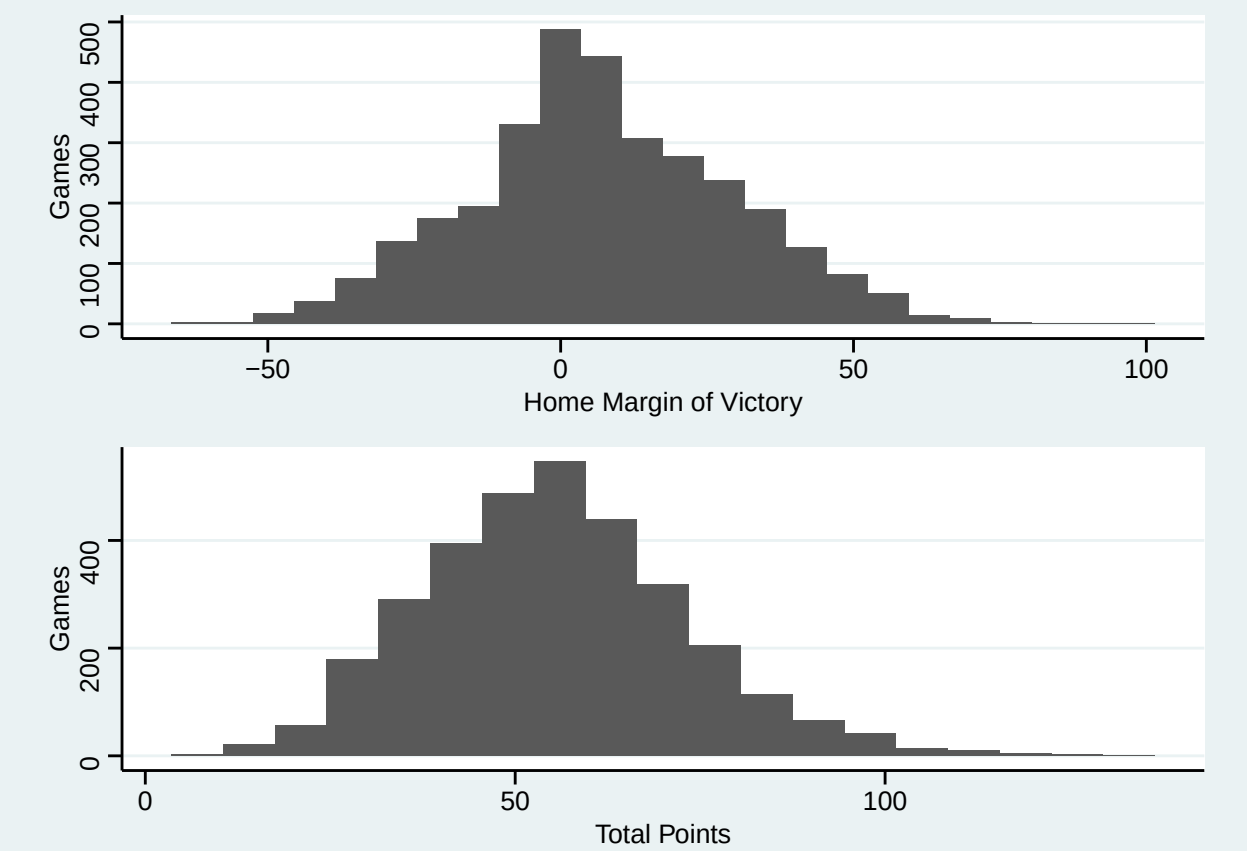
```
summary(XallWeeks1$TOTPTS)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6.00   44.00   55.00   55.64   66.00  130.00
```

```
# create histograms of target variables
PhistHMOV1 <- ggplot(
  data = XallWeeks1, mapping = aes(HMOV)) +
  geom_histogram(binwidth = 1) +
  theme_stata() +
  scale_colour_stata() +
  labs(x = "Home Margin of Victory", y ="Games")

PhistTOTPTS1 <- ggplot(
  data = XallWeeks1, mapping = aes(TOTPTS)) +
  geom_histogram(binwidth = 1) +
  theme_stata() +
  scale_colour_stata() +
  labs(x = "Total Points", y ="Games")

PhistHMOV7 <- ggplot(
  data = XallWeeks1, mapping = aes(HMOV)) +
  geom_histogram(binwidth = 7) +
  theme_stata() +
  scale_colour_stata() +
  labs(x = "Home Margin of Victory", y ="Games")

PhistTOTPTS7 <- ggplot(
  data = XallWeeks1, mapping = aes(TOTPTS)) +
  geom_histogram(binwidth = 7) +
  theme_stata() +
  scale_colour_stata() +
  labs(x = "Total Points", y ="Games")

# plot target variable histograms
  # we created 2 versions of each w/ bin/bar widths 1pt & 7pts, 4 graphs total
grid.arrange(PhistHMOV1, PhistTOTPTS1, nrow = 2)
```




```
grid.arrange(PhistHMOV7, PhistTOTPTS7, nrow = 2)
```



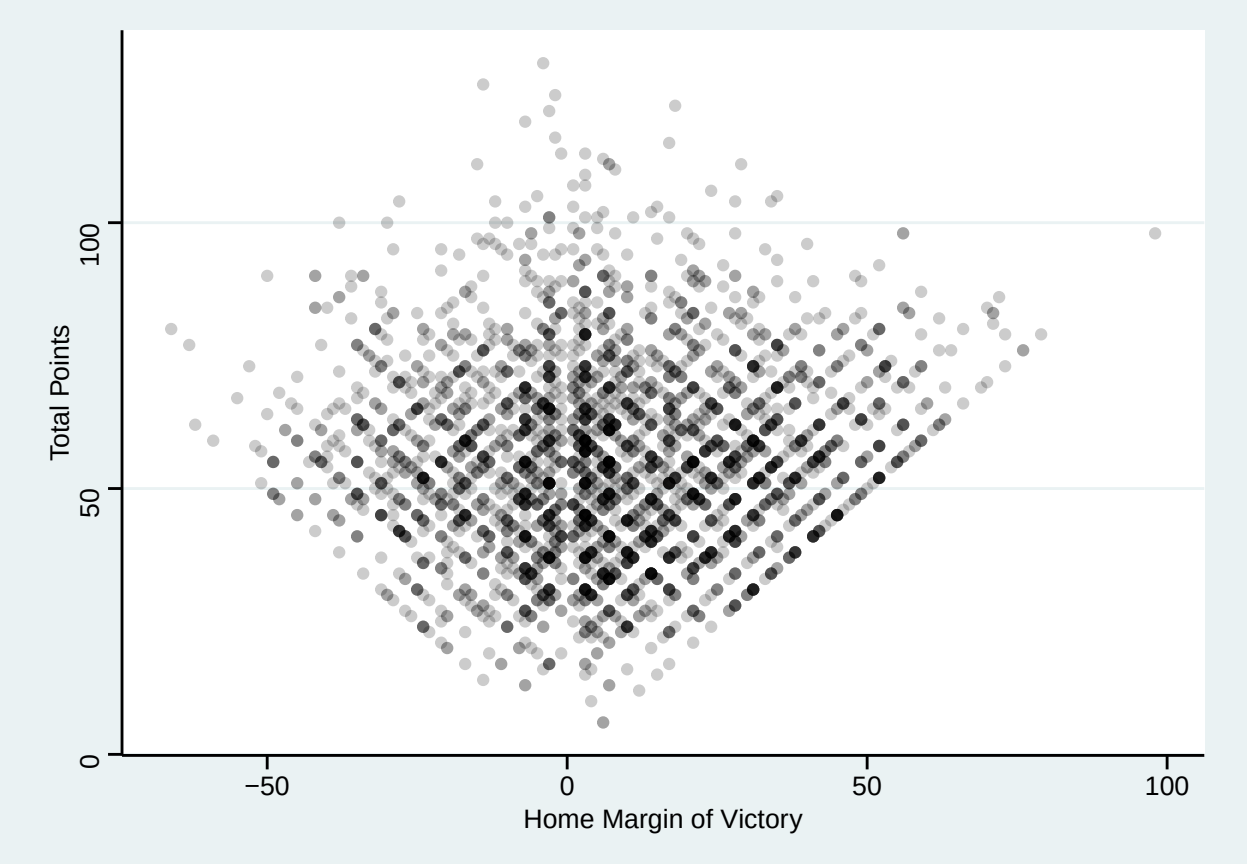

```
# find skew of target variables
  # this checks the similarity of our data to a normal distribution
  # + skew -> excessive right tail -> generally, (mode < median < mean)
  # - skew -> excessive left tail -> generally, (mean < median < mode)
    # if |skew| < 0.5, the data is GOOD for a normal dist assumption
    # if |skew| < 0.1, the data is GREAT for a normal dist assumption
skewness(XallWeeks1$HMOV)
```

```
## [1] 0.08606689
```

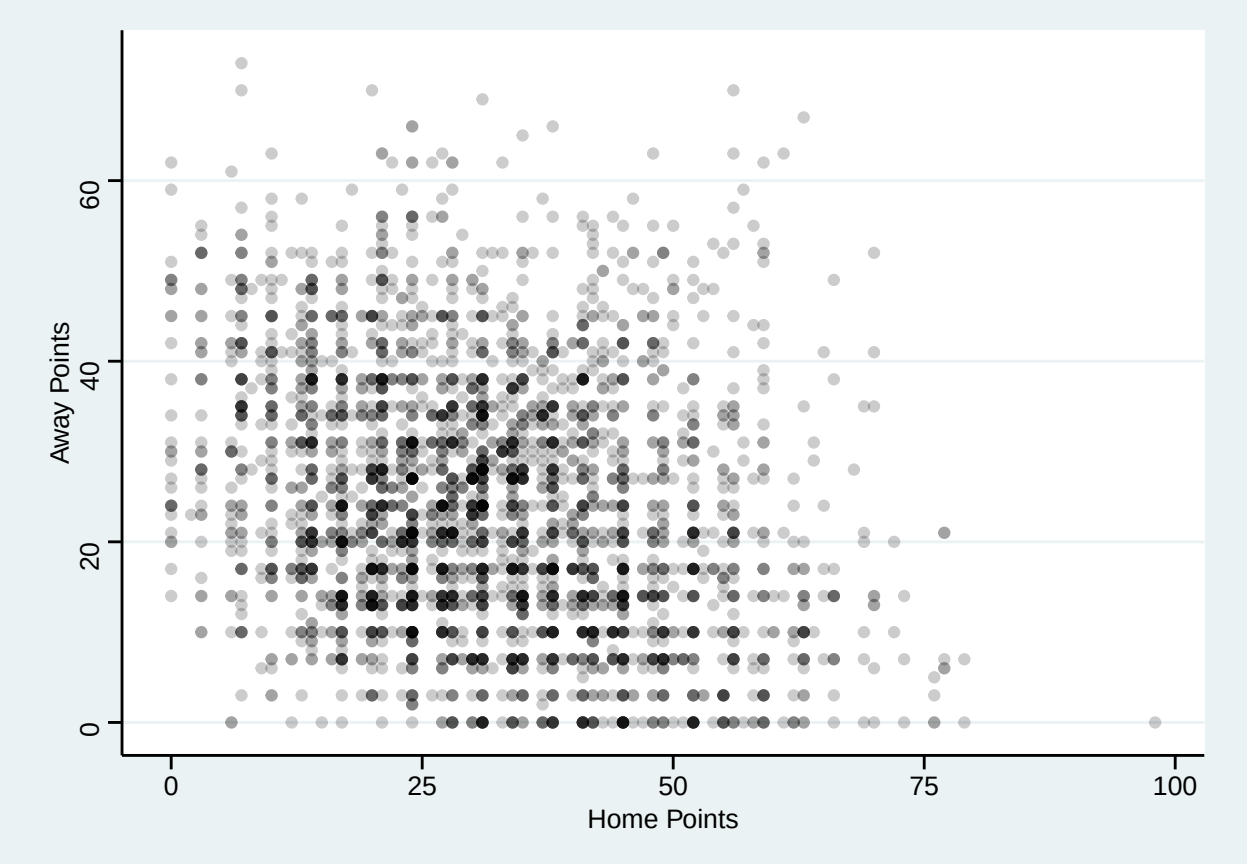
```
skewness(XallWeeks1$TOTPTS)
```

```
## [1] 0.3939959
```

```
# create scatter plot of target variables
PscatterTgtVars <- ggplot(
  data = XallWeeks1,
  mapping = aes(x = HMOV, y = TOTPTS)) +
  geom_point(alpha = .2) +
  theme_stata() +
  scale_colour_stata() +
  labs(x = "Home Margin of Victory", y ="Total Points")

# plot scatter plot of target variables
PscatterTgtVars
```

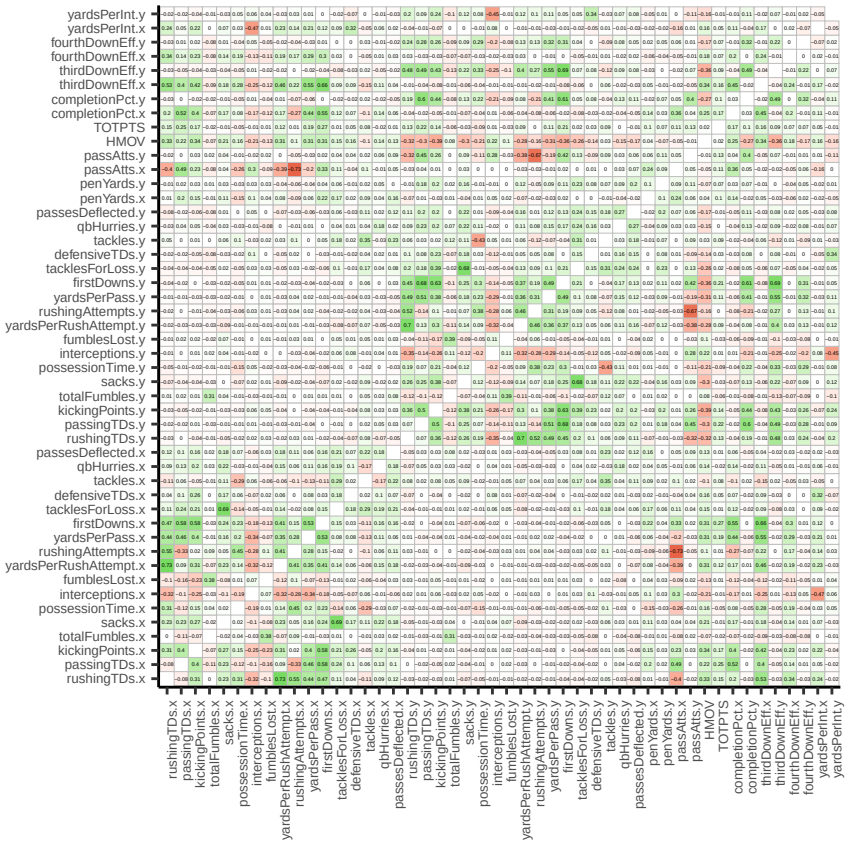



```
# create Scorigami chart (home points vs. away points)
  # https://nflscorigami.com/
PscatterScorigami <- ggplot(
  data = XallWeeks1,
  mapping = aes(x = points.x, y = points.y)) +
  geom_point(alpha = .2) +
  theme_stata() +
  scale_colour_stata() +
  labs(x = "Home Points", y ="Away Points")

# plot Scorigami chart
PscatterScorigami
```


Away Points
0
20
40
60
0
25
50
75
100
Home Points

correlationMatrix

```
# create correlation matrix for preDash rolling avg variables
cor <-  round(cor(preDash[,sapply(preDash,is.numeric)],
                  use="complete.obs", method="pearson"), 2)

# create correlation matrix plot for preDash variables
corPlot <- ggcorrplot(
  cor,
  show.diag = FALSE,
  type = c("full"),
  show.legend = FALSE,
  ggtheme = ggplot2::theme_classic,
  tl.cex = 5, tl.srt = 90,
  colors = c("red3", "white", "green3"), lab = TRUE,
  lab_size = .85)

# plot correlation matrix
corPlot
```




```
