Data Exploration: Cooperation

Your name here

September 16, 2021

To begin this assignment, **make sure you have downloaded all the materials in this week's folder on Canvas**. Before you begin, make sure you have the files `data_assignment_sept16.Rmd` and `data_assignment_sept16.pdf` as well as the folders `axelRod-master` and `rmd_photos` in the same folder on your computer. You will be using a Shiny app, developed by Simon Garnier and edited slightly by the TF team, that will only work if those things are all in the same place.

Next, **you must set your working directory to the source file location**. To do so, select 'Session' from the menu bar at the top of your screen, hover over 'Set Working Directory', then select 'To Source File Location'.

When knitting your RMarkdown file to a PDF, make sure to say "No" when it asks you if you would like to "Render and view this document using Shiny instead."

If you have trouble getting the Shiny app to work (or with anything else), please come to the teaching team. We are happy to help!

The Evolution of Cooperation

Axelrod's *Evolution of Cooperation* uses the construct of the Prisoner's Dilemma to illustrate how cooperation can emerge despite incentives not to. In the Prisoner's Dilemma game, players must choose whether to cooperate or defect. The payoffs for doing one or the other depend on what the other player does, but no matter if the other player cooperates or defects, it is always strictly better for players to defect.

This can be seen in the table below, which replicates the game Axelrod uses throughout his book. If player 2 cooperates, it's better for player 1 to defect, since then player 1 would receive 5 points instead of 3. If player 2 defects, player 1 definitely wants to defect and receive 1 point rather than 0. So no matter what each player expects the other to do, they will both choose to defect, yielding 1 point to each player.

P1 ↓; P2 →	C	D
C	R = 3, R = 3	S = 0, T = 5
D	T = 5, S = 0	P = 1, P = 1

But ideally, in the long run (in a repeated game) players would like to cooperate and receive 3 points on each round. Axelrod explains how strategies of cooperation can evolve even in circumstances where players are antagonists (like Allied and Axis soldiers in the trenches of World War I) or when the players are not capable of foresight (as is the case for cooperation in biological systems).

The Prisoner's Dilemma Simulator

For this week's Data Exploration assignment, you will be working with a Shiny app that simulates prisoner's dilemma games. To use it, simply run the code chunk above labeled 'setup', then run the following code (`shiny_tournament()`). Doing so will open the app in a separate window.

P.S., the "Instructions" tab in the app is broken. Don't worry if it doesn't display anything for you. Refer to this document for instructions.

P.P.S., when you close the app window, there may be some warnings or errors (like "no loop for break/next, jumping to top level"). You can just ignore them.

Setup

Now we're going to do a round-robin style tournament between strategies of your choosing, similar to the ones Axelrod conducted. All students must complete this part, as the subsequent questions are based on the tournament you conduct here.

First, choose at least 6 strategies from the menu that look interesting to you. Your task is to play each one against all the other opponents and record the results in the excel file available in the Google Drive called `prisoners_dilemma_data.xlsx`.

Second, once you have chosen your strategies, type all the pairwise combinations of those strategies into the columns `player1` and `player2`. Make sure you type the strategies exactly as they appear in the application, including the case of the letters! Your spreadsheet should look this like after you have done so (but with the strategies you choose):

	A	B	C	D
1	player1	player2	score1	score2
2	titfortat	inverser		
3	titfortat	appeaser		
4	titfortat	foolMeOnce		
5	titfortat	handshaker		
6	titfortat	grumpy		
7	inverser	appeaser		
8	inverser	foolMeOnce		
9	inverser	handshaker		
10	inverser	grumpy		
11	appeaser	foolMeOnce		
12	appeaser	handshaker		
13	appeaser	grumpy		
14	foolMeOnce	handshaker		
15	foolMeOnce	grumpy		
16	handshaker	grumpy		
17				

Figure 1: This is what your table should look like after step 2.

