DATA 607 Assignment 5B

Catherine Dube

Intro

In this project, I am given some chess tournament data. My goal is to calculate each player's expected score, and the difference from their actual score, then retrieve the top five overperformers and underperformers compared to their expected performance.

I had placed the tournament text file in my github repo in project 1. I'll retrieve it below and repeat (most of) the steps that I took to clean it in that project.

```
data_src <- "https://raw.githubusercontent.com/cdube89128/DATA-607/refs/heads/main/project-01/tournamenti
nfo.txt"

# Read file lines
lines <- readLines(data_src)
```

```
## Warning in readLines(data_src): incomplete final line found on
## 'https://raw.githubusercontent.com/cdube89128/DATA-607/refs/heads/main/project-01/tournamentinfo.txt'
```

I saw this warning in Project 1 when I read in the file. It is due to the lack of a newline character at the end of my text file, so I'm continuing onward as normal.

```r
# Remove header lines in file
lines <- lines[-c(1:4)]

# Remove divider lines
lines <- lines[!grepl("^-", lines)]

# Group into chunks of 2 lines per player
player_lines <- split(lines, ceiling(seq_along(lines)/2))

# Combine each pair into one string
combined <- sapply(player_lines, paste, collapse = "")

# The whitespace is messy, cleaning that up
combined <- gsub("\\s+", " ", combined)
combined <- trimws(combined)

# This looks more easily parsable. Almost all of the distinct values are separated by pipes (|).
split_data <- str_split(combined, "\\|")

# Create a function to parse each entry
parse_player <- function(x) {
  x <- str_trim(x)   # trim whitespace because it was still slightly irregular

  tibble(
    Pair = as.numeric(x[1]),
    Name = x[2],
    Total = as.numeric(x[3]),
    Round_1 = as.numeric(str_extract(x[4], "\\d+")),
    Round_2 = as.numeric(str_extract(x[5], "\\d+")),
    Round_3 = as.numeric(str_extract(x[6], "\\d+")),
    Round_4 = as.numeric(str_extract(x[7], "\\d+")),
    Round_5 = as.numeric(str_extract(x[8], "\\d+")),
    Round_6 = as.numeric(str_extract(x[9], "\\d+")),
    Round_7 = as.numeric(str_extract(x[10], "\\d+")),
    State = x[11],
    #After this, more complicated parsing is needed
    ID = str_extract(x[12], "\\d+"),                          # get 1st group of digits
    Pre_Rating = as.numeric(str_extract(x[12], "(?<=R: )\\d+")),     # get group of digits after R:
    Post_Rating = as.numeric(str_extract(x[12], "(?<=->)\\d+"))      # get group of digits after ->
  )
}

# Apply my function to each element of split_data
# Bind the resulting rows together into a new dataframe
my_df <- bind_rows(lapply(split_data, parse_player))

# Checking in
head(my_df, 5)
```

```
## # A tibble: 5 × 14
##    Pair Name       Total Round_1 Round_2 Round_3 Round_4 Round_5 Round_6 Round_7
##   <dbl> <chr>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1     1 GARY HUA     6        39      21      18      14       7      12       4
## 2     2 DAKSHESH …   6        63      58       4      17      16      20       7
## 3     3 ADITYA BA…   6         8      61      25      21      11      13      12
## 4     4 PATRICK H…   5.5      23      28       2      26       5      19       1
## 5     5 HANSHI ZUO   5.5      45      37      12      13       4      14      17
## # ℹ 4 more variables: State <chr>, ID <chr>, Pre_Rating <dbl>,
## #   Post_Rating <dbl>
```

Up until now I have repeated things from Project 1 in order to read in and clean the chess tournament data. I did **not** repeat the step from Project 1 where I calculated the average pre chess rating of opponents for each entry/player. Instead, I will be looking at each opponents score individually as part of the process of predicting each player's final chess rating.

