CS506 Deliverable 4 Report - 04/18/2021

Analyzing children's interactions in a Repeated Prisoner's Dilemma Game

Team 1 - Jessica Chen, Andrew Coughlan, Brendan Leap, Abhishek Samal, Maha Alrashed

1. Introduction

In this project, our purpose is to gain insight into children's interactions in a Repeated Prisoner's Dilemma Game. We explore a variety of statistical models that attempt to predict a child's traits based on their in-game decisions, including Drift Diffusion models, Random Forest Regression, Support Vector Regression, Logistic Regression, Linear Regression and a Naive Bayes classifier for multivariate Bernoulli models. The result of this study is expected to help us understand the cognitive and social mechanisms that support and drive social interactions.

1.1 Game description

In the game, children between the ages of 9 and 17 play 10 rounds against a pre-programmed partner. In each round, the child has to choose between cooperating with their partner thus earning no points and giving the partner three, or defecting and earning one point while the partner gets none. The game includes three pre-programmed partners designed to elicit aggressive and possibly forgiving responses:

- Cooperative partner: This partner cooperates on all rounds except for rounds 3 and 7.
- Defecting partner: This partner defects on all rounds except for rounds 3 and 7.
- Tit-for-tat partner: This partner cooperates in the 1st round and then mimics the child's choice in the previous round.

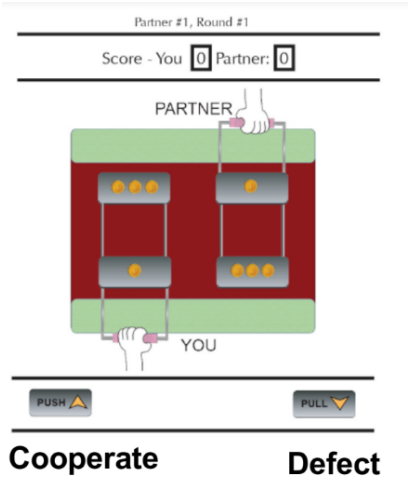


Figure 1: Visual representation of the Prisoner's Dilemma Game

1.2 Data description

The Social Development & Learning Lab has presented us with a data set containing 150 data points. For each child, we have data such as age, gender, reaction time for each trial, and measures of the child's traits. We also have each child's binary response to their preprogrammed partner (0 for Cooperation, 1 for Defection). The traits we will focus on in this report the *proactive aggression* score which measures the child's tendency to behave aggressively with no provocation, the *reactive aggression* score which measures the child's tendency to behave aggressively in reaction to provocation, and *total aggression* which is the combination of both. Here, proactive aggression is on a scale of 10 while reactive aggression is on a scale of 10. This data was collected by the lab through surveying the children's parents.

2. Data Exploration

For our team's data, we carefully visualized and analyzed the features we were given from the study. Initially, our goal was to manipulate the data into various forms to look for any patterns or correlations that could have been helpful in our continued research. During this stage, we plotted some very general statistics such as the average decisions and reaction times of the children in the study at any given point in the game. We also went through the amount of children with various aggression scores finding that we had very little data on children with higher reported "proactive" aggression. Quickly, though, we continued to refine and refocus our exploration again and again discovering many interesting things and even being able to help answer some of our clients critical questions. For example, the first question he sought to answer was whether we could notice a change in the child's preference to cooperate or defect after their computer opponent deviated from their normal pattern. To this goal, we honed in on and graphed out how the opponents as a whole reacted in the rounds following this betrayal of expectation, and we carefully adjusted the variables to look at this response for every possible category of child based on our other features. We looked at the data separating the children by whether they had above or below average aggression ratings for each of the three types of aggression being looked into for both the cooperative and defecting partners as they had pre-programmed patterns with rare deviations in the way the question considers as can be seen in Figure 2. We repeated this analysis using median as our metric for dividing our aggression groups and groupings of the children in three categories as low, medium, and high aggression as well just to cover another possibility. Both these showed a very drastic change in the average response amongst almost all groups of children, though the largest change was in the children with more "reactive" aggression and it is worth noting that the change was larger in the decrease in average cooperation after an opponent who usually cooperated defected as opposed to the increase in cooperation that followed an opponent who usually defected cooperating. Still, the children of all aggression levels did slowly return to their original pattern of responses in general over the next few rounds of the opponent following their normal behavior again. While this is perhaps one of our most impactful findings in exploring our data, we did continue digging creating similar graphs following the children over every round of the game as well as measuring our various groupings reaction times each

round. For that analysis, we also further divided all our previous categories into two depending on whether the child had cooperated or defected in that round to see if there were differences in reaction time based on that as well among the children.

