MinneMUDAC Final

Brandon Winder

3/16/2021

### Collect stats for each game played of the top 150 teams ranked by win percent.

# Create function to find the schedule and stats for a team.  
### Inputs: team as it would appear in the URL of the website below  
### Output: A data frame with each game and stats for the team entered.  
  
games\_collect <- function(team){  
 Sports\_ref <- paste("https://www.sports-reference.com/cbb/schools/", team, "/2021-gamelogs.html", sep = "")  
 html <- read\_html(as.character(Sports\_ref))  
  
 dat <- data.frame(Team = team,  
 Opp = html\_text(html\_nodes(html, "td:nth-child(4)")),  
 W\_L = html\_text(html\_nodes(html, "td:nth-child(5)")),  
 TM = html\_text(html\_nodes(html, "td:nth-child(6)")),  
 Opp\_Points = html\_text(html\_nodes(html, "td:nth-child(7)")),  
 FG = html\_text(html\_nodes(html, "td:nth-child(8)")),  
 FGA =html\_text(html\_nodes(html, "td:nth-child(9)")),  
 FG\_perc = html\_text(html\_nodes(html, "td:nth-child(10)")),  
 P3 = html\_text(html\_nodes(html, "td:nth-child(11)")),  
 PA3 = html\_text(html\_nodes(html, "td:nth-child(12)")),  
 perc\_3 = html\_text(html\_nodes(html, "td:nth-child(13)")),  
 FT = html\_text(html\_nodes(html, "td:nth-child(14)")),  
 FTA = html\_text(html\_nodes(html, "td:nth-child(15)")),  
 FT\_perc = html\_text(html\_nodes(html, "td:nth-child(16)")),  
 ORB = html\_text(html\_nodes(html, "td:nth-child(17)")),  
 TRB = html\_text(html\_nodes(html, "td:nth-child(18)")),  
 AST = html\_text(html\_nodes(html, "td:nth-child(19)")),  
 STL = html\_text(html\_nodes(html, "td:nth-child(20)")),  
 BLK = html\_text(html\_nodes(html, "td:nth-child(21)")),  
 TOV = html\_text(html\_nodes(html, "td:nth-child(22)")),  
 PF = html\_text(html\_nodes(html, "td:nth-child(23)")))  
   
 return(dat)  
}

# Collect the team names and store in data frame.  
teams2 <- data.frame()  
  
url <- paste("https://www.sports-reference.com/cbb/seasons/2021-school-stats.html")  
html <- read\_html(url)  
x <- html\_nodes(html, "td:nth-child(2)")  
wins <- html\_nodes(html, "td:nth-child(6)")  
teams2 <- data.frame(Team = html\_text(x, trim = TRUE), Win = html\_text(wins, trim = TRUE))  
  
# Order that data frame by win percentage.  
teams2 <- teams2[order(teams2$Win, decreasing = TRUE), ]  
  
# Run code to clean the team names so they appear the same in the URL.  
trial <- teams2  
trial$Team <- str\_replace\_all(trial$Team, "UC-", "California ")  
  
trial$Team <- gsub('[[:punct:] ]+', ' ', trial$Team)  
trial$Team <- str\_replace\_all(trial$Team, "A M", "AM")  
  
trial$Team <- str\_to\_lower(trial$Team)  
which\_ncaa <- which(str\_detect(trial$Team, "ncaa"))  
for(i in 1:length(which\_ncaa)){  
 trial$Team[which\_ncaa[i]] <- substr(trial$Team[which\_ncaa[i]], 1, nchar(trial$Team[which\_ncaa[i]])-5)  
}  
trial$Team <- str\_replace\_all(trial$Team, " ", "-")  
  
extra <- which(grepl("^.+(-)$",trial$Team))  
for(i in 1:length(extra)){  
 name <- trial$Team[extra[i]]  
 trial$Team[extra[i]] <- substr(name,1,nchar(name)-1)  
}  
  
# Fix teams that did not appear in correct form for the URL.  
trial$Team[which(trial$Team == "louisiana")] <- str\_replace\_all(trial$Team[which(trial$Team == "louisiana")], "louisiana", "louisiana-lafayette")  
trial$Team[which(trial$Team == "st-john-s-ny")] <- str\_replace\_all(trial$Team[which(trial$Team == "st-john-s-ny")], "st-john-s-ny", "st-johns-ny")  
trial$Team[which(trial$Team == "saint-mary-s-ca")] <- str\_replace\_all(trial$Team[which(trial$Team == "saint-mary-s-ca")], "saint-mary-s-ca", "saint-marys-ca")  
trial$Team[which(trial$Team == "saint-peter-s")] <- str\_replace\_all(trial$Team[which(trial$Team == "saint-peter-s")], "saint-peter-s", "saint-peters")  
  
# Make a final data frame with the team names as it appears in URL.  
Final\_Teams <- trial

# Collect data for the schedule and stats for each game for the top 150 teams ranked by win%.  
teams\_sched <- data.frame()  
for(i in 1:150){  
 one\_team <- games\_collect(Final\_Teams$Team[i])  
 teams\_sched <- rbind(teams\_sched, one\_team)  
}  
  
