March Madness 2017

This notebook uses historical March Madness data to predict the results of the 2017 tournament.

Reading the data

The data comes in the following files:

```
ls | grep csv$
```

```
## RegularSeasonCompactResults.csv
## RegularSeasonDetailedResults.csv
## Seasons.csv
## Teams.csv
## TourneyCompactResults.csv
## TourneyDetailedResults.csv
## TourneySeeds.csv
## TourneySlots.csv
## sample_submission.csv
```

When training the model, we mostly care about using the detailed results to develop features. The detailed data, both tournament and regular season, contains game by game information as follows:

```
# Setup
library(data.table)
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.2.5

## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr

## Warning: package 'tidyr' was built under R version 3.2.5

## Warning: package 'readr' was built under R version 3.2.5

## Warning: package 'purrr' was built under R version 3.2.5

## Warning: package 'dplyr' was built under R version 3.2.5

## Conflicts with tidy packages ----------------------------------------------

## between():   dplyr, data.table
## filter():    dplyr, stats
## lag():       dplyr, stats
## last():      dplyr, data.table
## transpose(): purrr, data.table
```

```
regularSeason <- fread("RegularSeasonDetailedResults.csv")
tournament <- fread("TourneyDetailedResults.csv")
head(regularSeason)
```

```
##    Season Daynum Wteam Wscore Lteam Lscore Wloc Numot Wfgm Wfga Wfgm3
## 1:   2003     10  1104     68  1328     62    N     0   27   58     3
## 2:   2003     10  1272     70  1393     63    N     0   26   62     8
## 3:   2003     11  1266     73  1437     61    N     0   24   58     8
## 4:   2003     11  1296     56  1457     50    N     0   18   38     3
## 5:   2003     11  1400     77  1208     71    N     0   30   61     6
## 6:   2003     11  1458     81  1186     55    H     0   26   57     6
##    Wfga3 Wftm Wfta Wor Wdr Wast Wto Wstl Wblk Wpf Lfgm Lfga Lfgm3 Lfga3
## 1:    14   11   18  14  24   13  23    7    1  22   22   53     2    10
## 2:    20   10   19  15  28   16  13    4    4  18   24   67     6    24
## 3:    18   17   29  17  26   15  10    5    2  25   22   73     3    26
## 4:     9   17   31   6  19   11  12   14    2  18   18   49     6    22
## 5:    14   11   13  17  22   12  14    4    4  20   24   62     6    16
## 6:    12   23   27  12  24   12   9    9    3  18   20   46     3    11
##    Lftm Lfta Lor Ldr Last Lto Lstl Lblk Lpf
## 1:   16   22  10  22    8  18    9    2  20
## 2:    9   20  20  25    7  12    8    6  16
## 3:   14   23  31  22    9  12    2    5  23
## 4:    8   15  17  20    9  19    4    3  23
## 5:   17   27  21  15   12  10    7    1  14
## 6:   12   17   6  22    8  19    4    3  25
```

```r
head(tournament)
```

```
##    Season Daynum Wteam Wscore Lteam Lscore Wloc Numot Wfgm Wfga Wfgm3
## 1:   2003    134  1421     92  1411     84    N     1   32   69    11
## 2:   2003    136  1112     80  1436     51    N     0   31   66     7
## 3:   2003    136  1113     84  1272     71    N     0   31   59     6
## 4:   2003    136  1141     79  1166     73    N     0   29   53     3
## 5:   2003    136  1143     76  1301     74    N     1   27   64     7
## 6:   2003    136  1163     58  1140     53    N     0   17   52     4
##    Wfga3 Wftm Wfta Wor Wdr Wast Wto Wstl Wblk Wpf Lfgm Lfga Lfgm3 Lfga3
## 1:    29   17   26  14  30   17  12    5    3  22   29   67    12    31
## 2:    23   11   14  11  36   22  16   10    7   8   20   64     4    16
## 3:    14   16   22  10  27   18   9    7    4  19   25   69     7    28
## 4:     7   18   25  11  20   15  18   13    1  19   27   60     7    17
## 5:    20   15   23  18  20   17  13    8    2  14   25   56     9    21
## 6:    14   20   27  12  29    8  14    3    8  16   20   64     2    17
##    Lftm Lfta Lor Ldr Last Lto Lstl Lblk Lpf
## 1:   14   31  17  28   16  15    5    0  22
## 2:    7    7   8  26   12  17   10    3  15
## 3:   14   21  20  22   11  12    2    5  18
## 4:   12   17  14  17   20  21    6    6  21
## 5:   15   20  10  26   16  14    5    8  19
## 6:   11   13  15  26   11  11    8    4  22
```