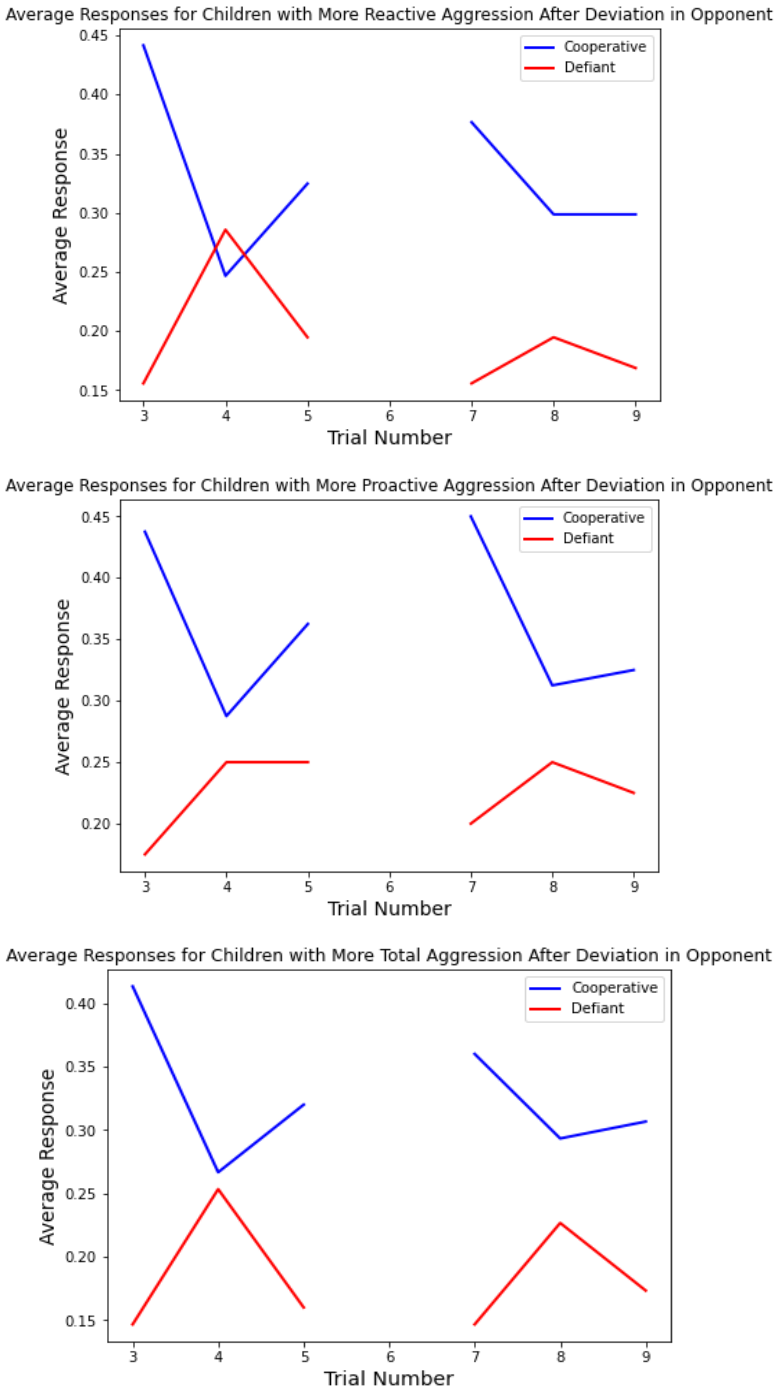


Figure 2: The Average Responses of Children that had higher than average Aggression scores after a deviation in the opponents' typical actions in both the Cooperative and Defective opponents. From top to bottom, the types of aggression being focused on are Reactive, Proactive, and Total.

3. Models

3.1 Drift Diffusion

During literature research, our team found that the drift diffusion model should only be applied to single-stage decisions made within 1-1.5 s (Ratcliff and Mackoon 2008). Most reaction times in our data set are much longer than the ideal range, and decisions made in the prisoners' dilemma game are multiple-staged decisions involving reasoning and strategy planning. Thus, after meeting with the client, we all agreed to move away from the Drift Diffusion Model.