Note that there are 15 ways to combine 6 elements into pairs[1], so if you don't have 15 pairs, check your work. Also note that the more strategies you choose, the more typing you will have to do.

Third, set the app so that "Tournament Type" = "Repeated", "Number of Rounds" = 100, and "Number of Replications" = 100. Just as in Axelrod's simulation, we are playing repeated games (this is determined by the "Number of Rounds"). The "Number of Replications" changes how many times the computer plays each repeated game. So in the example above, the computer will repeat the 100-round game of titfortat vs. inverser 100 times over and take the average outcome of each of those replications. This is useful because some of the strategies rely on probability (e.g. play "Defect" with probability .5) and so the outcome will be

[1] In math terms, this results from the fact that $\binom{6}{2} = 15$

different each time. We average over many outcomes to see which strategy wins on average. Once you're done, your spreadsheet should look something like this, but with different strategies:

	A	B	C	D
1	player1	player2	score1	score2
2	titfortat	inverser	300	300
3	titfortat	appeaser	300	300
4	titfortat	foolMeOnce	300	300
5	titfortat	handshaker	101	106
6	titfortat	grumpy	300	300
7	inverser	appeaser	300	300
8	inverser	foolMeOnce	300	300
9	inverser	handshaker	3	498
10	inverser	grumpy	300	300
11	appeaser	foolMeOnce	300	300
12	appeaser	handshaker	52	302
13	appeaser	grumpy	300	300
14	foolMeOnce	handshaker	100	110
15	foolMeOnce	grumpy	300	300
16	handshaker	grumpy	122	97
17				

Figure 2: This is what your table should look like after step 3.

Of course, don't just copy these numbers, since they're made up.

Fourth, save the file as a CSV. To do so, go to File > Save As, then set the File Format to be `CSV UTF-8 (Comma delimited) (.csv)`. Make sure to save it with the name `prisoners_dilemma_data.csv`.

Now you can finally read the data you created into R using the following code and start answering the questions that follow.

```
pd_data <- read_csv("data/prisoners_dilemma_data.csv",
                    col_types = cols(player1 = col_character(),
                                     player2 = col_character(),
                                     score1 = col_double(),
                                     score2 = col_double()
                                     )) %>%
  mutate(winner = case_when( # if you are interested, case_when() is a very useful
    score1 > score2 ~ player1, # function to create new variables. check out how it
    score1 < score2 ~ player2, # works by googling.
    score1 == score2 ~ "tie"
  ))
```

Question 1

How do the strategies you chose rank against each other based on the number of wins? How do they rank based on the number of points won? Discuss the patterns you see here as they relate to what you read from Axelrod. Keep in mind the concepts of niceness, retaliation, forgiveness, and clarity.

The following code makes a data frame called `player_data_long` that you can use to rank the strategies based on the number of points won during the tournament. As a hint, you may want to try using `group_by()`, `summarize()`, and `arrange()` on `player_data_long`.

If you want, try to figure out what each line of code does. Cleaning the data and rearranging it like this is an important part of data science, not just running regressions and taking means. This is why we are leaving some of it to you, via `group_by()` and `summarize()`.

To rank the strategies based on the number of wins, think about making use of the `winner` variable in the `pda_data` data frame.

```
player1_data <- pd_data %>% select(player = player1, score = score1, opponent = player2)
player2_data <- pd_data %>% select(player = player2, score = score2, opponent = player1)
player_data_long <- bind_rows(player1_data, player2_data)

# Best strategy based on wins
# I do not count a tie as a win (all ties include titfortat)
pd_data %>%
  count(winner) %>%
  arrange(desc(n)) %>%
  filter(winner != "tie") %>%
  gt() %>%
  tab_header(title = "Best Strategy Based on Wins") %>%
  cols_label(
    winner = "Strategy",
    n = "Wins"
  )
```

