Main R Markdown Document For NCAA Basketball Analysis

Ben Wasserman and Connor Kennedy

Table of Contents

Read in the files

```r
# remove environment variables
rm(list = ls())

# read in files
tourney_games <- read.csv("ConferenceTourneyGames.csv")
tourney_compact_results <- read.csv("NCAATourneyCompactResults.csv")
regular_season_compact_results <- read.csv("RegularSeasonCompactResults.csv")
teams <- read.csv("Teams.csv")
team_conferences <- read.csv("TeamConferences.csv")
tourney_seeds <- read.csv("NCAATourneySeeds.csv")
tourney_seeds_round_slots <- read.csv("NCAATourneySeedRoundSlots.csv")
tourney_detailed_result <- read.csv("NCAATourneyDetailedResults.csv")
tourney_slots <- read.csv("NCAATourneySlots.csv")
regular_detailed_result <- read.csv("RegularSeasonDetailedResults.csv")
rankings <- read.csv("MasseyOrdinals.csv")
rating_round_viewers <- read.csv("rating_round_viewers.csv")
tourney_semi_views_ratings <- read.csv("semi_rating_round_viewers.csv")
```

Parse 7 different rankings

```r
tourney_rankings <- rankings %>% filter(RankingDayNum > 132) %>%
  rename (DayNum = RankingDayNum) %>% select(
  Season, SystemName, TeamID, everything()
)

# get rid of duplicates by system name, season, and team ID
tourney_rankings <- tourney_rankings[!duplicated(tourney_rankings[1:3]),]

# parse rankings
bih_rankings <- tourney_rankings %>% filter(SystemName == 'BIH')
col_rankings <- tourney_rankings %>% filter(SystemName == 'COL')
dol_rankings <- tourney_rankings %>% filter(SystemName == 'DOL')
mor_rankings <- tourney_rankings %>% filter(SystemName == 'MOR')
wlk_rankings <- tourney_rankings %>% filter(SystemName == 'WLK')
wob_rankings <- tourney_rankings %>% filter(SystemName == 'WOB')
wol_rankings <- tourney_rankings %>% filter(SystemName == 'WOL')

# rename each the team IDs and the ranks to prepare for join
bih_rankings <- bih_rankings %>% rename(WTeamID = TeamID,
                                        WRankBih = OrdinalRank)

col_rankings <- col_rankings %>% rename(WTeamID = TeamID,
                                        WRankCol = OrdinalRank)

dol_rankings <- dol_rankings %>% rename(WTeamID = TeamID,
                                        WRankDol = OrdinalRank)


wlk_rankings <- wlk_rankings %>% rename(WTeamID = TeamID,
                                        WRankWlk = OrdinalRank)

wob_rankings <- wob_rankings %>% rename(WTeamID = TeamID,
                                        WRankWob = OrdinalRank)
```

```r
wol_rankings <- wol_rankings %>% rename(WTeamID = TeamID,
                                        WRankWol = OrdinalRank)

mor_rankings <- mor_rankings %>% rename(WTeamID = TeamID,
                                        WRankMor = OrdinalRank)

# join with tourney detailed result
tourney_d_result <-left_join(x = tourney_detailed_result,
                                    y = bih_rankings,
                                    by = c("Season", "WTeamID"))


tourney_d_result <-left_join(x = tourney_d_result,
                                    y = wob_rankings,
                                    by = c("Season", "WTeamID"))


tourney_d_result <-left_join(x = tourney_d_result,
                                    y = wol_rankings,
                                    by = c("Season", "WTeamID"))


tourney_d_result <-left_join(x = tourney_d_result,
                                    y = wlk_rankings,
                                    by = c("Season", "WTeamID"))


tourney_d_result <-left_join(x = tourney_d_result,
                                    y = dol_rankings,
                                    by = c("Season", "WTeamID"))


tourney_d_result <-left_join(x = tourney_d_result,
                                    y = col_rankings,
                                    by = c("Season", "WTeamID"))


tourney_d_result <-left_join(x = tourney_d_result,
                                    y = mor_rankings,
                                    by = c("Season", "WTeamID"))


# rename rankings to prepare for join with losing team
bih_rankings <- bih_rankings %>% rename(LTeamID = WTeamID,
                                        LRankBih = WRankBih)

col_rankings <- col_rankings %>% rename(LTeamID = WTeamID,
                                        LRankCol = WRankCol)

dol_rankings <- dol_rankings %>% rename(LTeamID = WTeamID,
                                        LRankDol = WRankDol)
```

```
wlk_rankings <- wlk_rankings %>% rename(LTeamID = WTeamID,
                                       LRankWlk = WRankWlk)

wob_rankings <- wob_rankings %>% rename(LTeamID = WTeamID,
                                       LRankWob = WRankWob)

wol_rankings <- wol_rankings %>% rename(LTeamID = WTeamID,
                                       LRankWol = WRankWol)

mor_rankings <- mor_rankings %>% rename(LTeamID = WTeamID,
                                       LRankMor = WRankMor)


tourney_d_result <-left_join(x = tourney_d_result,
                                    y = bih_rankings,
                                    by = c("Season", "LTeamID"))

tourney_d_result <-left_join(x = tourney_d_result,
                                    y = wob_rankings,
                                    by = c("Season", "LTeamID"))

tourney_d_result <-left_join(x = tourney_d_result,
                                    y = wol_rankings,
                                    by = c("Season", "LTeamID"))

tourney_d_result <-left_join(x = tourney_d_result,
                                    y = wlk_rankings,
                                    by = c("Season", "LTeamID"))

tourney_d_result <-left_join(x = tourney_d_result,
                                    y = dol_rankings,
                                    by = c("Season", "LTeamID"))

tourney_d_result <-left_join(x = tourney_d_result,
                                    y = col_rankings,
                                    by = c("Season", "LTeamID"))

tourney_d_result <-left_join(x = tourney_d_result,
                                    y = mor_rankings,
                                    by = c("Season", "LTeamID"))

# remove unecessary columns
tourney_d_result = tourney_d_result[,-grep("^DayNum.y",colnames(tourney_d_result))]
tourney_d_result = tourney_d_result[,-grep("^DayNum.x.",colnames(tourney_d_result))]
tourney_d_result = tourney_d_result[,-grep("^SystemName",colnames(tourney_d_result))]

tourney_d_result <- tourney_d_result %>% select(-DayNum)

tourney_d_result <- tourney_d_result %>% rename(DayNum = DayNum.x)
```

Add rankings to the detailed results

```
# just get the average ranking for each
tourney_d_result <- tourney_d_result %>% mutate(
  WAvgSeed = ((WRankMor + WRankWol + WRankBih + WRankWob + WRankWlk + WRankDol
               + WRankCol)/7),
  LAvgSeed = ((LRankMor + LRankWol + LRankBih + LRankWob + LRankWlk + LRankBih
               + LRankCol)/7),
  AvgSeedDiff = WAvgSeed - LAvgSeed
)

# get rid of tournament seeds prior to 2002 for consistency
tourney_seeds <- subset(tourney_seeds, Season > 2002)
tourney_seeds <- tourney_seeds %>% rename(WTeamID = TeamID)

tourney_d_result <-left_join(x = tourney_d_result,
                                    y = tourney_seeds,
                                    by = c("WTeamID", "Season"))

tourney_seeds <- tourney_seeds %>% rename(LTeamID = WTeamID)

tourney_d_result <-left_join(x = tourney_d_result,
                                    y = tourney_seeds,
                                    by = c("LTeamID", "Season"))

tourney_d_result <- tourney_d_result %>% rename(
  WTeamSeed = Seed.x,
  LTeamSeed = Seed.y
)

# make all the seeds only numeric
tourney_d_result$WTeamSeed <- as.integer(gsub('[a-zA-Z]', '',
                                              tourney_d_result$WTeamSeed))
tourney_d_result$LTeamSeed <- as.integer(gsub('[a-zA-Z]', '',
                                              tourney_d_result$LTeamSeed))
```

Add team conferences to detailed result

```
team_conferences <- team_conferences %>% rename(WTeamID = TeamID)

# join with conferences
tourney_d_result <- left_join(x = tourney_d_result,
                                    y = team_conferences,
                                    by = c("WTeamID", "Season"))

team_conferences <- team_conferences %>% rename(LTeamID = WTeamID)

tourney_d_result <- left_join(x = tourney_d_result,
                                    y = team_conferences,
                                    by = c("LTeamID", "Season"))

tourney_d_result <- tourney_d_result %>% rename(
  WTeamConf = ConfAbbrev.x,
  LTeamConf = ConfAbbrev.y
)
```

```
tourney_d_result <- tourney_d_result %>% mutate(LTeamConfFactor = fct_lump(LTeamConf, 10))

tourney_d_result <- tourney_d_result %>% mutate(WTeamConfFactor = fct_lump(WTeamConf, 10))
```

Randomly code each team to prepare for a binomial model

```
# higher_team_won is the variable we will be predicting on
tourney_d_result <- tourney_d_result %>% mutate(
  lower_team = pmin(WTeamID, LTeamID),
  higher_team = pmax(WTeamID, LTeamID),
  higher_team_won = ifelse(higher_team == WTeamID, 1, 0),
  HTSeed = ifelse(higher_team == WTeamID, WTeamSeed, LTeamSeed),
  LTSeed = ifelse(lower_team == WTeamID, WTeamSeed, LTeamSeed),
  HTScore = ifelse(higher_team == WTeamID, WScore, LScore),
  LTScore = ifelse(lower_team == WTeamID, WScore, LScore),
  HTConf = ifelse(higher_team == WTeamID, as.character(WTeamConf), as.character(LTeamConf)),
  LTConf = ifelse(lower_team == WTeamID, as.character(WTeamConf), as.character(LTeamConf)),
  HTFGM = ifelse(higher_team == WTeamID, WFGM, LFGM),
  LTFGM = ifelse(lower_team == WTeamID, WFGM, LFGM),
  HTFGA = ifelse(higher_team == WTeamID, WFGA, LFGA),
  LTFGA = ifelse(lower_team == WTeamID, WFGA, LFGA),
  HTFGM3 = ifelse(higher_team == WTeamID, WFGM3, LFGM3),
  LTFGM3 = ifelse(lower_team == WTeamID, WFGM3, LFGM3),
  HTFGA3 = ifelse(higher_team == WTeamID, WFGA3, LFGA3),
  LTFGA3 = ifelse(lower_team == WTeamID, WFGA3, LFGA3),
  HTFTM = ifelse(higher_team == WTeamID, WFTM, LFTM),
  LTFTM = ifelse(lower_team == WTeamID, WFTM, LFTM),
  HTFTA = ifelse(higher_team == WTeamID, WFTA, LFTA),
  LTFTA = ifelse(lower_team == WTeamID, WFTA, LFTA),
  HTOR = ifelse(higher_team == WTeamID, WOR, LOR),
  LTOR = ifelse(lower_team == WTeamID, WOR, LOR),
  HTDR = ifelse(higher_team == WTeamID, WDR, LDR),
  LTDR = ifelse(lower_team == WTeamID, WDR, LDR),
  HTAst = ifelse(higher_team == WTeamID, WAst, LAst),
  LTAst = ifelse(lower_team == WTeamID, WAst, LAst),
  HTTO = ifelse(higher_team == WTeamID, WTO, LTO),
  LTTO = ifelse(lower_team == WTeamID, WTO, LTO),
  HTStl = ifelse(higher_team == WTeamID, WStl, LStl),
  LTStl = ifelse(lower_team == WTeamID, WStl, LStl),
  HTBlk = ifelse(higher_team == WTeamID, WBlk, LBlk),
  LTBlk = ifelse(lower_team == WTeamID, WBlk, LBlk),
  HTPF = ifelse(higher_team == WTeamID, WPF, LPF),
  LTPF = ifelse(lower_team == WTeamID, WPF, LPF),
  HTBih = ifelse(higher_team == WTeamID, WRankBih, LRankBih),
  LTBih = ifelse(lower_team == WTeamID, WRankBih, LRankBih),
  HTCol = ifelse(higher_team == WTeamID, WRankCol, LRankCol),
  LTCol = ifelse(lower_team == WTeamID, WRankCol, LRankCol),
  HTDol = ifelse(higher_team == WTeamID, WRankDol, LRankDol),
  LTDol = ifelse(lower_team == WTeamID, WRankDol, LRankDol),
  HTMor = ifelse(higher_team == WTeamID, WRankMor, LRankMor),
  LTMor = ifelse(lower_team == WTeamID, WRankMor, LRankMor),
  HTWlk = ifelse(higher_team == WTeamID, WRankWlk, LRankWlk),
  LTWlk = ifelse(lower_team == WTeamID, WRankWlk, LRankWlk),
```

```
  HTWob = ifelse(higher_team == WTeamID, WRankWob, LRankWob),
  LTWob = ifelse(lower_team == WTeamID, WRankWob, LRankWob),
  HTWol = ifelse(higher_team == WTeamID, WRankWol, LRankWol),
  LTWol = ifelse(lower_team == WTeamID, WRankWol, LRankWol),
  HTAvgRank = ifelse(higher_team == WTeamID, WAvgSeed, LAvgSeed),
  LTAvgRank = ifelse(lower_team == WTeamID, WAvgSeed, LAvgSeed),
  DefaultTest = ifelse(WTeamSeed > LTeamSeed, 1, 0)
)
```