3.2 Bayesian Model

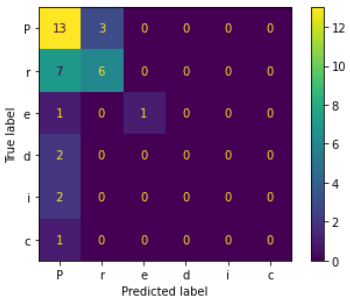
During a previous meeting, the client expressed interest in a Bayesian model. However, after discussion with Prof. Galletti, Spark! Manager Gowtham and Spark! staff lead Steve, we agreed that it might not be a good fit for this data set with such a small data set. Thus, we are planning to tell the client to move away from the Bayesian model in the next meeting. Furthermore, based on a literature review of previous research, we note that this model usually requires a number as large as 10,000 trials in order to be able to successfully predict a child's next move (Deng and Deng 2015).

3.3 Naive Bayes Classifier for Multivariate Bernoulli models

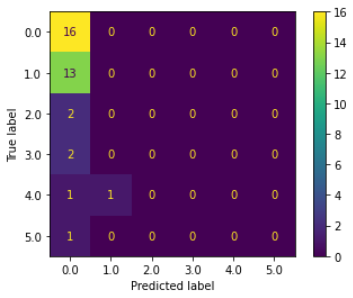
We used sklearn's Naive Bayes Classifier for Multivariate Bernoulli models to predict a child's aggression score as it implements the naive Bayes classification algorithm for data that is distributed according to multivariate Bernoulli distributions. This may be a promising model as it is best suited for data that has multiple binary features, much like our data. Training this model has produced the following results.

Results:

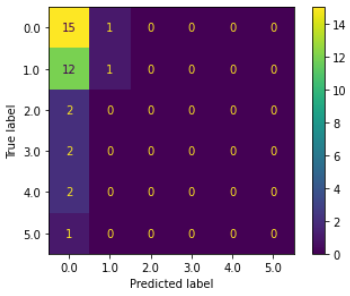
The model shows a low accuracy score of 55.5% at most when trained to predict proactive aggression scores using a TFT partner (Table 1, Exhibit A).



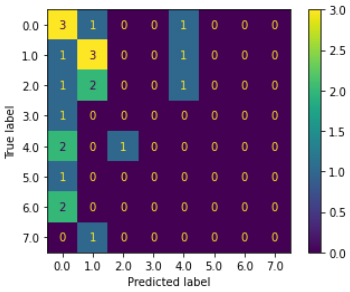
(A)
Predicting Proactive aggression score using TFT partner
Accuracy: 55.56%



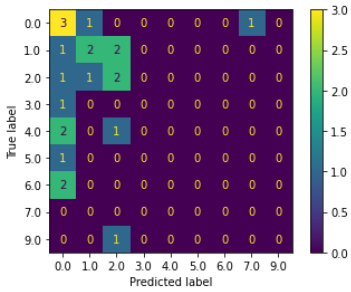
(B)
Predicting Proactive aggression score using cooperative partner
Accuracy: 44.44%



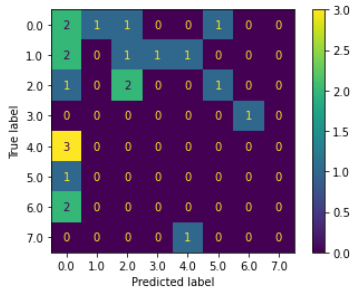
(C)
Predicting proactive aggression score using defecting partner
Accuracy: 44.44%



(D)
Predicting Reactive aggression score using TFT partner
Accuracy: 27.27%



(E)
Predicting Reactive aggression score using cooperative partner
Accuracy: 31.8%



(F)
Predicting Reactive aggression score using defecting partner
Accuracy: 18.18%

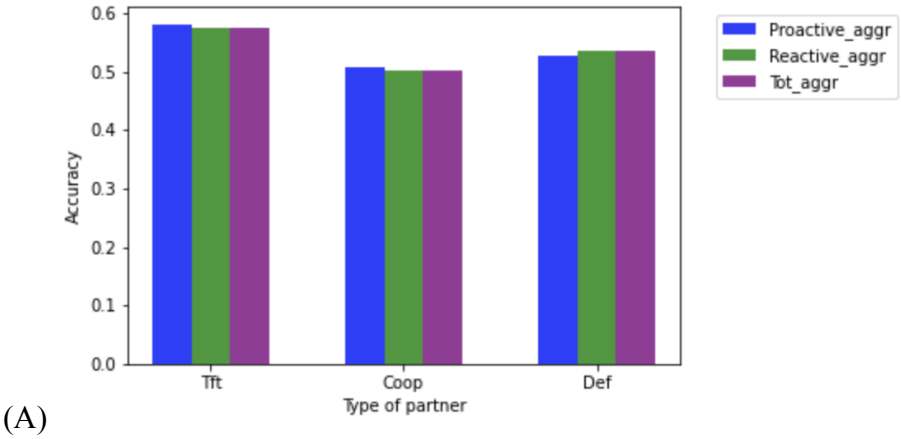
Table 1: Confusion matrix of the results of the Naive Bayes Classifier