# notable correlations include:
  # negative:
    # intrateam "passAtts" vs. "rushingAttempts"
    # intrateam "possessionTime" vs. "tackles"
```

```
  # positive:
    # intrateam "yardsPerRushAttempt" & "rushingTDs"
    # intrateam "tacklesForLoss" & "sacks"
# "noisy" variables: "firstDowns", effs, TDs
```

splitTrainingTest

```
# arrange data by sequential/ascending "start_date" column values
XallWeeks <- arrange(XallWeeks, XallWeeks$start_date)

# create training & test data frames
  # some observations may be lost when slicing
XallWeeksTrain <- XallWeeks %>% slice_head(prop = 0.70)
XallWeeksTest  <- XallWeeks %>% slice_tail(prop = 0.30)

# split training & test data into separate data frames for "HMOV" & "TOTPTS"

# remove columns for predicting 'HMOV"
XallWeeksTrainHMOV <- XallWeeks
XallWeeksTrainHMOV <-  select(XallWeeksTrain, -start_date, -TOTPTS)
XallWeeksTestHMOV <- XallWeeks
XallWeeksTestHMOV <-  select(XallWeeksTest, -start_date, -TOTPTS)

# remove columns for predicting "TOTPTS"
XallWeeksTrainTOTPTS <- XallWeeks
XallWeeksTrainTOTPTS <-  select(XallWeeksTrain, -start_date, -HMOV)
XallWeeksTestTOTPTS <- XallWeeks
XallWeeksTestTOTPTS <-  select(XallWeeksTest, -start_date, -HMOV)

# remove helper data frames that w/ no specified target variable
  # we do not want one target variable in our regression model for the other
remove(XallWeeksTest)
remove(XallWeeksTrain)

# our date info is baked into the model w/ rolling avgs -> remove dates
```

randomForests

```
# https://bradleyboehmke.github.io/HOML/random-forest.html
# ranger automatically generates 1 ensemble decision tree for regression

# create # of features (for estimating # of decision trees to create)
nFeaturesHMOV <- length(setdiff(names(XallWeeksTrainHMOV), "HMOV"))
nFeaturesTOTPTS <- length(setdiff(names(XallWeeksTrainTOTPTS), "TOTPTS"))

# we can optimize our regression w/ hyperparameters

# create hyperparameter grid for optimization of "HMOV" regression
gridHMOV <- expand.grid(
  mtry = floor(c(40, 42, 44)),
  min.node.size = c(5, 6, 7),
  replace = c(TRUE, FALSE),
  sample.fraction = c(0.61, 0.64, 0.67),
  rmse = NA)

# create hyperparameter grid for optimization of "TOTPTS" regression
gridTOTPTS <- expand.grid(
  mtry = floor(c(33, 35, 37)),
  min.node.size = c(5),
  replace = c(TRUE, FALSE),
  sample.fraction = c(0.52, 0.55, 0.58),
  rmse = NA)

# execute Cartesian grid search for optimal "HMOV" regressions
for(i in seq_len(nrow(gridHMOV))) {
  # fit model for ith hyperparameter combination
  XallWeeksHMOV_Rf <- ranger(formula = HMOV ~ .,
    data            = select(XallWeeksTrainHMOV,
                             -game_id, -conference.x, -conference.y),
    num.trees       = nFeaturesHMOV * 10,
    mtry            = gridHMOV$mtry[i],
    min.node.size   = gridHMOV$min.node.size[i],
    replace         = gridHMOV$replace[i],
    sample.fraction = gridHMOV$sample.fraction[i],
    verbose         = FALSE,
    seed            = 123,
    classification = FALSE,
    respect.unordered.factors = 'order')
  # export rmse
  gridHMOV$rmse[i] <- sqrt(XallWeeksHMOV_Rf$prediction.error)
  }