Tidying the data

To generate training and testing data, we need to work with data tables with the same formatting as the sample submission, which is formatted as follows:

```r
sampleSub <- fread("sample_submission.csv")
head(sampleSub)
```

```
##                id pred
## 1: 2013_1103_1107  0.5
## 2: 2013_1103_1112  0.5
## 3: 2013_1103_1125  0.5
## 4: 2013_1103_1129  0.5
## 5: 2013_1103_1137  0.5
## 6: 2013_1103_1139  0.5
```

We need to get ids for all the games that contain the season, followed by the team with the lower team id, then the team with the higher team id. The label is 1 when the team with the lower team id wins, and 0 otherwise. To create this data table, we use the following function:

```
# Converts detailed data to submission data
convertToSub <- function(dt) {
    dt <- copy(dt)
    dt[, id := paste(Season, ifelse(Wteam > Lteam, Lteam, Wteam), ifelse(Wteam < Lteam, Lteam, Wteam), s
    dt[, label := as.numeric(Wteam < Lteam)]
    dt[, .(id, label)]
}

labeledSeason <- convertToSub(regularSeason)
labeledTournament <- convertToSub(tournament)
head(labeledSeason)
```

```
##                id label
## 1: 2003_1104_1328     1
## 2: 2003_1272_1393     1
## 3: 2003_1266_1437     1
## 4: 2003_1296_1457     1
## 5: 2003_1208_1400     0
## 6: 2003_1186_1458     0
```

```
head(labeledTournament)
```

```
##                id label
## 1: 2003_1411_1421     0
## 2: 2003_1112_1436     1
## 3: 2003_1113_1272     1
## 4: 2003_1141_1166     1
## 5: 2003_1143_1301     1
## 6: 2003_1140_1163     0
```

Now that we have labeled data that we can add features to, we use the historical data to generate features.

Working with the Historical Data

The game-by-game records from the detailed data sets contain a lot of information that we cannot use in its current form.

To create features that can be added to our labeled data, we look at the following approaches:

1. Get season averages from the available statistics
    - Compute advanced statistics, like an adjusted score for each game based on the opponents allowed points per game, then take the season average again
2. Create a score for how "hot" a team is using the data as a time series, possibly using the advanced statistics calculated previously

Computing Season Averages

The following function computes the season averages for each of the major basketball statistical categories using any detailed data.

```r
compressStats <- function(dt) {
    # Helper function to calculate for a single feature
    calcFeatWithString <- function(dt, featStr) {
        wFeat <- dt[, .(wfeat = mean(get(featStr)), nwins = .N), by = c("Season", "Wteam")]
        featStr2 <- gsub("^W", "L", featStr)
        lFeat <- dt[, .(lfeat = mean(get(featStr2)), nloss = .N), by = c("Season", "Lteam")]
        feat <- merge(wFeat, lFeat, by.x = c("Season", "Wteam"), by.y = c("Season", "Lteam"), all = T)
        feat[is.na(feat)] <- 0
        feat[, ft := (wfeat*nwins +  lfeat*nloss)/ (nwins+nloss)]
        ret <- feat[, .(Season, Wteam, ft)]
        names(ret) <- c("Season", "Team", featStr)
        ret
    }
    dt <- copy(dt)
    dt[, Wloc := NULL]
    statNames <- grep("^W", names(dt), value = T)
    statNames <- statNames[statNames != "Wteam"]
    ret <- dt[, .(Season, Team = Wteam)] %>% unique()
    for(stat in statNames) {
        ret <- merge(ret,
                     calcFeatWithString(dt, stat), by = c("Season", "Team"), all = T)
    }
    return(ret)
}
```

