Project Proposal

Due November 17 at 11:59pm

Alejandro Paredes La Torre, Liangcheng (Jay) Liu

Nzarama Michaella Desire Kouadio, Jahnavi Maddhuri

Load Packages

Dataset 1

Data source: Tennis atp, by Jeff Sackmann: https://github.com/JeffSackmann/tennis_atp/tree/master?tab=r
ov-file

Brief description:

This dataset is a comprehensive archive of ATP player information, rankings, match results, and stats. It includes: a player file containing biographical data (e.g., player_id, name, hand, birth date, country, height); ranking files that track historical ATP rankings; a results file covering tour-level, challenger, and futures matches; match stats for tour-level matches. Some match stats may be missing due to either lack of ATP data or data validation filters.

Research question 1:

How does the difference in ranking predict the length in minutes for the match, and does this predictive power vary across different tournament levels?

- Outcome Variable: minutes (continuous). This represents the length of a match.
- Explanatory Variable: winner_rank - loser_rank (continuous). This represents the difference in rank between the winner and loser. Let difference = winner_rank - loser_rank.
- Interaction Term: surface (categorical). This represents the surface that the match was played on.

$$minutes = \beta 0 + \beta 1 \cdot (difference) + \beta 2 \cdot surface + \beta 3(difference * surface) + \epsilon$$

- Question Rationale: We believe there is a relationship between rank difference and match duration where players that are closely matched will have a longer match. The surface could also play a role in how fast the players are able to react.

Step 1: Selecting only relevant columns to study

```
#glimpse(tennis)
```

A. Using Correlation Matrix to identify confounding variables for our numerical variable

- What I realised is that I cannot clean all columns, so my goal is to select the relevant columns for our analysis first and then do the cleaning for those columns.
- We need to find confounding variables.

```
# Create the rank difference varaiable and put it in our dataset
tennis <- tennis %>%
  mutate(rank_diff = winner_rank - loser_rank)
```

- Our first technique to finding confounding variables is to do a correlation analysis. By performing a correlation matrix, we're identifying potential confounders as variables that are highly correlated with both your response variable (Y) and your predictors (Xs). ==> this works for continous variables