Now I will head into the ELO calculations. For these, I referenced the provided video: [The Elo Rating System for Chess and Beyond] (https://www.youtube.com/watch?v=AsYfbmp0To0 (https://www.youtube.com/watch?v=AsYfbmp0To0)). Feb 15, 2019

```
# Rename data frame for clarity
tournament <- my_df

# Creating a function to calculate the expected score (elo)
elo_expected <- function(rA, rB) {
  1 / (1 + 10^((rB - rA) / 400))
}

# using pivot_longer to give each player/round its own row
long_matches <- tournament %>%
  pivot_longer(cols = starts_with("Round_"),
               names_to = "Round",
               values_to = "Opponent") %>%
  left_join(tournament %>% select(Pair, Pre_Rating),
            by = c("Opponent" = "Pair"),
            suffix = c("", "_Opp"))

# Calc expected scores for every match
# (technically doing this twice, once for each player)
long_matches <- long_matches %>%
  mutate(Expected = elo_expected(Pre_Rating, Pre_Rating_Opp))

# Checking In
head(long_matches)
```

```
## # A tibble: 6 × 11
##    Pair Name     Total State ID       Pre_Rating Post_Rating Round   Opponent
##   <dbl> <chr>    <dbl> <chr> <chr>         <dbl>       <dbl> <chr>      <dbl>
## 1     1 GARY HUA     6 ON    15445895       1794        1817 Round_1       39
## 2     1 GARY HUA     6 ON    15445895       1794        1817 Round_2       21
## 3     1 GARY HUA     6 ON    15445895       1794        1817 Round_3       18
## 4     1 GARY HUA     6 ON    15445895       1794        1817 Round_4       14
## 5     1 GARY HUA     6 ON    15445895       1794        1817 Round_5        7
## 6     1 GARY HUA     6 ON    15445895       1794        1817 Round_6       12
## # ℹ 2 more variables: Pre_Rating_Opp <dbl>, Expected <dbl>
```

I now have the expected score from every match; I will roll this back up into the expected score for every player. (I will also get the predicted rating for each player, just becuase I am interested in it.)

```
# Elo factor can apparently sometimes vary, so declaring as 32 here
K <- 32

# Get predicted player scores and ratings
players <- long_matches %>%
  group_by(Pair) %>%
  summarise(
    Name = first(Name),
    Pre_Rating = first(Pre_Rating),
    Post_Rating = first(Post_Rating),
    Actual_Score = first(Total),
    Expected_Score = round(sum(Expected, na.rm = TRUE), 2), # NA to deal with empty rounds
    Predicted_Rating = round(Pre_Rating + K * (Actual_Score - Expected_Score), 0),
    .groups = "drop"
  )

# Checking In
head(players)
```

```
## # A tibble: 6 × 7
##    Pair Name                Pre_Rating Post_Rating Actual_Score Expected_Score
##   <dbl> <chr>                    <dbl>       <dbl>        <dbl>          <dbl>
## 1     1 GARY HUA                  1794        1817            6           5.16
## 2     2 DAKSHESH DARURI           1553        1663            6           3.78
## 3     3 ADITYA BAJAJ              1384        1640            6           1.95
## 4     4 PATRICK H SCHILLING       1716        1744          5.5           4.74
## 5     5 HANSHI ZUO                1655        1690          5.5           4.38
## 6     6 HANSEN SONG               1686        1687            5           4.94
## # ℹ 1 more variable: Predicted_Rating <dbl>
```

Looks lovely! Now I'm going to look at the difference between actual and expected scores, and see who outperformed or underperformed the most

```
# Getting the difference between actual and expected scores from the tournament
players <- players %>%
  mutate(Score_Difference = Expected_Score - Actual_Score) %>%
  arrange(desc(Score_Difference))

players <- players[, c("Pair", "Name", "Pre_Rating", "Predicted_Rating", "Post_Rating",
                       "Expected_Score", "Actual_Score", "Score_Difference")]
```

```
kable(head(players, 5), caption = "Top 5 Players Who Scored Higher than Expected")
```

Top 5 Players Who Scored Higher than Expected

Pair	Name	Pre_Rating	Predicted_Rating	Post_Rating	Expected_Score	Actual_Score	Score_Difference
25	LOREN SCHWIEBERT	1745	1656	1681	6.28	3.5	2.78

Pair	Name	Pre_Rating	Predicted_Rating	Post_Rating	Expected_Score	Actual_Score	Score_Difference
30	GEORGE AVERY JONES	1522	1441	1444	6.02	3.5	2.52
42	JARED GE	1332	1268	1256	5.01	3.0	2.01
31	RISHI SHETTY	1494	1443	1444	5.09	3.5	1.59
35	JOSHUA DAVID LEE	1438	1391	1392	4.96	3.5	1.46

```
paste0("Average point difference between expected and actual for top 5: ",mean(head(players, 5)$Score_Dif
ference))
```

```
## [1] "Average point difference between expected and actual for top 5: 2.072"
```

```
#kable(tail(players, 5), caption = "Bottom 5 Players Who Scored Lower than Expected")

players <- players %>%
  arrange((Score_Difference))

kable(head(players, 5), caption = "Bottom 5 Players Who Scored Lower than Expected")
```