The output is as follows:

```r
seasonStats <- compressStats(regularSeason)
tournamentStats <- compressStats(tournament)
head(seasonStats)
```

```
##    Season Team   Wscore     Wfgm     Wfga    Wfgm3    Wfga3      Wftm
## 1:   2003 1102 57.25000 19.14286 39.78571 7.821429 20.82143 11.142857
## 2:   2003 1103 78.77778 27.14815 55.85185 5.444444 16.07407 19.037037
## 3:   2003 1104 69.28571 24.03571 57.17857 6.357143 19.85714 14.857143
## 4:   2003 1105 71.76923 24.38462 61.61538 7.576923 20.76923 15.423077
## 5:   2003 1106 63.60714 23.42857 55.28571 6.107143 17.64286 10.642857
## 6:   2003 1107 65.92857 24.03571 57.46429 7.928571 22.17857  9.928571
##        Wfta       Wor      Wdr     Wast      Wto     Wstl     Wblk
## 1: 17.10714  4.178571 16.82143 13.00000 11.42857 5.964286 1.785714
## 2: 25.85185  9.777778 19.92593 15.22222 12.62963 7.259259 2.333333
## 3: 20.92857 13.571429 23.92857 12.10714 13.28571 6.607143 3.785714
## 4: 21.84615 13.500000 23.11538 14.53846 18.65385 9.307692 2.076923
## 5: 16.46429 12.285714 23.85714 11.67857 17.03571 8.357143 3.142857
## 6: 13.53571  8.250000 20.25000 11.92857 12.57143 6.857143 2.035714
##         Wpf
## 1: 18.75000
## 2: 19.85185
## 3: 18.03571
## 4: 20.23077
## 5: 18.17857
## 6: 15.89286
```

In addition to the team statistics calculated above, there are others that can be calculated.

Additional Statistics

The following function takes in the season statistics calculated above and modifies statistics in-place. Note that free throw attempts and makes are removed since they have a correlation of 0.9290291, which would not be suited for model training.

```r
addFeatures <- function(dt) {
    createFreeThrowPercentage <- function(dt) {
        dt[, Wftp := Wftm/Wfta]
        dt$Wftm <- dt$Wfta <- NULL
    }
    addSeeds <- function(dt) {
        seeds <- fread("TourneySeeds.csv")
        seeds[, SeedNum := gsub('[a-zA-Z]', '', Seed) %>% as.numeric]
        combined <- merge(dt[, .(Season, Team)],
                          seeds[, .(Season, Team, SeedNum)],
                          all.x = T, by = c("Season", "Team"))
        dt[, Seed := combined$SeedNum]
    }
    createFreeThrowPercentage(dt)
    addSeeds(dt)
    invisible(dt)
}

addFeatures(seasonStats)
addFeatures(tournamentStats)
head(seasonStats)
```

```
##    Season Team   Wscore     Wfgm     Wfga    Wfgm3    Wfga3      Wftm
## 1:   2003 1102 57.25000 19.14286 39.78571 7.821429 20.82143 11.142857
## 2:   2003 1103 78.77778 27.14815 55.85185 5.444444 16.07407 19.037037
## 3:   2003 1104 69.28571 24.03571 57.17857 6.357143 19.85714 14.857143
## 4:   2003 1105 71.76923 24.38462 61.61538 7.576923 20.76923 15.423077
## 5:   2003 1106 63.60714 23.42857 55.28571 6.107143 17.64286 10.642857
## 6:   2003 1107 65.92857 24.03571 57.46429 7.928571 22.17857  9.928571
##        Wfta       Wor      Wdr     Wast      Wto     Wstl     Wblk
## 1: 17.10714  4.178571 16.82143 13.00000 11.42857 5.964286 1.785714
## 2: 25.85185  9.777778 19.92593 15.22222 12.62963 7.259259 2.333333
## 3: 20.92857 13.571429 23.92857 12.10714 13.28571 6.607143 3.785714
## 4: 21.84615 13.500000 23.11538 14.53846 18.65385 9.307692 2.076923
## 5: 16.46429 12.285714 23.85714 11.67857 17.03571 8.357143 3.142857
## 6: 13.53571  8.250000 20.25000 11.92857 12.57143 6.857143 2.035714
##         Wpf      Wftp Seed
## 1: 18.75000 0.6513570   NA
## 2: 19.85185 0.7363897   NA
## 3: 18.03571 0.7098976   10
## 4: 20.23077 0.7059859   NA
## 5: 18.17857 0.6464208   NA
## 6: 15.89286 0.7335092   NA
```