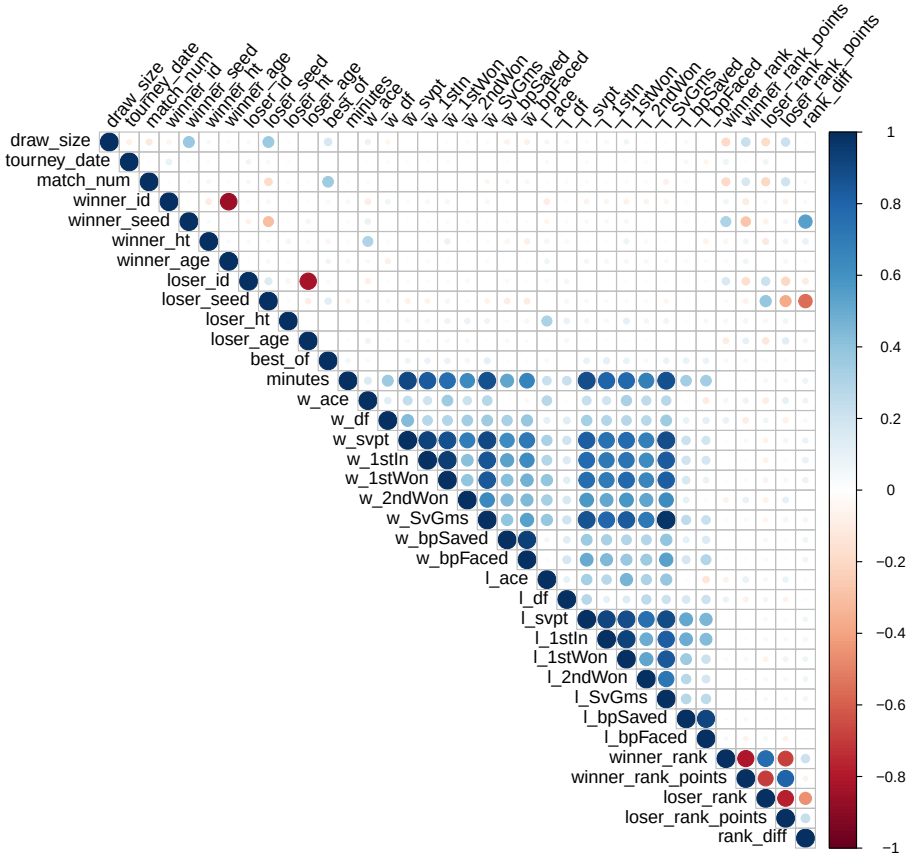
Conclusion: None of the numerical varaible are correlated to minutes and rank difference at the same time.

```
# Step 1: Build the correlation matrix

  # Filter to only select numerical variables
numerical_vars <- tennis[, sapply(tennis, is.numeric)]

  # Compute the correlation matrix
correlation_matrix <- cor(numerical_vars, use = "complete.obs")

  # Visualize the correlation matrix
corrplot(correlation_matrix, method = "circle", type = "upper", tl.col = "black", tl.srt = 5
```




```r
# Step 2:  Flag the variable that are highly correlated to both minutes and rank diff at the

  # Compute correlations with minutes (Y) and rank_diff (X)
cor_minutes <- correlation_matrix["minutes", ] # Correlation with Y
cor_rank_diff <- correlation_matrix["rank_diff", ] # Correlation with X

  # Combine into a dataframe for easier filtering
cor_data <- data.frame(
  variable = colnames(correlation_matrix),
  cor_with_minutes = cor_minutes,
  cor_with_rank_diff = cor_rank_diff
)

  # Add a flag for high correlation ( I chose a very low treshold of |0.5|)
cor_data <- cor_data %>%
  mutate(
    high_cor_minutes = abs(cor_with_minutes) > 0.5,
    high_cor_rank_diff = abs(cor_with_rank_diff) > 0.5,
```

```
    high_cor_both = high_cor_minutes & high_cor_rank_diff
  )

# Filter and format the output to show only relevant columns
cor_data <- cor_data %>%
  select(cor_with_minutes, cor_with_rank_diff, high_cor_both)

  # View the flagged variables
print(cor_data)
```

```
                cor_with_minutes cor_with_rank_diff high_cor_both
draw_size           0.004305643        0.018687712         FALSE
tourney_date        0.007027013        0.005360109         FALSE
match_num          -0.013309204        0.036532248         FALSE
winner_id          -0.040710039        0.031548416         FALSE
winner_seed         0.029599619        0.543106281         FALSE
winner_ht          -0.026855036        0.091059240         FALSE
winner_age          0.033950097       -0.029942730         FALSE
loser_id           -0.001283556       -0.100813758         FALSE
loser_seed         -0.055542556       -0.553965491         FALSE
loser_ht           -0.008238139        0.005670147         FALSE
loser_age           0.013641636        0.056965452         FALSE
best_of             0.054748962       -0.001953848         FALSE
minutes             1.000000000        0.077374946         FALSE
w_ace               0.150099370        0.073266578         FALSE
w_df                0.365576181        0.031344732         FALSE
w_svpt              0.907319527        0.082476313         FALSE
w_1stIn             0.834718392        0.093180319         FALSE
w_1stWon            0.769626733        0.087202150         FALSE
w_2ndWon            0.634558193        0.009771748         FALSE
w_SvGms             0.870661655        0.053945734         FALSE
w_bpSaved           0.522353356        0.068426535         FALSE
w_bpFaced           0.662633276        0.068720050         FALSE
l_ace               0.227353273       -0.024657790         FALSE
l_df                0.228483743        0.028060556         FALSE
l_svpt              0.881895330        0.060463762         FALSE
l_1stIn             0.809579446        0.048486436         FALSE
l_1stWon            0.783762269        0.067313165         FALSE
l_2ndWon            0.685030080        0.063327411         FALSE
l_SvGms             0.879717392        0.053567451         FALSE
l_bpSaved           0.341371613        0.001564181         FALSE
l_bpFaced           0.343887974       -0.013553311         FALSE
```

