Chess Banter Project Checkin

Reed Evans, Jared Feldman, Jerry Gonzalez, Randy Louie

10/24/2023

1 Treatment Design

In addition to the treatment of banter that creates a "sense of rivalry" with the opponent we would like to explore the possibility of a placebo treatment if we have time. If randomized into this group, the participant will see irrelevant chats from us. This could be something like random fun animal facts like "Did you know giraffes can grow to 19 ft tall?" This will allow us to further isolate what is causing the participants to play better or worse against us. By having this placebo we will be answering the question of whether the cause of the treatment effect is the opponent reading the chats or the actual content of those chats.

2 Experiment Design

We were able to use Lichess.org's game API to play ranked games against random opponents on the website without the opponent knowing we were using a program. Since this API is python based, we are fully able to control the randomization, the playing skill level, and chats. This will enable us to lower the active workload of our team.

For every game we win or lose during our experiment, the rating of our account will rise or lower accordingly. This will mean that the player that we play next may be affected by whether we won the previous game. If we have a win streak, the next player will be higher rated than if we had a losing streak. We think that the randomization of our treatment and control should eliminate this bias.

Based on our power analysis, we would like to implement two points in which we look at our data. We think 200 and 400 total played games are reasonable stopping points.

3 Measurement Design

We are currently planning to use Lichess.org's built in 'Accuracy' measurement as our analysis metric. A 100% accuracy score means that the player played every move exactly how the Lichess engine would have played in that board scenario. Although chess is not a completely 'solved' game i.e., a chess engine still has a chance to be beaten by an elite human, it will be impossible for the population of players we will be playing to beat the chess engine. So we can assume that the chess engine is a perfect player for our population. The chess engine we will be using for our automated playing will make mistakes and try to replicate the skill of the average 1500 level player.
If after further progress we deem Lichess's Accuracy algorithm to be flawed, we have the option of downloading every move during the game and running the moves through another chess game analysis tool to generate a different metric.

4 Pilot Data

For pilot data, we have run the chess bot against a sample of 15 real lichess players around the 1500 range and captured the Lichess.org game analysis for these games. For the first 10 games, these were all untreated.

id	treat	status	winner	op_player	op_color	op_rating	op_ratingDiff	op_blunder	op_accuracy
lhZtpt54	1	mate	black	Khanny_V	black	1374	6	0	95
tFlJvUnb	1	mate	black	Aarav123champ	black	1490	5	0	92
c4ThXpIs	1	resign	white	MVRoglic	black	1435	-22	4	78
dd2ybSZe	1	mate	white	teolop72	white	1490	5	2	92
F4ke8xON	1	mate	black	HoomanFa1993	white	1494	-6	3	81
kMuSf84w	0	mate	black	lynn83	black	1461	6	2	57
xtZ9zuqx	0	resign	white	lynn83	white	1455	6	4	53
34jsuOnC	0	resign	black	juan47	black	1590	5	3	83
XilD5V1U	0	outoftime	white	SAVExistChess	white	1564	2	3	77
uMzbyFj9	0	mate	white	valeriy5454	black	1522	-5	3	57

The last 5 games, we treated the players to light, unscripted banter.

```
kable(head(d, 10), booktabs = TRUE) %>%
  kable_styling(font_size = 8)
```

```
# run a regression and build stargazer

pilot_test <- d[ , lm(op_accuracy ~ treat)]
pilot_test_robust <- coefci(pilot_test, vcov. = vcovHC(pilot_test))

stargazer(pilot_test,
          pilot_test_robust,
          type = 'text')
```

```
##
## ===============================================
##                         Dependent variable:
##                     ---------------------------
##                             op_accuracy
## -----------------------------------------------
## treat                         16.100*
##                               (7.479)
##
## Constant                     71.500***
##                               (4.318)
##
## -----------------------------------------------
## Observations                    15
## R2                             0.263
## Adjusted R2                    0.206
## Residual Std. Error      13.654 (df = 13)
## F Statistic             4.634* (df = 1; 13)
## ===============================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
##
## =========================
##             2.5 %  97.5 %
## -------------------------
## (Intercept) 60.255 82.745
## treat       2.197  30.003
## -------------------------
```