Add team names to the detailed data frame for more interpretability

```
# add team names
teams_tourney <- teams %>% select(TeamName, TeamID)
teams_tourney <- teams_tourney %>% rename(lower_team = TeamID)
tourney_d_result <-left_join(x = tourney_d_result,
                                    y = teams_tourney,
                                    by = "lower_team")


teams_tourney <- teams_tourney %>% rename(higher_team = lower_team)
tourney_d_result <- left_join(x = tourney_d_result,
                                     y = teams_tourney,
                                     by = "higher_team")


# renaming
tourney_d_result %<>% rename(LowerTeamName = TeamName.x,
                                    HigherTeamName = TeamName.y)
```

Add score difference, seed difference and rounds

```
# add score difference
tourney_d_result <- tourney_d_result %>% mutate(scoreDiff = HTScore - LTScore)

# add rounds to tourney_d_results
tourney_d_result <- tourney_d_result %>% mutate(Round = ifelse(
  DayNum %in% 134:135, 0,
  ifelse(DayNum %in% 136:137, 1,
         ifelse(DayNum %in% 138:139, 2,
                ifelse(DayNum %in% 143:144, 3,
                       ifelse(DayNum %in% 145:146, 4,
                              ifelse(DayNum == 152, 5,
                                     ifelse(DayNum == 154, 6, 7))))))
))

tourney_d_result <- tourney_d_result %>% mutate(
  SeedDifference = HTSeed - LTSeed
)
```

Factor some variables

```
# factoring
tourney_d_result %<>% mutate(Season = factor(Season),
                                    DayNum = factor(DayNum),
                                    HigherTeamName = factor(HigherTeamName),
                                    LowerTeamName = factor(LowerTeamName),
                                    NumOT = factor(NumOT))

tourney_d_result <- tourney_d_result %>% mutate(HigherTeamConfFactor = fct_lump(HTConf, 10))

tourney_d_result <- tourney_d_result %>% mutate(LowerTeamConfFactor =
                                                    fct_lump(LTConf, 10))
```

Make all the in game statistics deltas between the two teams

```
tourney_d_result <- tourney_d_result %>% mutate(
  DFTM = HTFTM - LTFTM,
  DStl = HTStl - LTStl,
  DFGA = HTFGA - LTFGA,
  DFGA3 = HTFGA3 - LTFGA3,
  DFGM3 = HTFGM3 - LTFGM3,
  DFTM = HTFTM - LTFTM,
  DFTA = HTFTA - LTFTA,
  DOR = HTOR - LTOR,
  DDR = HTDR - LTDR,
  DTO = HTTO - LTTO,
  DAst = HTAst - LTAst,
  DBlk = HTBlk - LTBlk,
  DPF = HTPF - LTPF
)
```

Parse the regular season data. Since the regular season tables have slightly different attributes and we won't always use both datasets combined, we will initially parse them separately and then join them as needed.

```
# select all rankings for the last 20 days of the regular season
reg_rankings <- rankings %>% filter(between(rankings$RankingDayNum, 110, 130)) %>% select(Season,
                                                                                          SystemName,
                                                                                          TeamID,
                                                                                          everything())


# get rid of duplicates by system name, season, and team ID
reg_rankings <- reg_rankings[!duplicated(reg_rankings[1:3]),]

# parse rankings by the ranking system
bih_rankings <- reg_rankings %>% filter(SystemName == 'BIH')
col_rankings <- reg_rankings %>% filter(SystemName == 'COL')
dol_rankings <- reg_rankings %>% filter(SystemName == 'DOL')
mor_rankings <- reg_rankings %>% filter(SystemName == 'MOR')
wlk_rankings <- reg_rankings %>% filter(SystemName == 'WLK')
wob_rankings <- reg_rankings %>% filter(SystemName == 'WOB')
wol_rankings <- reg_rankings %>% filter(SystemName == 'WOL')

# rename to prepare for join based
bih_rankings <- bih_rankings %>% rename(WTeamID = TeamID,
```

```
                                        WRankBih = OrdinalRank)

col_rankings <- col_rankings %>% rename(WTeamID = TeamID,
                                        WRankCol = OrdinalRank)

dol_rankings <- dol_rankings %>% rename(WTeamID = TeamID,
                                        WRankDol = OrdinalRank)


wlk_rankings <- wlk_rankings %>% rename(WTeamID = TeamID,
                                        WRankWlk = OrdinalRank)

wob_rankings <- wob_rankings %>% rename(WTeamID = TeamID,
                                        WRankWob = OrdinalRank)


wol_rankings <- wol_rankings %>% rename(WTeamID = TeamID,
                                        WRankWol = OrdinalRank)

mor_rankings <- mor_rankings %>% rename(WTeamID = TeamID,
                                        WRankMor = OrdinalRank)

# join with regular season data based on winning team id and season
reg_detailed_result <-left_join(x = regular_detailed_result,
                                    y = bih_rankings,
                                    by = c("Season", "WTeamID"))


reg_detailed_result <-left_join(x = reg_detailed_result,
                                    y = wob_rankings,
                                    by = c("Season", "WTeamID"))


reg_detailed_result <-left_join(x = reg_detailed_result,
                                    y = wol_rankings,
                                    by = c("Season", "WTeamID"))


reg_detailed_result <-left_join(x = reg_detailed_result,
                                    y = wlk_rankings,
                                    by = c("Season", "WTeamID"))


reg_detailed_result <-left_join(x = reg_detailed_result,
                                    y = dol_rankings,
                                    by = c("Season", "WTeamID"))

reg_detailed_result <-left_join(x = reg_detailed_result,
                                    y = col_rankings,
                                    by = c("Season", "WTeamID"))

reg_detailed_result <-left_join(x = reg_detailed_result,
                                    y = mor_rankings,
```

```
                                    by = c("Season", "WTeamID"))


# add losing teams rankings to prepare for join
bih_rankings <- bih_rankings %>% rename(LTeamID = WTeamID,
                                        LRankBih = WRankBih)

col_rankings <- col_rankings %>% rename(LTeamID = WTeamID,
                                        LRankCol = WRankCol)

dol_rankings <- dol_rankings %>% rename(LTeamID = WTeamID,
                                        LRankDol = WRankDol)

wlk_rankings <- wlk_rankings %>% rename(LTeamID = WTeamID,
                                        LRankWlk = WRankWlk)

wob_rankings <- wob_rankings %>% rename(LTeamID = WTeamID,
                                        LRankWob = WRankWob)

wol_rankings <- wol_rankings %>% rename(LTeamID = WTeamID,
                                        LRankWol = WRankWol)

mor_rankings <- mor_rankings %>% rename(LTeamID = WTeamID,
                                        LRankMor = WRankMor)


# join with regular season data based on losing team id and the season
reg_detailed_result <-left_join(x = reg_detailed_result,
                                y = bih_rankings,
                                by = c("Season", "LTeamID"))

reg_detailed_result <-left_join(x = reg_detailed_result,
                                y = wob_rankings,
                                by = c("Season", "LTeamID"))

reg_detailed_result <-left_join(x = reg_detailed_result,
                                y = wol_rankings,
                                by = c("Season", "LTeamID"))

reg_detailed_result <-left_join(x = reg_detailed_result,
                                y = wlk_rankings,
                                by = c("Season", "LTeamID"))

reg_detailed_result <-left_join(x = reg_detailed_result,
                                y = dol_rankings,
                                by = c("Season", "LTeamID"))

reg_detailed_result <-left_join(x = reg_detailed_result,
                                y = col_rankings,
                                by = c("Season", "LTeamID"))

reg_detailed_result <-left_join(x = reg_detailed_result,
                                y = mor_rankings,
```

```
                                    by = c("Season", "LTeamID"))

# remove unecessary columns
reg_detailed_result = reg_detailed_result[,-grep("^RankingDayNum",colnames(reg_detailed_result))]
reg_detailed_result = reg_detailed_result[,-grep("^SystemName",colnames(reg_detailed_result))]
```

```
# get a certain number of days out of the regular season
reg_detailed_result <- reg_detailed_result %>% filter(between(reg_detailed_result$DayNum, 1, 132))

# get the average seed for each
reg_detailed_result <- reg_detailed_result %>% mutate(
  WAvgSeed = ((WRankMor + WRankWol + WRankWob + WRankWlk + WRankDol + WRankBih
               + WRankCol)/7),
  LAvgSeed = ((LRankMor + LRankWol + LRankWob + LRankWlk + LRankBih + LRankBih
               + LRankCol)/7),
  AvgSeedDiff = WAvgSeed - LAvgSeed
)
```

Add conferences

```
# team conferences to tourney results
team_conferences <- read.csv("TeamConferences.csv")
team_conferences <- team_conferences %>% rename(WTeamID = TeamID)

# join with conferences
reg_detailed_result <- left_join(x = reg_detailed_result,
                                 y = team_conferences,
                                 by = c("WTeamID", "Season"))

team_conferences <- team_conferences %>% rename(LTeamID = WTeamID)

reg_detailed_result <- left_join(x = reg_detailed_result,
                                 y = team_conferences,
                                 by = c("LTeamID", "Season"))

reg_detailed_result <- reg_detailed_result %>% rename(
  WTeamConf = ConfAbbrev.x,
  LTeamConf = ConfAbbrev.y
)
```

