Game of Thrones Analysis

Game of Thrones is an HBO action-adventure-drama series adapted from George R. R. Martin's *A Song of Ice and Fire* fantasy book series. The television series aired for eight seasons from 2011 to 2019, and consisted of $N = 73$ episodes. The file `GOT.csv` contains information about each of these episodes. The table below describes the 12 variates recorded for each episode.

Variate	Description
`Season`	the season number of the episode (1,2. . . ,8)
`Episode_Number`	the episode's number in the series (1,2,. . . ,73)
`Number_in_Season`	the episode's number in the season
`Episode_Name`	the episode's name
`Director`	the episode's director
`Writer`	the episode's writer
`Original_Air_Date`	the episode's original air date
`US_Viewers`	the episode's number of US cable viewers (in millions)
`Runtime`	the duration of the episode (in minutes)
`IMDb_Description`	the episode's description as it appears on IMDb
`IMDb_Rating`	the episode's rating [1,10] as determined by IMDb registered voters
`Notable_Death_Count`	the number of notable characters who died in the episode

```r
### First, we read in the 'GOT' .csv file
got_csv <- read.csv(file = "GOT.csv", header = TRUE)
colnames(got_csv)[1] <- "Season"
```

Determining which directors directed the five most highly rated episodes,

```r
got_csv$Director[which(got_csv$IMDb_Rating >= sort(x = got_csv$IMDb_Rating, decreasing = TRUE)[5])]
```

```
## [1] "David Nutter"     "Miguel Sapochnik" "Miguel Sapochnik" "Miguel Sapochnik"
## [5] "Matt Shakman"
```

∴ We see that:

- David Nutter directed the first most highly rated episode
- Miguel Sapochnik directed the second, third and fourth most highly rated episodes
- Matt Shakman directed the fifth most highly rated episode.

Determining the names of the highest and lowest rated episodes,

```
# highest rated episodes
got_csv$Episode_Name[which(got_csv$IMDb_Rating == max(got_csv$IMDb_Rating))]
```

```
## [1] "The Rains of Castamere" "Hardhome"               "Battle of the Bastards"
## [4] "The Winds of Winter"
```

```
# lowest rated episodes
got_csv$Episode_Name[which(got_csv$IMDb_Rating == min(got_csv$IMDb_Rating))]
```

```
## [1] "The Iron Throne"
```

∴ We see that:

The names of the highest rate episodes (9.9 rating) are:

- The Rains of Castamere
- Hardhome
- Battle of the Bastards
- The Winds of Winter

The names of the lowest rate episode (4.1 rating) is:

- The Iron Throne

Calculating and reporting the average IMDb_Rating for each unique writer-director pairing. Which pairing produces the highest rated episodes, on average? Which pairing produces the lowest rated episodes, on average?

```r
# here, we report the average `IMDb_Rating` for each unique writer-director pairing
# using the aggregate function in R
temp<- aggregate(x=got_csv$IMDb_Rating,
                 by=list(Writer=got_csv$Writer, Director=got_csv$Director),
                 FUN=mean)
temp
```

```
##                            Writer                    Director        x
## 1  David BenioffÂ &Â D. B. Weiss                 Alan Taylor 9.157143
## 2                    Bryan Cogman                 Alex Graves 9.000000
## 3  David BenioffÂ &Â D. B. Weiss                 Alex Graves 9.475000
## 4             George R. R. Martin                 Alex Graves 9.700000
## 5                    Bryan Cogman               Alik Sakharov 9.300000
## 6  David BenioffÂ &Â D. B. Weiss               Alik Sakharov 9.050000
## 7                    Bryan Cogman                  Brian Kirk 8.800000
## 8  David BenioffÂ &Â D. B. Weiss                  Brian Kirk 8.900000
## 9  David BenioffÂ &Â D. B. Weiss                 D. B. Weiss 9.100000
## 10 David BenioffÂ &Â D. B. Weiss              Daniel Minahan 9.133333
## 11            George R. R. Martin              Daniel Minahan 9.100000
## 12                 Vanessa Taylor              Daniel Minahan 8.700000
## 13 David BenioffÂ &Â D. B. Weiss             Daniel Sackheim 9.000000
## 14 David BenioffÂ &Â D. B. Weiss               David Benioff 8.900000
## 15 David BenioffÂ &Â D. B. Weiss David Benioff & D. B. Weiss 4.100000
## 16                   Bryan Cogman                David Nutter 7.900000
## 17                      Dave Hill                David Nutter 7.600000
## 18 David BenioffÂ &Â D. B. Weiss                David Nutter 8.700000
## 19                 Vanessa Taylor                David Nutter 9.100000
## 20 David BenioffÂ &Â D. B. Weiss              David Petrarca 8.900000
## 21                 Vanessa Taylor              David Petrarca 8.900000
## 22                   Bryan Cogman                 Jack Bender 8.500000
## 23 David BenioffÂ &Â D. B. Weiss                 Jack Bender 9.700000
## 24                   Bryan Cogman              Jeremy Podeswa 8.400000
## 25                      Dave Hill              Jeremy Podeswa 9.500000
## 26 David BenioffÂ &Â D. B. Weiss              Jeremy Podeswa 8.966667
## 27                   Bryan Cogman                  Mark Mylod 8.850000
## 28                      Dave Hill                  Mark Mylod 8.800000
## 29 David BenioffÂ &Â D. B. Weiss                  Mark Mylod 8.766667
## 30                      Dave Hill                Matt Shakman 9.000000
## 31 David BenioffÂ &Â D. B. Weiss                Matt Shakman 9.800000
## 32 David BenioffÂ &Â D. B. Weiss              Michael Slovis 8.600000
## 33                   Bryan Cogman           Michelle MacLaren 8.900000
## 34 David BenioffÂ &Â D. B. Weiss           Michelle MacLaren 8.950000
## 35            George R. R. Martin           Michelle MacLaren 8.800000
## 36 David BenioffÂ &Â D. B. Weiss            Miguel Sapochnik 8.716667
## 37 David BenioffÂ &Â D. B. Weiss               Neil Marshall 9.600000
## 38            George R. R. Martin               Neil Marshall 9.700000
## 39 David BenioffÂ &Â D. B. Weiss              Tim Van Patten 8.900000
```

```r
# pairing who produces the highest rated episodes on average
temp$Writer[which(temp$x >= sort(temp$x, decreasing = TRUE)[1])]
```

```
## [1] "David BenioffÂ &Â D. B. Weiss"
```

```r
temp$Director[which(temp$x >= sort(temp$x, decreasing = TRUE)[1])]
```

```
## [1] "Matt Shakman"
```

```r
# pairing who produces the lowest rated episodes on average
temp$Writer[which(temp$x <= sort(temp$x, decreasing = FALSE)[1])]
```

```
## [1] "David BenioffÂ &Â D. B. Weiss"
```

```r
temp$Director[which(temp$x <= sort(temp$x, decreasing = FALSE)[1])]
```

```
## [1] "David Benioff & D. B. Weiss"
```

∴ We see that:

The unique writer-director pairing who produces the **highest** rated episodes on average:

- David Benioff & D. B. Weiss (Writer) and Matt Shakman (Director)

The unique writer-director pairing who produces the **lowest** rated episodes on average:

- David Benioff & D. B. Weiss (Writer) and David Benioff & D. B. Weiss (Director)

