Project 1: Chess Tournament Data Analysis

Alina Vikhnevich

2025-03-03

Introduction

Chess tournaments generate detailed records of player performance, including match pairings, scores, and pre-tournament ratings. However, this data is often stored in an unstructured text format, making it difficult to analyze. The goal of this project is to clean, transform, and structure this raw data into a usable dataset for analysis.

Project Objective

This project focuses on extracting key player statistics from a chess tournament dataset and computing average opponent ratings. The final dataset will be saved as a CSV file containing the following key details for each player:

- **Player's Name**: The full name of the chess player.
- **State**: The state (or province) the player represents.
- **Total Points**: The total number of points the player earned in the tournament.
- **Pre-Tournament Rating**: The player's chess rating before the tournament.
- **Average Opponent Rating**: The average pre-tournament rating of all opponents faced.

By organizing the data in this way, we can assess player performance, compare ratings, and analyze competition levels in a structured format.

Dataset Overview

The dataset comes from a chess tournament featuring 64 players, each competing in seven rounds. The raw data includes player names, pairing numbers, game results, and ratings. The information is scattered across multiple lines and includes extra characters that need to be removed.

Data Processing Steps

To transform this data into a structured format, we will follow three parts:

1. **Data Transformation** – Read the text file, remove unnecessary characters, and convert it into a structured table.
2. **Extracting Pre-Tournament Ratings** – Use regular expressions to extract player ratings from the text data.
3. **Calculating Average Opponent Ratings** – Match each player's opponents and compute their average pre-tournament rating.

After completing these steps, the cleaned dataset will be saved as a CSV file for further analysis. This structured dataset provides valuable insights into player performance, rating comparisons, and tournament trends.

1. Data Transformation

This section focuses on transforming the raw chess tournament text data into a structured format suitable for further analysis. The raw data is stored in an unstructured text file, where player details are split across multiple lines, and unnecessary characters such as game result indicators (W, L, B, D) and separators (|, /) are present.

The following steps are taken to clean and structure the data:

- **Step 1: Read the raw text data** from the provided URL.
- **Step 2: Remove separator lines** consisting entirely of hyphens (`-`).
- **Step 3: Remove unnecessary characters** that are not required for analysis.
- **Step 4: Combine player information lines**, ensuring that each player's data appears in a single row.
- **Step 5: Convert the cleaned text into a data frame**, allowing easy manipulation in R.

Implementation: The code below performs these steps:

```
# Load necessary library for data manipulation and text processing
library(tidyverse)

# Step 1
tournamentinfo <- read_lines("https://raw.githubusercontent.com/AlinaVikhnevich/data_607/main/tournamen

# Step 2
hyphens <- str_detect(tournamentinfo, '^[-]{2,}$')
tournamentinfo <- tournamentinfo[!hyphens == "TRUE"]

# Step 3
 # Remove characters 'W', 'L', 'B', 'D'
tournamentinfo <- str_remove_all(tournamentinfo, "[WLBD]")
# Replace characters '|' and '/' with commas for CSV structure
tournamentinfo <- str_replace_all(tournamentinfo, "[|/]",",")

# Step 4
empty <- c("")
for (i in seq(1, length(tournamentinfo)-1, by = 2)){
   empty <- c(empty, paste(tournamentinfo[i], tournamentinfo[i+1], sep = "", collapse = NULL))
}
empty <- empty[-1]

# Step 5
TournamentResults <- as.data.frame(do.call(rbind, strsplit(empty, ",")), stringsAsFactors = FALSE)

# Display the first few rows of the transformed data frame to inspect the initial structure
head(TournamentResults)
```

```
##       V1                                V2    V3    V4    V5    V6    V7    V8
## 1  Pair   Player Name                      Total Round Round Round Round Round
## 2     1   GARY HUA                         6.0      39    21    18    14     7
## 3     2     AKSHESH ARURI                  6.0      63    58     4    17    16
## 4     3     AITYA AJAJ                     6.0       8    61    25    21    11
## 5     4     PATRICK H SCHIING              5.5      23    28     2    26     5
## 6     5   HANSHI ZUO                       5.5      45    37    12    13     4
##      V9   V10     V11        V12                     V13   V14   V15   V16
## 1 Round Round   Num      USCF I   Rtg (Pre->Post)         Pts    1     2
## 2    12     4     ON   15445895    R: 1794   ->1817      N:2
## 3    20     7     MI   14598900    R: 1553   ->1663      N:2
## 4    13    12     MI   14959604    R: 1384   ->1640      N:2
## 5    19     1     MI   12616049    R: 1716   ->1744      N:2
## 6    14    17     MI   14601533    R: 1655   ->1690      N:2
##     V17   V18   V19   V20   V21    V22
## 1   3     4     5     6     7
## 2                                   1
## 3                                   2
## 4                                   3
## 5                                   4
## 6                                   5
```