Randomly code data for a binary variable to predict on

```
# now I'm going to look to make a model for lower team wins and higher team wins
reg_detailed_result <- reg_detailed_result %>% mutate(
  lower_team = pmin(WTeamID, LTeamID),
  higher_team = pmax(WTeamID, LTeamID),
  higher_team_won = ifelse(higher_team == WTeamID, 1, 0),
  HTScore = ifelse(higher_team == WTeamID, WScore, LScore),
  LTScore = ifelse(lower_team == WTeamID, WScore, LScore),
  HTConf = ifelse(higher_team == WTeamID, as.character(WTeamConf), as.character(LTeamConf)),
  LTConf = ifelse(lower_team == WTeamID, as.character(WTeamConf), as.character(LTeamConf)),
  HTFGM = ifelse(higher_team == WTeamID, WFGM, LFGM),
  LTFGM = ifelse(lower_team == WTeamID, WFGM, LFGM),
```

```
  HTFGA = ifelse(higher_team == WTeamID, WFGA, LFGA),
  LTFGA = ifelse(lower_team == WTeamID, WFGA, LFGA),
  HTFGM3 = ifelse(higher_team == WTeamID, WFGM3, LFGM3),
  LTFGM3 = ifelse(lower_team == WTeamID, WFGM3, LFGM3),
  HTFGA3 = ifelse(higher_team == WTeamID, WFGA3, LFGA3),
  LTFGA3 = ifelse(lower_team == WTeamID, WFGA3, LFGA3),
  HTFTM = ifelse(higher_team == WTeamID, WFTM, LFTM),
  LTFTM = ifelse(lower_team == WTeamID, WFTM, LFTM),
  HTFTA = ifelse(higher_team == WTeamID, WFTA, LFTA),
  LTFTA = ifelse(lower_team == WTeamID, WFTA, LFTA),
  HTOR = ifelse(higher_team == WTeamID, WOR, LOR),
  LTOR = ifelse(lower_team == WTeamID, WOR, LOR),
  HTDR = ifelse(higher_team == WTeamID, WDR, LDR),
  LTDR = ifelse(lower_team == WTeamID, WDR, LDR),
  HTAst = ifelse(higher_team == WTeamID, WAst, LAst),
  LTAst = ifelse(lower_team == WTeamID, WAst, LAst),
  HTTO = ifelse(higher_team == WTeamID, WTO, LTO),
  LTTO = ifelse(lower_team == WTeamID, WTO, LTO),
  HTStl = ifelse(higher_team == WTeamID, WStl, LStl),
  LTStl = ifelse(lower_team == WTeamID, WStl, LStl),
  HTBlk = ifelse(higher_team == WTeamID, WBlk, LBlk),
  LTBlk = ifelse(lower_team == WTeamID, WBlk, LBlk),
  HTPF = ifelse(higher_team == WTeamID, WPF, LPF),
  LTPF = ifelse(lower_team == WTeamID, WPF, LPF),
  HTBih = ifelse(higher_team == WTeamID, WRankBih, LRankBih),
  LTBih = ifelse(lower_team == WTeamID, WRankBih, LRankBih),
  HTCol = ifelse(higher_team == WTeamID, WRankCol, LRankCol),
  LTCol = ifelse(lower_team == WTeamID, WRankCol, LRankCol),
  HTDol = ifelse(higher_team == WTeamID, WRankDol, LRankDol),
  LTDol = ifelse(lower_team == WTeamID, WRankDol, LRankDol),
  HTMor = ifelse(higher_team == WTeamID, WRankMor, LRankMor),
  LTMor = ifelse(lower_team == WTeamID, WRankMor, LRankMor),
  HTWlk = ifelse(higher_team == WTeamID, WRankWlk, LRankWlk),
  LTWlk = ifelse(lower_team == WTeamID, WRankWlk, LRankWlk),
  HTWol = ifelse(higher_team == WTeamID, WRankWol, LRankWol),
  LTWol = ifelse(lower_team == WTeamID, WRankWol, LRankWol)
)
```

Add team names for interpretability and add score difference

```
# add team names
teams <- read.csv("Teams.csv")
teams <- teams %>% select(TeamName, TeamID)
teams <- teams %>% rename(lower_team = TeamID)
reg_detailed_result <-left_join(x = reg_detailed_result,
                                y = teams,
                                by = "lower_team")


teams <- teams %>% rename(higher_team = lower_team)
reg_detailed_result <- left_join(x = reg_detailed_result,
                                 y = teams,
                                 by = "higher_team")
```

```
# renaming
reg_detailed_result %<>% rename(LowerTeamName = TeamName.x,
                                HigherTeamName = TeamName.y)

# add score difference
reg_detailed_result <- reg_detailed_result %>% mutate(scoreDiff = HTScore - LTScore)
```

Factor a variety of variables

```
# factoring
reg_detailed_result %<>% mutate(Season = factor(Season),
                                DayNum = factor(DayNum),
                                HigherTeamName = factor(HigherTeamName),
                                LowerTeamName = factor(LowerTeamName),
                                NumOT = factor(NumOT))


reg_detailed_result <- reg_detailed_result %>% mutate(HigherTeamConfFactor = fct_lump(HTConf, 10))

reg_detailed_result <- reg_detailed_result %>% mutate(LowerTeamConfFactor =
                                                        fct_lump(LTConf, 10))
```

Add deltas for the in game statistics

```
# transition all wins and losses into deltas
reg_detailed_result <- reg_detailed_result %>% mutate(
  DFTM = HTFTM - LTFTM,
  DFGM = HTFGM - LTFGM,
  DStl = HTStl - LTStl,
  DFGA = HTFGA - LTFGA,
  DFGA3 = HTFGA3 - LTFGA3,
  DFGM3 = HTFGM3 - LTFGM3,
  DFTM = HTFTM - LTFTM,
  DFTA = HTFTA - LTFTA,
  DOR = HTOR - LTOR,
  DDR = HTDR - LTDR,
  DTO = HTTO - LTTO,
  DAst = HTAst - LTAst,
  DBlk = HTBlk - LTBlk,
  DPF = HTPF - LTPF
)
```

Get averages

```
averages <- reg_detailed_result %>%
  group_by(higher_team, Season) %>%
  summarize(DFTMReg = mean(DFTM, na.rm = TRUE),
            DFGMReg = mean(DFGM, na.rm = TRUE),
            DStlReg = mean(DStl, na.rm = TRUE),
            DFGAReg = mean(DFGA, na.rm = TRUE),
            DFGA3Reg = mean(DFGA3, na.rm = TRUE),
```

```
            DFGM3Reg = mean(DFGM3, na.rm = TRUE),
            DFTAReg = mean(DFTA, na.rm = TRUE),
            DORReg = mean(DOR, na.rm = TRUE),
            DDRReg = mean(DDR, na.rm = TRUE),
            DTOReg = mean(DTO, na.rm = TRUE),
            DAstReg = mean(DAst, na.rm =TRUE),
            DBlkReg = mean(DBlk, na.rm = TRUE),
            DPFAReg = mean(DPF, na.rm = TRUE))
```

Combine with tourney detailed result

```
tourney_d_result <- left_join(x = tourney_d_result,
                                    y = averages,
                                    by = "higher_team", "Season")

# remove duplicates caused by join
tourney_d_result <- tourney_d_result[!duplicated(tourney_d_result[1:10]),]


tourney_model_df <- tourney_d_result %>% select(
  higher_team, lower_team, higher_team_won, HigherTeamConfFactor,
  LowerTeamConfFactor, DFTMReg, DStlReg, DFGAReg, DFGA3Reg, DFGMReg,
  DStlReg, DFGAReg, DFGA3Reg, DFGM3Reg, DFTAReg, DORReg, DDRReg,
  DTOReg, DAstReg, DBlkReg, DPFAReg, AvgSeedDiff, Season.y, DayNum
)

tourney_model_df <- tourney_model_df %>% rename(
  DFTM = DFTMReg,
  DStl = DStlReg,
  DFGA = DFGAReg,
  DFGA3 = DFGA3Reg,
  DFGM3 = DFGM3Reg,
  DFGM = DFGMReg,
  DFTA = DFTAReg,
  DOR = DORReg,
  DDR = DDRReg,
  DTO = DTOReg,
  DAst = DAstReg,
  DBlk = DBlkReg,
  DPF = DPFAReg,
  Season = Season.y
)

reg_model_df <- reg_detailed_result %>% select(
  higher_team, lower_team, higher_team_won, HigherTeamConfFactor,
  LowerTeamConfFactor, DFTM, DStl, DFGA, DFGA3, DFGM3, DFGM, DFTA,
  DOR, DDR, DTO, DAst, DBlk, DPF, Season, DayNum, AvgSeedDiff
)
```

Some initial summary statistics

```r
# get original data frame to help with some of the summary stats
tourney_result_unmodified <- tourney_detailed_result

# wins per team
wins_per_team <- tourney_result_unmodified %>% group_by(WTeamID) %>% summarize(num_wins = n()) %>% arra
wins_per_team
```

```
## # A tibble: 143 x 2
##    WTeamID num_wins
##      <int>    <int>
##  1    1314       42
##  2    1242       38
##  3    1246       35
##  4    1181       33
##  5    1277       31
##  6    1196       30
##  7    1163       29
##  8    1257       28
##  9    1458       28
## 10    1393       25
## # ... with 133 more rows
```

```r
wins_per_team %<>% rename(higher_team = WTeamID)
wins_per_team <- left_join(x = wins_per_team,
                           y = teams,
                           by = "higher_team")
wins_per_team <- wins_per_team %>% rename(WTeamID = higher_team) %>% select(WTeamID, TeamName, num_wins
head(wins_per_team)
```

```
## # A tibble: 6 x 3
##   WTeamID TeamName       num_wins
##     <int> <fct>             <int>
## 1    1314 North Carolina       42
## 2    1242 Kansas               38
## 3    1246 Kentucky             35
## 4    1181 Duke                 33
## 5    1277 Michigan St          31
## 6    1196 Florida              30
```

```r
# winning summary
winning_pts_summary <- tourney_result_unmodified %>% summarize(min_fg_made = min(WFGM),
                                                               mean_fg_made = mean(WFGM),
                                                               max_fg_made = max(WFGM),
                                                               sd_fg_made = sd(WFGM),
                                                               min_3pts_made = min(WFGM3),
                                                               mean_3pts_made = mean(WFGM3),
                                                               max_3pts_made = max(WFGM3),
                                                               sd_3pts_made = sd(WFGM3)
                                                               )
winning_pts_summary
```

```
##   min_fg_made mean_fg_made max_fg_made sd_fg_made min_3pts_made
```

```
## 1          13     26.17431          44   4.782749             0
##   mean_3pts_made max_3pts_made sd_3pts_made
## 1       6.787971            16     2.804133
```

```
# mean winner points summary
mean_pts_summary <- tourney_result_unmodified %>% summarize(win_mean_fg_made = mean(WFGM),
                                                            win_mean_3pts_made = mean(WFGM3),
                                                            lose_mean_fg_made = mean(LFGM),
                                                            lose_mean_3pts_made = mean(LFGM3)
                                                            )
mean_pts_summary
```

```
##   win_mean_fg_made win_mean_3pts_made lose_mean_fg_made
## 1         26.17431           6.787971          22.90622
##   lose_mean_3pts_made
## 1            6.124363
```

```
# season summary - not sure how this DF was altered for winner
# season <- tourney_detailed_result1 %>% group_by(Season) %>% group_by(winner) %>% summarize(mean(HTSco
# season

# summary of Scores and Points
summary_scores_points <- tourney_d_result %>%
  select(HTScore, LTScore, HTFGM, HTFGA, LTFGM, LTFGA, HTFGM3, HTFGA3, LTFGM3, LTFGA3)
summary(summary_scores_points)
```

```
##     HTScore         LTScore         HTFGM           HTFGA
##  Min.   : 39.00   Min.   : 29.00   Min.   :12.00   Min.   :34.00
##  1st Qu.: 61.00   1st Qu.: 61.00   1st Qu.:21.00   1st Qu.:51.00
##  Median : 69.00   Median : 69.00   Median :24.00   Median :56.00
##  Mean   : 69.67   Mean   : 69.08   Mean   :24.59   Mean   :56.48
##  3rd Qu.: 77.00   3rd Qu.: 77.00   3rd Qu.:27.00   3rd Qu.:62.00
##  Max.   :121.00   Max.   :112.00   Max.   :44.00   Max.   :80.00
##      LTFGM           LTFGA          HTFGM3          HTFGA3
##  Min.   :11.00   Min.   :37.0   Min.   : 0.000   Min.   : 5.00
##  1st Qu.:21.00   1st Qu.:51.0   1st Qu.: 4.000   1st Qu.:15.00
##  Median :24.00   Median :56.0   Median : 6.000   Median :18.00
##  Mean   :24.49   Mean   :56.1   Mean   : 6.468   Mean   :18.89
##  3rd Qu.:27.00   3rd Qu.:61.0   3rd Qu.: 8.000   3rd Qu.:22.00
##  Max.   :43.00   Max.   :85.0   Max.   :18.000   Max.   :38.00
##      LTFGM3          LTFGA3
##  Min.   : 0.000   Min.   : 4.00
##  1st Qu.: 4.000   1st Qu.:15.00
##  Median : 6.000   Median :19.00
##  Mean   : 6.444   Mean   :18.68
##  3rd Qu.: 8.000   3rd Qu.:22.00
##  Max.   :16.000   Max.   :42.00
```

```
# summary details of score by HTSeed
HTSeed_details <- tourney_d_result %>% group_by(HTSeed) %>%
  summarize(avg_points_HTSeed = mean(HTScore),
            max_points_HTSeed = max(HTScore),
```

```
            min_points_HTSeed = min(HTScore))
HTSeed_details
```