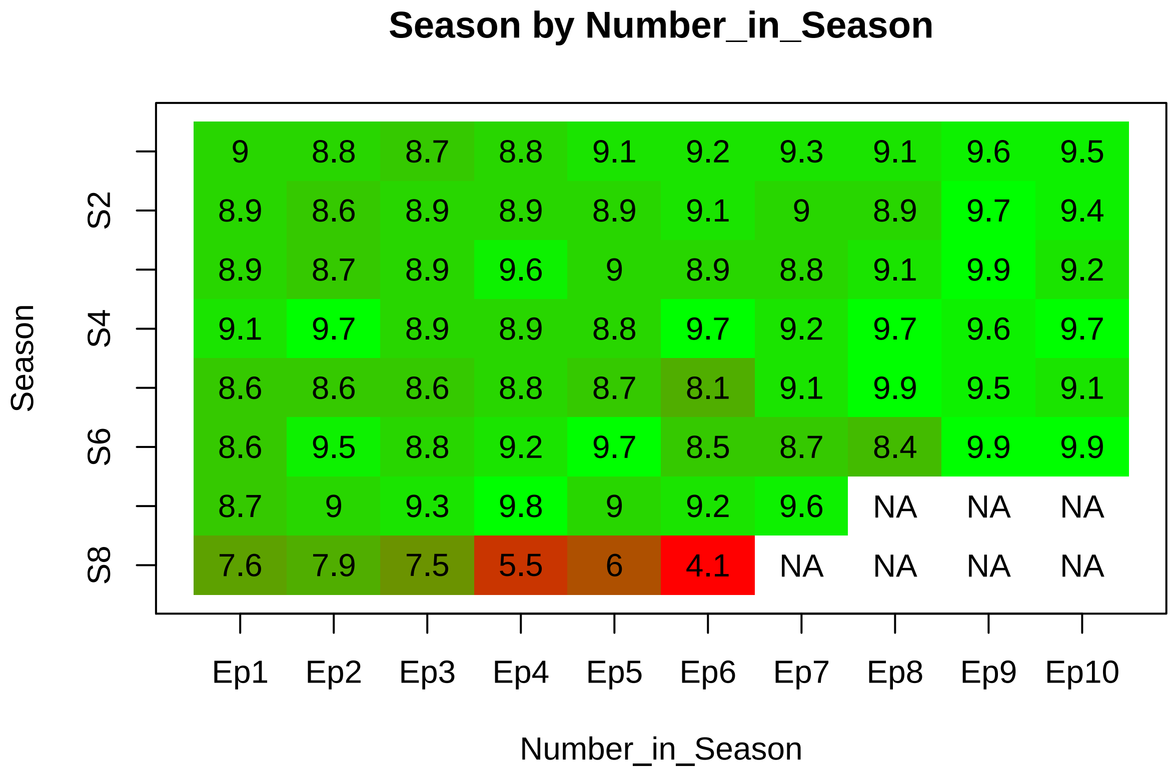
Construting an 8×10 matrix of `IMDb_Rating` values where rows correspond to `Season` and columns correspond to `Number_in_Season` and element (i, j) corresponds to the IMDb rating of episode j within season i,

```r
## make an empty matrix
X <- matrix(data = NA, # set all the entries as NA values
            nrow = 8, ncol = 10,
            byrow = TRUE,
            dimnames = list(c("S1", "S2", "S3", "S4", "S5", "S6", "S7", "S8"),
                            c("Ep1", "Ep2", "Ep3", "Ep4", "Ep5", "Ep6", "Ep7",
                              "Ep8", "Ep9", "Ep10")))

## set up some constants for brevity
n = nrow(X)
m = ncol(X)

## filling in the corresponding IMDb Ratings in the appropriate entries
for (i in 1:n) {
  for (j in 1:m) {
    # make a condition that the command doesn't run if the episode does not exist
    # e.g. Season 8 Episode 7 does not exist, so the command below shouldn't run
    # at that iteration
    if(length(which(got_csv$Season == i & got_csv$Number_in_Season == j)) > 0) {
      X[i,j] = got_csv$IMDb_Rating[which(got_csv$Season == i & got_csv$Number_in_Season == j)]
    }
  }
}
## outputting matrix
X
```

```
##    Ep1 Ep2 Ep3 Ep4 Ep5 Ep6 Ep7 Ep8 Ep9 Ep10
## S1 9.0 8.8 8.7 8.8 9.1 9.2 9.3 9.1 9.6  9.5
## S2 8.9 8.6 8.9 8.9 8.9 9.1 9.0 8.9 9.7  9.4
## S3 8.9 8.7 8.9 9.6 9.0 8.9 8.8 9.1 9.9  9.2
## S4 9.1 9.7 8.9 8.9 8.8 9.7 9.2 9.7 9.6  9.7
## S5 8.6 8.6 8.6 8.8 8.7 8.1 9.1 9.9 9.5  9.1
## S6 8.6 9.5 8.8 9.2 9.7 8.5 8.7 8.4 9.9  9.9
## S7 8.7 9.0 9.3 9.8 9.0 9.2 9.6  NA  NA   NA
## S8 7.6 7.9 7.5 5.5 6.0 4.1  NA  NA  NA   NA
```