# execute Cartesian grid search for optimal "TOTPTS" regressions
for(i in seq_len(nrow(gridTOTPTS))) {
  # fit model for ith hyperparameter combination
  XallWeeksTOTPTS_Rf <- ranger(formula = TOTPTS ~ .,
    data            = select(XallWeeksTrainTOTPTS,
                             -game_id, -conference.x, -conference.y),
    num.trees       = nFeaturesTOTPTS * 10,
```

```
    mtry            = gridTOTPTS$mtry[i],
    min.node.size   = gridTOTPTS$min.node.size[i],
    replace         = gridTOTPTS$replace[i],
    sample.fraction = gridTOTPTS$sample.fraction[i],
    verbose         = FALSE,
    seed            = 123,
    classification = FALSE,
    respect.unordered.factors = 'order')
  # export rmse
  gridTOTPTS$rmse[i] <- sqrt(XallWeeksTOTPTS_Rf$prediction.error)
  }

# assess top 10 regression models for each target variable
gridHMOV %>% arrange(rmse) %>% head(10)
```

```
##    mtry min.node.size replace sample.fraction     rmse
## 1    42             6    TRUE            0.64 14.79368
## 2    40             6    TRUE            0.64 14.80906
## 3    42             7    TRUE            0.64 14.81701
## 4    42             5    TRUE            0.67 14.82594
## 5    42             5    TRUE            0.64 14.83021
## 6    40             7    TRUE            0.64 14.83038
## 7    42             6    TRUE            0.67 14.83248
## 8    40             5    TRUE            0.64 14.83629
## 9    42             7    TRUE            0.67 14.84265
## 10   42             5    TRUE            0.61 14.84652
```

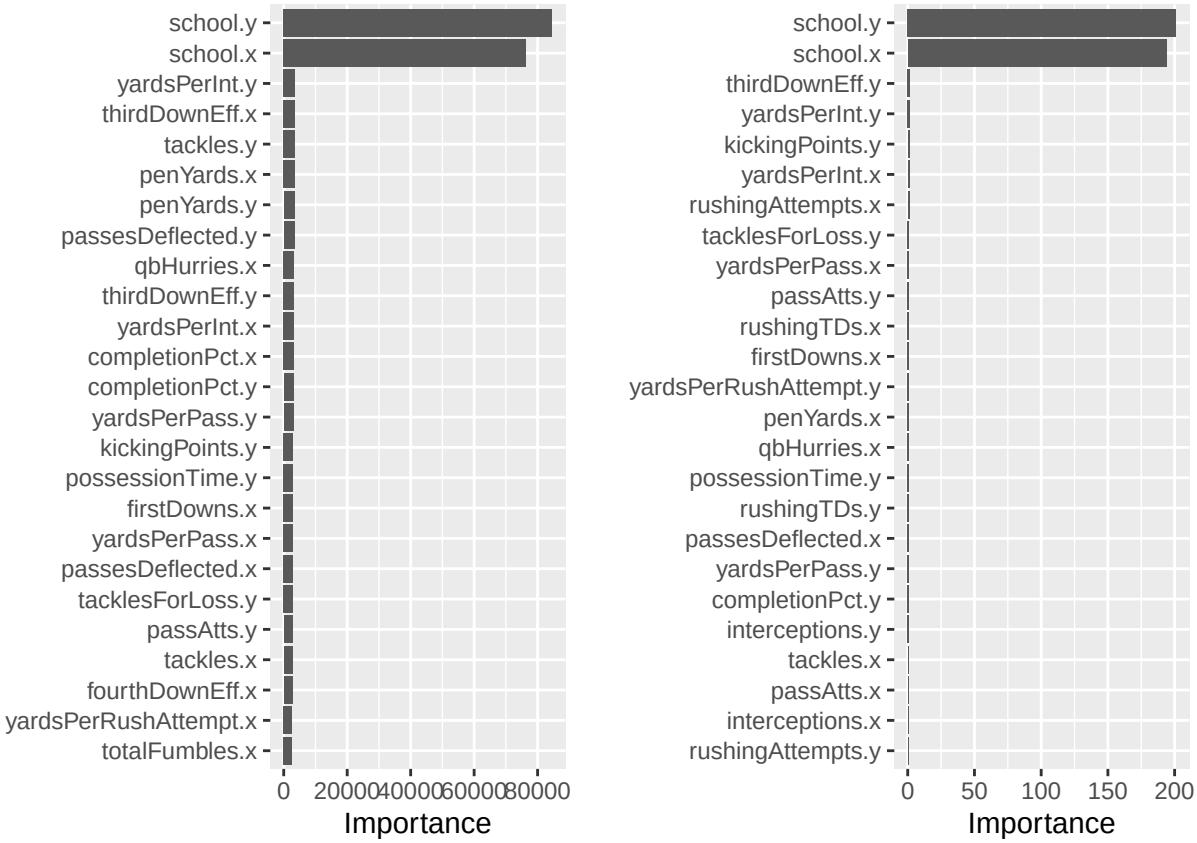
```
gridTOTPTS %>% arrange(rmse) %>% head(10)
```

```
##    mtry min.node.size replace sample.fraction     rmse
## 1    37             5    TRUE            0.55 15.17089
## 2    33             5    TRUE            0.55 15.18927
## 3    35             5    TRUE            0.52 15.18973
## 4    35             5    TRUE            0.58 15.19318
## 5    35             5    TRUE            0.55 15.20328
## 6    33             5    TRUE            0.58 15.21451
## 7    37             5    TRUE            0.52 15.22573
## 8    33             5    TRUE            0.52 15.23412
## 9    37             5    TRUE            0.58 15.25782
## 10   35             5   FALSE            0.52 15.26605
```

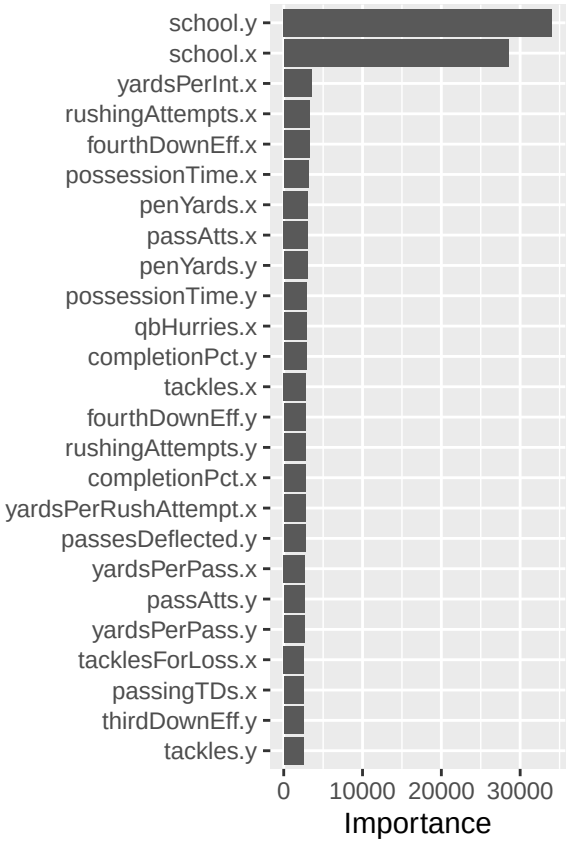
randomForestAnalysis

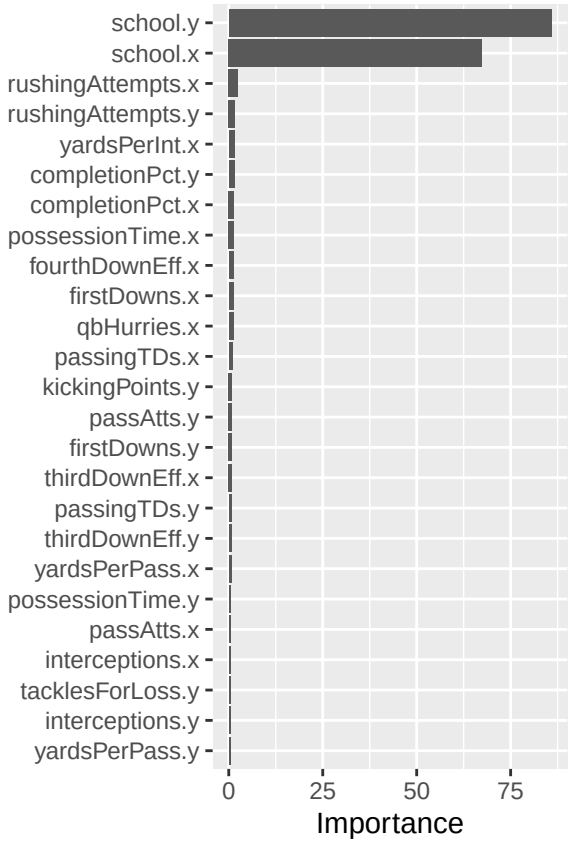
```
# re-run optimized "HMOV" model w/ selected hyperparamter values
XallWeeksHMOV_Rf <- ranger(formula = HMOV ~ .,
  data = select(XallWeeksTrainHMOV,
                              -game_id, -conference.x, -conference.y),
  num.trees = nFeaturesHMOV * 10,
  mtry = 42,
  min.node.size = 6,
  sample.fraction = 0.64,
  replace = TRUE,
  respect.unordered.factors = "order",
  verbose = FALSE,
  seed  = 123,
  classification = FALSE)

# re-run "HMOV" model with impurity-based variable importance
XallWeeksHMOV_RfImpurity <- ranger(formula = HMOV ~ .,
  data = select(XallWeeksTrainHMOV,
                              -game_id, -conference.x, -conference.y),
  num.trees = nFeaturesHMOV * 10,
  mtry = 42, min.node.size = 6,
  sample.fraction = 0.64,
  replace = TRUE,
  importance = "impurity",