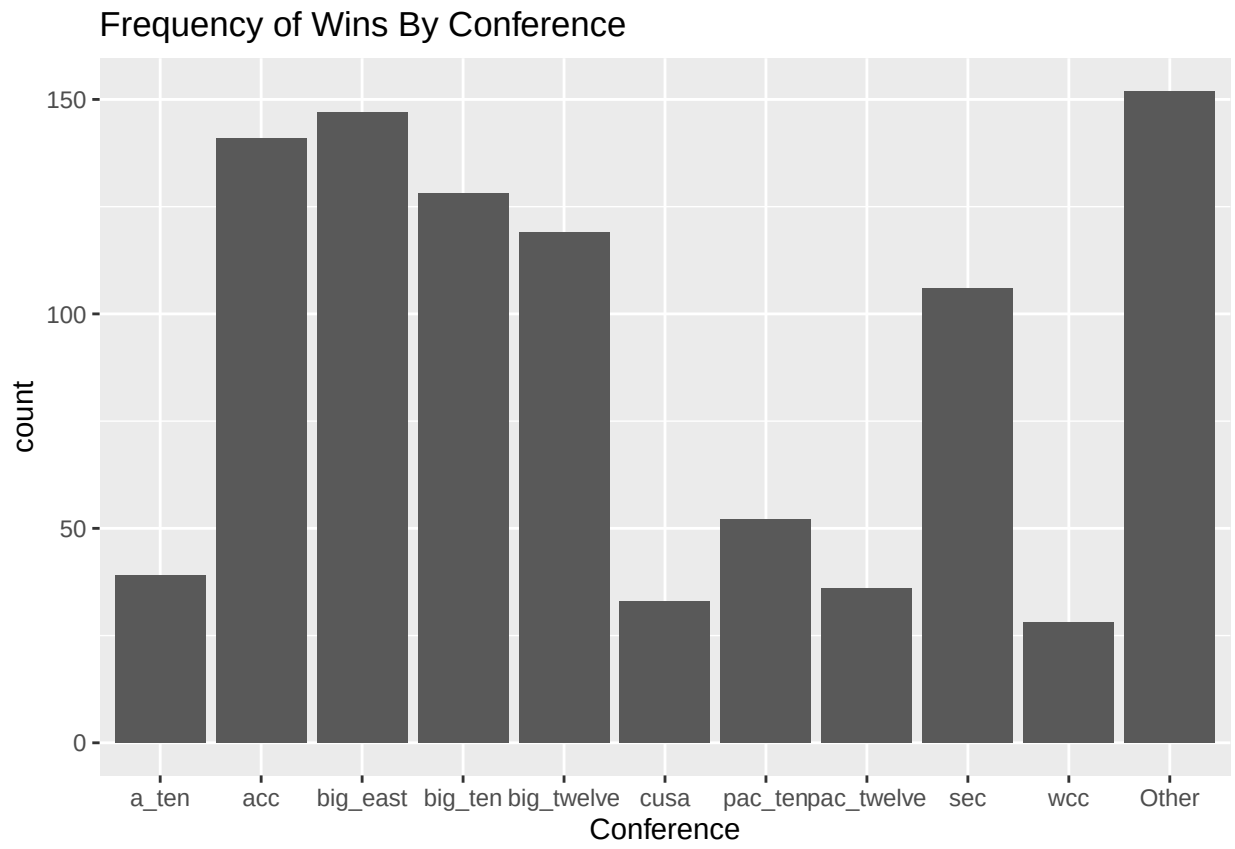
```
## # A tibble: 16 x 4
##    HTSeed avg_points_HTSeed max_points_HTSeed min_points_HTSeed
##     <int>             <dbl>             <int>             <int>
##  1      1              77.5               113                44
##  2      2              72.0               108                50
##  3      3              71.4               102                51
##  4      4              71.0                94                47
##  5      5              68.9               121                39
##  6      6              68.1                96                48
##  7      7              71.4               111                47
##  8      8              68.4               100                49
##  9      9              68.0                95                45
## 10     10              68.0                92                43
## 11     11              68.9                94                48
## 12     12              67.0               101                46
## 13     13              64.7                85                40
## 14     14              62.3                97                43
## 15     15              60.3                90                40
## 16     16              63.0                96                44
```

```
# summary details of score by LTSeed
LTSeed_details <- tourney_d_result %>% group_by(LTSeed) %>%
  summarize(avg_points_LTSeed = mean(LTScore),
            max_points_LTSeed = max(LTScore),
            min_points_LTSeed = min(LTScore))
LTSeed_details
```

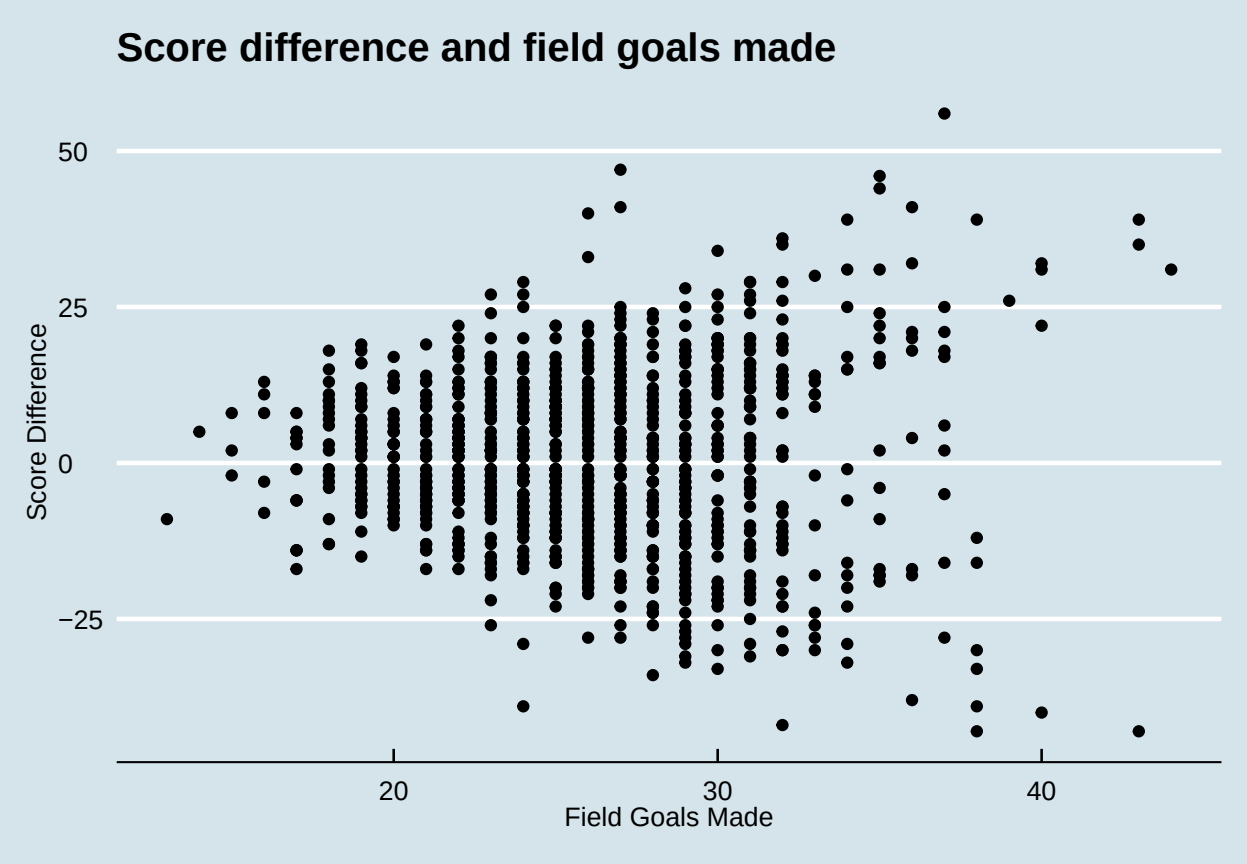
```
## # A tibble: 16 x 4
##    LTSeed avg_points_LTSeed max_points_LTSeed min_points_LTSeed
##     <int>             <dbl>             <int>             <int>
##  1      1              75.8               112                45
##  2      2              74.4               105                46
##  3      3              72.3               101                39
##  4      4              70.6               100                45
##  5      5              68.8                99                41
##  6      6              68.1                87                48
##  7      7              67.9                99                49
##  8      8              68.5                94                41
##  9      9              68.4               102                49
## 10     10              67.1                90                43
## 11     11              64.7                95                41
## 12     12              65.7                87                42
## 13     13              65.4                84                34
## 14     14              63.0                80                44
## 15     15              60.3                87                35
## 16     16              61.6                84                29
```

Let's look at a few plots that might start to reveal some interesting patterns

```
# top ten conferences frequency of wins
p4 <- ggplot(tourney_d_result, aes(x=WTeamConfFactor)) +
  geom_bar() +
  labs(x = "Conference",
       title = "Frequency of Wins By Conference")

p4
```




```
# field goals made vs score difference
p1 <- ggplot(
  tourney_d_result,
  aes(x = WFGM, y = scoreDiff)) +
  geom_point() +
  labs(x = "Field Goals Made",
       y = "Score Difference",
       title = "Score difference and field goals made") +
  theme_economist()
p1
```



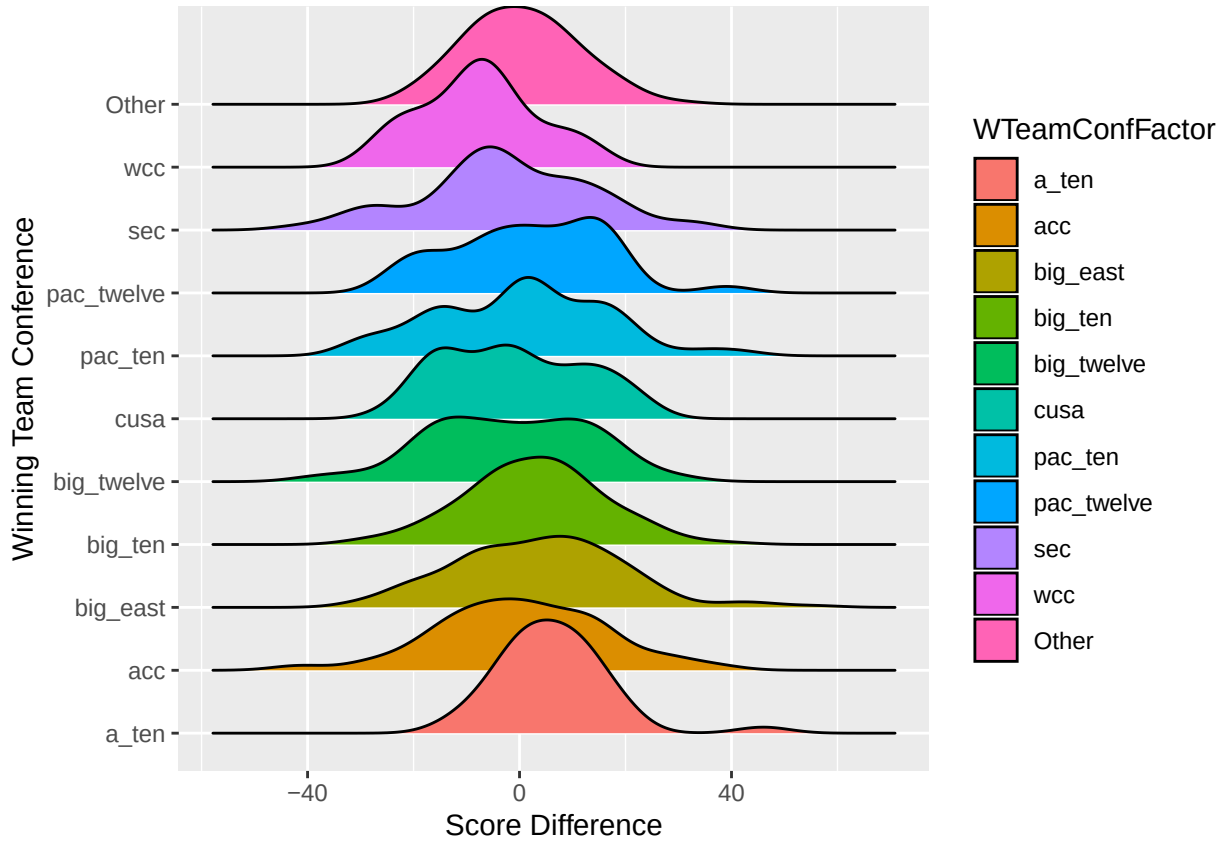

```
# density plot by conference
library(ggridges)
```

```
##
## Attaching package: 'ggridges'
```