Making a heatmap,

```r
### make.heatmap() function
### our matrx function will denote our n x m matrix input
make.heatmap <- function(matrx) {
    # first, we save the min and max values in the matrix input
    max_matrx <- max(matrx, na.rm = TRUE)
    min_matrx <- min(matrx, na.rm = TRUE)
    # initialize a variable for the number or rows and columns in the matrix
    n <- nrow(matrx)
    m <- ncol(matrx)
    # next, we write code to make a blank canvas as was demonstrated in
    # the week 2 tutorial video
    plot(NA, main = "",
     # notice that here, we set the x and y limits to be the number of rows
     # and columns of the input matrix respectively
     xlim = c(0,m), xlab = "", xaxt = "n",
     ylim = c(0,n), ylab = "", yaxt = "n")
    # plotting the axis ONLY IF there is one
    if(!is.null(rownames(matrx))){
        # (minus 0.5 in the axis for centering and
        # reversed the rownames for correct ordered output)
        axis(side = 2, at = 1:n - 0.5, labels = rev((rownames(matrx))))
    }
    if(!is.null(colnames(matrx))){
        axis(side = 1, at = 1:m - 0.5, labels = colnames(matrx))
    }

    # first, we set up the bin intervals (1 to 20 intervals)
    # length.out = 21 as we want 20 intervals
    bin_interval <- seq(min_matrx,max_matrx,length.out = 21)

    # plotting rectangles using for loops
    for (i in 1:n) {
        for (j in 1:m) {
            # set up the gradation of hues between red and gree
            colour_fun <- colorRampPalette(colors = c("red", "green"))
            # set up the number of entries in the colour_grad vector
            # in Slack, the professor mentioned 1 to 20
            colour_grad <- colour_fun(n = 20)
            # consider if the value is NA or not for the colours
            if(is.na(matrx[i,j])){
                colour = "white"
            } else {
                # finding which bin should we allocate for the value of
                # the index in the matrix
                for (l in 1:(length(bin_interval) - 1)){
                    if(matrx[i,j] == max_matrx) {
                        num = 20
                    } else if(matrx[i,j] >= bin_interval[l]
                       & matrx[i,j] < bin_interval[l+1]){
                        num = l
                    }
                }
                # set the colour if the value is not NA according to the
                # bin allocation loop above
                colour = colour_grad[num]
            }
            # after configurations, plot the polygon at the appropriate position
            polygon(x = c(j-1,j-1,j,j),
                    y = c(n-i+1,n-i,n-i,n-i+1),
                    col = colour,
                    border = FALSE)
            # label for displaying value of the (i,j)th element of the
            # input matrix
            if(is.na(matrx[i,j])){
                # if the value is NA, print out the "NA" string as the display label
                label = "NA"
            } else {
                label = matrx[i,j]
            }
            text(x = j-0.5, y = nrow(matrx)-i+0.5, labels = label)
        }
    }
}

### Pass in our matrix X from part d to produce the output
make.heatmap(X)
### Finally, we use the title function to plot an informative title
title(ylab="Season", xlab="Number_in_Season", main="Season by Number_in_Season")
```



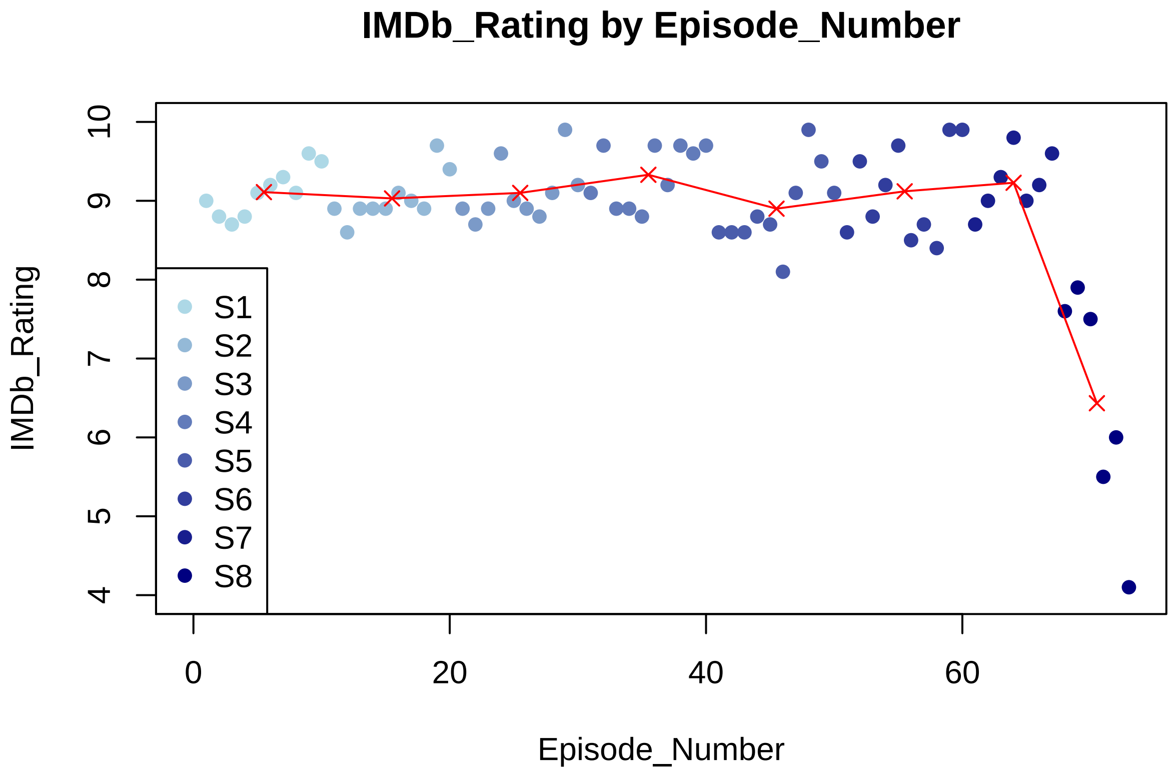
Based on the plot above, we can discuss some of the insights that we draw from the plot above.