  respect.unordered.factors = "order",
  verbose = FALSE,
  seed  = 123,
  classification = FALSE)

# re-run "HMOV" model with permutation-based variable importance
XallWeeksHMOV_RfPermutation <- ranger(
  formula = HMOV ~ .,
  data = select(XallWeeksTrainHMOV,
                              -game_id, -conference.x, -conference.y),
  num.trees = nFeaturesHMOV * 10,
  mtry = 42,
  min.node.size = 6,
  sample.fraction = 0.64,
  replace = TRUE,
  importance = "permutation",
  respect.unordered.factors = "order",
  verbose = FALSE,
  seed  = 123,
  classification = FALSE)

# re-run optimized "TOTPTS" model w/ selected hyperparamter values
XallWeeksTOTPTS_Rf <- ranger(
  formula = TOTPTS ~ .,
  data = select(XallWeeksTrainTOTPTS,
                              -game_id, -conference.x, -conference.y),
  num.trees = nFeaturesTOTPTS * 10,
  mtry = 35,
```

```
  min.node.size = 5,
  sample.fraction = 0.55,
  replace = TRUE,
  respect.unordered.factors = "order",
  verbose = FALSE,
  seed  = 123,
  classification = FALSE)

# re-run "TOTPTS" model with impurity-based variable importance
XallWeeksTOTPTS_RfImpurity <- ranger(
  formula = TOTPTS ~ .,
  data = select(XallWeeksTrainTOTPTS,
                              -game_id, -conference.x, -conference.y),
  num.trees = nFeaturesTOTPTS * 10,
  mtry = 35,
  min.node.size = 5,
  sample.fraction = 0.55,
  replace = TRUE,
  importance = "impurity",
  respect.unordered.factors = "order",
  verbose = FALSE,
  seed  = 123,
  classification = FALSE)

# re-run "TOTPTS" model with permutation-based variable importance
XallWeeksTOTPTS_RfPermutation <- ranger(
  formula = TOTPTS ~ .,
  data = select(XallWeeksTrainTOTPTS,
                              -game_id, -conference.x, -conference.y),
  num.trees = nFeaturesTOTPTS * 10,
  mtry = 35,
  min.node.size = 5,
  sample.fraction = 0.55,
  replace = TRUE,
  importance = "permutation",
  respect.unordered.factors = "order",
  verbose = FALSE,
  seed  = 123,
  classification = FALSE)

# "HMOV" model
p1 <- vip::vip(XallWeeksHMOV_RfImpurity, num_features = 25, bar = FALSE)
p2 <- vip::vip(XallWeeksHMOV_RfPermutation, num_features = 25, bar = FALSE)
gridExtra::grid.arrange(p1, p2, nrow = 1)
```




```
# "TOTPTS" model
p3 <- vip::vip(XallWeeksTOTPTS_RfImpurity, num_features = 25, bar = FALSE)
p4 <- vip::vip(XallWeeksTOTPTS_RfPermutation, num_features = 25, bar = FALSE)
gridExtra::grid.arrange(p3, p4, nrow = 1)
```






```
# get optimized "HMOV" & "TOTPTS" RMSEs
RMSE_HMOV <- sqrt(XallWeeksHMOV_Rf$prediction.error)
RMSE_TOTPTS <- sqrt(XallWeeksTOTPTS_Rf$prediction.error)
RMSE_HMOV
```

```
## [1] 14.79368
```

```
RMSE_TOTPTS
```

```
## [1] 15.20328
```

fittedValues

```
# extract fitted values from regression
predHMOV = predict(XallWeeksHMOV_Rf, data = XallWeeksTestHMOV)
predTOTPTS = predict(XallWeeksTOTPTS_Rf, data = XallWeeksTestTOTPTS)

# add fitted values from regression to test data frames
XallWeeksTestHMOV <- XallWeeksTestHMOV %>%
  mutate(fittedHMOV = predHMOV$predictions)
XallWeeksTestTOTPTS <- XallWeeksTestTOTPTS %>%
  mutate(fittedTOTPTS = predTOTPTS$predictions)
```

betting

```