winner_rank	0.005636379	0.219333106	FALSE
winner_rank_points	0.011930529	-0.067808621	FALSE
loser_rank	-0.045543318	-0.454895447	FALSE
loser_rank_points	0.039918429	0.232095818	FALSE
rank_diff	0.077374946	1.000000000	FALSE

Using ANOVA TEST to find confounding variables for our categorical variable

- Now we look at the categorical variables that could have a relationship with minutes and surface at the same time.

The following categorical variables where flaggeg as potential cofounders: - tourney_id - tourney_name
- winner_name
- winner_ioc
- loser_entry
- loser_name
- loser_hand
- loser_ioc
- score
- round

> But we have to question if those variables are true cofounders or if they are naturally related to the structure of the data, so we need to think about their role in the dataset

Not a true confounder: - tourney_id, tourney_name, winner_name, loser_name, winner_ioc, loser_ioc: These are identifiers or descriptive variables that don't directly influence minutes. Their association with both surface and minutes might just reflect the structure of the dataset, not a confounding relationship.

- score: score lies in the causal pathway because surface influences how matches are played—different surfaces (like clay or grass) affect rally lengths and playing styles, which determine the number of games or sets in a match (score). This score then directly impacts minutes, as more games or sets naturally result in longer match durations, connecting surface to minutes through score

Potential confounder: `round`: Certain rounds (e.g., finals) may occur more often on specific surfaces, plus later rounds tend to have longer matches due to competitiveness and higher level skillsets.

```r
# Convert all character variables to factors
tennis <- tennis %>%
  mutate(across(where(is.character), as.factor))

# Identify categorical variables in the dataset
categorical_vars <- names(tennis)[sapply(tennis, is.factor)]

# Exclude surface from the list (we're testing against it)
categorical_vars <- setdiff(categorical_vars, "surface")

# Initialize a results dataframe
results <- data.frame(
  variable = categorical_vars,
  associated_with_minutes = NA,
  associated_with_surface = NA
)

# Loop through each categorical variable
for (var in categorical_vars) {
  # Test association with minutes using ANOVA
  anova_test <- summary(aov(tennis$minutes ~ tennis[[var]], data = tennis))
  results$associated_with_minutes[results$variable == var] <-
    anova_test[[1]]["Pr(>F)"][1] < 0.05

  # Test association with surface using Chi-Square
  table_surface <- table(tennis$surface, tennis[[var]])
  chisq_test <- chisq.test(table_surface)
  results$associated_with_surface[results$variable == var] <-
    chisq_test$p.value < 0.05
}

# Filter variables associated with both minutes and surface
potential_confounders <- results %>%
  filter(associated_with_minutes == TRUE & associated_with_surface == TRUE)

# View the results
print(potential_confounders)
```

```
       variable associated_with_minutes associated_with_surface
1    tourney_id                    TRUE                    TRUE
2  tourney_name                    TRUE                    TRUE
3   winner_name                    TRUE                    TRUE
```

```
4    winner_ioc                  TRUE                  TRUE
5   loser_entry                  TRUE                  TRUE
6    loser_name                  TRUE                  TRUE
7    loser_hand                  TRUE                  TRUE
8     loser_ioc                  TRUE                  TRUE
9         score                  TRUE                  TRUE
10        round                  TRUE                  TRUE
```

C. Relevant column announced

Thus the relationship we will be analysing will include the following columns: - minutes - rank difference - surface - round

