Project 1 for 607

Ethan Haley

2/21/2021

Transform a text representation of a chess tourney and extract info we want

```
# Check the format of the input file
tourney <- read.csv("../Project1/tournamentinfo.txt", nrows = 7)
tourney
```

```
##   X.....................................................................................
## 1  Pair | Player Name                     |Total|Round|Round|Round|Round|Round|Round|Round|
## 2  Num  | USCF ID / Rtg (Pre->Post)       | Pts |  1  |  2  |  3  |  4  |  5  |  6  |  7  |
## 3  ---------------------------------------------------------------------------------------
## 4     1 | GARY HUA                        |6.0  |W  39|W  21|W  18|W  14|W   7|D  12|D   4|
## 5    ON | 15445895 / R: 1794   ->1817     |N:2  |W    |B    |W    |B    |W    |B    |W    |
## 6  ---------------------------------------------------------------------------------------
## 7     2 | DAKSHESH DARURI                 |6.0  |W  63|W  58|L   4|W  17|W  16|W  20|W   7|
```

The task here

We just want to get the player names, states, points, and ratings, and keep track of who they played. The rest of the printout above is just noise, for our purposes here. The two approaches that occur to me, to accomplish the task, are

1) Spend some time parsing the file into a sensible and clean `data.frame`, and then use frame operations to calculate the desired output, or
2) Keep the table exactly as read in by the `read.csv` defaults, and use `regex` and whatever regular structure the messy table provides us to pick out the details we need.

Since we've recently been focusing on `regex`, I'm going to go with the second option....

Explore the structure

```
tourney <- read.csv("../Project1/tournamentinfo.txt", header = FALSE, skip = 2)
names <- tourney %>% filter(row_number() %% 3 == 0)
head(names, n=4)
```

```
##                                                                                                  V1
## 1     1 | GARY HUA                        |6.0  |W  39|W  21|W  18|W  14|W   7|D  12|D   4|
## 2     2 | DAKSHESH DARURI                 |6.0  |W  63|W  58|L   4|W  17|W  16|W  20|W   7|
## 3     3 | ADITYA BAJAJ                    |6.0  |L   8|W  61|W  25|W  21|W  11|W  13|W  12|
## 4     4 | PATRICK H SCHILLING             |5.5  |W  23|D  28|W   2|W  26|D   5|W  19|D   1|
```

```
glue("{dim(names)[1]} player names")
```

```
## 64 player names
```

We need just the name, points won, and opponents from those rows.

Names:

```
extract_name <- function(string) {
  step1 <- str_match(string, "\\|[a-zA-Z -]+\\|")
  str_remove_all(step1, "\\|\\s*|\\s*\\|")
}
extract_name(names[4,])
```

```
## [1] "PATRICK H SCHILLING"
```

Points:

```
pull_points <- function(string) {
  step1 <- str_match(string, "\\|[0-9\\.]+")
  as.numeric(str_remove(step1, "\\|"))
}
pull_points(names[4,])
```

```
## [1] 5.5
```

Opponents:

```
get_opps <- function(string) {
  step1 <- str_match_all(string, "\\b[WDL] *[0-9]+\\b")
  f <- function(s){as.numeric(str_remove(s, "[WDL]\\s*"))}
  sapply(step1, f)
}
as.vector(get_opps(names[4,]))
```

```
## [1] 23 28  2 26  5 19  1
```

And now we need homes and pre-rankings from the other rows

```
ranks <- tourney %>% filter(row_number() %% 3 == 1)
head(ranks, n=5)
```

```
##                                                                                           V1
## 1  Num  | USCF ID / Rtg (Pre->Post)       | Pts |  1  |  2  |  3  |  4  |  5  |  6  |  7  |
## 2     ON | 15445895 / R: 1794   ->1817     |N:2  |W    |B    |W    |B    |W    |B    |W    |
## 3     MI | 14598900 / R: 1553   ->1663     |N:2  |B    |W    |B    |W    |B    |W    |B    |
## 4     MI | 14959604 / R: 1384   ->1640     |N:2  |W    |B    |W    |B    |W    |B    |W    |
## 5     MI | 12616049 / R: 1716   ->1744     |N:2  |W    |B    |W    |B    |W    |B    |B    |
```

```
glue("{dim(ranks)[1]} player rankings")
```

```
## 65 player rankings
```