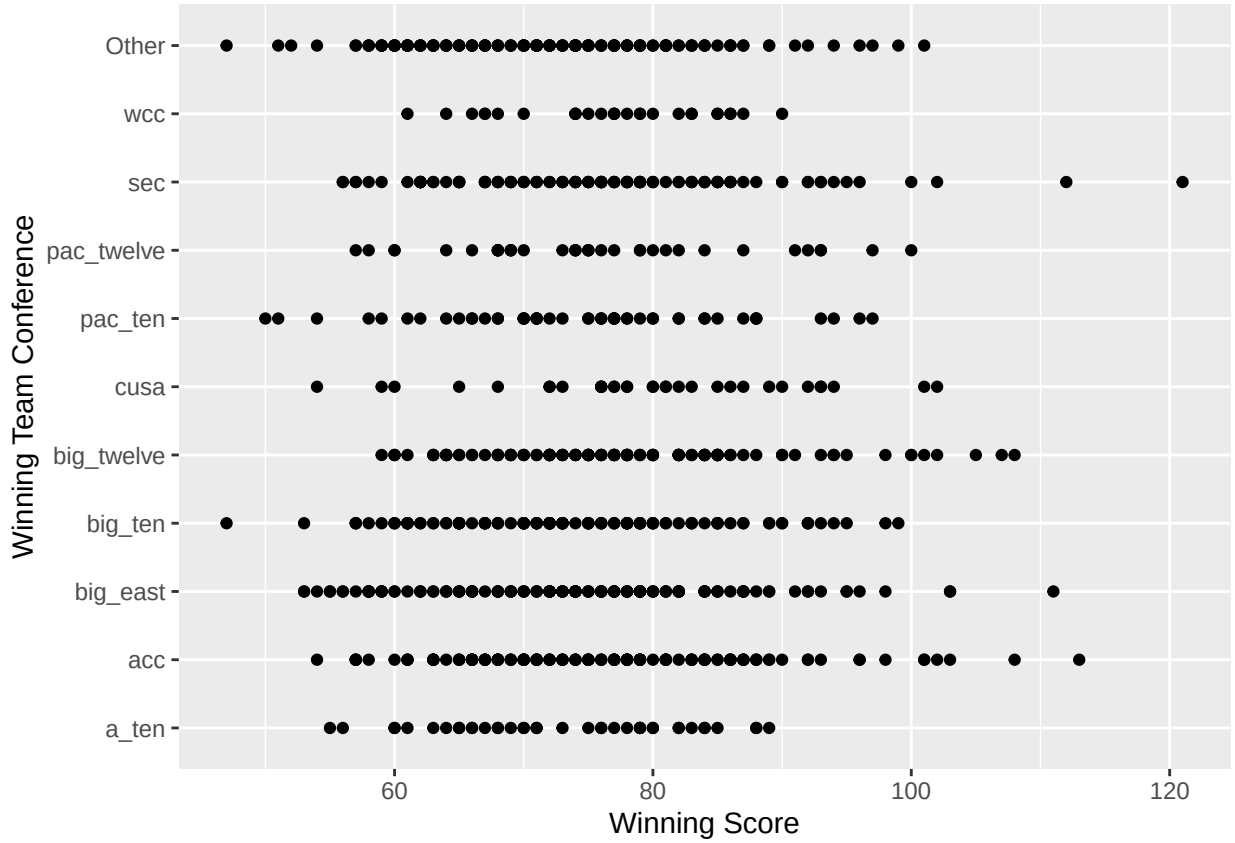
```
## The following object is masked from 'package:ggplot2':
##
##     scale_discrete_manual
```

```
p2 <- ggplot(tourney_d_result, aes(x = scoreDiff, y = WTeamConfFactor, fill = WTeamConfFactor)) +
  geom_density_ridges() +
  labs(x = "Score Difference",
       y = "Winning Team Conference")

p2
```

```
## Picking joint bandwidth of 4.96
```



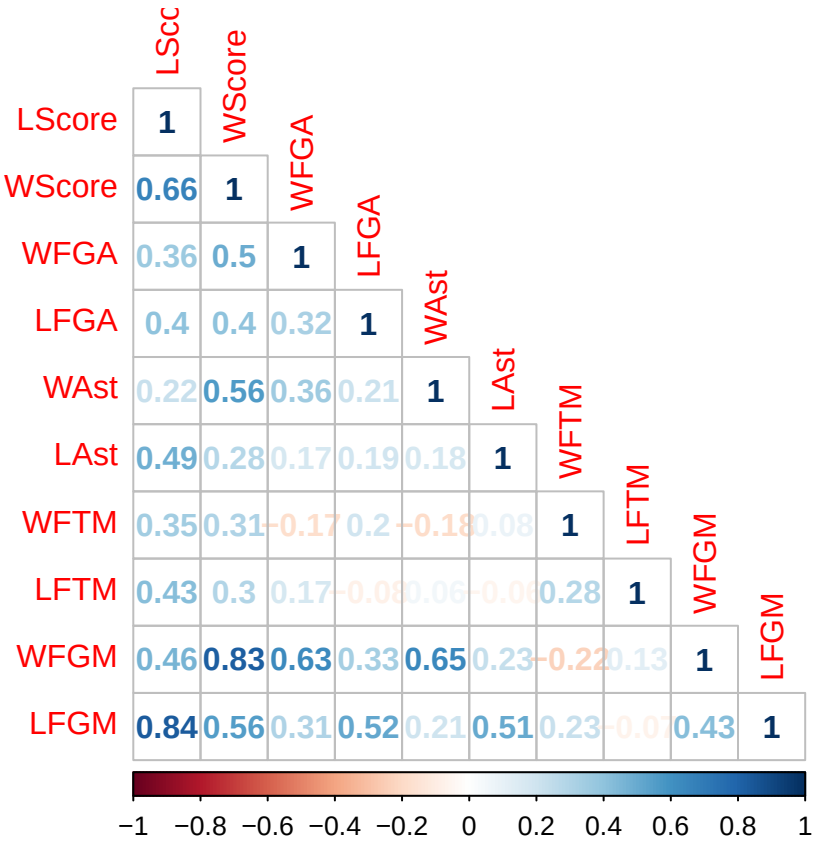

```
# scatter plot of winning scores and conferences
p3 <- ggplot(tourney_d_result, aes(x = WScore, y = WTeamConfFactor)) +
  geom_point() +
  labs(x = "Winning Score",
       y = "Winning Team Conference")

p3
```




```
# correlation plot of a variety of game stats - used as averages later on
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
cormat <- cor(tourney_d_result %>% select(LScore, WScore,
                                          WFGA, LFGA,
                                        WAst, LAst, WFTM, LFTM,
                                          WFGM, LFGM) %>% drop_na())
corrplot(cormat, method = "number", type="lower")
```



Combine with views and ratings data

```
# factor the season
rating_round_viewers <- rating_round_viewers %>% mutate(Season = factor(Season))
View(rating_round_viewers)
tourney_semi_views_ratings <- tourney_semi_views_ratings %>% mutate(
  Season = factor(Season)
)

# join the ratings with the tourney detailed result
tourney_d_result <- tourney_d_result %>% mutate(Season.x = as.numeric(as.character(Season.x)))
rating_round_viewers <- rating_round_viewers %>% mutate(Season = as.numeric(as.character(Season)))
tourney_d_result <- tourney_d_result %>% rename(Season = Season.x)

tourney_d_result <- tourney_d_result %>% select(Round, DayNum, everything())

tourney_finals_views_ratings <- left_join(x = tourney_d_result,
                                  y = rating_round_viewers,
                                  by = c("Season", "Round"))



tourney_semi_views_ratings <- tourney_semi_views_ratings %>% rename(WTeamID = TeamID1)
tourney_semi_views_ratings <- tourney_semi_views_ratings %>% rename(LTeamID = TeamID2)

tourney_semi_views_ratings <- tourney_semi_views_ratings %>% mutate(Season =
```

```
                                                    as.numeric(as.character(Season)))

# data frame with the semi final ratings and viewers
tourney_semi_views_ratings <- left_join(x = tourney_d_result,
                                        y = tourney_semi_views_ratings,
                                        by= c("Season", "WTeamID", "LTeamID"))

# data frame with the finals ratings and viewers
tourney_finals_views_ratings <- tourney_finals_views_ratings %>% filter(
  Round == 6
)

tourney_semi_views_ratings <- tourney_semi_views_ratings %>% rename(Round = Round.y)
tourney_semi_views_ratings <- tourney_semi_views_ratings %>% select(-Round.x)

tourney_views_ratings <- rbind(tourney_semi_views_ratings, tourney_finals_views_ratings)

tourney_views_ratings <- tourney_views_ratings %>% mutate(
  Network = factor(Network)
)

tourney_views_ratings <- tourney_views_ratings %>% select(Viewers, Rating, Network, Round,
                                                          everything())

tourney_views_ratings <- na.omit(tourney_views_ratings)
```

Add some variables that might be interesting to analyze in relation to the views and ratings

```
tourney_views_ratings <- tourney_views_ratings %>% mutate(
  IsUpset = ifelse(WTeamSeed < LTeamSeed, 1, 0)
)

tourney_views_ratings <- tourney_views_ratings %>% mutate(
  SeedDifference = abs(WTeamSeed - LTeamSeed)
)

tourney_views_ratings <- tourney_views_ratings %>% mutate(
  TotalThreesMade = WFGM3 + LFGM3,
  TotalFieldGoals = WFGM + LFGM,
  TotalFieldGoalsAttempts = WFGA + LFGA,
  TotalPointsScored = WScore + LScore
)
```

Baseline model based only on seed

```
preds_seed_df <- data.frame(
  tourney_d_result
)

preds_seed_df <- preds_seed_df %>% mutate(
  class_pred = ifelse(DefaultTest == 1, "Yes", "No")
)
```

```
table(preds_seed_df$class_pred, preds_seed_df$higher_team_won)
```

```
##
##         0   1
##   No  353 365
##   Yes 126 137
```