So our equation looks like this:

$$minutes = \beta 0 + \beta 1(surface) + \beta 2(rank_diff) + \beta 3(round) + \beta 4(surface x rank_diff) + \epsilon$$

Step 2: Cleaning Relevant Columns

Create a new dataframe with only relevant columns

```
# Create a dataframe with only the 4 selected columns
tennis_filtered <- tennis %>% select(minutes, rank_diff, surface, round)
glimpse(tennis_filtered)
```

```
Rows: 10,663
Columns: 4
$ minutes   <int> 88, 44, 67, 71, 85, 127, 139, 87, 79, 121, 86, 77, 66, 104, ~
$ rank_diff <int> 99, NA, -1152, -234, -1300, 224, 311, 193, -270, 143, 39, -1~
$ surface   <fct> Hard, Hard, Hard, Hard, Hard, Hard, Hard, Hard, Hard, Hard, ~
$ round     <fct> Q1, Q1, Q1, Q1, Q1, Q1, Q1, Q1, Q2, Q2, Q2, Q2, Q2, Q2, R32,~
```

Show the number of missing values for each column. Only minutes and rank difference have missing values.

```
# Count the number of NAs in each column
na_counts <- tennis_filtered  %>%
  summarise(across(everything(), ~ sum(is.na(.))))
# View the NA counts
na_counts
```

```
  minutes rank_diff surface round
1     382       334       0     0
```

Deal with the missing values for the rank difference column.

- Out of all 10663, only 3% of the values are missing.
- Complex solution to fill in missing values: Filling missing rank_diff values by looking at other matches (where the same player appears) and taking the mean of their rank to fill in. This assumes that their rank remains constant across all matches where the data is missing. This assumption is unrealistic because rankings fluctuate based on, number of tournament they participated in, types of tournamnets, performance, tournament outcomes, and points earned…
- With just 3% of the data missing, the amount of information lost by removing these rows is minimal.

```
# Calculate the percentage of missing values in rank_diff
percent_missing_rank_diff <- sum(is.na(tennis_filtered$rank_diff)) / nrow(tennis_filtered) *
percent_missing_rank_diff
```

```
[1] 3.132327
```

```
# Drop rows with NAs in rank_diff
tennis_filtered <- tennis_filtered %>%
  filter(!is.na(rank_diff))

# Check if NAs were removed from rank_diff
sum(is.na(tennis_filtered$rank_diff))
```

```
[1] 0
```