# Change variables to numeric.  
for(i in 4:21){  
 teams\_sched[,i] <- as.numeric(teams\_sched[,i])  
}  
  
# Remove rows with NA values.  
teams\_sched <- na.omit(teams\_sched)

### Create logistic regression model for prediction.

# Make win a factor variable.  
teams\_sched$W\_L <- substr(teams\_sched$W\_L, 1, 1)  
teams\_sched$W\_L <- ifelse(teams\_sched$W\_L == "W", 1, 0)  
teams\_sched$W\_L <- as.factor(teams\_sched$W\_L)  
  
# Set up training/validation data.   
set.seed(111)  
train\_index <- sample(row.names(teams\_sched), 0.6\*dim(teams\_sched)[1])  
valid\_index <- setdiff(row.names(teams\_sched), train\_index)   
training\_data <- teams\_sched[train\_index, ]  
valid\_data <- teams\_sched[valid\_index, ]  
  
  
# Run logistic regression on training data. Remove repetitive percent variables.  
logit\_data <- glm(W\_L ~ FG + FGA + P3 + PA3 + FT + FTA + ORB + TRB + AST + STL + BLK + TOV + PF, data = training\_data, family = "binomial")  
summary(logit\_data)

##   
## Call:  
## glm(formula = W\_L ~ FG + FGA + P3 + PA3 + FT + FTA + ORB + TRB +   
## AST + STL + BLK + TOV + PF, family = "binomial", data = training\_data)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.0557 -0.2872 0.1222 0.3904 3.0928   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.54629 0.82417 -0.663 0.507438   
## FG 0.38699 0.02926 13.227 < 2e-16 \*\*\*  
## FGA -0.34858 0.02305 -15.125 < 2e-16 \*\*\*  
## P3 0.21433 0.04046 5.297 1.18e-07 \*\*\*  
## PA3 -0.04758 0.02024 -2.350 0.018755 \*   
## FT 0.20462 0.03573 5.726 1.03e-08 \*\*\*  
## FTA -0.08018 0.02833 -2.830 0.004654 \*\*   
## ORB -0.03991 0.02933 -1.361 0.173574   
## TRB 0.36909 0.02190 16.852 < 2e-16 \*\*\*  
## AST 0.09023 0.02483 3.634 0.000279 \*\*\*  
## STL 0.39209 0.03127 12.539 < 2e-16 \*\*\*  
## BLK 0.11302 0.03558 3.176 0.001492 \*\*   
## TOV -0.31832 0.02549 -12.487 < 2e-16 \*\*\*  
## PF -0.10872 0.01974 -5.507 3.65e-08 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 2849.6 on 2261 degrees of freedom  
## Residual deviance: 1274.1 on 2248 degrees of freedom  
## AIC: 1302.1  
##   
## Number of Fisher Scoring iterations: 6

### Collect data for the average stats over the season.

# Collect the team names and store in data frame.  
teams <- data.frame()  
  
for(i in 1:7){  
 NCAA\_url <- paste("https://www.ncaa.com/stats/basketball-men/d1/current/team/168/p", i, sep = "")  
 html <- read\_html(NCAA\_url)  
 x <- html\_nodes(html, "td:nth-child(2)")  
 page <- data.frame(Team = html\_text(x, trim = TRUE))  
   
 teams <- rbind(teams, page)  
}

# Function for collecting Team data from NCAA.com.  
  
### Input: url\_num for team stats page  
### col1 for team name column of data on page  
### col2 for stat column of data on page  
### Output: Returns a data frame with the team and specific stat  
  
collect <- function(url\_num, col1, col2){  
 dat <- data.frame()  
 for(i in 1:7){  
 NCAA\_url <- paste("https://www.ncaa.com/stats/basketball-men/d1/current/team/", url\_num, "/p", i, sep = "")  
  
 html <- read\_html(NCAA\_url)  
 team <- html\_nodes(html, paste("td:nth-child(", col1, ")", sep = ""))  
 vars <- html\_nodes(html, paste("td:nth-child(", col2, ")", sep = ""))  
   
 page <- data.frame(Team = html\_text(team, trim = TRUE), Stat = html\_text(vars, trim = TRUE))  
   
 dat <- rbind(dat, page)  
 }  
 return(dat)  
}

# Collect Team data to use for predictions.  
GAMES <- collect(147, 2, 3)  
TM <- collect(147, 2, 5)  
Opp\_points <- collect(147, 2, 7)  
FG <- collect(148, 2, 4)  
FGA <- collect(148, 2, 5)  
FG\_perc <- collect(148, 2, 6)  
P3 <- collect(152, 2, 4)  
PA3 <- collect(152, 2, 5)  
Perc\_3 <- collect(152, 2, 6)  
FT <- collect(150, 2, 4)  
FTA <- collect(150, 2, 5)  
FT\_perc <- collect(150, 2, 6)  
ORB <- collect(602, 2, 4)  
TRB <- collect(602, 2, 6)  
AST <- collect(216, 2, 5)  
STL <- collect(215, 2, 5)  
BLK <- collect(214, 2, 5)  
TOV <- collect(217, 2, 5)  
PF <- collect(286, 2, 5)  
  