Bottom 5 Players Who Scored Lower than Expected

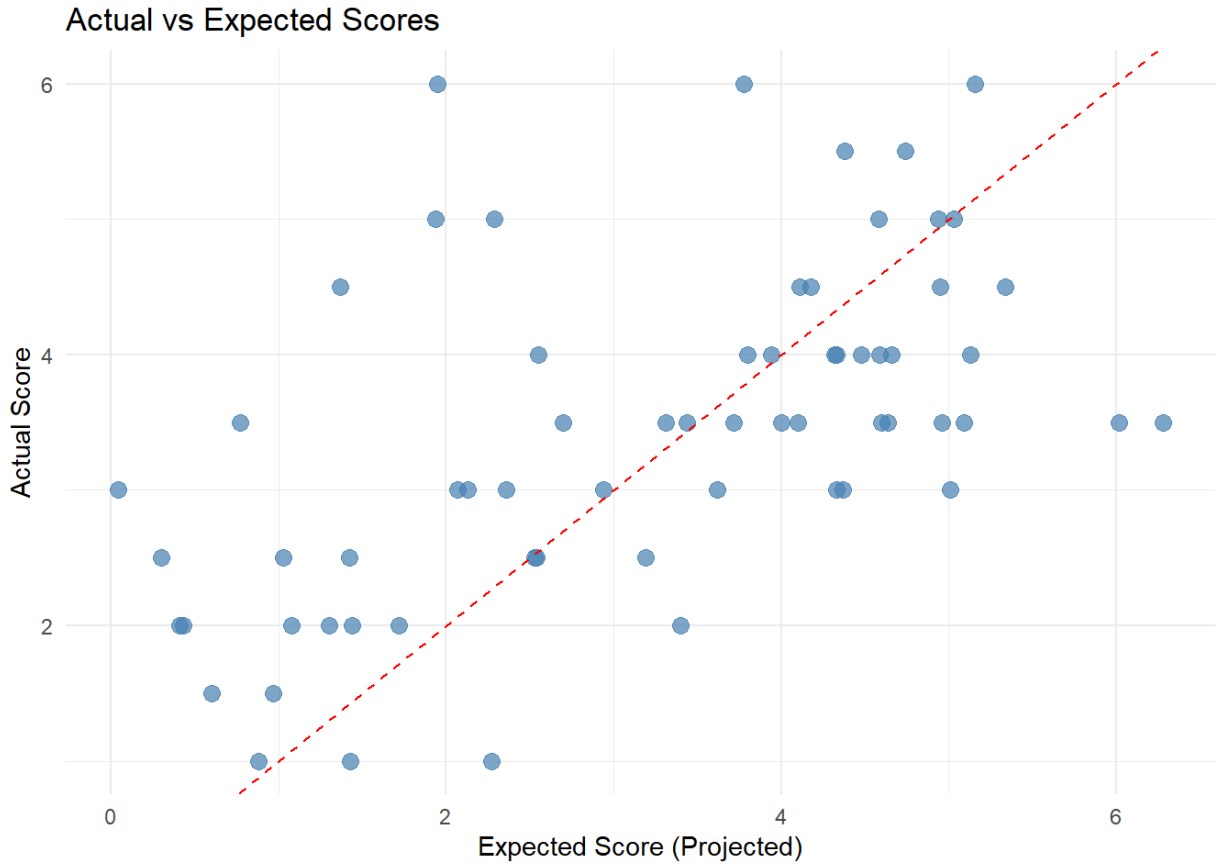
Pair	Name	Pre_Rating	Predicted_Rating	Post_Rating	Expected_Score	Actual_Score	Score_Difference
3	ADITYA BAJAJ	1384	1514	1640	1.95	6.0	-4.05
15	ZACHARY JAMES HOUGHTON	1220	1320	1416	1.37	4.5	-3.13
10	ANVIT RAO	1365	1463	1544	1.94	5.0	-3.06
46	JACOB ALEXANDER LAVALLEY	377	472	1076	0.04	3.0	-2.96
37	AMIYATOSH PWNANANDAM	980	1067	1077	0.77	3.5	-2.73

```
paste0("Average point difference between expected and actual for top 5: ",mean(head(players, 5)$Score_Dif
ference))
```

```
## [1] "Average point difference between expected and actual for top 5: -3.186"
```

I’m going to generate a visual as well, just so that I can get a better idea of the distribution.

```
ggplot(players, aes(x = Expected_Score, y = Actual_Score)) +
  geom_point(color = "steelblue", size = 3, alpha = 0.7) +
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  theme_minimal() +
  labs(
    title = "Actual vs Expected Scores",
    x = "Expected Score (Projected)",
    y = "Actual Score"
  )
```



Conclusion

The top five players who scored higher than expected, scored higher by 2.1 points. The bottom five players who scored lower than expected, scored lower by 3.2 points. Looking at the plot above, the projected ELO scores were not the best predictors, but they did capture the general trend.