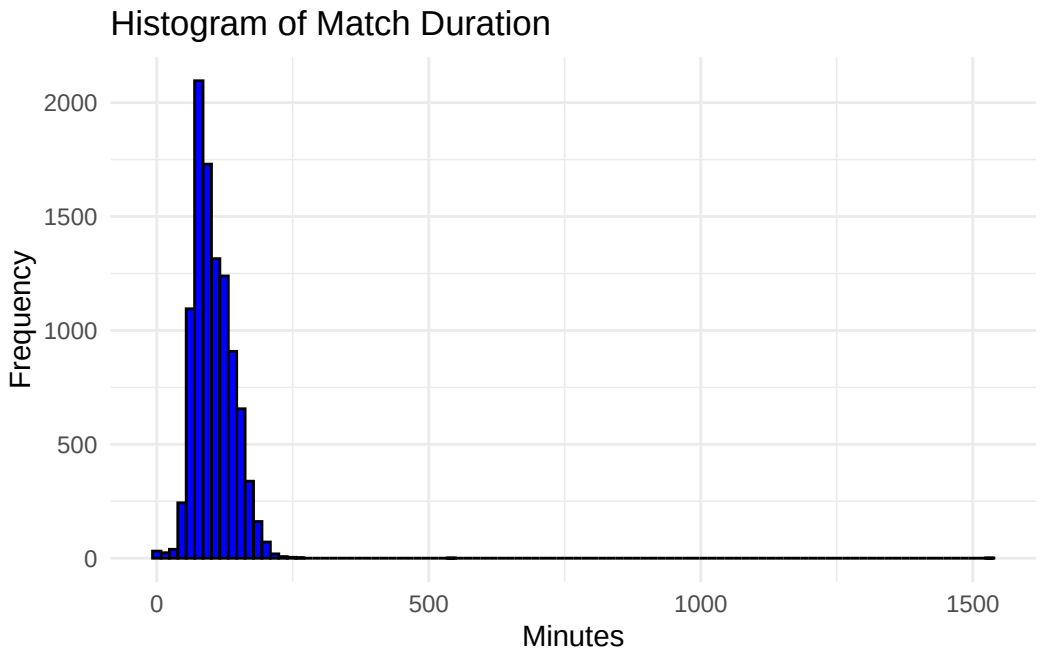
Deal with the missing values for minutes column - Out of all 10663, only 3.5% of the values are missing. - This is also not significant, but we can still fill in the missing values. We can fill missing values by grouping matches by round and calculating the median or mean of match duration for each round, then assigning this median/mean value to missing entries within the same round.

```
# Calculate the percentage of missing values in rank_diff
percent_missing_minutes<- sum(is.na(tennis_filtered$minutes)) / nrow(tennis_filtered) * 100
# Print the result
print(percent_missing_minutes)
```

```
[1] 3.378836
```

- We checked the distribution of minutes (match durations), it shows that it is highly skewed. Most matches had short durations, but a few extreme outliers (very long matches) stretched the scale. Before filling missing values, it's crucial to know if the data is normally distributed or skewed to determine if it's best to use the mean or the median. In our case we will be using the mdeian.

```
# Plot histogram using ggplot2
ggplot(na.omit(tennis_filtered), aes(x = minutes)) +
  geom_histogram(bins = 100, fill = "blue", color = "black") +
  labs(title = "Histogram of Match Duration",
       x = "Minutes",
       y = "Frequency") +
  theme_minimal()
```



- To account for differences in match durations across tournament stages (e.g., earlier rounds may have shorter matches), we filled missing values using the median duration within each round.

```
# Fill missing minutes with the median of the respective round
tennis_filtered <- tennis_filtered %>%
  group_by(round) %>%
  mutate(minutes = ifelse(is.na(minutes), median(minutes, na.rm = TRUE), minutes))
```

```r
sum(is.na(tennis_filtered$minutes))
```

```
[1] 0
```

Step 3: Run the Model

```r
# Fit the linear model
model_q1 <- lm(minutes ~ rank_diff * surface + round, data = tennis_filtered)

# View the summary of the model
summary(model_q1)
```

```
Call:
lm(formula = minutes ~ rank_diff * surface + round, data = tennis_filtered)

Residuals:
    Min      1Q  Median      3Q     Max
-113.48  -24.29   -6.51   21.55 1419.26

Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)            103.27202    6.06062  17.040  < 2e-16 ***
rank_diff                0.02547    0.02106   1.209  0.22660
surfaceClay             11.28901    5.51144   2.048  0.04056 *
surfaceGrass             3.67427    5.87932   0.625  0.53202
surfaceHard              6.58983    5.51056   1.196  0.23178
roundQ1                -11.02498    2.72499  -4.046 5.25e-05 ***
roundQ2                 -5.81876    2.78902  -2.086  0.03697 *
roundQ3                  2.27512    5.30928   0.429  0.66828
roundQF                 -5.37624    2.93745  -1.830  0.06724 .
roundR16                -3.47764    2.78736  -1.248  0.21219
roundR32                -7.23360    2.70967  -2.670  0.00761 **
roundSF                 -8.65895    3.21630  -2.692  0.00711 **
rank_diff:surfaceClay   -0.01259    0.02113  -0.596  0.55117
rank_diff:surfaceGrass  -0.01076    0.02322  -0.463  0.64315
rank_diff:surfaceHard   -0.01298    0.02112  -0.615  0.53873
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 36.69 on 10314 degrees of freedom
Multiple R-squared:  0.02426,   Adjusted R-squared:  0.02293
F-statistic: 18.32 on 14 and 10314 DF,  p-value: < 2.2e-16
```