We just want the first 2 letters plus the "Pre" part of that, and that first line will throw off a lot of things, so let's start by removing it.

```
ranks <- ranks %>% filter(row_number() > 1)
# other subsetting is changing d.f to strings(??)
from <- function(string) {
  str_match(string, "[A-Z]+")
}

get_ranks <- function(string) {
  step1 <- str_match(string, " R:\\s*[0-9]+")
  as.numeric(str_remove_all(step1, " R:\\s*"))
}
from(ranks[4,])
```

```
##      [,1]
## [1,] "MI"
```

```
get_ranks(ranks[4,])
```

```
## [1] 1716
```

With players connected to their rankings, we can now substitute opponents with their rankings and find means.

```
meanranks <- function(opplist, rankvec) {
  # for each opponent list, map to avg ranking in list
  opplist <- map(opplist, function(x){round(mean(rankvec[x], 0))})
  unlist(opplist)
}
```

Now build a frame that has what we need, using those 6 functions, because the required output is a .csv

```
players <- extract_name(names$V1)
points <- pull_points(names$V1)
opponents <- get_opps(names$V1)
from <- from(ranks$V1)
rankings <- get_ranks(ranks$V1)
oppranks <- meanranks(opponents, rankings)

chess <- data.frame(player = players, home = from, points = points,
                    prerank = rankings, opp_ranks = oppranks)
chess
```

```
##                       player home points prerank opp_ranks
## 1                   GARY HUA   ON    6.0    1794      1605
## 2             DAKSHESH DARURI   MI    6.0    1553      1469
## 3                ADITYA BAJAJ   MI    6.0    1384      1564
## 4         PATRICK H SCHILLING   MI    5.5    1716      1574
## 5                  HANSHI ZUO   MI    5.5    1655      1501
## 6                HANSEN SONG   OH    5.0    1686      1519
## 7           GARY DEE SWATHELL   MI    5.0    1649      1372
## 8            EZEKIEL HOUGHTON   MI    5.0    1641      1468
## 9                 STEFANO LEE   ON    5.0    1411      1523
## 10                  ANVIT RAO   MI    5.0    1365      1554
## 11   CAMERON WILLIAM MC LEMAN   MI    4.5    1712      1468
## 12             KENNETH J TACK   MI    4.5    1663      1506
```

```
## 13          TORRANCE HENRY JR   MI    4.5    1666      1498
## 14               BRADLEY SHAW   MI    4.5    1610      1515
## 15     ZACHARY JAMES HOUGHTON   MI    4.5    1220      1484
## 16               MIKE NIKITIN   MI    4.0    1604      1386
## 17         RONALD GRZEGORCZYK   MI    4.0    1629      1499
## 18              DAVID SUNDEEN   MI    4.0    1600      1480
## 19               DIPANKAR ROY   MI    4.0    1564      1426
## 20                JASON ZHENG   MI    4.0    1595      1411
## 21              DINH DANG BUI   ON    4.0    1563      1470
## 22           EUGENE L MCCLURE   MI    4.0    1555      1300
## 23                   ALAN BUI   ON    4.0    1363      1214
## 24          MICHAEL R ALDRICH   MI    4.0    1229      1357
## 25           LOREN SCHWIEBERT   MI    3.5    1745      1363
## 26                    MAX ZHU   ON    3.5    1579      1507
## 27              GAURAV GIDWANI   MI    3.5    1552      1222
## 28 SOFIA ADINA STANESCU-BELLU   MI    3.5    1507      1522
## 29           CHIEDOZIE OKORIE   MI    3.5    1602      1314
## 30         GEORGE AVERY JONES   ON    3.5    1522      1144
## 31               RISHI SHETTY   MI    3.5    1494      1260
## 32      JOSHUA PHILIP MATHEWS   ON    3.5    1441      1379
## 33                   JADE GE   MI    3.5    1449      1277
## 34     MICHAEL JEFFERY THOMAS   MI    3.5    1399      1375
## 35           JOSHUA DAVID LEE   MI    3.5    1438      1150
## 36              SIDDHARTH JHA   MI    3.5    1355      1388
## 37       AMIYATOSH PWNANANDAM   MI    3.5     980      1385
## 38                 BRIAN LIU   MI    3.0    1423      1539
## 39              JOEL R HENDON   MI    3.0    1436      1430
## 40               FOREST ZHANG   MI    3.0    1348      1391
## 41        KYLE WILLIAM MURPHY   MI    3.0    1403      1248
## 42                   JARED GE   MI    3.0    1332      1150
## 43          ROBERT GLEN VASEY   MI    3.0    1283      1107
## 44         JUSTIN D SCHILLING   MI    3.0    1199      1327
## 45                  DEREK YAN   MI    3.0    1242      1152
## 46   JACOB ALEXANDER LAVALLEY   MI    3.0     377      1358
## 47                ERIC WRIGHT   MI    2.5    1362      1392
## 48               DANIEL KHAIN   MI    2.5    1382      1356
## 49           MICHAEL J MARTIN   MI    2.5    1291      1286
## 50                 SHIVAM JHA   MI    2.5    1056      1296
## 51             TEJAS AYYAGARI   MI    2.5    1011      1356
## 52                  ETHAN GUO   MI    2.5     935      1495
## 53              JOSE C YBARRA   MI    2.0    1393      1345
## 54                LARRY HODGE   MI    2.0    1270      1206
## 55                  ALEX KONG   MI    2.0    1186      1406
## 56               MARISA RICCI   MI    2.0    1153      1414
## 57                 MICHAEL LU   MI    2.0    1092      1363
## 58               VIRAJ MOHILE   MI    2.0     917      1391
## 59          SEAN M MC CORMICK   MI    2.0     853      1319
## 60                 JULIA SHEN   MI    1.5     967      1330
## 61               JEZZEL FARKAS   ON    1.5     955      1327
## 62              ASHWIN BALAJI   MI    1.0    1530      1186
## 63       THOMAS JOSEPH HOSMER   MI    1.0    1175      1350
## 64                    BEN LI   MI    1.0    1163      1263
```