3.4 Logistic Regression Model

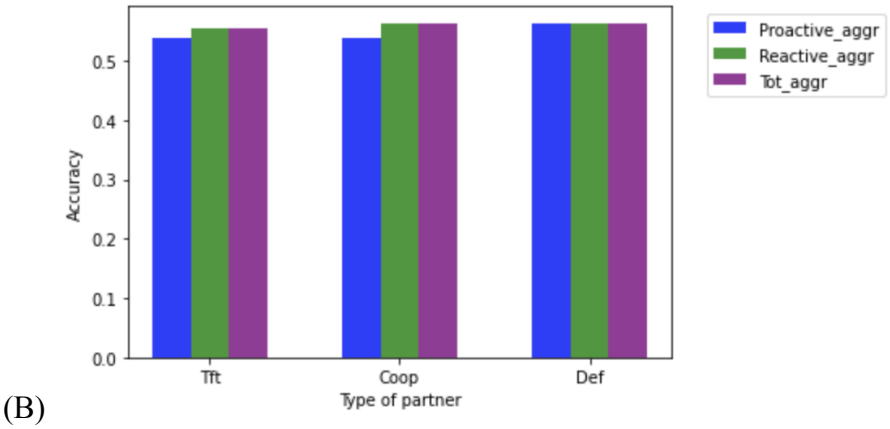
We used the sklearn library to train a logistic regression classifier to predict the aggression level of the children. Since this classifier can only predict discrete categorical variables, and some aggression score group has too few members, we first classified the children into above-the-median and below-the-median in respective to each aggression score (active, proactive, and total); samples with missing aggression scores are dropped. We will train a classifier for each kind of aggression score and each kind of opponent and report their accuracies by conducting 5-fold cross validation experiments.

Results:

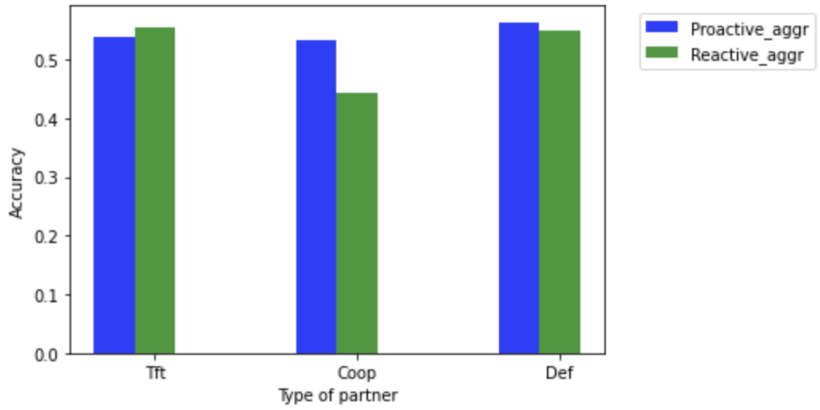
Most trained classifiers had an accuracy between 0.5 and 0.6 (Figure 1). When trained using decisions from all 10 rounds, the classifier trained using decisions against the TFT opponent had the highest accuracy. When trained using decisions of the latter 7 rounds, the accuracies of COOP and DEF classifiers noticeably increased, whereas the accuracies of TFT classifiers decreased. No significant improvements were seen in the classifier trained using round 4 and round 7 decisions only, and the accuracies of COOP classifiers decreased to only 0.44, indicating that just the two immediate rounds following the unexpected moves of the opponents are not sufficient to predict the kids' reactive or total aggression scores.



(A)



(B)



(C)

Figure 1: The average accuracy of the trained logistic regression classifiers for the respective type of partners and type of aggression scores in 5-fold cross validation experiments. (A) The classifier was trained using the kids' decisions from all 10 rounds. (B) The classifier was trained using the kids' decisions from rounds 4 to 10. (C) The classifier was trained using the kids' decisions on Round 4 and Round 8.