Best Strategy Based on Wins

Strategy	Wins
handshaker	4
winStayer	4
thuemorser	2
cycler	1
inverser	1

```
# Best strategy based on number of points
player_data_long %>%
  group_by(player) %>%
  summarize(min_score = min(score),
            med_score = median(score),
            avg_score = mean(score),
            max_score = max(score),
            total_score = sum(score),
            .groups = "drop") %>%
  arrange(desc(avg_score)) %>%
  gt() %>%
  tab_header(title = "Best Strategy Based on Points") %>%
  cols_label(
    player = "Strategy",
    min_score = "Lowest Score",
    med_score = "Median Score",
    avg_score = "Mean Score",
    max_score = "Highest Score",
```

```
    total_score = "Total Score"
)
```

Best Strategy Based on Points

Strategy	Lowest Score	Median Score	Mean Score	Highest Score	Total Score
winStayer	201	350	358.0	498	1790
handshaker	7	299	261.6	498	1308
titfortat	101	233	221.6	300	1108
thuemorser	54	233	213.4	313	1067
inverser	3	186	158.0	300	790
cycler	28	188	157.8	278	789

Based on the total number of wins (not including ties), the handshaker and winStayer strategies tied for first with 4 wins each. cycler and inverser both came in last with only 1 win each. Notice that titfortat didn't even place in this challenge - it only had ties and losses. However, when we look at the total number of points, we can see that titfortat actually ranks third with winStayer coming out way ahead of handshaker in total score. cycler and inverser are actually very close to each other with only a one score difference in total.

These rankings make a lot of sense, following the methods and reasoning in Axelrod (2006). First, most of these strategies implement some level of kindness, as in cooperating to start with. The only outlier would be thuemorser, which follows the Thue-Morse sequence in a seemingly random sequence. The top three strategies: winStayer, handshaker, and titfortat all have some type of retaliation strategy. inverser also has a retaliation strategy, but it is based on a probability of how much its opponent defects so it is not always a certain retaliation. When it comes to forgiveness, I think this is what sets winStayer apart from handshaker: winStayer has a forgiveness method, but handshaker does not. The top three strategies are also great in terms of clarity in that they do not try to sneak in any defects. They only do when the other oppondent does first, or in terms of handshaker just defects or cooperates forever.

I think that the reason titfortat didn't do as well in this round is the fewer number of other strategies that it encountered. If there were more of a variety of strategies, especially more unforgiving strategies, some of the others would not do as well as titfortat. The overall winner, winStayer is also very similar to titfortat in that it also reacts to what its opponent does first.

Question 2

Make a plot like the one below that displays how your winning strategy (according to total points) fared against the strategies it played against. Using `player_data_long` is probably easiest, but complete this question however makes most sense to you. Comment on what you find.