# load yearly betting lines/odds .csv files
X2019lines <- read_csv("2019lines.csv")
X2020lines <- read_csv("2020lines.csv")
X2021lines <- read_csv("2021lines.csv")
X2022lines <- read_csv("2022lines.csv")

# append yearly betting lines/odds data frames
xLines <- rbind(X2019lines, X2020lines, X2021lines, X2022lines)

# remove yearly betting lines/odds data frames
remove(X2019lines, X2020lines, X2021lines, X2022lines)

# rename "id" to "game_id"
  # column names & formats need to match for merging of data frames
xLines <- rename(xLines, game_id = id)

# select "game_id", "spread", & "overUnder" columns
xLines <- select(xLines, game_id, overUnder, spread)

# find betting lines/odds averages among bookmakers
  # comparing betting lines/odds among various bookmakers is crucial!
  # better betting lines/odds = free EV!
xLines <- group_by(xLines, game_id) %>%
  mutate(overUnder = mean(overUnder))
xLines <- group_by(xLines, game_id) %>%
  mutate(spread = mean(spread))

# format "game_id" as character
xLines <-  xLines %>%
  mutate_at(vars(game_id), as.character)

# ungroup by "game_id"
xLines <- xLines %>%
  ungroup(game_id)

# filter by rows w/ distinct "game_id"
xLines <- distinct(xLines, game_id, .keep_all = TRUE)

# merge test data frame with betting lines/odds data frame
XallWeeksTestHMOV <- left_join(XallWeeksTestHMOV,
                               select(xLines, -overUnder),
                               by = "game_id")
XallWeeksTestTOTPTS <- left_join(XallWeeksTestTOTPTS,
                                 select(xLines, -spread),
                                 by = "game_id")

# find "estWinPct"
XallWeeksTestHMOV <- XallWeeksTestHMOV %>%
  mutate(estWinPct = pnorm(-XallWeeksTestHMOV$spread,
                           mean = XallWeeksTestHMOV$fittedHMOV,
                           sd = RMSE_HMOV,
```

```
                            lower.tail = FALSE))

# find "estOverPct"
XallWeeksTestTOTPTS <- XallWeeksTestTOTPTS %>%
  mutate(estOverPct = pnorm(XallWeeksTestTOTPTS$overUnder,
                            mean = XallWeeksTestTOTPTS$fittedTOTPTS,
                            sd = RMSE_TOTPTS,
                            lower.tail = FALSE))

# create global betting lines/odds place holder
bookOdds <- 1.909

# EV = expected value (% of $ risked) = (probability * bookOdds) - 1
  # ex. 75% chance of success, 1.5 decimal odds (67%)
    # EV = (0.75(1.5) - 1) = 0.125 = 12.5%

# find "spreadEV"
XallWeeksTestHMOV <- XallWeeksTestHMOV %>%
  mutate(spreadEV = (bookOdds*XallWeeksTestHMOV$estWinPct) - 1)

# find "overUnderEV"
XallWeeksTestTOTPTS <- XallWeeksTestTOTPTS %>%
  mutate(overUnderEV = (bookOdds*XallWeeksTestTOTPTS$estOverPct) - 1)

# set "evLimit" for bet decisions for 'spread" & "overUnder"
  # this is the threshold EV for which we will place bets
  # in this model, we will set the threshold as the std dev of calculated EVs
sd(XallWeeksTestHMOV$spreadEV)
```

```
## [1] 0.4809482
```

```
sd(XallWeeksTestTOTPTS$overUnderEV)
```

```
## [1] 0.3572086
```

```
evLimitHMOV <- sd(XallWeeksTestHMOV$estWinPct)
evLimitTOTPTS <- sd(XallWeeksTestTOTPTS$estOverPct)