3.5 Assorted Continuous Regression Models

Linear regression, support vector regression, and random forest regression were implemented to predict aggression values on a continuous scale. Participants were classified on a 1-10 aggression scale based on their game decisions and reaction times. In

an attempt to obtain independence between decisions for each round, the input data was altered to include the context of the pre-programmed player's previous decision. Instead of using 0 and 1 corresponding to defection and cooperation, -1 corresponded to a defection when the simulation chose to cooperate previously, +1 when the participant chose to cooperate after the simulation defected, and 0 when the participant's response mirrored the simulation. Separate classifications occurred for each partner type. In the future the classification will take into account all three rounds played by each participant, but given the poor classification results for each round, the outcomes are unlikely to improve.

Results:

Under all circumstances the regression models performed poorly. None of the models were capable of classifications much better than random. This likely is the result of the lack of variability between the choices of high and low aggression players, wide differences in the actions of players within the same aggression category, and the small number of data points. Figures 2, 3, and 4 below are of the trends in the decisions of low, medium, and high aggression participants for every round.

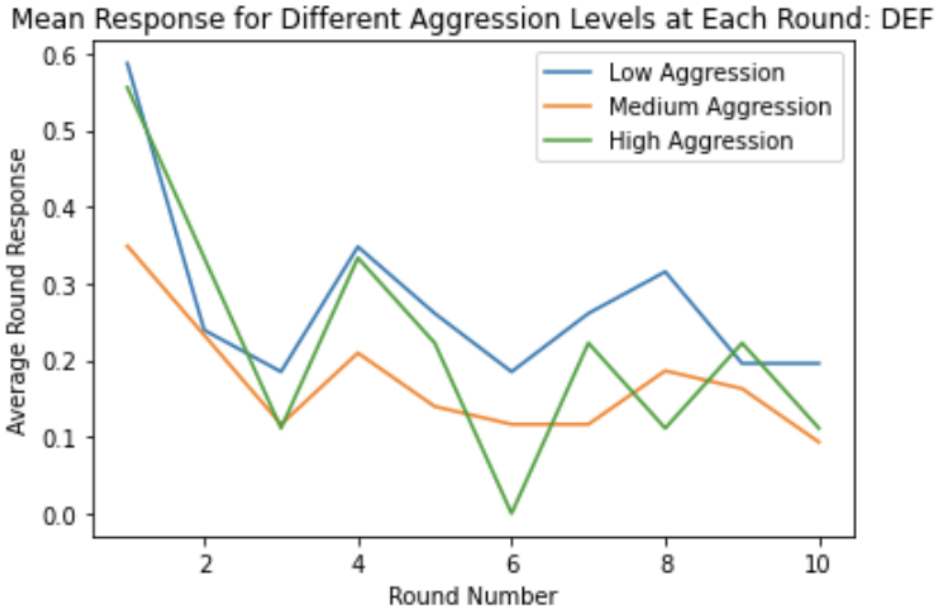


Figure 2: The mean game decision for each round against the defecting simulated partner sorted by reactive aggression level. Mean responses for all aggression categories all started neutral, with nearly half choosing to defect and the other half to cooperate. As the simulated player continued to defect, all aggression categories saw increased defections.