Now that we have the labeled data and the stats for each team, we can combine them with this function:

```
combineLabelsWithStats <- function(labeled, stats) {
    stats <- copy(stats)
    stats[, id := paste(Season, Team, sep = "_")]
    stats$Season <- stats$Team <- NULL
    labeled <- copy(labeled)
    labeled[, team1 := gsub("_[0-9]+$", "", id)]
    labeled[, team2 := gsub("_[0-9]+_", "_", id)]
    ret <- merge(labeled, stats, all.x = T, by.x = "team1", by.y ="id")
    ret <- merge(ret, stats, all.x = T, by.x = "team2", by.y ="id", suffixes = c(".1", ".2"))
    ret$team2 <- ret$team1 <- NULL
    ret
}

labeledSeasonStats <- combineLabelsWithStats(labeledSeason, seasonStats)
labeledTournamentStats <- combineLabelsWithStats(labeledTournament, tournamentStats)
head(labeledSeasonStats)
```

```
##                 id label Wscore.1   Wfgm.1   Wfga.1  Wfgm3.1  Wfga3.1
## 1: 2003_1104_1106     1 69.28571 24.03571 57.17857 6.357143 19.85714
## 2: 2003_1105_1106     0 71.76923 24.38462 61.61538 7.576923 20.76923
## 3: 2003_1105_1106     0 71.76923 24.38462 61.61538 7.576923 20.76923
## 4: 2003_1105_1108     0 71.76923 24.38462 61.61538 7.576923 20.76923
## 5: 2003_1105_1108     0 71.76923 24.38462 61.61538 7.576923 20.76923
## 6: 2003_1106_1108     1 63.60714 23.42857 55.28571 6.107143 17.64286
##      Wftm.1   Wfta.1    Wor.1    Wdr.1   Wast.1    Wto.1   Wstl.1   Wblk.1
## 1: 14.85714 20.92857 13.57143 23.92857 12.10714 13.28571 6.607143 3.785714
## 2: 15.42308 21.84615 13.50000 23.11538 14.53846 18.65385 9.307692 2.076923
## 3: 15.42308 21.84615 13.50000 23.11538 14.53846 18.65385 9.307692 2.076923
## 4: 15.42308 21.84615 13.50000 23.11538 14.53846 18.65385 9.307692 2.076923
## 5: 15.42308 21.84615 13.50000 23.11538 14.53846 18.65385 9.307692 2.076923
## 6: 10.64286 16.46429 12.28571 23.85714 11.67857 17.03571 8.357143 3.142857
##       Wpf.1    Wftp.1 Seed.1 Wscore.2   Wfgm.2   Wfga.2  Wfgm3.2  Wfga3.2
## 1: 18.03571 0.7098976     10 63.60714 23.42857 55.28571 6.107143 17.64286
## 2: 20.23077 0.7059859     NA 63.60714 23.42857 55.28571 6.107143 17.64286
## 3: 20.23077 0.7059859     NA 63.60714 23.42857 55.28571 6.107143 17.64286
## 4: 20.23077 0.7059859     NA 69.09091 24.93939 58.72727 5.212121 16.33333
## 5: 20.23077 0.7059859     NA 69.09091 24.93939 58.72727 5.212121 16.33333
## 6: 18.17857 0.6464208     NA 69.09091 24.93939 58.72727 5.212121 16.33333
##      Wftm.2   Wfta.2    Wor.2    Wdr.2   Wast.2    Wto.2   Wstl.2   Wblk.2
## 1: 10.64286 16.46429 12.28571 23.85714 11.67857 17.03571 8.357143 3.142857
## 2: 10.64286 16.46429 12.28571 23.85714 11.67857 17.03571 8.357143 3.142857
## 3: 10.64286 16.46429 12.28571 23.85714 11.67857 17.03571 