Logistic Regression. First we'll run logistic regression with just historical tournament data based on the NCAA pre-bracket seed difference between the two teams, the average ranking difference at the end of the regular season based on the average of 7 different ranking systems, and the factored conference for each team.

```
# logistic fit with a lot less data
tourney_d_result <- na.omit(tourney_d_result)

# Split into test and train
set.seed(1861)
(num_rows <- nrow(tourney_d_result))
```

```
## [1] 917
```

```
train_idx <- sample(1:num_rows, floor(.8 * num_rows))

tourney_d_train <- tourney_d_result %>% slice(train_idx)
tourney_d_test <- tourney_d_result %>% slice(-train_idx)

glm_fit <- glm(higher_team_won ~ SeedDifference + AvgSeedDiff + HigherTeamConfFactor +
                 LowerTeamConfFactor,
               data = tourney_d_train,
               family = binomial)

summary(glm_fit)
```

```
##
## Call:
## glm(formula = higher_team_won ~ SeedDifference + AvgSeedDiff +
##     HigherTeamConfFactor + LowerTeamConfFactor, family = binomial,
##     data = tourney_d_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1684  -0.8942   0.3143   0.8921   2.1911
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -0.154149   0.419103  -0.368   0.7130
## SeedDifference                -0.181467   0.020330  -8.926   <2e-16 ***
## AvgSeedDiff                   -0.001017   0.002056  -0.495   0.6207
## HigherTeamConfFactoracc        0.255539   0.436124   0.586   0.5579
## HigherTeamConfFactorbig_east   0.187853   0.422551   0.445   0.6566
## HigherTeamConfFactorbig_ten    0.263129   0.417173   0.631   0.5282
```

```
## HigherTeamConfFactorbig_twelve -0.279365   0.448557  -0.623   0.5334
## HigherTeamConfFactormvc         0.345817   0.561954   0.615   0.5383
## HigherTeamConfFactormwc        -0.886722   0.586121  -1.513   0.1303
## HigherTeamConfFactorpac_ten     0.334041   0.520408   0.642   0.5209
## HigherTeamConfFactorpac_twelve  0.402391   0.556435   0.723   0.4696
## HigherTeamConfFactorsec        -0.093865   0.473078  -0.198   0.8427
## HigherTeamConfFactorOther       0.292245   0.407055   0.718   0.4728
## LowerTeamConfFactorbig_east     0.360402   0.356084   1.012   0.3115
## LowerTeamConfFactorbig_ten     -0.066794   0.381598  -0.175   0.8611
## LowerTeamConfFactorbig_twelve   0.229776   0.342479   0.671   0.5023
## LowerTeamConfFactorcusa         0.194370   0.505608   0.384   0.7007
## LowerTeamConfFactormwc          0.824018   0.559912   1.472   0.1411
## LowerTeamConfFactorpac_ten     -0.107019   0.501301  -0.213   0.8310
## LowerTeamConfFactorpac_twelve   0.357571   0.526353   0.679   0.4969
## LowerTeamConfFactorsec         -0.798912   0.371999  -2.148   0.0317 *
## LowerTeamConfFactorwcc          0.024467   0.500034   0.049   0.9610
## LowerTeamConfFactorOther       -0.032117   0.335060  -0.096   0.9236
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1015.92  on 732  degrees of freedom
## Residual deviance:  794.55  on 710  degrees of freedom
## AIC: 840.55
##
## Number of Fisher Scoring iterations: 4
```

```r
preds_test_DF <- data.frame(
  scores_logit = predict(glm_fit, type = "response", newdata = tourney_d_test),
  tourney_d_test
)

preds_train_DF <- data.frame(
  scores_logit = predict(glm_fit, type = "response", newdata = tourney_d_train),
  tourney_d_train
)

preds_test_DF <- preds_test_DF %>% mutate(
  class_pred05 = ifelse(scores_logit > .5, "Yes", "No")
)

preds_test_DF <- preds_test_DF %>% mutate(
  class_pred05 = ifelse(scores_logit > .5, "Yes", "No")
)

table(preds_test_DF$class_pred05, preds_test_DF$higher_team_won)
```

```
##
##        0  1
##   No  72 28
##   Yes 23 61
```

Next we will still only use the historical tournament data to train, but we will attempt to add in the average

in game statistics from the regular season. Adding these in will add a lot of noise, so it will be interesting to see how the model performs with these attributes added. Then try and use historical tournament data and leave out a couple seasons to test on.

```
tourney_d_2017 <- subset(tourney_d_result, Season == 2017)
tourney_d_2017_train <- subset(tourney_d_result, as.numeric(Season) < 2017)
tourney_d_2016 <- subset(tourney_d_result, Season == 2016)
tourney_d_2016_train <- subset(tourney_d_result, as.numeric(Season) < 2016)

glm_fit <- glm(higher_team_won ~ SeedDifference + AvgSeedDiff + HigherTeamConfFactor +
                 LowerTeamConfFactor + DFTMReg + DStlReg + DFGAReg + DFGA3Reg + DFGM3Reg + DFTAReg + DO
               + DORReg + DTOReg + DAstReg + DBlkReg,
               data = tourney_d_train,
               family = binomial)

summary(glm_fit)
```

```
##
## Call:
## glm(formula = higher_team_won ~ SeedDifference + AvgSeedDiff +
##     HigherTeamConfFactor + LowerTeamConfFactor + DFTMReg + DStlReg +
##     DFGAReg + DFGA3Reg + DFGM3Reg + DFTAReg + DORReg + DDRReg +
##     DORReg + DTOReg + DAstReg + DBlkReg, family = binomial, data = tourney_d_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0901  -0.8940   0.2931   0.8540   2.4095
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -0.3288727  0.4633334  -0.710   0.4778
## SeedDifference                -0.1854649  0.0209541  -8.851   <2e-16 ***
## AvgSeedDiff                   -0.0007831  0.0020801  -0.376   0.7065
## HigherTeamConfFactoracc        0.3232376  0.4741300   0.682   0.4954
## HigherTeamConfFactorbig_east   0.3448738  0.4499041   0.767   0.4433
## HigherTeamConfFactorbig_ten    0.3078108  0.4700439   0.655   0.5126
## HigherTeamConfFactorbig_twelve -0.4559007  0.4843969  -0.941   0.3466
## HigherTeamConfFactormvc        0.3541907  0.6139894   0.577   0.5640
## HigherTeamConfFactormwc       -0.7120869  0.6499357  -1.096   0.2732
## HigherTeamConfFactorpac_ten    0.5640567  0.5929267   0.951   0.3414
## HigherTeamConfFactorpac_twelve 0.5722915  0.6097615   0.939   0.3480
## HigherTeamConfFactorsec       -0.1577531  0.5088472  -0.310   0.7565
## HigherTeamConfFactorOther      0.4735240  0.4483063   1.056   0.2909
## LowerTeamConfFactorbig_east    0.3321620  0.3640745   0.912   0.3616
## LowerTeamConfFactorbig_ten    -0.0399966  0.3918171  -0.102   0.9187
## LowerTeamConfFactorbig_twelve  0.2336529  0.3500999   0.667   0.5045
## LowerTeamConfFactorcusa        0.1121865  0.5169653   0.217   0.8282
## LowerTeamConfFactormwc         0.7685315  0.5660559   1.358   0.1746
## LowerTeamConfFactorpac_ten    -0.1793321  0.5127207  -0.350   0.7265
## LowerTeamConfFactorpac_twelve  0.2758602  0.5357746   0.515   0.6066
## LowerTeamConfFactorsec        -0.8317630  0.3787646  -2.196   0.0281 *
## LowerTeamConfFactorwcc         0.0998335  0.5099112   0.196   0.8448
## LowerTeamConfFactorOther       0.0203548  0.3435522   0.059   0.9528
## DFTMReg                       -0.1814139  0.1260957  -1.439   0.1502
```

```
## DStlReg                        -0.2324503  0.1113841  -2.087   0.0369 *
## DFGAReg                         0.1739338  0.1839956   0.945   0.3445
## DFGA3Reg                        0.0147943  0.0771457   0.192   0.8479
## DFGM3Reg                       -0.0895769  0.1822652  -0.491   0.6231
## DFTAReg                         0.2177762  0.1463054   1.489   0.1366
## DORReg                         -0.2318532  0.1982289  -1.170   0.2422
## DDRReg                          0.1002140  0.0753818   1.329   0.1837
## DTOReg                         -0.0148095  0.1715846  -0.086   0.9312
## DAstReg                         0.0230204  0.0621898   0.370   0.7113
## DBlkReg                        -0.1332695  0.0947601  -1.406   0.1596
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1015.92  on 732  degrees of freedom
## Residual deviance:  782.06  on 699  degrees of freedom
## AIC: 850.06
##
## Number of Fisher Scoring iterations: 4
```

```
preds_test_DF <- data.frame(
  scores_logit = predict(glm_fit, type = "response", newdata = tourney_d_test),
  tourney_d_test
)

preds_2017_DF <- data.frame(
  scores_logit = predict(glm_fit, type = "response", newdata = tourney_d_2017),
  tourney_d_2017
)

preds_test_DF <- preds_test_DF %>% mutate(
  class_pred05 = ifelse(scores_logit > .5, "Yes", "No")
)

preds_test_2017 <- preds_2017_DF %>% mutate(
  class_pred05 = ifelse(scores_logit > .5, "Yes", "No")
)

table(preds_test_DF$class_pred05, preds_test_DF$higher_team_won)
```

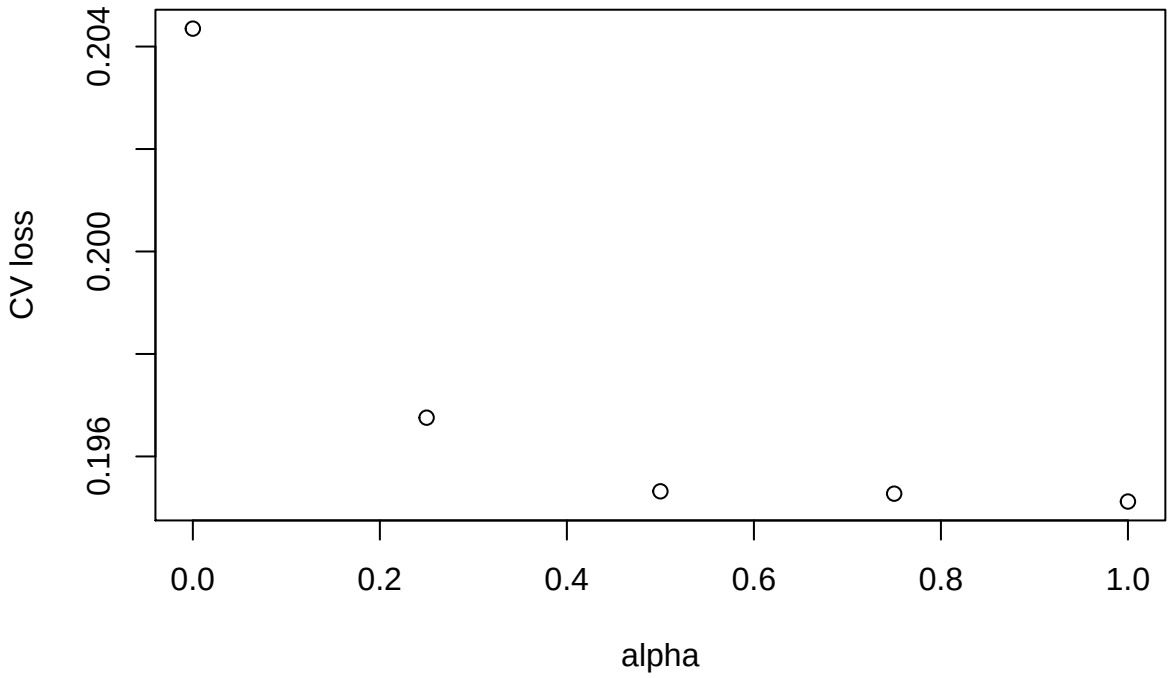
```
##
##        0  1
##   No  72 34
##   Yes 23 55
```

```
table(preds_test_2017$class_pred05, preds_test_2017$higher_team_won)
```

```
##
##        0  1
##   No  31 15
##   Yes  2 19
```