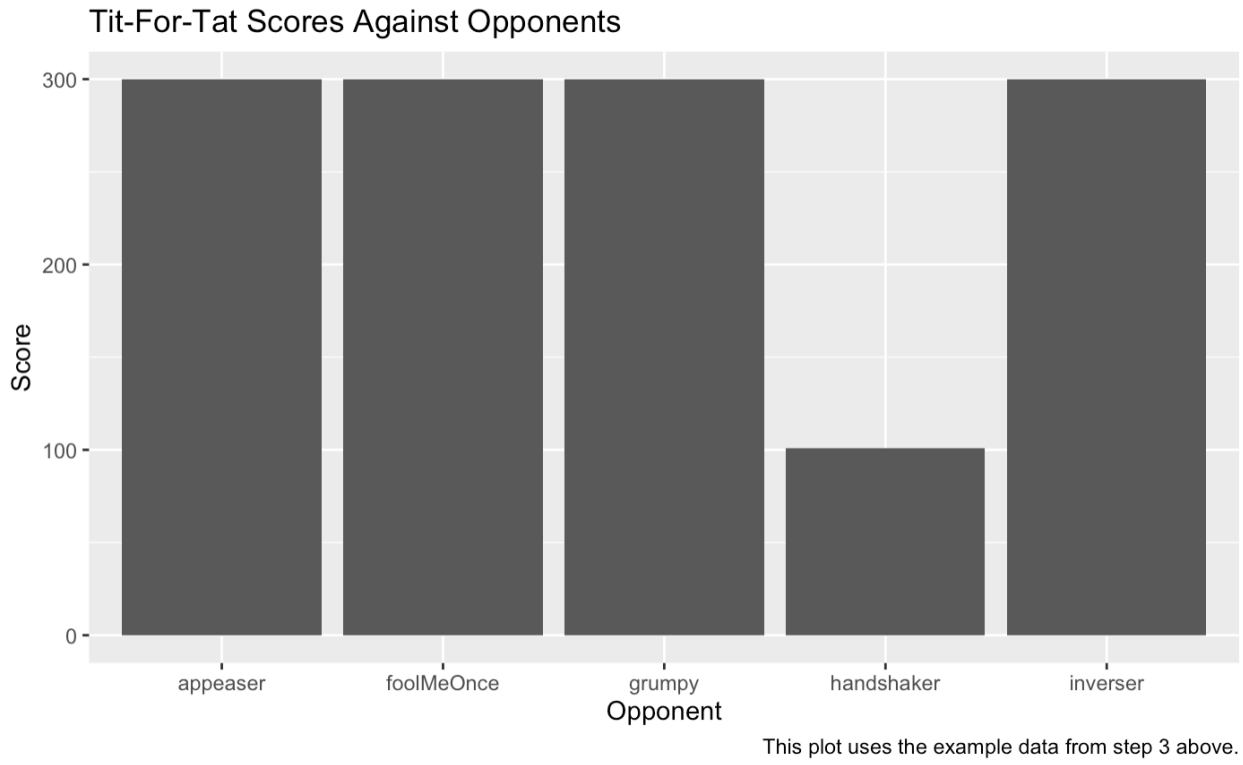