Firstly, it is clear that Season 8 of Game of Thrones didn't do very well overall in terms of IMDb ratings. Each episode in Season 8 has been consistently poor as shown by the graph above. Notice that the 6 episodes in Season 8 are actually the worst 6 episodes in all of the GOT seasons based on IMDb Ratings, particularly the last episode which had the lowest IMDb Rating (4.1).

Next, apart from Season 8, every other season in Game of Thrones did very well based on their IMDb Ratings. Particularly, Season 6 did very well towards the last 2 episodes in the season (which happens to be among the select few episodes which have the highest IMDb Rating, 9.9, among the GOT franchise).

Constructing a scatterplot of `IMDb_Rating` vs. `Episode_Number`,

```r
### first, we plot the colour palette for the episode’s season
### WLOG, I choose my palette colours to be between lightblue and navyblue
colour_fun <- colorRampPalette(colors = c("lightblue", "navyblue"))
colour_grad <- colour_fun(n = 8) # 8 GOT seasons

### plot the blank baseline plot
plot(NA, main = "",
     # notice that here, we set the x and y limits to be the number of rows
     # and columns of the input matrix respectively
     xlim = c(0,73), xlab = "",
     ylim = c(4,10), ylab = "")
     # set the ylim to 4 to 10 as the lowest IMDb rating is 4.1
     # and the highest is 9.9, so we could ignore IMDb ratings from 0 to 3

### now, to plot the average episode rating in each season, we use the aggregate
### function (cbind variation) to calculate the average episode rating in each
### season and store it in the temporary table
temp_table <- aggregate(cbind(Episode_Number, IMDb_Rating) ~ Season,
                        data = got_csv, FUN = mean)

### next, we plot the scatterplot of IMDb_Rating vs Episode_Number with pch=16
for(i in 1:8){
  # plot points
  points(x = got_csv$Episode_Number[got_csv$Season == i],
         y = got_csv$IMDb_Rating[got_csv$Season == i],
         col = colour_grad[i],
         pch = 16)
  # plot average points with red crosses for each Season
  points(x = temp_table$Episode_Number[temp_table$Season == i],
         y = temp_table$IMDb_Rating[temp_table$Season == i],
         col = "red",
         pch = 4)
}

### now, to plot the red line across the red 'x' marks in the scatterplot
lines(x = temp_table$Episode_Number,
      y = temp_table$IMDb_Rating,
      col = "red")

### Finally, we now add a legend, axis labels and an informative title to the plot
legend(x="bottomleft", legend=c("S1", "S2", "S3", "S4", "S5", "S6", "S7", "S8"),
       col=colour_grad, pch=16)
title(ylab="IMDb_Rating", xlab="Episode_Number", main="IMDb_Rating by Episode_Number")
```



Trends observed in the plot above:

Firstly, we can see that across Season 1 to Season 7 of Game of Thrones, the average IMDb ratings per episode appear fairly constant, not varying by a very large margin. They appear to stay the 8 to 9.9 IMDb ratings range, which would mean that the episodes in Season 1 to 7 are a good hit among viewers.