# add binary "bet"
  # 1 = bet on home team
  # -1 = bet on away team
XallWeeksTestHMOV <- XallWeeksTestHMOV %>%
  mutate(bet = ifelse(spreadEV > evLimitHMOV, 1,
                      ifelse(spreadEV < -evLimitHMOV, -1, 0)))
XallWeeksTestTOTPTS <- XallWeeksTestTOTPTS %>%
  mutate(bet = ifelse(overUnderEV > evLimitTOTPTS, 1,
                      ifelse(overUnderEV < -evLimitTOTPTS, -1, 0)))

# format "bet" as numerical
XallWeeksTestHMOV <- XallWeeksTestHMOV %>%
  mutate_at(vars(bet), as.numeric)
XallWeeksTestTOTPTS <- XallWeeksTestTOTPTS %>%
```

```
  mutate_at(vars(bet), as.numeric)

# add binary "result"
  # points ATS = points at the spread (sum of points & spread)
    # 1 = home team win/away team loss ATS
    # -1 = home team loss/away team win ATS
    # 0 = push/tie ($ risked is returned)
      # EV = 0% in a push/tie
XallWeeksTestHMOV <- XallWeeksTestHMOV %>%
  mutate(result = ifelse(HMOV + spread > 0, 1,
                         ifelse(HMOV + spread  < 0, -1, 0)))
XallWeeksTestTOTPTS <- XallWeeksTestTOTPTS %>%
  mutate(result = ifelse(TOTPTS > overUnder, 1,
                         ifelse(TOTPTS < overUnder, -1, 0)))

# format "result" as numerical
XallWeeksTestHMOV <- XallWeeksTestHMOV %>%
  mutate_at(vars(result), as.numeric)
XallWeeksTestTOTPTS <- XallWeeksTestTOTPTS %>%
  mutate_at(vars(result), as.numeric)

# create results data frames of games where bets would have been placed
resultsHMOV <- XallWeeksTestHMOV %>%
  filter(bet == -1 | bet == 1)
resultsTOTPTS <- XallWeeksTestTOTPTS %>%
  filter(bet == -1 | bet == 1)

# select relevant columns for new results data frame
resultsHMOV <- select(resultsHMOV, game_id, school.x, school.y,
                      HMOV, fittedHMOV, spread, estWinPct,
                      spreadEV, bet, result)
resultsTOTPTS <- select(resultsTOTPTS, game_id, school.x, school.y,
                      TOTPTS, fittedTOTPTS, overUnder, estOverPct,
                      overUnderEV, bet, result)

# create "betOutcome"
  # 1 = winning bet
  # -1 = losing bet
  # 0 = push/tie
resultsHMOV <- resultsHMOV %>%
  mutate(betOutcome = ifelse(result == 0, 0,
                      ifelse(bet == result, 1, -1)))
resultsTOTPTS <- resultsTOTPTS %>%
  mutate(betOutcome = ifelse(result == 0, 0,
                      ifelse(bet == result, 1, -1)))

# find accuracy
accuracyHMOV <- sum(resultsHMOV$betOutcome == 1) /
  (sum(resultsHMOV$betOutcome == 1) + sum(resultsHMOV$betOutcome == -1))
accuracyTOTPTS <- sum(resultsTOTPTS$betOutcome == 1) /
  (sum(resultsTOTPTS$betOutcome == 1) + sum(resultsTOTPTS$betOutcome == -1))
```

dashboard

```
# run from here to repeatedly analyze different scenarios

# arrange data by sequential/ascending "start_date" column values
preDash <- arrange(preDash, preDash$start_date)

# create preDash data frames

# remove columns for predicting 'HMOV"
preDashHMOV <- preDash
preDashHMOV <- select(preDash, -start_date, -TOTPTS)

# remove columns for predicting "TOTPTS"
preDashTOTPTS <- preDash
preDashTOTPTS <- select(preDash, -start_date, -HMOV)

# extract fitted values from regression
predDashHMOV = predict(XallWeeksHMOV_Rf, data = preDashHMOV)