In summary, the routine starts with the above components, and ends with a `.csv` output, which can all be encapsulated as follows:

```
text2csv <- function(tourneyFile, toFile) {
  # read in the textfile, which of course has to be formatted exactly like ours:)
  tourney <- read.csv(tourneyFile, header = FALSE, skip = 2)
  # subset the names rows
  names <- tourney %>% filter(row_number() %% 3 == 0)
  #----helper functions for name rows-----
  extract_name <- function(string) {
    step1 <- str_match(string, "\\|[a-zA-Z -]+\\|")
    str_remove_all(step1, "\\|\\s*|\\s*\\|")
  }
  pull_points <- function(string) {
    step1 <- str_match(string, "\\|[0-9\\.]+")
    as.numeric(str_remove(step1, "\\|"))
  }
  get_opps <- function(string) {
    step1 <- str_match_all(string, "\\b[WDL] *[0-9]+\\b")
    f <- function(s){as.numeric(str_remove(s, "[WDL]\\s*"))}
    sapply(step1, f)
  }
  # subset ranking rows
  ranks <- tourney %>% filter(row_number() %% 3 == 1)
  # remove header
  ranks <- ranks %>% filter(row_number() > 1)
  #----helper funcs for ranking rows------
  from <- function(string) {
    str_match(string, "[A-Z]+")
  }
  get_ranks <- function(string) {
    step1 <- str_match(string, " R:\\s*[0-9]+")
    as.numeric(str_remove_all(step1, " R:\\s*"))
  }
  meanranks <- function(opplist, rankvec) {
    # for each opponent list, map to avg ranking in list
    opplist <- map(opplist, function(x){round(mean(rankvec[x], 0))})
    unlist(opplist)
  }
  # build the frame
  players <- extract_name(names$V1)
  points <- pull_points(names$V1)
  opponents <- get_opps(names$V1)
  from <- from(ranks$V1)
  rankings <- get_ranks(ranks$V1)
  oppranks <- meanranks(opponents, rankings)

  chess <- data.frame(player = players, home = from, points = points,
                      prerank = rankings, opp_ranks = oppranks)
  # Output to csv
  write_csv(chess, toFile)
}
```

Test if it works:

```
infile = "../Project1/tournamentinfo.txt"
tempfile = "tmp.csv"
text2csv(infile, tempfile)
chess <- read_csv(tempfile)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   player = col_character(),
##   home = col_character(),
##   points = col_double(),
##   prerank = col_double(),
##   opp_ranks = col_double()
## )
```

```
chess
```

```
## # A tibble: 64 x 5
##    player              home  points prerank opp_ranks
##    <chr>               <chr>  <dbl>   <dbl>     <dbl>
##  1 GARY HUA            ON       6      1794      1605
##  2 DAKSHESH DARURI     MI       6      1553      1469
##  3 ADITYA BAJAJ        MI       6      1384      1564
##  4 PATRICK H SCHILLING MI       5.5    1716      1574
##  5 HANSHI ZUO          MI       5.5    1655      1501
##  6 HANSEN SONG         OH       5      1686      1519
##  7 GARY DEE SWATHELL   MI       5      1649      1372
##  8 EZEKIEL HOUGHTON    MI       5      1641      1468
##  9 STEFANO LEE         ON       5      1411      1523
## 10 ANVIT RAO           MI       5      1365      1554
## # ... with 54 more rows
```