Explanation:

- **Step 1:** Loads the raw text data from the given URL.
- **Step 2:** Identifies and removes separator lines (`-`), which do not contain any useful data.
- **Step 3:** Cleans the text by removing unnecessary game result indicators (`W`, `L`, `B`, `D`) and formatting the data into a CSV-compatible format by replacing separators (`|` and `/`) with commas.
- **Step 4:** Since each player's information spans two lines, the loop merges them into a single line per player to make further processing easier.
- **Step 5:** Splits the combined text into columns and stores it as a structured data frame, allowing further data cleaning and transformation.

Why was it done? Without this transformation, the dataset remains inconsistent and difficult to process. By restructuring the data into a clean and uniform format, we ensure that each player's details are stored in a single row, making it much easier to extract necessary details and perform calculations in the next steps.

---

2. Extracting Pre-Tournament Ratings

The goal of this step is to extract each player's **pre-tournament rating**, which is listed in the raw text data as a numerical value prefixed with `"R:"`. This value represents the player's ranking before the tournament began and is essential for computing the **average opponent rating** later.

Since the raw text is unstructured and contains other characters like pairing numbers and special markers (e.g., `"->"`), **text processing techniques** must be applied to extract only the **clean numeric ratings**.

Steps Taken:

- **Step 1: Identify and extract the ratings** using a regular expression (regex) pattern that targets values prefixed with `"R:"`.
- **Step 2: Remove unnecessary characters**, such as `"R:"`, `"->"`, and any pairing-related identifiers like `"P01"`, `"P02"`, etc.
- **Step 3: Ensure only numeric values remain**, by filtering out any remaining non-numeric elements.
- **Step 4: Convert the extracted values into a numeric format**, handling any missing or incorrect values.

Implementation: The code below performs these steps:

```r
# Step 1
Preratings <- str_extract_all(tournamentinfo, "R:\\d*(.*?)\\d*->", simplify = TRUE)

# Step 2
Preratings <- str_remove_all(Preratings, "R:")
Preratings <- str_remove_all(Preratings, "->")
Preratings <- str_remove_all(Preratings, "P\\d{2}")
Preratings <- str_remove_all(Preratings, "P\\d")

# Step 3
Preratings <- str_match_all(Preratings, "\\d+")
Preratings <- str_extract_all(Preratings, "\\d+", simplify = TRUE)

# Step 4
stuff <- unlist(Preratings)
stuff <- as.numeric(as.character(stuff))
stuff <- stuff - 1
stuff <- gsub("-1", NA, stuff)
stuff <- na.omit(stuff)

Preratings <- stuff
Preratings <- as.data.frame(Preratings)
Preratings <- as.numeric(as.character(unlist(Preratings[[1]])))
Preratings <- as.data.frame(Preratings)
Preratings <- na.omit(Preratings)

# Display the extracted pre-ratings
head(Preratings)
```

```
##   Preratings
## 1       1793
## 2       1552
## 3        1383
## 4        1715
## 5        1654
## 6        1685
```

Explanation:

- **Step 1:** Uses a regular expression (`"R:\\d*(.*?)\\d*->"`) to isolate pre-tournament ratings, which are prefixed by `"R:"` and followed by `"->"`.
- **Step 2:** Cleans the extracted values by removing non-numeric elements such as `"R:"`, `"->"`, and pairing-related labels (e.g., `"P01"`, `"P02"`).
- **Step 3:** Ensures that only numeric values remain, removing any unexpected characters.
- **Step 4:** Converts the extracted text into a numeric format, handling missing values and ensuring cleaned pre-ratings are stored in a structured format.

