Project1

Umais Siddiqui

September 16, 2017

Analyzing Chess Tournament data

Rubs:

http://rpubs.com/umais/data607_Project1

GitHub:

https://github.com/umais/DATA-607/tree/master/Project1

Requirements

In this project, you're given a text file with chess tournament results where the information has some structure. Your job is to create an R Markdown file that generates a .CSV file (that could for example be imported into a SQL database) with the following information for all of the players:

Player's Name, Player's State, Total Number of Points, Player's Pre-Rating, and Average Pre Chess Rating of Opponents For the first player, the information would be:

Gary Hua, ON, 6.0, 1794, 1605

Given

A text file by the name of "tournamentinfo.txt". This tet file will contain all of the information we need to get our information.

Approach

We have to read the file clean the data, calculate the average pre ratings of the opponents and then write the results to a file.

Solution

1) Loading the Required Libraries.

We are loading the required libraries. Note that one of the libraries that I will be using is the sqldf library. The purpose of using sqldf library is so that I can use sql like queries to aggregate and construct the result set that I need at the end.

```
library(stringr)
library(sqldf)
```

```
## Loading required package: gsubfn
```

```
## Loading required package: proto
```

```
## Loading required package: RSQLite
## Loading required package: DBI
```

2) Setting up the column names for the tournament Data

In this step we are setting up the columns for the Data Frame that will be read from the tournament file.

```
tournament_fields <- c("Number", "Name", "Points", "Round1", "Round2", "Round3", "Round4", "Round5", "R
```

3) Reading the text file

In this step we will be reading the tournamentinfo.txt file and setting the header parameter to false since the file does not have a header . We are also going to skip 4 lines before reading the data because the data we are interested in starts after skipping four lines. The columns are seperated by a pipe '|' character that is why we supplied that as the seperator.

```
tournament <- read.table("tournamentinfo.txt", header = FALSE, skip = 4, sep = "|", fill = TRUE, string
```

4) Scrubbing file and removing the unneccesary data.

In this step I am cleaing up the data so that I can just get the data I need for my analysis. As you can observe below I have taken two data frames tournament and PlayerGames.The tournament dataframe contains rows of different record types meaning that there is a row type 'A' which has the information for all the rounds and then there is row type 'B' which contains information about the players rating,state and other information. I am constructing a PlayerGames dataframe by just extracting the information about the rounds so that I can use it in my analysis further in the project.

```
tournament <- subset(tournament, !Name == "", select = c(Number:Round7))
PlayersGames<-subset(tournament, grepl("[:0-9:]+", Number))
PlayersGames$Id<-PlayersGames$Number

head(tournament)
```

```
##   Number                                Name Points Round1 Round2 Round3
## 1      1   GARY HUA                          6.0    W  39  W  21  W  18
## 2     ON   15445895 / R: 1794   ->1817       N:2    W      B      W
## 4      2   DAKSHESH DARURI                   6.0    W  63  W  58  L   4
## 5     MI   14598900 / R: 1553   ->1663       N:2    B      W      B
## 7      3   ADITYA BAJAJ                      6.0    L   8  W  61  W  25
## 8     MI   14959604 / R: 1384   ->1640       N:2    W      B      W
##   Round4 Round5 Round6 Round7
## 1  W  14  W   7  D  12  D   4
## 2  B      W      B      W
## 4  W  17  W  16  W  20  W   7
## 5  W      B      W      B
## 7  W  21  W  11  W  13  W  12
## 8  B      W      B      W
```

5) Creating a Function getRoundsData

getRoundsData is a function being defined to return the opponent's id for that round for all the players. I will be using this function later on to get a clean list of the id's for the players and their opponents. This will

be helpful for analysis later on.

```
getRoundsData=function(games)
{
  Round=c()
for(i in 1:length(games))
{
  val=games[[i]][1]
  if(is.na(val))
    val=0
    Round[i]=val

}
  return(Round)
}
```

