White Advantage in Chess and How to Counter It

Jun Won (Lakon) Park (Group Leader) *79453940, Group Leader, data collection, data analysis, report-writing*
Sarah Li *60136959, data collection, data analysis, report-writing*

Research question: Is white at an advantage in chess and if so, what are some optimal strategies for black to increase their winning probability?

Introduction

For several centuries, millions of people worldwide have been playing chess as a recreational and competitive board game at their homes, in clubs, in tournaments, and even online nowadays. In the recent decades, chess has been one of the most popular topic in machine learning and artificial intelligence. The first move advantage has been researched extensively since the end of 18th century, and many studies have been shown that white has an inherent advantage.

Although there are general set chess openings for black according to white's first move, less research has been done on the effects of those openings on the final outcome. This paper intends to confirm white's first move advantage and study the relationship between the openings and the victory status.

This paper's data collection consists basic player information and game information of over 20000 chess games obtained exclusively from Lichess, a very popular internet chess platform. The data includes game length, number of turns, winner, player elo*, all moves in Standard Chess Notation, Opening Eco*, Opening Name, and Opening Ply* (some stuff about sampling method and target population)

Elo : A numerical measurement to quantify a player's skill level
Eco : Standardised code for any given opening
Ply : Number of moves in the opening phasenewline

Analysis

```r
# Load data
df <- read.csv("games.csv")

# Calculate the average elo of the game
df <- mutate(df %>% rowwise(),
       average_elo = rowMeans(cbind(black_rating,white_rating)))

# Filter games by average elo
df <- filter(df, average_elo >= 1200)

# Filter games by average elo
df <- filter(df, victory_status != "outoftime")

df <- subset(df,
             select = c(turns, white_rating, black_rating, victory_status,
                        winner, moves, opening_eco, opening_name, opening_ply, average_elo ))

# Simple Random Sampling
N <- nrow(df)
n <- 1000
set.seed(1234)
sample.index <- sample(1:N, size=n, replace = FALSE)
srs.sample <- df[sample.index,]
```

```r
# Determine minimum and maximum before stratifying
min(df$average_elo)
```

```
## [1] 1200
```

```r
max(df$average_elo)
```

```
## [1] 2475.5
```

```r
# Stratified sampling
df$elo_range <- cut(df$average_elo,
                    c(1200, 1400, 1600, 1800, 2000, 2200, 2400, 2600))
levels(df$elo_range) <- c("1200-1400", "1400-1600", "1600-1800", "1800-2000",
                          "2000-2200", "2200-2400", "2400+")
df$winner <- as.factor(df$winner)
# levels(df$winner) <- list("white"=c("white"),
```

```r
#                              "black"=c("black")
#                              "not_white"=c("black", "draw"))

# Check if standard deviations of the strata are identical
se <- aggregate(as.numeric(df$winner), by=list(df$elo_range), FUN=sd)
se
```

```
##     Group.1         x
## 1 1200-1400 0.9783955
## 2 1400-1600 0.9777180
## 3 1600-1800 0.9737201
## 4 1800-2000 0.9697871
## 5 2000-2200 0.9618646
## 6 2200-2400 0.9323589
## 7     2400+ 0.8232726
```

```r
# Standard deviations within strata are not identical, \
# so find optimal sample sizes
pop_sizes <- aggregate(df$winner, by=list(df$elo_range), FUN=length)
denom <- sum(pop_sizes[2] * se[2])
sample_sizes <- (pop_sizes[2] * se[2]) / denom


# Sample from each strata
stratified_sample <- df[FALSE,]
colnames(stratified_sample) <- names(df)
for (i in 1:length(levels(df$elo_range))) {
  strata <- which(df$elo_range == levels(df$elo_range)[i])
  sample_indices <- sample(strata,
                           size = ceiling(sample_sizes$x[i] * n),
                           replace = FALSE)
  sample <- df[sample_indices,]
  stratified_sample <- rbind(stratified_sample, sample)
}
table <- table(stratified_sample$elo_range)
table
```

```
##
## 1200-1400 1400-1600 1600-1800 1800-2000 2000-2200 2200-2400     2400+
##       199       326       245       155        63        14         1
```

```r
# Stratified sample contains 1003 samples due to rounding of the proportions,
# so we randomly remove three from random strata
strata_for_removal <- sample(1:7, 3)
for (s in strata_for_removal) {
  to_remove <- sample(which(stratified_sample$elo_range == levels(df$elo_range)[s]), 1)
  stratified_sample <- stratified_sample[-to_remove,]
```

```
}
table(stratified_sample$elo_range)
```

```
##
## 1200-1400 1400-1600 1600-1800 1800-2000 2000-2200 2200-2400     2400+
##       198       326       243       155        63        14         1
```

```
# Calculate white's win rate
win.prop <- srs.sample %>%
  count(winner) %>%
  group_by(winner) %>%
  mutate(win.prop = n / 1000)

white.p <- as.numeric(win.prop[3,3])
black.p <- as.numeric(win.prop[1,3])

srs.se <- sqrt((1-n/N)*(white.p * (1-white.p) + black.p * (1-black.p) - 2*white.p*black.p)/n)

(white.p-black.p) + 1.96 * srs.se * c(-1, 1)
```

```
## [1] -0.006236205  0.020236205
```

```
strata <- c("1200-1400", "1400-1600", "1600-1800", "1800-2000",
                          "2000-2200", "2200-2400", "2400+")
stratified_sample <- stratified_sample %>% group_by(elo_range)

# Calculate Nh/N, the strata proportion
nh <- stratified_sample %>% count(elo_range)
strata.size.prop <- nh[2]/n

# Calculate white's win proportion by each strata
win.prop <- stratified_sample %>%
  count(winner) %>%
  group_by(elo_range) %>%
  mutate(win.prop = n / sum(n))

# The estimated aggregated win proportion for white
white.prop <- win.prop[win.prop$winner == "white", ]
white.p.str.est <- sum(white.prop$win.prop * strata.size.prop)

black.prop <- win.prop[win.prop$winner == "black", ]
black.p.str.est <- sum(black.prop$win.prop * strata.size.prop)

# The estimated se
white.se.by.strata <- sqrt((1-nh[2]/N) * white.prop$win.prop * (1-white.prop$win.prop)/nh[2])
white.str.se <- sum(strata.size.prop^2 * (1 - nh[2]/pop_sizes[2]) * white.se.by.strata^2)
```

```
black.se.by.strata <- sqrt((1-nh[2]/N) * black.prop$win.prop * (1-black.prop$win.prop)/nh[2])
black.str.se <- sum(strata.size.prop^2 * (1 - nh[2]/pop_sizes[2]) * black.se.by.strata^2)

# Their difference,
diff.se <- sqrt(1-n/N) *sqrt(white.p.str.est*(1-white.p.str.est) + black.p.str.est*(1-black.p.s
(white.p.str.est - black.p.str.est) + 1.96 * diff.se * c(-1, 1)
```

```
## [1] 0.03531683 0.06299630
```

```
# The confidence interval does not contain 0 so we can reject the null hypothesis in favour of
```

Conclusion

References