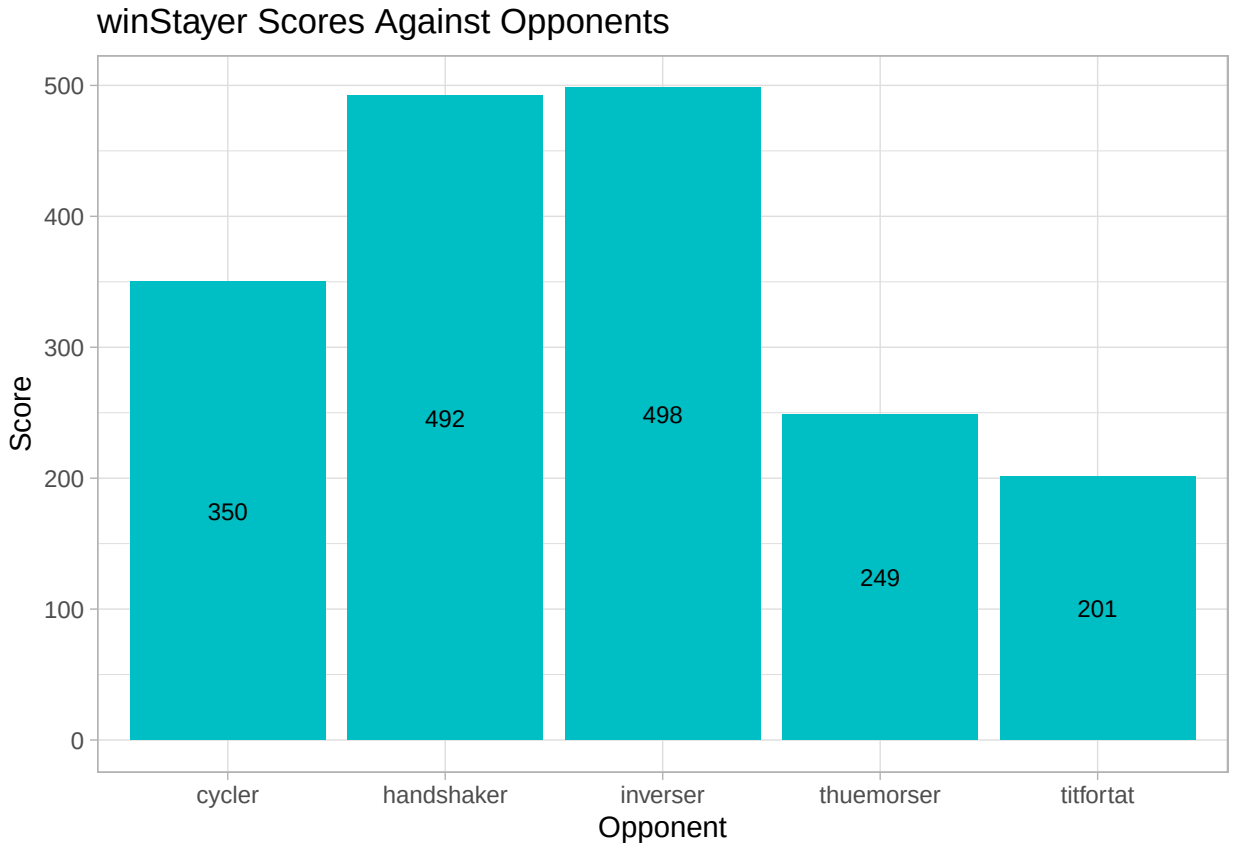