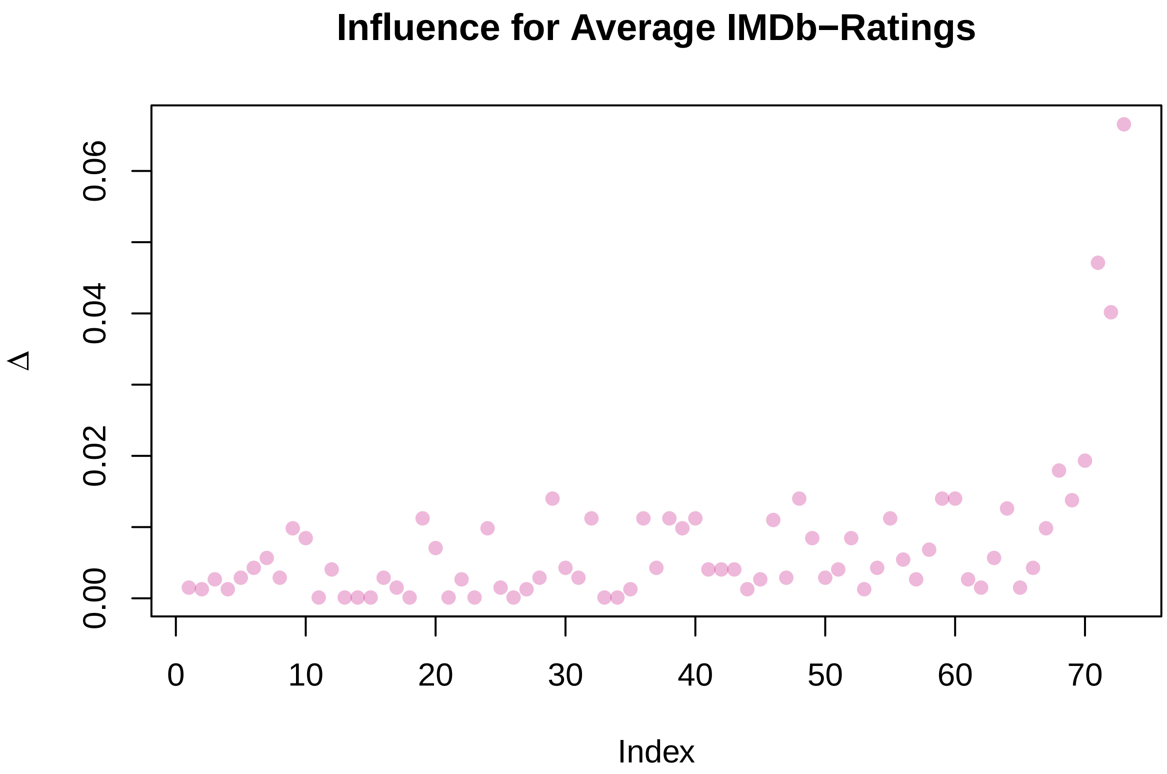
Secondly, we observe that the average IMDb Ratings per episode in Season 8 drops drastically after each episode. Specifically, Season 8 saw a huge decline in average IMDb ratings per episode, which would suggest that the last season of GOT was disliked by a majority of viewers. GOT viewers especially dislike the finale of GOT due to it having the lowest IMDb Rating in the plot.

Let y denote the IMDb rating (`IMDb_Rating`) of an episode, and let $a(\mathcal{P}) = \overline{y}$ be the attribute of interest. Define the influence of episode u on $a(\mathcal{P})$ to be:

$$\Delta(a,u) = |a(y_1,\ldots,y_{u-1},y_u,y_{u+1},\ldots,y_N) - a(y_1,\ldots,y_{u-1},y_{u+1},\ldots,y_N)|$$

Here I construct an influence plot of Δ vs. observation number and identify the episode with the largest influence on the average `IMDb_Rating` attribute.

```
y <- got_csv$IMDb_Rating

delta <- abs((y - mean(y))/(length(y) - 1))
### plot the influence plot
plot(delta,
     main="Influence for Average IMDb-Ratings", ylab=bquote(Delta),
     pch=16, col=adjustcolor("mediumvioletred", 0.3))
```



```
### Now, identifying which episode has the largest influence
got_csv$Episode_Name[which(delta > 0.06)]
```

```
## [1] "The Iron Throne"
```

```
got_csv$Episode_Number[which(delta > 0.06)]
```