Why was it done? Without extracting the pre-tournament ratings correctly, we cannot calculate the average rating of a player's opponents, which is a critical metric in this analysis. By cleaning and structuring the data properly, we ensure that the ratings are accurate, consistent, and ready for further processing in the next step.

---

3. Calculation of Average Opponent Rating

In this step, we integrate the pre-tournament ratings extracted earlier with the main tournament results and compute the average rating of each player's opponents. This metric helps evaluate player performance by considering the strength of their competition.

Since each player competes against different opponents, we must match each opponent's pair number to their pre-tournament rating and then compute the mean rating of those opponents. The process involves multiple transformations to ensure that the calculations are accurate and formatted correctly.

Steps Taken:

- **Step 1: Remove the header row from TournamentResults**
  - Eliminates any unnecessary first-row artifacts that may have been introduced during text parsing.
- **Step 2:Combine TournamentResults with the extracted Preratings**
  - Merges the cleaned player information with their pre-tournament rating for better reference.
- **Step 3: Convert opponent pair numbers for each round into numeric format**
  - Extracts and converts the round-based opponent pair numbers into numeric values to enable calculations.
- **Step 4: Subset TournamentResults to retain only necessary columns**
  - Filters the dataset to include only the relevant columns before computing the opponent ratings.
- **Step 5: Assign meaningful column names to TournamentResults**
  - Renames variables to ensure clear interpretation of the data.
- **Step 6: Calculate opponent pre-ratings for each round by looking up ratings based on opponent Pair numbers**
  - Matches each player's opponent pair numbers to their respective pre-tournament ratings.
- **Step 7: Calculate the average opponent rating for each player across all rounds**
  - Computes the mean rating of all opponents for each player, ensuring missing values (if any) are handled appropriately.
- **Step 8: Assign the calculated average opponent rating and perform final data frame subsetting**
  - Saves the computed values under the column `Avg_Opponent_Rating` and keeps only the final required columns for output.

Implementation: The code below performs these steps:

```
# Step 1
TournamentResults <- TournamentResults[2:65, ]

# Step 2
TournamentResults <- cbind.data.frame(TournamentResults, Preratings)

# Step 3
Round1 <- as.numeric(as.character(unlist(TournamentResults$V4)))
Round2 <- as.numeric(as.character(unlist(TournamentResults$V5)))
Round3 <- as.numeric(as.character(unlist(TournamentResults$V6)))
```

```r
Round4 <- as.numeric(as.character(unlist(TournamentResults$V7)))
Round5 <- as.numeric(as.character(unlist(TournamentResults$V8)))
Round6 <- as.numeric(as.character(unlist(TournamentResults$V9)))
Round7 <- as.numeric(as.character(unlist(TournamentResults$V10)))

# Step 4
TournamentResults <- subset(TournamentResults, select = c(
  "V1",
  "V2",
  "V3",
  "V11",
  "Preratings")
  )
TournamentResults <- cbind(TournamentResults, Round1, Round2, Round3, Round4, Round5, Round6, Round7)

# Step 5
colnames(TournamentResults) <- c(
  "Pair",
  "Name",
  "Total",
  "State",
  "Preratings",
  "Round1",
  "Round2",
  "Round3",
  "Round4",
  "Round5",
  "Round6",
  "Round7"
  )

# Step 6
for (i in 1:64) {
  TournamentResults$Round1[i] <- TournamentResults$Preratings[TournamentResults$Round1[i]]
  TournamentResults$Round2[i] <- TournamentResults$Preratings[TournamentResults$Round2[i]]
  TournamentResults$Round3[i] <- TournamentResults$Preratings[TournamentResults$Round3[i]]
  TournamentResults$Round4[i] <- TournamentResults$Preratings[TournamentResults$Round4[i]]
  TournamentResults$Round5[i] <- TournamentResults$Preratings[TournamentResults$Round5[i]]
  TournamentResults$Round6[i] <- TournamentResults$Preratings[TournamentResults$Round6[i]]
  TournamentResults$Round7[i] <- TournamentResults$Preratings[TournamentResults$Round7[i]]
}

# Step 7
for (i in 1:64) {
  TournamentResults$Means [i] <- rowMeans(TournamentResults[i, 6:12], na.rm = TRUE)
}

# Step 8
TournamentResults$Avg_Opponent_Rating <- TournamentResults$Means
TournamentResults <- subset(TournamentResults, select = c(
  "Name",
  "State",
  "Total",
  "Preratings",
  "Avg_Opponent_Rating"
))

# Display sample results
head(TournamentResults)
```

```
##                                 Name  State Total Preratings Avg_Opponent_Rating
## 2  GARY HUA                            ON   6.0         1793            1604.286
## 3    AKSHESH ARURI                     MI   6.0         1552            1468.286
## 4    AITYA AJAJ                        MI   6.0         1383            1562.571
## 5    PATRICK H SCHIING                 MI   5.5         1715            1572.571
## 6  HANSHI ZUO                          MI   5.5         1654            1499.857
## 7  HANSEN SONG                         OH   5.0         1685            1517.714
```