Figure 3: Here is an example of a plot you might make.

```
score_winStayer_plot <- player_data_long %>%
  filter(player == "winStayer") %>%
  ggplot(aes(x = opponent, y = score, label = score)) +
  geom_col(fill = "#00bfc4") +
  geom_text(size = 3, position = position_stack(vjust = 0.5)) +
  theme_light() +
  labs(title = "winStayer Scores Against Opponents",
       x = "Opponent",
       y = "Score")

score_winStayer_plot
```




```
# png("score_winStayer_plot.png", units="in", width=8, height=5, res=300)
# print(score_winStayer_plot)
# dev.off()
```

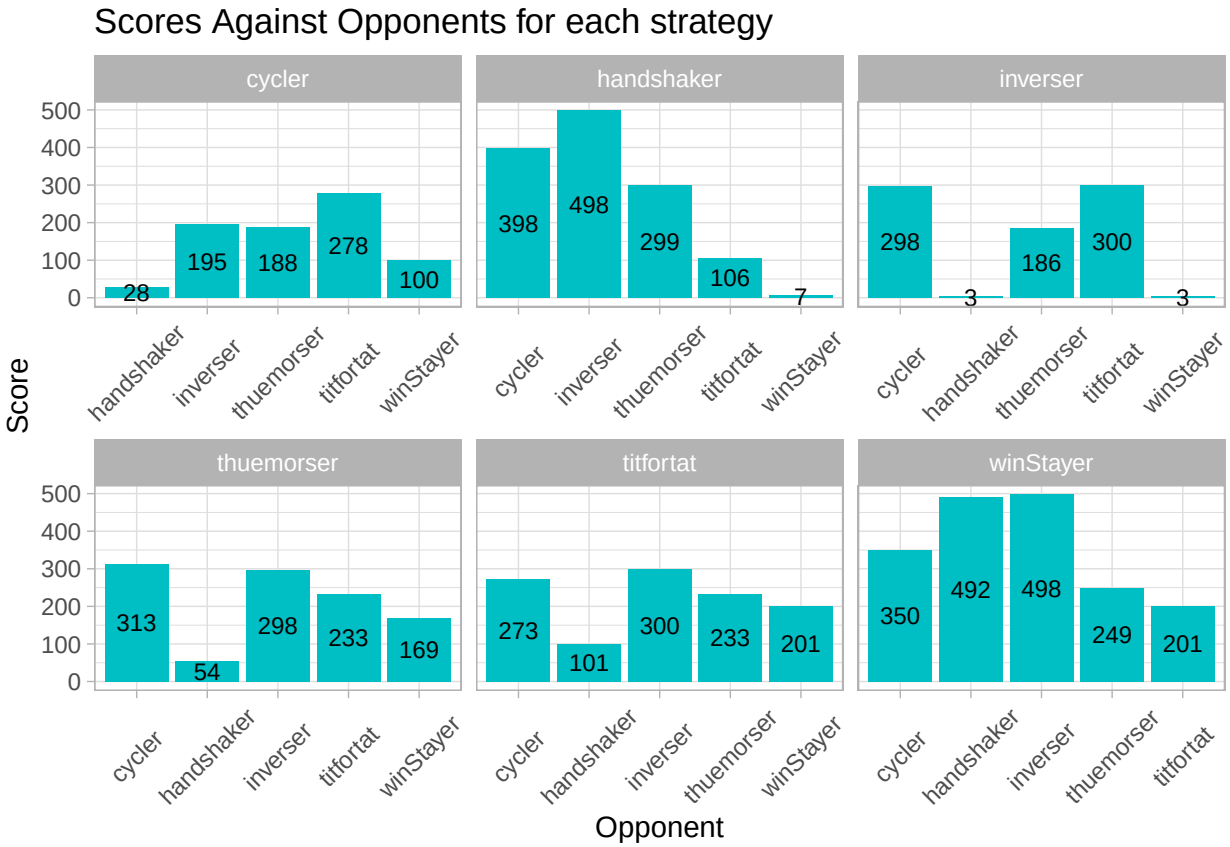
From this plot, we see that winStayer actually did best against inverser and handshaker. This makes sense as inverser doesn't have an absolute retaliation strategy and handshaker defected forever once it realized that winStayer was cooperating. It did worst against thuemorser and titfortat. This also makes sense because thuemorser is based on an almost random sequence and titfortat has a very similar strategy to winStayer (so they probably went back and forth switching moves against each other).

Question 3: Data Science Question

Create a plot, similar to the one above, that displays how each strategy fared against all the other strategies. Which strategies were most successful against which other ones? Comment on the patterns that you see.

```
score_all_plot <- player_data_long %>%
  ggplot(aes(x = opponent, y = score, label = score)) +
  geom_col(fill = "#00bfc4") +
  facet_wrap(~player, scales = "free_x") +
  geom_text(size = 3, position = position_stack(vjust = 0.5)) +
  theme_light() +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5, hjust = 0.5)) +
  labs(title = "Scores Against Opponents for each strategy",
       x = "Opponent",
       y = "Score")

score_all_plot
```




```
# png("score_all_plot.png", units="in", width=8, height=5, res=300)
# print(score_all_plot)
# dev.off()
```

Looking at the outcomes for all six strategies, we can first see that winStayer actually tied with titfortat, which makes sense given how similar their strategies are to one another. We can also see that when titfortat did lose, it lost only by a very small margin. In the other times it tied, so we can say that titfortat is definitely the most consistent strategy, which is why it is the best when competing against more strategies. winStayer, on the other hand, won with much higher margins most of the time against the other strategies other than titfortat (tied). We can also see that handshaker, the only strategy here without forgiveness, did the worst against titfortat and winStayer, the two strategies that act based on the move of the opponent and have forgiveness.

Question 4

What is the main difference between the game you played here and the tournament Axelrod implemented? How do you think this difference might matter?

The main difference between the game I played and the tournament Axelrod implemented is worded exactly in the question - a tournament. I only tested six strategies here, but Axelrod's tournament had many more different strategies competing against each other. As we saw in the graph from Question 3, titfortat was the most consistent performer, making it much more likely to succeed than other strategies when facing more opponents. With a limited number of strategies, titfortat lost out to other strategies that focus more on the win rather than only basing off the move of its opponent.