predDashTOTPTS = predict(XallWeeksTOTPTS_Rf, data = preDashTOTPTS)

# add fitted values from regression to preDash data frames
preDashHMOV <- preDashHMOV %>%
  mutate(fittedPreDashHMOV = predDashHMOV$predictions)
preDashTOTPTS <- preDashTOTPTS %>%
  mutate(fittedPreDashTOTPTS = predDashTOTPTS$predictions)

# merge preDash data frames with betting lines/odds data frame
preDashHMOV <- left_join(preDashHMOV,
                         select(xLines, -overUnder),
                         by = "game_id")
preDashTOTPTS <- left_join(preDashTOTPTS,
                           select(xLines, -spread),
                           by = "game_id")

# find "preDashEstWinPct"
preDashHMOV <- preDashHMOV %>%
  mutate(preDashEstWinPct = pnorm(-preDashHMOV$spread,
                                  mean = preDashHMOV$fittedPreDashHMOV,
                                  sd = RMSE_HMOV,
                                  lower.tail = FALSE))

# find "preDashEstOverPct"
preDashTOTPTS <- preDashTOTPTS %>%
  mutate(preDashEstOverPct = pnorm(preDashTOTPTS$overUnder,
                                   mean = preDashTOTPTS$fittedPreDashTOTPTS,
                                   sd = RMSE_TOTPTS,
                                   lower.tail = FALSE))

# find "spreadEV"
preDashHMOV <- preDashHMOV %>%
  mutate(spreadEV = (bookOdds*preDashHMOV$preDashEstWinPct) - 1)
```

```
# find "overUnderEV"
preDashTOTPTS <- preDashTOTPTS %>%
  mutate(overUnderEV = (bookOdds*preDashTOTPTS$preDashEstOverPct) - 1)

#### INPUT/ENTRY ####

home = "Arkansas"
away = "Auburn"
spread = "3.5"
spreadOdds = "1.87"
overUnder = "61.5"
overUnderOdds = "1.952"

# split preDash for home & away "HMOV"
homePreDashHMOV <- preDashHMOV
homePreDashHMOV <- homePreDashHMOV %>%
  select(-ends_with(".y"), -fittedPreDashHMOV,
         -spread, -preDashEstWinPct, -spreadEV)
awayPreDashHMOV <- preDashHMOV
awayPreDashHMOV <- awayPreDashHMOV %>%
  select(-ends_with(".x"), -fittedPreDashHMOV,
         -spread, -preDashEstWinPct, -spreadEV)

# select most recent entries by team
homePreDashHMOV <- homePreDashHMOV %>%
  group_by(school.x) %>%
  top_n(1, game_id)
awayPreDashHMOV <- awayPreDashHMOV %>%
  group_by(school.y) %>%
  top_n(1, game_id)

# create preDashRowHMOV
  # this "row" contains our entries
homePreDashHMOV_row <- filter(homePreDashHMOV, school.x == home)
awayPreDashHMOV_row <- filter(awayPreDashHMOV, school.y == away)

# rename "game_id" to "entry"

homePreDashHMOV_row <- homePreDashHMOV_row %>%
  mutate(game_id = "entry")
awayPreDashHMOV_row <- awayPreDashHMOV_row %>%
  mutate(game_id = "entry")

# merge
preDashHMOV <- left_join(homePreDashHMOV_row,
                         awayPreDashHMOV_row,
                         by = "game_id")

# remove extra "HMOV" & rename other
preDashHMOV <- select(preDashHMOV, -HMOV.y)
preDashHMOV <- rename(preDashHMOV, HMOV = HMOV.x)

# split preDash for home & away "TOTPTS"
```

```
homePreDashTOTPTS <- preDashTOTPTS
homePreDashTOTPTS <- homePreDashTOTPTS %>%
  select(-ends_with(".y"), -fittedPreDashTOTPTS,
         -overUnder, -preDashEstOverPct, -overUnderEV)
awayPreDashTOTPTS <- preDashTOTPTS