8.357143 3.142857
## 4: 14.00000 20.93939 13.12121 23.21212 13.84848 18.45455 8.181818 3.515152
## 5: 14.00000 20.93939 13.12121 23.21212 13.84848 18.45455 8.181818 3.515152
## 6: 14.00000 20.93939 13.12121 23.21212 13.84848 18.45455 8.181818 3.515152
##       Wpf.2    Wftp.2 Seed.2
## 1: 18.17857 0.6464208     NA
## 2: 18.17857 0.6464208     NA
## 3: 18.17857 0.6464208     NA
## 4: 19.66667 0.6685962     NA
## 5: 19.66667 0.6685962     NA
## 6: 19.66667 0.6685962     NA
```

```r
head(labeledTournamentStats)
```

```
##                 id label Wscore.1 Wfgm.1 Wfga.1 Wfgm3.1 Wfga3.1 Wftm.1
## 1: 2003_1140_1163     0    53.00     20  64.00    2.00   17.00     11
## 2: 2003_1141_1166     1    69.50     25  55.00    5.50   12.00     14
## 3: 2003_1141_1181     0    69.50     25  55.00    5.50   12.00     14
## 4: 2003_1161_1181     0    57.00     18  54.00    3.00   11.00     18
## 5: 2003_1112_1211     1    84.75     31  67.75    7.75   20.75     15
## 6: 2003_1153_1211     0    69.00     26  66.00   10.00   27.00      7
##    Wfta.1 Wor.1 Wdr.1 Wast.1 Wto.1 Wstl.1 Wblk.1 Wpf.1    Wftp.1 Seed.1
## 1:  13.00  15.0 26.00  11.00  11.0   8.00    4.0  22.0 0.8461538     12
## 2:  19.50  12.5 19.50  11.50  19.5   8.50    1.5  16.5 0.7179487     11
## 3:  19.50  12.5 19.50  11.50  19.5   8.50    1.5  16.5 0.7179487     11
## 4:  22.00  11.0 24.00   8.00  19.0   5.00    4.0  19.0 0.8181818     14
## 5:  19.25  13.5 30.25  18.75  13.5   9.25    4.5  15.5 0.7792208      1
## 6:  10.00  13.0 22.00  13.00  10.0   7.00    6.0  24.0 0.7000000      8
##    Wscore.2   Wfgm.2   Wfga.2  Wfgm3.2  Wfga3.2 Wftm.2   Wfta.2     Wor.2
## 1: 73.66667 26.00000 63.66667 4.666667 12.33333     17 24.33333 14.333333
## 2: 73.00000 27.00000 60.00000 7.000000 17.00000     12 17.00000 14.000000
## 3: 72.66667 25.33333 57.00000 8.000000 18.00000     14 18.33333  9.333333
## 4: 72.66667 25.33333 57.00000 8.000000 18.00000     14 18.33333  9.333333
## 5: 84.50000 27.00000 62.00000 8.500000 22.00000     22 29.00000 12.000000
## 6: 84.50000 27.00000 62.00000 8.500000 22.00000     22 29.00000 12.000000
##       Wdr.2    Wast.2    Wto.2    Wstl.2   Wblk.2    Wpf.2    Wftp.2
## 1: 27.33333 11.000000 11.33333  4.666667 5.666667 16.66667 0.6986301
## 2: 17.00000 20.000000 21.00000  6.000000 6.000000 21.00000 0.7058824
## 3: 24.00000  9.666667 13.66667 10.333333 8.000000 19.66667 0.7636364
## 4: 24.00000  9.666667 13.66667 10.333333 8.000000 19.66667 0.7636364
## 5: 27.50000 16.000000 11.50000  3.500000 3.500000 20.00000 0.7586207
## 6: 27.50000 16.000000 11.50000  3.500000 3.500000 20.00000 0.7586207
##    Seed.2
## 1:      5
## 2:      6
## 3:      3
## 4:      3
## 5:      9
## 6:      9
```