team\_data <- list(teams, GAMES, TM, Opp\_points, FG, FGA, FG\_perc, P3, PA3, Perc\_3, FT, FTA, FT\_perc, ORB, TRB, AST, STL, BLK, TOV, PF) %>% reduce(left\_join, by = "Team")  
  
colnames(team\_data) <- c("Team", "GAMES", colnames(teams\_sched[,4:21]))  
  
for(i in 2:20){  
 team\_data[,i] <- as.numeric(team\_data[,i])  
}  
  
# Change variables to averages that are in totals for the year.  
to\_avg <- c("FG", "FGA", "P3", "PA3", "FT", "FTA", "ORB", "TRB")  
for(i in 1:length(to\_avg)){  
 team\_data[,to\_avg[i]] <- team\_data[,to\_avg[i]] / team\_data$GAMES  
}

# Find predicted probabilities for each team to win a game.  
pred\_probs <- predict(logit\_data, team\_data, type = "response")  
  
# Store it in a data fame and order it by win probability.  
prediction <- data.frame(Team = teams, Pred\_prob = pred\_probs)  
prediction <- prediction[order(prediction$Pred\_prob, decreasing = TRUE), ]

### Collect data from the Ken Pomeroy website.

# Create function to collect column from ken pomeroy's ranking table.   
### Input a column number.  
### Output the text in that column.  
kens\_data <- function(i){  
 Ken <- paste("https://kenpom.com/")  
 html <- read\_html(Ken)  
 x <- html\_nodes(html, paste("td:nth-child(", i, ")"))  
 page <- html\_text(x, trim = TRUE)  
}  
  
# Collect data.  
kens\_rank <- data.frame(Team = kens\_data(2),  
 Rank = kens\_data(1),  
 AdjEM = kens\_data(5),  
 AdjO = kens\_data(6),  
 AdjD = kens\_data(8))  
  
# Change the variables to numeric, remove +/- signs, and remove seeding.  
kens\_rank$AdjEM <- substr(kens\_rank$AdjEM, 2, nchar(kens\_rank$AdjEM))  
for(i in 2:length(kens\_rank[1,])){  
 kens\_rank[,i] <- as.numeric(kens\_rank[,i])  
}  
kens\_rank$Team <- gsub('[[:digit:]]+', '', kens\_rank$Team)  
kens\_rank$Team <- trimws(kens\_rank$Team)

### Code for seed difference model.

# load in data  
df <- read.csv("Big\_Dance\_CSV.csv")

# mutate in:  
# margin of victory  
# response variable = win/loss  
# new predictor = seed difference  
df <- df %>% mutate(MarginOfVictory = Score - Score.1)  
df <- df %>% mutate(SeedDifference = Seed.1 - Seed)  
df <- df %>% mutate(Win = ifelse(MarginOfVictory >= 1, 1, 0))

# set up algorithm  
model <- glm(Win ~ SeedDifference + Round, data = df, family = binomial)

### Create function to compare two teams.

# Compare teams in bracket,  
### Input: team1 - Lower seed team name as character string  
### team2 - Higher seed team name as character string  
### Output: Name of team predicted to win the game.  
compare\_teams <- function(team1, team2, seeddiff, round){  
 x\_team1 <- which(prediction$Team == team1)  
 x\_team2 <- which(prediction$Team == team2)  
 # For logistic regression.  
 if(prediction$Pred\_prob[x\_team1] > prediction$Pred\_prob[x\_team2]){  
 x <- 1  
 }  
 else{  
 x <- 0   
 }  
 # For kenpom ranking.  
 y\_team1 <- which(kens\_rank$Team == team1)  
 y\_team2 <- which(kens\_rank$Team == team2)  
 if(kens\_rank$Rank[y\_team1] < kens\_rank$Rank[y\_team2]){  
 y <- 1  
 }  
 else{  
 y <- 0  
 }  
   
 # Add in seed difference model.  
 new.df <- data.frame(SeedDifference = seeddiff, Round = round)  
 probs\_seed <- predict(model, newdata = new.df, type = "response")  
 if(probs\_seed > 0.5){  
 z = 1  
 }  
 else{  
 z = 0  
 }  
   
 if(sum(x, y, z) > 1){  
 winner <- team1  
 }  
 else{  
 winner <- team2  
 }  
 return(winner)  
}

# Modify the names in the data frames so they are the same and work in the compare teams function.  
prediction$Team[which(prediction$Team == "Southern California")] <- str\_replace\_all(prediction$Team[which(prediction$Team == "Southern California")], "Southern California", "USC")  
  
prediction$Team[which(prediction$Team == "Eastern Wash.")] <- str\_replace\_all(prediction$Team[which(prediction$Team == "Eastern Wash.")], "Eastern Wash.", "Eastern Washington")  
  
kens\_rank$Team[which(kens\_rank$Team == "Connecticut")] <- str\_replace\_all(kens\_rank$Team[which(kens\_rank$Team == "Connecticut")], "Connecticut", "UConn")