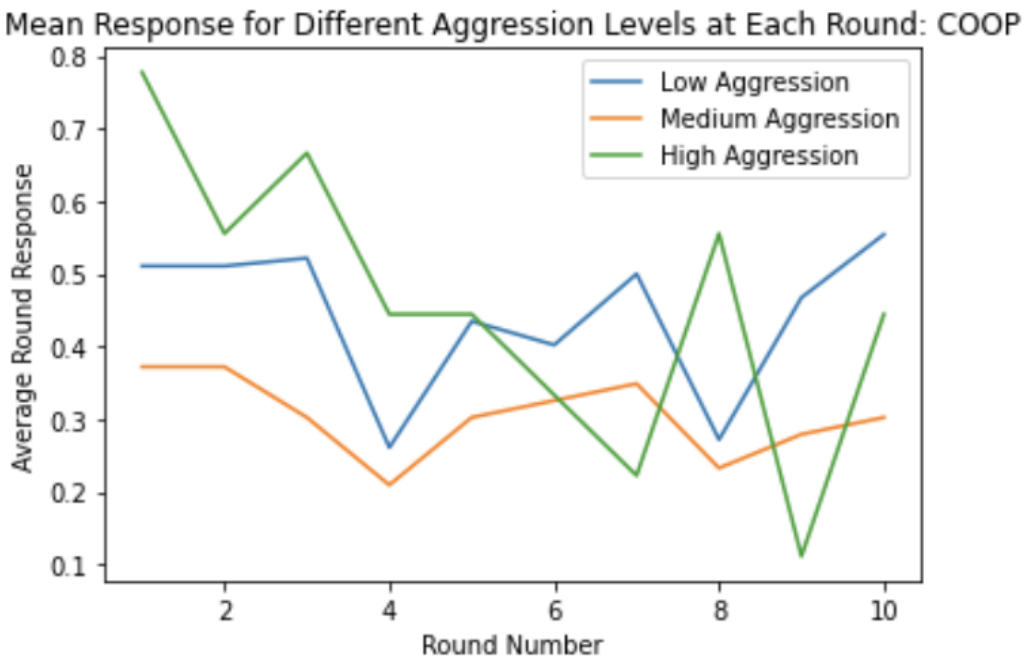


Figure 3: The mean game decision for each round against the cooperative simulated partner sorted by reactive aggression level. Mean responses started particularly cooperative for high-aggression participants, with nearly 80% choosing to cooperate in the first round. Cooperation decreased slightly throughout the round, possibly due to the two definition rounds programmed in the cooperative simulation.

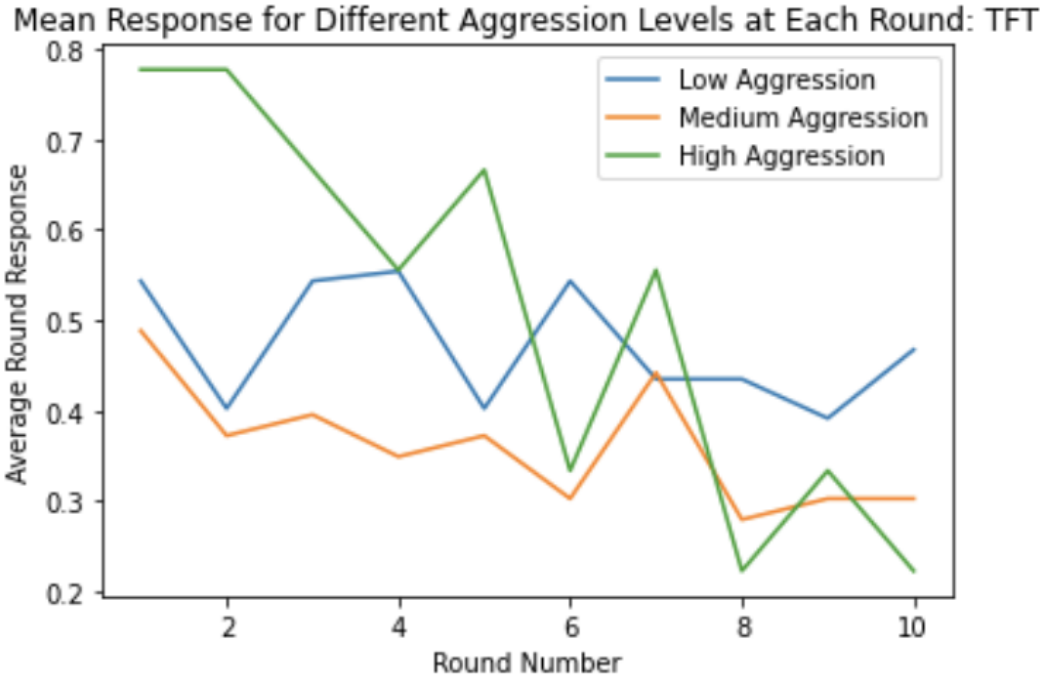


Figure 4: The mean game decision for each round against the Tit-for-Tat simulated partner sorted by reactive aggression level. Initial mean responses were particularly cooperative for high-aggression participants, with nearly 80% choosing to cooperate and the other half to cooperate. Cooperation decreased

significantly throughout the round, possibly due to the two definition rounds programmed in the cooperative simulation.

Are children with higher aggressive traits more less likely to return to cooperation over the three rounds after partner defection?