Elastic Net To Look for the Optimal Alpha

```
alpha_list <- seq(0,1,len = 5)

enet_fit <- cva.glmnet(higher_team_won ~ SeedDifference + AvgSeedDiff + HigherTeamConfFactor +
                LowerTeamConfFactor + DFTMReg + DStlReg + DFGAReg + DFGA3Reg + DFGM3Reg + DFTAReg + DO
             + DORReg + DTOReg + DAstReg + DBlkReg,
                       data = tourney_d_train,
                       alpha = alpha_list)

minlossplot(enet_fit)
```



Run at the optimal alpha

```
# alpha = 1 for lasso which appears to minimize CV Loss
lasso_fit <- cv.glmnet(higher_team_won ~ SeedDifference + AvgSeedDiff + HigherTeamConfFactor +
                LowerTeamConfFactor + DFTMReg + DStlReg + DFGAReg + DFGA3Reg + DFGM3Reg + DFTAReg + DO
            + DORReg + DTOReg + DAstReg + DBlkReg,
          data = tourney_d_train,
          family = "binomial",
          alpha = 1,
          nfolds = 10)

lasso_preds <- predict(lasso_fit, newdata = tourney_d_test, s = lasso_fit$lambda.min)
lasso_preds_2017 <- predict(lasso_fit, newdata = tourney_d_2017, s = lasso_fit$lambda.min)
```

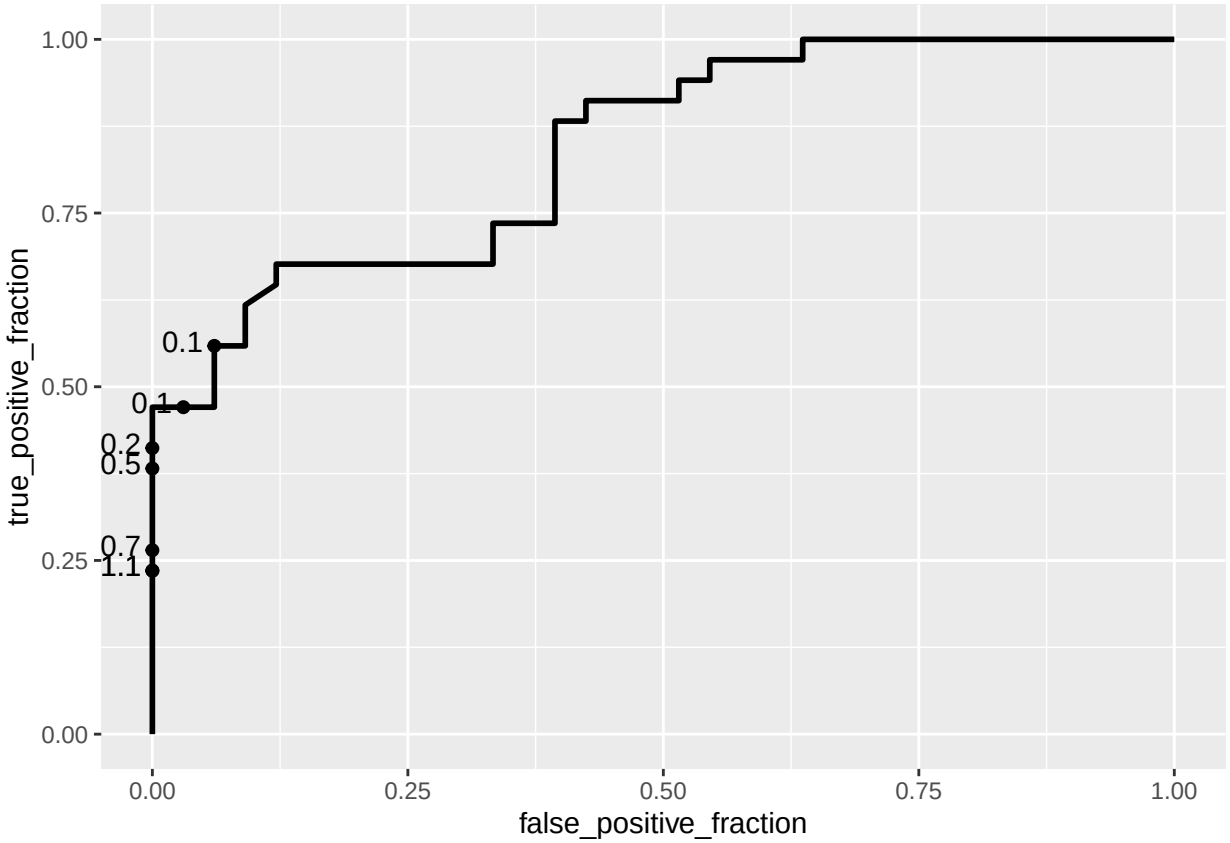
```r
preds_test_lasso <- data.frame(
  tourney_d_test,
  lasso_preds
)

preds_2017_lasso <- data.frame(
  tourney_d_2017,
  lasso_preds_2017
)

preds_test_lasso <- preds_test_lasso %>% mutate(
  class_pred05 = ifelse(lasso_preds > .5, "Yes", "No")
)

preds_2017_lasso <- preds_2017_lasso %>% mutate(
  class_pred05 = ifelse(lasso_preds_2017 > .05, "Yes", "No")
)


table(preds_test_lasso$class_pred05, preds_test_lasso$higher_team_won)
```

```
##
##        0  1
##   No  81 47
##   Yes 14 42
```

```r
table(preds_2017_lasso$class_pred05, preds_2017_lasso$higher_team_won)
```

```
##
##        0  1
##   No  31 15
##   Yes  2 19
```

```r
# how are these probabilities not scaled to 1?
(roc_test <- ggplot(preds_2017_lasso, aes(m = lasso_preds_2017,
                                          d = higher_team_won)) +
  geom_roc(cutoffs.at = c(.99, .9, .75, .5, .25, .1, .01)))
```

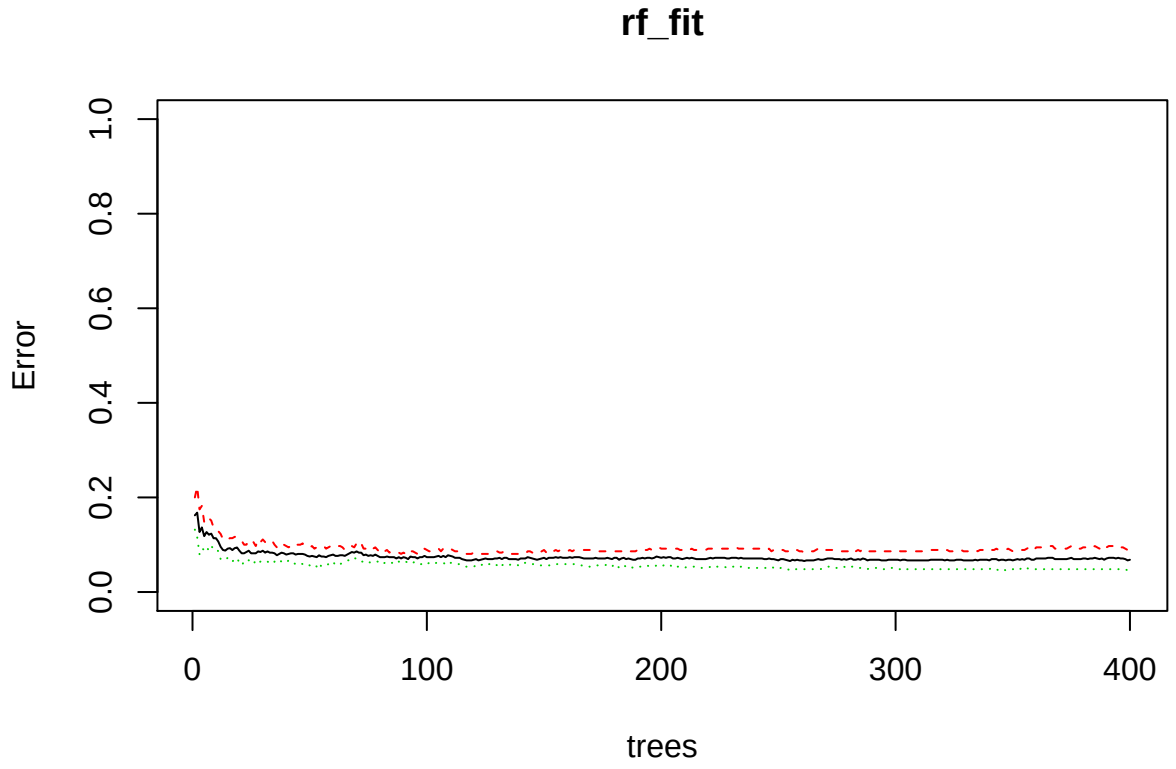


Random Forest

```
rf_fit <- randomForest(as.factor(higher_team_won) ~ SeedDifference + AvgSeedDiff + HigherTeamConfFactor
                  LowerTeamConfFactor,
                        data = tourney_d_train,
                        type = classification,
                        mrty = 3,
                        ntree = 400,
                        importance = TRUE,
                        localImp = TRUE)

rf_fit
```

```
##
## Call:
##  randomForest(formula = as.factor(higher_team_won) ~ SeedDifference +      AvgSeedDiff + HigherTeamC
##                Type of random forest: classification
##                      Number of trees: 400
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 6.82%
## Confusion matrix:
##     0   1 class.error
## 0 327  33  0.09166667
## 1  17 356  0.04557641
```

```
plot(rf_fit, ylim = c(0, 1))
```




```
# THIS LINE IS JUST COMMENTED OUT BECAUSE IT CAUSES ISSUES WITH KNITTING
#explain_forest(rf_fit, interactions = TRUE, data = tourney_d_train)

preds_test_DF <- data.frame(
  scores_logit = predict(rf_fit, type = "prob", newdata = tourney_d_test),
  tourney_d_test
)

View(preds_test_DF)

preds_test_DF <- preds_test_DF %>% mutate(
  class_pred05 = ifelse(scores_logit.1 > .5, "Yes", "No")
)

table(preds_test_DF$class_pred05, preds_test_DF$higher_team_won)
```

```
##
##         0  1
##   No   86  7
##   Yes   9 82
```

```
View(preds_test_DF)

potential_upsets <- preds_test_DF %>% filter(between(preds_test_DF$scores_logit.1, .4, .6))

View(potential_upsets)

View(teams)
```