Step 4: Interpretation of Model

We will only be interpreting the variables that are statistically significant

- For surface clay: Matches played on clay surfaces last, on average, 11.25 minutes longer than matches on carpet (reference), holding all other variables constant.
- Round Q1: Matches in Round Q1 are, on average, 11.14 minutes shorter than matches in the Final round, holding all other variables constant
- Round Q2: Matches in Round Q2 are, on average, 5.85 minutes shorter than matches in the final round, holding all other variables constant.
- Round R32: Matches in Round R32 are, on average, 7.23 minutes shorter than matches in final round, holding all other variables constant.
- Round SF: Matches in Semifinals are, on average, 8.66 minutes shorter than matches in final round, holding all other variables constant

Key Takeways:

- Clay Surface: Matches tend to last longer on clay.
- Rounds: Early rounds like Q1, Q2, R32, and even SF are significantly shorter in duration compared to the Final round.
- These findings suggest that both the tournament surface and the round significantly influence match duration.

Note:

- Q1: First round of qualification matches (Qualifier 1).
- Q2: Second round of qualification matches (Qualifier 2).
- Q3: Third round of qualification matches (Qualifier 3).
- R32: Round of 32 people
- R16: Round of 16 people
- QF: Quarterfinals

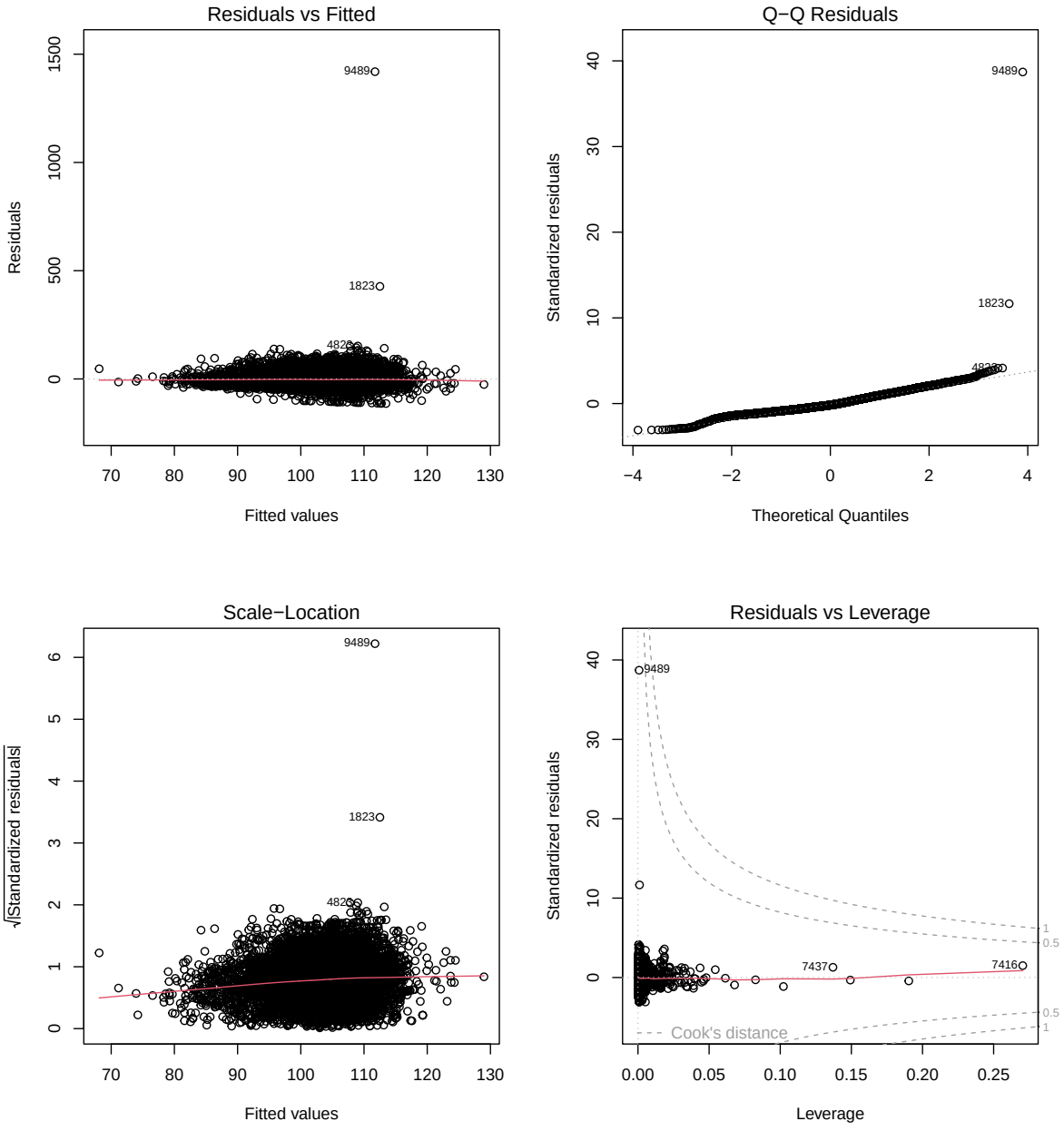
- SF: Semifinals
- F: Final (The reference category in regression model).