awayPreDashTOTPTS <- awayPreDashTOTPTS %>%
  select(-ends_with(".x"), -fittedPreDashTOTPTS,
         -overUnder, -preDashEstOverPct, -overUnderEV)

# select most recent entries by team
homePreDashTOTPTS <- homePreDashTOTPTS %>%
  group_by(school.x) %>%
  top_n(1, game_id)
awayPreDashTOTPTS <- awayPreDashTOTPTS %>%
  group_by(school.y) %>%
  top_n(1, game_id)

# create preDashRowTOTPTS
  # this "row" contains our entries
homePreDashTOTPTS_row <- filter(homePreDashTOTPTS, school.x == home)
awayPreDashTOTPTS_row <- filter(awayPreDashTOTPTS, school.y == away)

# rename "game_id" to "entry"
homePreDashTOTPTS_row <- homePreDashTOTPTS_row %>%
  mutate(game_id = "entry")
awayPreDashTOTPTS_row <- awayPreDashTOTPTS_row %>%
  mutate(game_id = "entry")

# merge
preDashTOTPTS <- left_join(homePreDashTOTPTS_row, awayPreDashTOTPTS_row,
                           by = "game_id")

# remove extra "TOTPTS" & rename other
preDashTOTPTS <- select(preDashTOTPTS, -TOTPTS.y)
preDashTOTPTS <- rename(preDashTOTPTS, TOTPTS = TOTPTS.x)

# apply regression to both preDashRows
finalHMOV <- predict(XallWeeksHMOV_Rf, data = preDashHMOV)
finalTOTPTS <- predict(XallWeeksTOTPTS_Rf, data = preDashTOTPTS)

# add (est)spread(odds)
preDashHMOV <- preDashHMOV %>%
  mutate(spread = spread)
preDashHMOV <- preDashHMOV %>%
  mutate(estSpread = finalHMOV$predictions)
preDashHMOV <- preDashHMOV %>%
  mutate(spreadOdds = spreadOdds)

# add (est)over/under(odds)
preDashTOTPTS <- preDashTOTPTS %>%
  mutate(overUnder = overUnder)
preDashTOTPTS <- preDashTOTPTS %>%
  mutate(estOverUnder = finalTOTPTS$predictions)
```

```
preDashTOTPTS <- preDashTOTPTS %>%
  mutate(overUnderOdds = overUnderOdds)

# create dashboards for "HMOV" & "TOTPTS"
dashHMOV <- preDashHMOV
dashTOTPTS <- preDashTOTPTS

# format entries as numerical
dashHMOV <- dashHMOV %>%
  mutate_at(vars(spread), as.numeric)
dashHMOV <- dashHMOV %>%
  mutate_at(vars(spreadOdds), as.numeric)
dashTOTPTS <- dashTOTPTS %>%
  mutate_at(vars(overUnder), as.numeric)
dashTOTPTS <- dashTOTPTS %>%
  mutate_at(vars(overUnderOdds), as.numeric)

# find dashboard "estWinPct"
dashHMOV <- dashHMOV %>%
  mutate(estWinPct = pnorm(-dashHMOV$spread,
                           mean = dashHMOV$estSpread,
                           sd = RMSE_HMOV,
                           lower.tail = FALSE))

# find dashboard "estOverPct"
dashTOTPTS <- dashTOTPTS %>%
  mutate(estOverPct = pnorm(dashTOTPTS$overUnder,
                            mean = dashTOTPTS$estOverUnder,
                            sd = RMSE_TOTPTS,
                            lower.tail = FALSE))

# find dashboard "spreadEV"
dashHMOV <- dashHMOV %>%
  mutate(spreadEV = (spreadOdds*dashHMOV$estWinPct) - 1)

# find dashboard "overUnderEV"
dashTOTPTS <- dashTOTPTS %>%
  mutate(overUnderEV = (overUnderOdds*dashTOTPTS$estOverPct) - 1)

# find final dashboard results

  # for "spread" / "HMOV"
dashHMOV$estSpread
```

```
## [1] 3.824739
```

```
dashHMOV$spreadEV
```

```
## [1] 0.2898222
```

```
  # for "overUnder" / "TOTPTS"
dashTOTPTS$estOverUnder
```

```
## [1] 47.56575
```

```
dashTOTPTS$overUnderEV
```

```
## [1] -0.6492359
```