When preparing a bracket, we do not have any detailed information about the teams' performances in the tournament. We can combine the regular season details with the tournament teams as such:

```r
tournamentWithSeasonStats <- combineLabelsWithStats(labeledTournament, seasonStats)
head(tournamentWithSeasonStats)
```

```
##                 id label Wscore.1   Wfgm.1   Wfga.1  Wfgm3.1  Wfga3.1
## 1: 2003_1140_1163     0 72.45161 24.03226 51.25806 6.193548 16.12903
## 2: 2003_1141_1166     1 79.34483 26.62069 52.68966 6.827586 17.93103
## 3: 2003_1141_1181     0 79.34483 26.62069 52.68966 6.827586 17.93103
## 4: 2003_1161_1181     0 74.00000 26.40000 52.10000 4.200000 11.56667
## 5: 2003_1112_1211     1 85.21429 30.32143 65.71429 7.035714 20.07143
## 6: 2003_1153_1211     0 67.32143 22.89286 56.67857 6.678571 19.50000
##      Wftm.1   Wfta.1    Wor.1    Wdr.1   Wast.1    Wto.1   Wstl.1   Wblk.1
## 1: 18.19355 24.16129 10.87097 24.41935 13.41935 13.74194 6.935484 2.516129
## 2: 19.27586 25.17241 10.58621 23.27586 15.62069 18.24138 7.103448 4.000000
## 3: 19.27586 25.17241 10.58621 23.27586 15.62069 18.24138 7.103448 4.000000
## 4: 17.00000 24.26667 10.80000 23.46667 15.50000 16.13333 5.333333 4.233333
## 5: 17.53571 25.00000 15.17857 27.64286 17.64286 14.78571 8.464286 4.214286
## 6: 14.85714 21.53571 12.14286 23.39286 12.28571 10.60714 5.178571 4.250000
##       Wpf.1    Wftp.1 Seed.1 Wscore.2   Wfgm.2   Wfga.2  Wfgm3.2  Wfga3.2
## 1: 21.41935 0.7530040     12 80.03333 29.53333 62.20000 6.066667 15.70000
## 2: 20.96552 0.7657534     11 79.24242 28.69697 57.45455 7.969697 20.48485
## 3: 20.96552 0.7657534     11 81.96667 27.36667 60.33333 7.333333 20.60000
## 4: 20.56667 0.7005495     14 81.96667 27.36667 60.33333 7.333333 20.60000
## 5: 17.75000 0.7014286      1 77.06452 26.06452 55.45161 7.161290 19.06452
## 6: 18.96429 0.6898839      8 77.06452 26.06452 55.45161 7.161290 19.06452
##      Wftm.2   Wfta.2    Wor.2    Wdr.2   Wast.2    Wto.2   Wstl.2   Wblk.2
## 1: 14.90000 22.10000 14.76667 27.90000 15.63333 15.80000 5.933333 7.733333
## 2: 13.87879 20.03030 10.87879 23.18182 16.81818 13.36364 8.393939 4.454545
## 3: 19.90000 28.06667 13.76667 23.10000 13.83333 14.03333 8.500000 5.133333
## 4: 19.90000 28.06667 13.76667 23.10000 13.83333 14.03333 8.500000 5.133333
## 5: 17.77419 24.64516 11.93548 25.32258 15.74194 14.54839 6.806452 3.516129
## 6: 17.77419 24.64516 11.93548 25.32258 15.74194 14.54839 6.806452 3.516129
##       Wpf.2    Wftp.2 Seed.2
## 1: 18.40000 0.6742081      5
## 2: 17.27273 0.6928896      6
## 3: 21.26667 0.7090261      3
## 4: 21.26667 0.7090261      3
## 5: 18.64516 0.7212042      9
## 6: 18.64516 0.7212042      9
```