The plots in Figures 2, 3, and 4 show that sustained defection by the simulation prompts participants of all aggression levels to defect in the next round. For all simulation patterns, low-aggression participants were slightly more likely to be cooperative in the final rounds than higher-aggression participants. Other than when the simulated partner was consistently defecting, low-aggression participants usually ended with the same or greater likelihood of cooperation compared to when they began. In contrast, high-aggression participants were very likely to cooperate in the first round, but always began to defect more often after later rounds. From the data, it seems that low-aggression participants are in fact more likely to continue to cooperate as long as the partner mostly cooperates in good faith. This is particularly impressive in the TFT games, where a choice to cooperate after the simulation defects is needed to keep the cooperation percentages from spiraling downwards.

3.6 LSTM Neural Network

During previous meetings, the client has expressed wish for a model that can predict a kids' next move in the game given the opponent type, their aggression scores, and their previous moves. Neural network would be a good fit for this task, since it stores information about previous information. We have tried to build a Long-Short Term Memory (LSTM) neural network for this task. However, we have encountered several difficulties in this process.

First, the size of our dataset is too small to train a good neural network, and there will likely be overfitting. The structure of our dataset is not ideal for neural network either, since it consists of many kids with each kid only playing three games, instead of one same kid playing hundreds and thousands of games. In other words, we don't have enough data for each individual kid for the LSTM to learn how the kid will likely respond to a particular opponent move.

Second , none of the team members has specialized in coding neural networks, so the technical barrier is quite high, and there is the probability that the LSTM might not get successfully built by the end of the semester. We were referencing a study titled "Predicting Human Decision Making in Psychological Tasks with Recurrent Neural Networks" (Lin *et al* 2020). The code associated with this study was made public at https://github.com/doerlbh/HumanLSTM. However, their code was not annotated at all, so it took us a long time to understand the code in the first place, and after we successfully trained the LSTM following their methods, we realized that our dataset structure was too different from theirs, which means an LSTM trained in their way will not offer meaningful insight into our dataset. We have spent a lot of time trying to reshape our data to train our own LSTM, and have actively tried to enlist help by asking around in friends and families and going to Prof. Galletti's office hours for help.

Nevertheless, there remains an error in the loss function in the LSTM training process that we were unable to debug without more technical help.

3.7 Binary Decision Tree

Due to difficulties producing accurate predictions using regression and machine learning models, a binary decision tree was created to visualize and manually identify gameplay patterns found in either the entire data set or a specified subset of participants. Each node represents a specific path taken during the course of the game by the participant. At each node, the participant can choose to defect or cooperate in the next round, leading to two new nodes in the next layer. This pattern creates a binary tree as seen in Figure 5 below. Although visualization can only show a single value for each node, many useful metrics can be calculated. For example, the number of participants, percent cooperation, and percent cooperation separated by aggression levels are calculated for every node. This model would be capable of finding points of divergence between participants with different aggression levels at a node of interest, identifying the most common gameplay strategies, and producing the percent likelihood of aggression or other indicators based on the participant's path. A data frame containing metrics for each node will accompany the visualized decision tree.

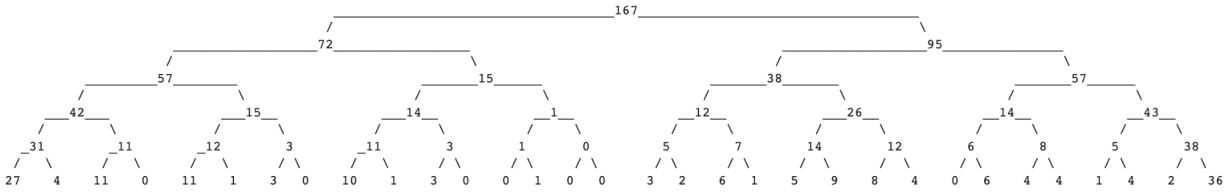


Figure 5: Binary decision tree for participant gameplay outcomes in the first five rounds against the Tit-for-Tat simulated partner. Each node displays the number of participants whose gameplay path passes through that node. At each node, a connection to the right signifies cooperation, while a connection to the left signifies defection. As expected with the TFT partner, players are likely to either mostly cooperate or mostly defect rather than alternate between cooperation and defection.

4. Conclusions

We started with an objective to study children's social interactions and understand what cognitive and social factors affect children's aggressive versus forgiving responses.
Initially, we visualized the data to gain insight into how players generally react to defection or cooperation by the pre-programmed partner in different rounds. We saw strong patterns among all players when the pre-programmed opponent made certain choices. For example when the partner deviates from its usual pattern of cooperation and defection we can see the average responses of children change in the next round.