Step 5: Model assessment and assumption violation

R_square = 0.02, the model struggles to explain the variability in our data this is a poor fit.

Now let's check if assumptions are violated

```
par(mfrow = c(2,2))
plots <- plot(model_q1)
```



Linearity Assumption: Looking the residual vs fitted plot, residuals are randomly scattered around 0 (red horizontal line) and show no clear pattern. We do see some outliers (data point 9489 and data point 1823), but the assumption is satisfied.

Independence Assumption: The predictors are independent from one another. So this assumption is not violated.

Normality Assumption: Looking at the Q-Q plot, residuals follow a straight line and only slightly deviate at the very end, due to the same two outliers mentioned above but the assumption is satisfied

Equal Variance: Looking at the scale location, we see that the residuals are equally spread out around a fairly horizontal line, so it's fair to asssume that homoscedasticity is maintained

Limitations of our model

- Low predictive power: R_square = 0.02 indicates, our model explains only 2% of the variability in the response variable (minutes). This suggests that the predictors rank_diff, surface, round, and their interaction are not doing a great job of explaining match duration. We should explore additional predictors in our dataset or consider external factors such as fatigue, weather…
- The interaction terms (rank_diff:surface) in our model is not statistically significant, suggesting it contributes nothing to explaining minutes. We should rethink or remove insignificant interaction terms to simplify the model.
- So our main issue comes from having picked none signicant predictors for match duration and not necessarily due to a violation of assumption.

Useful pieces of code to not delete (might be used later on)

- Checking if ID's are unique Since one row is a match, we can see if a player with a missing rank in one row appears in another row and fill up their rank with that.

```
# Combine winner_id and loser_id into a single vector
unique_players <- unique(c(tennis$winner_id, tennis$loser_id))

# Check the length of unique values
num_unique_players <- length(unique(unique_players))
num_total_players <- length(unique_players)

# Verify if the total is equal to the number of unique players
if (num_unique_players == num_total_players) {
  print("All player IDs are unique!")
} else {
  print("There are duplicate player IDs.")
}
```

```
[1] "All player IDs are unique!"
```