Analyzing the Time Series

To look at the points scored per game while controlling for the opposing team's defense, we create an adjusted score by dividing by the mean of the team's points per game and the opponent's allowed points per game:

$$\frac{Score}{\frac{PPG1+OPPG2}{2}}$$

Note: It might be better to look at the rolling means of points per game and opponent's allowed points per game when adjusting the score.

```r
getTidySeason <- function(regSeason) {
    # Get ppg and oppg for each team
    calcFeat <- function(dt, featCalc, featName = "feat") {
        # Get arguments
        arguments <- as.list(match.call())
        # Calculate number of wins and mean of arg when winning for each team
        wFeat <- dt[, .(wfeat = mean(eval(arguments$featCalc, dt)), nwins = .N), by = c("Season", "Wtea

        # Change the table so names that start with W start with L and vice versa
        dtc <- copy(dt)
        names(dtc) <- gsub("^W", "l", names(dtc))
        names(dtc) <- gsub("^L", "W", names(dtc))
        names(dtc) <- gsub("^l", "L", names(dtc))

        # Calculate number of losses and mean of arg when losing for each team
        lFeat <- dtc[, .(lfeat = mean(eval(arguments$featCalc, dtc)), nloss = .N), by = c("Season", "Wt

        # Combine tables for wins and losses
        feat <- merge(wFeat, lFeat, by = c("Season", "Wteam"), all = T)
        feat[is.na(feat)] <- 0
        feat[, ft := (wfeat*nwins +  lfeat*nloss)/ (nwins+nloss)]
        names(feat) <- c("Season", "Team", paste0("w.", featName), "nwins", paste0("l.", featName), "nl
        feat
    }

    ppg <- calcFeat(regSeason, Wscore, "ppg")[, .(Season, Team, ppg)]
    oppg <- calcFeat(regSeason, Lscore, "oppg")[, .(Season, Team, oppg)]

    wppg <- copy(ppg)
    names(wppg) <- c("Season", "Wteam", "Wppg")
    seasonWithppg <- merge(regSeason, wppg, by = c("Season", "Wteam"))
    lppg <- wppg
    names(lppg) <- c("Season", "Lteam", "Lppg")
    seasonWithppg <- merge(seasonWithppg, lppg, by = c("Season", "Lteam"))

    woppg <- copy(oppg)
    names(woppg) <- c("Season", "Wteam", "Woppg")
    seasonWithppg <- merge(seasonWithppg, woppg, by = c("Season", "Wteam"))
    loppg <- woppg
    names(loppg) <- c("Season", "Lteam", "Loppg")
    seasonWithppg <- merge(seasonWithppg, loppg, by = c("Season", "Lteam"))

    # Adjust teams score based on ppg
    seasonWithppg[, Wascore := 2 * Wscore / (Wppg + Loppg)]
    seasonWithppg[, Lascore := 2 * Lscore / (Woppg + Lppg)]

    # Tidy up the dataset
    tidySeason <- seasonWithppg[, .(Season, Team = Lteam, Daynum, Ascore = Lascore)] %>%
        rbind(seasonWithppg[, .(Season, Team = Wteam, Daynum, Ascore = Wascore)]) %>%
        arrange(Season, Team, Daynum) %>% data.table()

    tidySeason
}

scoreByDay <- getTidySeason(regularSeason)
head(scoreByDay)
```

```
##    Season Team Daynum    Ascore
## 1:   2003 1102     19 0.7477131
## 2:   2003 1102     22 1.0413016
## 3:   2003 1102     25 0.8972302
## 4:   2003 1102     27 0.7395049
## 5:   2003 1102     31 1.0290760
## 6:   2003 1102     34 1.0371848
```

We can develop a model