After going through available literature on Drift Diffusion Model we came in accordance with the client to explore other machine learning models due to the limitations of our data as mentioned in the previous section.

We tried to predict the levels of aggression in a participant based on their responses and reaction time in each round by implementing sklearn's logistic regression, Linear Regression, Support Vector Regression, Random Forest Regression and Naive Bayes Classifier for Multivariate Bernoulli. We further tried training a Long Short Term Memory Neural Network but due to the extremely small size of the dataset none of the methods show promising results.

As data is still being collected(intimated by client), the models that we have made could be used later on the larger dataset which might show promising results. Further if the reaction time is restricted to 1000-1500ms(ideal reaction times for Drift Diffusion Model) in the data that is currently being collected, then along with the models we have built, a DDM can also be used. Some kind of aggression test for testing each kind of aggression should be designed as the aggression scores from Parent's Questionnaire that we have in the data seems biased(skewed towards lower scores) and not reliable.

5. Limitations

5.1 Limitations

- The non-twin dataset for the children's Prisoner's Dilemma game has a relatively small number of participants(167),out of these participants some have missing entries which further shrinks the data being considered for analysis.
- Each participant has only 30 data points, 10 data points against each type of partner which is a very small data size for data analysis and the model mostly overfits the data.
- The Reactive and Proactive aggression scores obtained from the parent's questionnaire is based on the parent's impression and not formal evaluation. Also the Aggression scores are skewed towards lower scores with very few higher aggression scores in the dataset.

References:

[1] Ratcliff R and McKoon G (2008) The Diffusion Decision Model: Theory and Data for Two-Choice Decision Tasks. *Neural Comput.* **20**: 873–922.
[2] Deng, X. and Deng, J (2015) A study of prisoner's dilemma game model with incomplete information. *Mathematical Problems in Engineering*, 1-10. doi:10.1155/2015/452042
[3] Lin B, Bouneffouf D, and Cecchi G (2020) Predicting Human Decision Making in Psychological Tasks with Recurrent Neural Networks. arXiv:2010.11413.

Code
Our Analysis is split into 3 files :

1) data_processing.ipynb
2) Bayesian_model.ipynb
3) LSTM.ipynb

1) The data_processing.ipynb file has:
    - the data exploration and visualization steps and result.
    - Logistic Regression
    - Linear Regression, Random Forest Regression, Support Vector Regression.
    - Decision Tree

Link:
https://colab.research.google.com/drive/18Enl37WUrzT7PFp0nMSPfwtgmpxcn2t8?usp=sharing

2) Bayesian_model.ipynb
    - Naive Bayes Classifier for Multivariate Bernoulli

Link:
https://colab.research.google.com/drive/1UUCJBRccNLjaOuuuO388w9lpLnH18z4W?usp=sharing

3) LSTM_deliverable4.ipynb
    - LSTM Neural Network

Link:
https://colab.research.google.com/drive/1HYPde2JOD-JlNcvk-fLeRJ76jXzbpH8G?usp=sharing

Notes for Using CodeBase

If google colab is being used to run the jupyter notebook then the below line of codes is used to import the file location

**********Code**********

```
perfrom google.colab import drive
drive.mount('file_location_in_the_drive')
```

****

****Note****

If User is not using Jupyter notebook on colab and directly running the code in local jupyter notebook or running in a .py python file, then the above command/code is not required.

```
The user can directly mention the filename with full datapath in the
DATASET_PATH variable in the code as below.
DATASET_PATH = 'path/Blake_RPD_Dataset-2.xlsx'
```

Installing and loading any custom libraries that are not there in the User's Jupyter Kernel

**********Code**********

```
# Install a pip package in the current Jupyter kernel
import sys
!{sys.executable} -m pip install library_name
```

****

Here inplace of *library_name* (Libraries like numpy, pandas, statistics, sklearn, etc) any library that is not there in the User's Jupyter Kernel, can be added.

****Note****

If User is not using Jupyter notebook and directly running the code in .py file, then the user has to install the libraries in the python installed directory to make the code work using :

**********Code**********

```
pip install library_name
```

****

Other than this the code blocks in the Google Colab file has comments describing what that particular code block does.

The google colab file for each of the 3 links above can be accessed from the links but they won't be able to run the code in these colab files(as they don't have the privileges/access for running the code). For running the code the code has to be copied to a different python file/jupyter notebook, the data-file path has to be specified in the code, the above instructions

are supposed to be followed(if it applies) and then the code can be executed to see the required results.