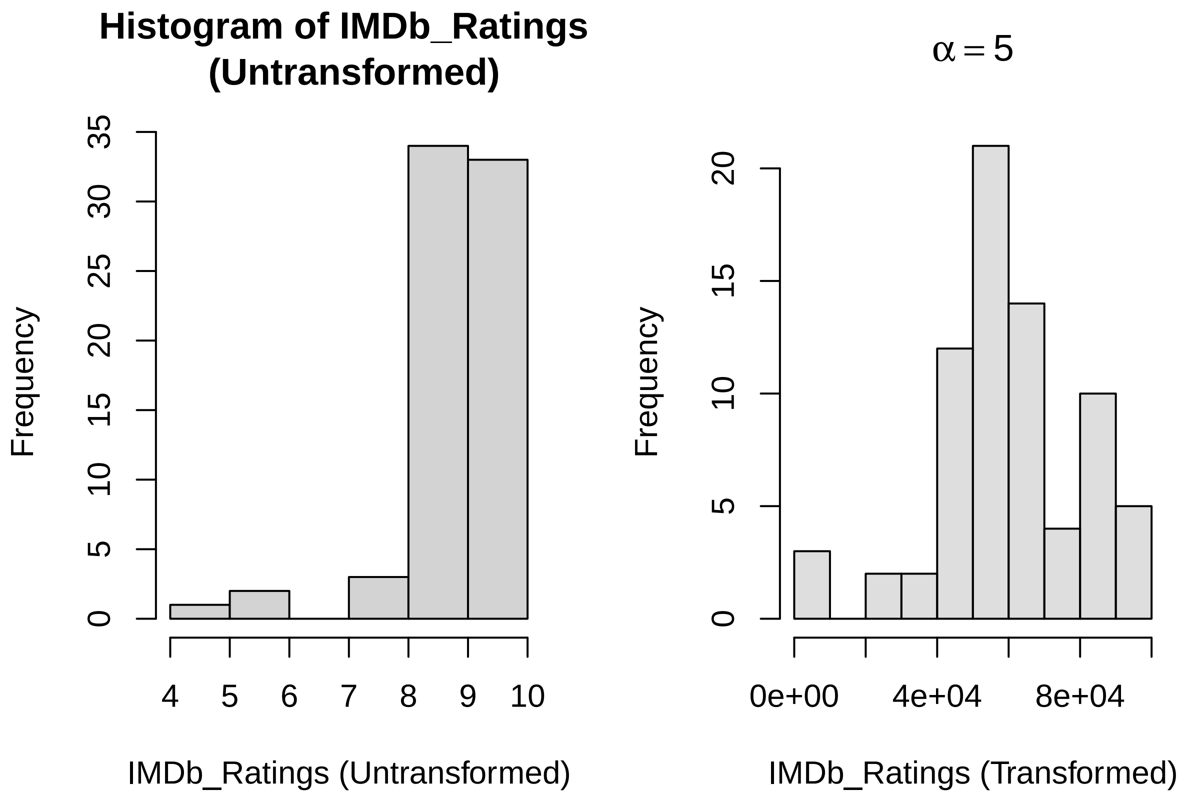
```
## [1] 73
```

Here, I will provide a rationale for why this particular episode is more influential than all of the others:

First of all, the episode with the highest influence is none other than ‘The Iron Throne’, the final episode of GOT. Logically, the ending to every TV Show is bound to leave more of an impression of the show compared to all other episodes. Moreover, not only was ‘The Iron Throne’ the final episode of the GOT series, it is to most viewers, not a very satisfying nor complete way to wrap up to show, which stirred up a lot of debate among the GOT community. Hence, this is why ‘The Iron Throne’ is more influential than any other episode.

Determining (and stating) a power α that makes the `IMDb_Rating` distribution more symmetric. Next, I construct a 1×2 plot which contains histograms of the untransformed ratings and the transformed ratings using what I feel is the best value of α.