6) Constructing the data frame Opponents

Using the function getRoundsData to populate the Opponents data frame.The purpose of Opponents data frame is to store the ratings for all the opponents of a player in all the rounds. This data frame will contain the player id as any times as the rounds that they played. In each row there will be the opponent Id and the pre rating for the opponent.

```
df= data.frame(PlayersGames$Id,getRoundsData(str_extract_all(PlayersGames$Round1, "[0-9]+")))
colnames(df)<-c("Id","OpponentsRounds")

df2 = data.frame(PlayersGames$Id,getRoundsData(str_extract_all(PlayersGames$Round2, "[0-9]+")))
colnames(df2)<-c("Id","OpponentsRounds")

df3 = data.frame(PlayersGames$Id,getRoundsData(str_extract_all(PlayersGames$Round3, "[0-9]+")))
colnames(df3)<-c("Id","OpponentsRounds")

df4 = data.frame(PlayersGames$Id,getRoundsData(str_extract_all(PlayersGames$Round4, "[0-9]+")))
colnames(df4)<-c("Id","OpponentsRounds")

df5 = data.frame(PlayersGames$Id,getRoundsData(str_extract_all(PlayersGames$Round5, "[0-9]+")))
colnames(df5)<-c("Id","OpponentsRounds")

df6 = data.frame(PlayersGames$Id,getRoundsData(str_extract_all(PlayersGames$Round6, "[0-9]+")))
colnames(df6)<-c("Id","OpponentsRounds")

df7 = data.frame(PlayersGames$Id,getRoundsData(str_extract_all(PlayersGames$Round7, "[0-9]+")))
colnames(df7)<-c("Id","OpponentsRounds")

Opponents=rbind(df,df2,df3,df4,df5,df6,df7)
head(Opponents)
```

```
##       Id OpponentsRounds
## 1     1               39
## 2     2               63
## 3     3                8
## 4     4               23
## 5     5               45
## 6     6               34
```

7) Getting the ratings data

We are now extracting the record type 'B' which contains the players rating information. We will use this in the final results to get the pre rating for the player.

```
PlayersRatings<-subset(tournament, grepl("[:A-Z:]+", Number))
PlayersRatings$State=PlayersRatings$Number
PlayersRatings$Id=PlayersGames$Id
head(PlayersRatings)
```

```
##    Number                                Name Points Round1 Round2 Round3
## 2     ON   15445895 / R: 1794   ->1817       N:2    W      B      W
## 5     MI   14598900 / R: 1553   ->1663       N:2    B      W      B
## 8     MI   14959604 / R: 1384   ->1640       N:2    W      B      W
## 11    MI   12616049 / R: 1716   ->1744       N:2    W      B      W
## 14    MI   14601533 / R: 1655   ->1690       N:2    B      W      B
## 17    OH   15055204 / R: 1686   ->1687       N:3    W      B      W
##    Round4 Round5 Round6 Round7  State     Id
## 2   B      W      B      W        ON      1
## 5   W      B      W      B        MI      2
## 8   B      W      B      W        MI      3
## 11  B      W      B      B        MI      4
## 14  W      B      W      B        MI      5
## 17  B      B      W      B        OH      6
```

8) Gathering the results

Using sqldf package to extract the Pre Ratings for each player's opponent for all rounds played and also including the players opponent id .

```
result1=sqldf("SELECT g.Id,r.Id as OppId,cast(trim(substr(r.Name,16,4)) as int) as Pre_Ratings
              FROM Opponents g
              JOIN PlayersRatings r on  cast(g.OpponentsRounds as int)=r.Id  ORDER BY g.Id ")
```

```
## Loading required package: tcltk
```

```
head(result1)
```

```
##       Id  OppId Pre_Ratings
## 1     1     39         1436
## 2     1     21         1563
## 3     1     18         1600
## 4     1     14         1610
## 5     1      7         1649
## 6     1     12         1663
```

9) Final Results

In this step we are aggregating all pre_ratings for all the player's opponents from result1 and storing that in avgRate data frame. Then we Join the PlayerGames data frame to the PlayerRatings data frame to get the Pre Rating for the player and then we join the results of that to the avgRate data frame to get the average.

```
avgRate=sqldf("SELECT Id ,cast(ROUND(AVG(Pre_Ratings)) as int) as average_opponent FROM result1 GROUP b
#avgRate
results <- sqldf("SELECT g.Name,r.State,g.Points,substr(r.Name,16,4) as Pre_Ratings,a.average_opponent
```

```
              FROM PlayersGames g
             JOIN PlayersRatings r on  g.Id=r.Id
              JOIN avgRate  a on g.Id=a.Id
                 ")
head(results)
```

```
##                                Name  State Points Pre_Ratings
## 1  GARY HUA                           ON   6.0          1794
## 2  DAKSHESH DARURI                    MI   6.0          1553
## 3  ADITYA BAJAJ                       MI   6.0          1384
## 4  PATRICK H SCHILLING                MI   5.5          1716
## 5  HANSHI ZUO                         MI   5.5          1655
## 6  HANSEN SONG                        OH   5.0          1686
##   average_opponent
## 1             1605
## 2             1469
## 3             1564
## 4             1574
## 5             1501
## 6             1519
```

10) Writing the Results to the File

Finally the results data frame contains the data we need to write to the file. In this final step we will be writing these results in to a csv file called “FinalResults.csv”"

```
write.table(results, file = "FinalResults.csv", sep = ",", row.names = FALSE)
```