Explanation:

- **Step 1:** Cleans up unnecessary rows that might exist from the initial transformation.
- **Step 2:** Merges the TournamentResults table with the extracted pre-tournament ratings to ensure all players have their correct ratings.
- **Step 3:** Extracts opponent pair numbers for each round, so we can later match them with their corresponding pre-tournament ratings.
- **Step 4:** Selects only the relevant columns needed for calculations and further processing.
- **Step 5:** Renames columns appropriately for better readability.
- **Step 6:** Loops through each round and replaces opponent pair numbers with their actual pre-tournament ratings.
- **Step 7:** Computes the mean opponent rating for each player, ignoring any missing values.
- **Step 8:** Selects only the final required columns for output and renames the `Avg_Opponent_Rating` column.

Why was it done? The Average Opponent Rating is a critical metric used in chess tournaments to evaluate player performance. A high score against lower-rated players may not indicate as strong a performance as the same score achieved against higher-rated opponents. This calculation helps normalize a player's success by considering the strength of their competition.

Export to CSV:

Once the tournament data has been transformed, cleaned, and structured into a proper format, it is essential to save it for further analysis or sharing. Exporting the data as a CSV file ensures compatibility with various analytical tools, including Excel, SQL databases, and visualization software.

The write.csv() function is used to write the TournamentResults dataframe into a CSV file. The file path specified should be adjusted based on where you want to save the output. The argument `row.names = FALSE` ensures that row numbers are not included in the saved file.

```
write.csv(TournamentResults, "C:/Users/alina/Desktop/Spring 2025/DATA 607/DATA607/tournament_results.csv
```

Conclusion

The first five players statistics are shown below:

```r
# Display data frame with first 5 players stats
TournamentResults[1:5,]
```

```
##                                 Name  State Total Preratings Avg_Opponent_Rating
## 2  GARY HUA                            ON  6.0         1793            1604.286
## 3    AKSHESH ARURI                     MI  6.0         1552            1468.286
## 4    AITYA AJAJ                        MI  6.0         1383            1562.571
## 5    PATRICK H SCHIING                 MI  5.5         1715            1572.571
## 6  HANSHI ZUO                          MI  5.5         1654            1499.857
```

This project successfully processed raw chess tournament data into a structured and analyzable format. By leveraging R's text manipulation and data transformation capabilities, we extracted key player statistics, computed average opponent ratings, and formatted the dataset for further analysis.

Key takeaways from this analysis:

- **Data Cleaning & Structuring:** The original raw text data contained unnecessary characters and required restructuring into a tabular format.
- **Pre-Tournament Rating Extraction:** Using regular expressions, we accurately extracted and assigned pre-tournament ratings for all players.
- **Opponent Rating Calculation:** The algorithm successfully mapped each player's opponents and calculated their `Avg_Opponent_Rating`, providing insight into tournament difficulty.

The final dataset presents a cleaned, well-structured summary of chess tournament results, which can be further analyzed or integrated into external ranking systems. This approach demonstrates the power of data wrangling in R, ensuring that even unstructured data can be transformed into meaningful insights.