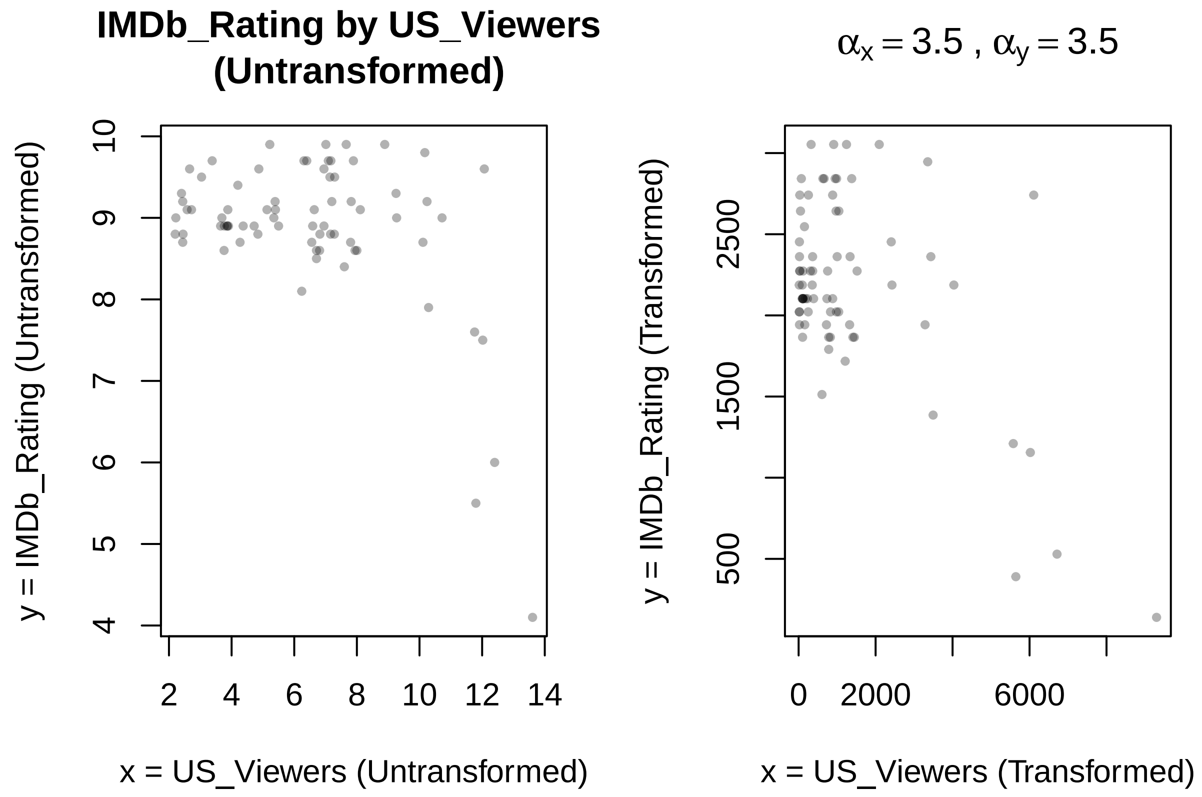
```r
powerfun <- function(x, alpha){
  if(sum(x <= 0) > 0) stop ("x must be positive")
  if(alpha == 0)
    log(x)
  else if (alpha > 0) {
    x^alpha
  } else -x^alpha
}

## By bump rule 1, since the untransformed histogram is leftly-skewed,
## we move higher on the ladder

## I used a method of trial and error
# par(mfrow = c(3,3))
# a = seq(3, 7, length.out = 9)
# for(i in 1:9){
# hist(powerfun(got_csv$IMDb_Rating, a[i]), col=adjustcolor("grey", alpha=0.5),
#      main=bquote(alpha == .(a[i])), xlab="")
# }

# From the transformation steps above, if alpha = 5, it really makes
# the IMDb_Rating attribute seem more symmetric
# Now, we construct a 1x2 plot which contains histograms of the
#  untransformed and transformed ratings (when alpha = 5)
par(mfrow = c(1,2))
# untransformed histogram
hist(got_csv$IMDb_Rating,
     xlab="IMDb_Ratings (Untransformed)",
     main="Histogram of IMDb_Ratings \n (Untransformed)")
# transformed histogram
# we don't have 0's in our dataset, so we do not need to do + 1 in our powerfun function
hist(powerfun(got_csv$IMDb_Rating, 5), col=adjustcolor("grey", alpha=0.5),
     main=bquote(alpha == .(5)), xlab="IMDb_Ratings (Transformed)")
```


Histogram of IMDb_Ratings (Untransformed)

$\alpha = 5$

Determining (and stating) powers α_x and α_y that “straighten” the scatterplot of `IMDb_Rating` vs. `US_Viewers`. Next, I construct a `IMDb_Rating` vs. `US_Viewers` scatterplot using what I feel are the *best* values of α_x and α_y.

```r
par(mfrow = c(1,2))
### First, we plot the untransformed scatterplot of IMDb_Rating vs US_Viewers

# untransformed scatterplot
plot(got_csv$US_Viewers, got_csv$IMDb_Rating, pch = 19, cex = 0.5,
     col = adjustcolor("black", alpha=0.3),
     xlab="x = US_Viewers (Untransformed)",
     ylab="y = IMDb_Rating (Untransformed)",
     main="IMDb_Rating by US_Viewers \n (Untransformed)")

# transformations (commented as not required to show transformations steps)
# By bump rule 2, since the untransformed scatterplot seems to be in the
# first quadrant of the circle, we go up on x and up on y
# we don't have 0's in our dataset, so we do not need to do + 1 in our powerfun function

# transformed scatterplot
plot(powerfun(got_csv$US_Viewers, 3.5), powerfun(got_csv$IMDb_Rating, 3.5),
     pch =19, cex=0.5 ,col=adjustcolor("black",alpha =0.3),
     xlab ="x = US_Viewers (Transformed)",ylab ="y = IMDb_Rating (Transformed)",
     main=bquote(alpha[x]==3.5~","~alpha[y]==3.5))
```


IMDb_Rating by US_Viewers
(Untransformed)
y = IMDb_Rating (Untransformed)
x = US_Viewers (Untransformed)
$\alpha_x = 3.5$, $\alpha_y = 3.5$
y = IMDb_Rating (Transformed)
x = US_Viewers (Transformed)