TV Viewership Linear Regression

```
##### Will need to do cross validation here since there isn't a ton of data ########

names(tourney_views_ratings)
```

```
##   [1] "Viewers"                "Rating"
##   [3] "Network"                "Round"
##   [5] "DayNum"                 "Season"
##   [7] "WTeamID"                "WScore"
##   [9] "LTeamID"                "LScore"
##  [11] "WLoc"                   "NumOT"
##  [13] "WFGM"                   "WFGA"
##  [15] "WFGM3"                  "WFGA3"
##  [17] "WFTM"                   "WFTA"
##  [19] "WOR"                    "WDR"
##  [21] "WAst"                   "WTO"
##  [23] "WStl"                   "WBlk"
##  [25] "WPF"                    "LFGM"
##  [27] "LFGA"                   "LFGM3"
##  [29] "LFGA3"                  "LFTM"
##  [31] "LFTA"                   "LOR"
##  [33] "LDR"                    "LAst"
##  [35] "LTO"                    "LStl"
##  [37] "LBlk"                   "LPF"
##  [39] "WRankBih"               "WRankWob"
##  [41] "WRankWol"               "WRankWlk"
##  [43] "WRankDol"               "WRankCol"
##  [45] "WRankMor"               "LRankBih"
##  [47] "LRankWob"               "LRankWol"
##  [49] "LRankWlk"               "LRankDol"
##  [51] "LRankCol"               "LRankMor"
##  [53] "WAvgSeed"               "LAvgSeed"
##  [55] "AvgSeedDiff"            "WTeamSeed"
##  [57] "LTeamSeed"              "WTeamConf"
##  [59] "LTeamConf"              "LTeamConfFactor"
##  [61] "WTeamConfFactor"        "lower_team"
##  [63] "higher_team"            "higher_team_won"
##  [65] "HTSeed"                 "LTSeed"
##  [67] "HTScore"                "LTScore"
##  [69] "HTConf"                 "LTConf"
##  [71] "HTFGM"                  "LTFGM"
##  [73] "HTFGA"                  "LTFGA"
##  [75] "HTFGM3"                 "LTFGM3"
##  [77] "HTFGA3"                 "LTFGA3"
```

```
##  [79] "HTFTM"                    "LTFTM"
##  [81] "HTFTA"                    "LTFTA"
##  [83] "HTOR"                     "LTOR"
##  [85] "HTDR"                     "LTDR"
##  [87] "HTAst"                    "LTAst"
##  [89] "HTTO"                     "LTTO"
##  [91] "HTStl"                    "LTStl"
##  [93] "HTBlk"                    "LTBlk"
##  [95] "HTPF"                     "LTPF"
##  [97] "HTBih"                    "LTBih"
##  [99] "HTCol"                    "LTCol"
## [101] "HTDol"                    "LTDol"
## [103] "HTMor"                    "LTMor"
## [105] "HTWlk"                    "LTWlk"
## [107] "HTWob"                    "LTWob"
## [109] "HTWol"                    "LTWol"
## [111] "HTAvgRank"                "LTAvgRank"
## [113] "DefaultTest"              "LowerTeamName"
## [115] "HigherTeamName"           "scoreDiff"
## [117] "SeedDifference"           "HigherTeamConfFactor"
## [119] "LowerTeamConfFactor"      "DFTM"
## [121] "DStl"                     "DFGA"
## [123] "DFGA3"                    "DFGM3"
## [125] "DFTA"                     "DOR"
## [127] "DDR"                      "DTO"
## [129] "DAst"                     "DBlk"
## [131] "DPF"                      "Season.y"
## [133] "DFTMReg"                  "DFGMReg"
## [135] "DStlReg"                  "DFGAReg"
## [137] "DFGA3Reg"                 "DFGM3Reg"
## [139] "DFTAReg"                  "DORReg"
## [141] "DDRReg"                   "DTOReg"
## [143] "DAstReg"                  "DBlkReg"
## [145] "DPFAReg"                  "IsUpset"
## [147] "TotalThreesMade"          "TotalFieldGoals"
## [149] "TotalFieldGoalsAttempts" "TotalPointsScored"
```

```
lin_mod1 <- lm(Viewers ~ IsUpset + SeedDifference + TotalThreesMade + TotalPointsScored,
               data = tourney_views_ratings)

summary(lin_mod1)
```

```
##
## Call:
## lm(formula = Viewers ~ IsUpset + SeedDifference + TotalThreesMade +
##     TotalPointsScored, data = tourney_views_ratings)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.1961 -2.9348 -0.8219  2.2906 10.5290
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       20.75736    5.17798   4.009 0.000284 ***
```

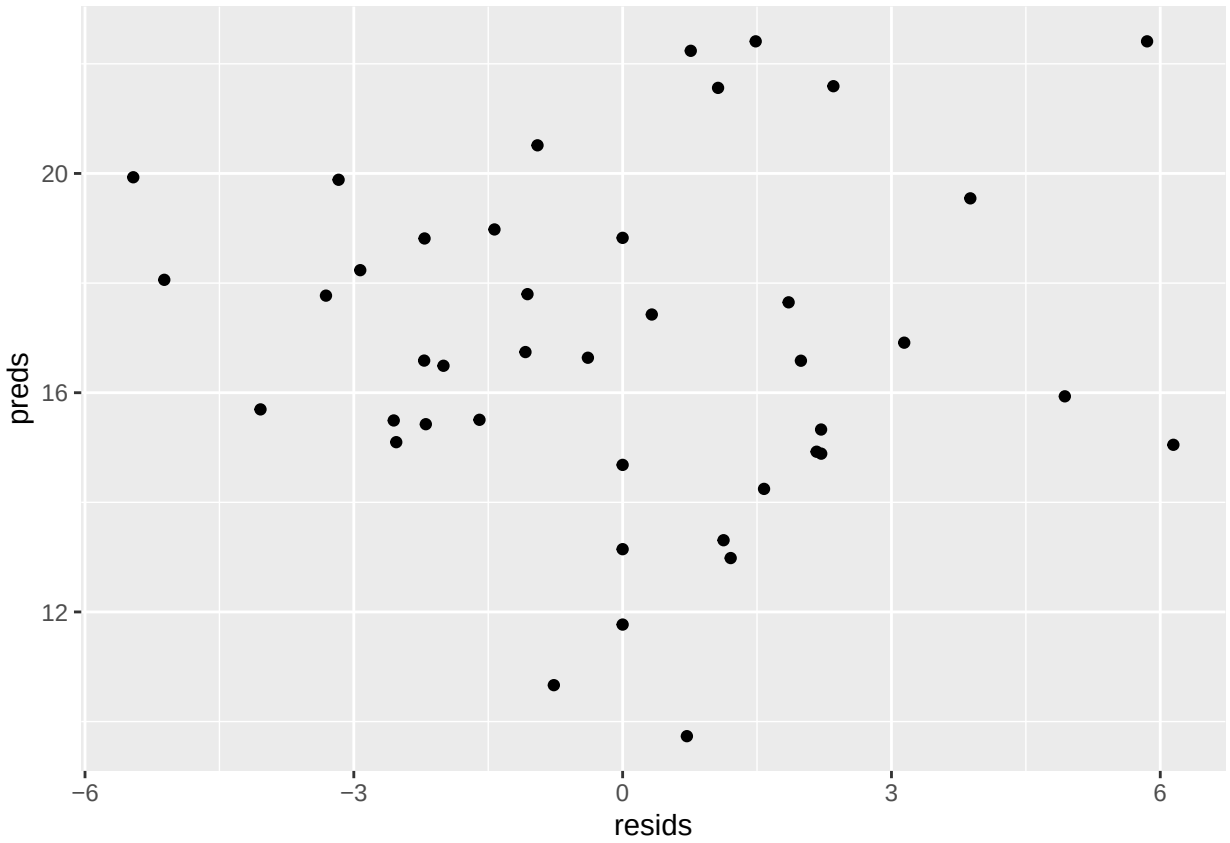
```
## IsUpset           -0.36209    1.48505  -0.244 0.808715
## SeedDifference    -0.34587    0.28784  -1.202 0.237152
## TotalThreesMade    0.26565    0.20610   1.289 0.205416
## TotalPointsScored -0.04539    0.04244  -1.069 0.291822
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.078 on 37 degrees of freedom
## Multiple R-squared:  0.1024, Adjusted R-squared:  0.005325
## F-statistic: 1.055 on 4 and 37 DF,  p-value: 0.3925
```

```
tourney_views_ratings <- tourney_views_ratings %>% mutate(
  scoreDiff = WScore - LScore
)

# score difference and round are significant, ACC games are relevant
lin_mod2 <- lm(Viewers ~ relevel(WTeamConfFactor, ref="Other") + scoreDiff + IsUpset + WTeamSeed +
                relevel(LTeamConfFactor, ref="Other"),
              data = tourney_views_ratings)

summary(lin_mod2)
```

```
##
## Call:
## lm(formula = Viewers ~ relevel(WTeamConfFactor, ref = "Other") +
##     scoreDiff + IsUpset + WTeamSeed + relevel(LTeamConfFactor,
##     ref = "Other"), data = tourney_views_ratings)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -5.461 -2.147  0.000  1.784  6.146
##
## Coefficients:
##                                                 Estimate Std. Error
## (Intercept)                                     21.98508    4.51090
## relevel(WTeamConfFactor, ref = "Other")acc       2.13290    3.65317
## relevel(WTeamConfFactor, ref = "Other")big_east  0.84283    3.32473
## relevel(WTeamConfFactor, ref = "Other")big_ten   0.71295    3.93091
## relevel(WTeamConfFactor, ref = "Other")big_twelve -1.25470    4.15852
## relevel(WTeamConfFactor, ref = "Other")cusa      -4.29686    5.90045
## relevel(WTeamConfFactor, ref = "Other")pac_ten   -4.24832    4.89197
## relevel(WTeamConfFactor, ref = "Other")sec       0.97894    3.34956
## relevel(WTeamConfFactor, ref = "Other")wcc       -5.33362    5.15252
## scoreDiff                                       -0.18574    0.07776
## IsUpset                                         -1.38872    1.50647
## WTeamSeed                                       -0.76591    0.49863
## relevel(LTeamConfFactor, ref = "Other")acc       -3.31237    2.45387
## relevel(LTeamConfFactor, ref = "Other")big_east  -3.81780    2.75972
## relevel(LTeamConfFactor, ref = "Other")big_ten   -0.01325    2.11167
## relevel(LTeamConfFactor, ref = "Other")big_twelve -3.38956    2.58549
## relevel(LTeamConfFactor, ref = "Other")cusa      -1.01569    3.44721
## relevel(LTeamConfFactor, ref = "Other")pac_ten   -2.36623    3.55932
## relevel(LTeamConfFactor, ref = "Other")pac_twelve -2.95260    4.20026
## relevel(LTeamConfFactor, ref = "Other")sec       0.92913    2.64513
```

```
##                                                t value Pr(>|t|)
## (Intercept)                                      4.874 7.16e-05 ***
## relevel(WTeamConfFactor, ref = "Other")acc       0.584   0.5653
## relevel(WTeamConfFactor, ref = "Other")big_east  0.254   0.8022
## relevel(WTeamConfFactor, ref = "Other")big_ten   0.181   0.8577
## relevel(WTeamConfFactor, ref = "Other")big_twelve -0.302   0.7657
## relevel(WTeamConfFactor, ref = "Other")cusa      -0.728   0.4742
## relevel(WTeamConfFactor, ref = "Other")pac_ten   -0.868   0.3945
## relevel(WTeamConfFactor, ref = "Other")sec       0.292   0.7728
## relevel(WTeamConfFactor, ref = "Other")wcc       -1.035   0.3118
## scoreDiff                                        -2.388   0.0259 *
## IsUpset                                          -0.922   0.3666
## WTeamSeed                                        -1.536   0.1388
## relevel(LTeamConfFactor, ref = "Other")acc       -1.350   0.1908
## relevel(LTeamConfFactor, ref = "Other")big_east  -1.383   0.1804
## relevel(LTeamConfFactor, ref = "Other")big_ten   -0.006   0.9951
## relevel(LTeamConfFactor, ref = "Other")big_twelve -1.311   0.2034
## relevel(LTeamConfFactor, ref = "Other")cusa      -0.295   0.7710
## relevel(LTeamConfFactor, ref = "Other")pac_ten   -0.665   0.5131
## relevel(LTeamConfFactor, ref = "Other")pac_twelve -0.703   0.4895
## relevel(LTeamConfFactor, ref = "Other")sec       0.351   0.7287
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.696 on 22 degrees of freedom
## Multiple R-squared:  0.5617, Adjusted R-squared:  0.1832
## F-statistic: 1.484 on 19 and 22 DF,  p-value: 0.186
```

```r
preds_train_DF <- data.frame(
  preds = predict(lin_mod2),
  resids = tourney_views_ratings$Viewers - predict(lin_mod2),
  tourney_views_ratings
)

# Check for Heteroschedasticity
(ggplot(preds_train_DF, aes(x = resids, y = preds)) +
  geom_point())
```



Conclusion: