DATA OVERVIEW AND DIRECTION

MOTIVATION

The goal of the project is to gain insight into how aggressive behavior manifests in the iterated Prisoner's Dilemma game that client's lab ran with children. The client is particularly interested in "grudges" vs "forgiveness", or more concretely cooperation recovery after a defection vs continued defection as punishment/retaliation.

VARIABLES OF INTEREST

There are several options to consider as explanatory and response variables. Reasonable choices for a response variable are cooperation or defection, the relative frequency of cooperation, the recovery/forgiveness rate after the partner defects (how long it takes to get back to cooperation), and whether the partner "forgives" or not. These can be based on aggression scores from the questionnaire, reaction times, other characteristics (age, gender, income), or all of the above.

LIMITATIONS OF DATA

Some complications are the fact that the decision times are unbounded; the researchers did not set a time limit for the children to select cooperation or defection, and so there is a good deal of variation which may not be reflective of the children's thinking. One potential remedy is to remove outliers that took too long, but it may be more effective to adopt a new metric: the *relative* quickness of decisions made following the partner's defection compared to speed of prior decisions within the same round. The time in which the partner's defection occurs also presents another complication; since the partner randomly defects 20% of the time, naturally there will be (potentially significant) variation among individuals who would have otherwise behaved similarly had the initial defection occurred at the same time. Any analysis linking cooperation recovery and aggressiveness must be attentive to these concerns.

DRIFT DIFFUSION MODEL

Though the dataset the client provided contained a lot of information and presented many directions to go for analysis, the client had a specific model in mind: a drift diffusion model which would use decision times to uncover "true preferences" of cooperation or defection in the children playing the "forgiveness" round of the game. Drift diffusion models (DDM's) are often used in behavioral studies to model cognitive processing in dichotomous discrimination tasks. The "drift" in the title refers to the noisy accumulation of evidence in favor of one of two alternate choices[1], though it seems to most commonly be applied to tasks where there is a "correct" answer, which is not representative of this experiment. Nonetheless the model has recently been applied to more economic games like the Ultimatum game and the Dictator game, as well as a similar Iterated Prisoner's Dilemma game performed on adults instead of children[2].

REGRESSION MODEL

[1] Ratcliff and McKoon
[2] Gallotti and Grujic

Since we are interested in predicting something a regression model is appropriate. Depending on what covariates we include and what response variable we select, the model can be logistic, linear, quadratic, etc.. Some of the questionnaire data is categorical or quasi-categorical, which makes interpreting a multivariate regression model less intuitive, and perhaps merits

Since the client acquired data from two sets of twins, a logical path forward is to fit the regression model on one set of twins (probably the more complete one) and use the second set to evaluate its performance.

SOURCES CITED

Gallotti, Riccardo, and Jelena Grujic. "A Quantitative Description of the Transition between Intuitive Altruism and Rational Deliberation in Iterated Prisoner's Dilemma Experiments." *Nature: Scientific Reports*.

Ratcliff, Roger, and Gail McKoon. "The Diffusion Decision Model: Theory and Data for Two-Choice Decision Tasks." *PMC*.

DATA EXPLORATION / PRELIMINARY ANALYSIS

We examine cooperation proportion as the response variable with different covariates on the next pages.

In [63]:
```
import matplotlib.pyplot as plt
from cs506 import read
```

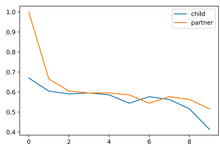
In [51]:
```
# Use CS506 library to import the CSV with the twins data
data = read.read_csv("Blake_RPD_Dataset-1_Twins.csv")

# remove blank columns
for i in range(len(data)):
    data[i] = data[i][:82]

# remove rows with blank entries
data_cleaned = []
for row in data:
    valid = True
    for item in row:
        if item == "" or item == " ":
            valid = False
    if valid:
        data_cleaned.append(row)

# remove header
data = data[1:]
```

In [69]:
```
print(data[0][13])
```

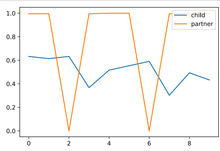
```
14
```

In [54]:
```
def getAverageDecisions(data, col):
    average_child = 0
    average_partner = 0
    for row in data:
        if row[col] != " ":
            if row[col][0] == "C":
                average_child += 1
            if row[col][1] == "C":
                average_partner += 1

    average_child = average_child/len(data)
    average_partner = average_partner/len(data)

    return average_child, average_partner
```

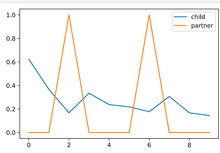
In [62]:
```
# Plot average cooperation proportion for tit-for-tat partner
tft_averages_child = []
tft_averages_partner = []

for col in range(22, 41, 2):
    average_child, average_partner = getAverageDecisions(data, col)
    tft_averages_child.append(average_child)
    tft_averages_partner.append(average_partner)

plt.plot(tft_averages_child, label="child")
plt.plot(tft_averages_partner, label="partner")
plt.legend(loc="upper right")
plt.show()
```



In [66]:
```
# Plot average cooperation proportion for cooperative partner
coop_averages_child = []
coop_averages_partner = []

for col in range(42, 61, 2):
    average_child, average_partner = getAverageDecisions(data, col)
    coop_averages_child.append(average_child)
    coop_averages_partner.append(average_partner)

plt.plot(coop_averages_child, label="child")
plt.plot(coop_averages_partner, label="partner")
plt.legend(loc="upper right")
plt.show()
```

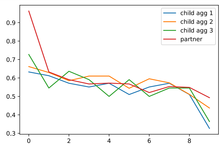


In [68]:
```
# Plot average cooperation proportion for defector partner
def_averages_child = []
def_averages_partner = []

for col in range(62, 81, 2):
    average_child, average_partner = getAverageDecisions(data, col)
    def_averages_child.append(average_child)
    def_averages_partner.append(average_partner)

plt.plot(def_averages_child, label="child")
plt.plot(def_averages_partner, label="partner")
plt.legend(loc="upper right")
plt.show()
```

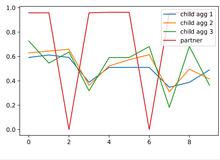


In [83]:
```
# Get average decisions for 3 different categories
def getAverageDecisionsAgg(data, col, aggCol, aggMax):
    child_agg1 = 0
    agg1_count = 0
    child_agg2 = 0
    agg2_count = 0
    child_agg3 = 0
    agg3_count = 0
    agg1_max = aggMax/3
    agg2_max = agg1_max*2
    average_partner = 0
    for row in data:
        if row[col] != " " and row[aggCol] != "":
            if row[aggCol] <= agg1_max:
                agg1_count += 1
                if row[col][0] == "C":
                    child_agg1 += 1
                if row[col][1] == "C":
                    average_partner += 1
            elif row[aggCol] <= agg2_max:
                agg2_count += 1
                if row[col][0] == "C":
                    child_agg2 += 1
                if row[col][1] == "C":
                    average_partner += 1
            else:
                agg3_count += 1
                if row[col][0] == "C":
                    child_agg3 += 1
                if row[col][1] == "C":
                    average_partner += 1

    child_agg1 = child_agg1/agg1_count
    child_agg2 = child_agg2/agg2_count
    child_agg3 = child_agg3/agg3_count
    average_partner = average_partner/len(data)

    return child_agg1, child_agg2, child_agg3, average_partner
```

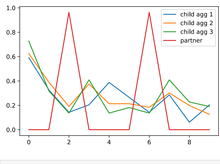
In [84]:
```
# Plot average cooperation proportion for 3 different aggression categories (from parent data) for tit-for-tat partner
tft_child_agg1 = []
tft_child_agg2 = []
tft_child_agg3 = []
tft_averages_partner = []

for col in range(22, 41, 2):
    child_agg1, child_agg2, child_agg3, average_partner = getAverageDecisionsAgg(data, col, 13, 31)
    tft_child_agg1.append(child_agg1)
    tft_child_agg2.append(child_agg2)
    tft_child_agg3.append(child_agg3)
    tft_averages_partner.append(average_partner)

plt.plot(tft_child_agg1, label="child agg 1")
plt.plot(tft_child_agg2, label="child agg 2")
plt.plot(tft_child_agg3, label="child agg 3")
plt.plot(tft_averages_partner, label="partner")
plt.legend(loc="upper right")
plt.show()
```



In [85]:
```
# Plot average cooperation proportion for 3 different aggression categories (from parent data) for cooperative partner
tft_child_agg1 = []
tft_child_agg2 = []
tft_child_agg3 = []
tft_averages_partner = []

for col in range(42, 61, 2):
    child_agg1, child_agg2, child_agg3, average_partner = getAverageDecisionsAgg(data, col, 13, 31)
    tft_child_agg1.append(child_agg1)
    tft_child_agg2.append(child_agg2)
    tft_child_agg3.append(child_agg3)
    tft_averages_partner.append(average_partner)

plt.plot(tft_child_agg1, label="child agg 1")
plt.plot(tft_child_agg2, label="child agg 2")
plt.plot(tft_child_agg3, label="child agg 3")
plt.plot(tft_averages_partner, label="partner")
plt.legend(loc="upper right")
plt.show()
```



In [86]:
```
# Plot average cooperation proportion for 3 different aggression categories (from parent data) for defector partner
tft_child_agg1 = []
tft_child_agg2 = []
tft_child_agg3 = []
tft_averages_partner = []

for col in range(62, 81, 2):
    child_agg1, child_agg2, child_agg3, average_partner = getAverageDecisionsAgg(data, col, 13, 31)
    tft_child_agg1.append(child_agg1)
    tft_child_agg2.append(child_agg2)
    tft_child_agg3.append(child_agg3)
    tft_averages_partner.append(average_partner)

plt.plot(tft_child_agg1, label="child agg 1")
plt.plot(tft_child_agg2, label="child agg 2")
plt.plot(tft_child_agg3, label="child agg 3")
plt.plot(tft_averages_partner, label="partner")
plt.legend(loc="upper right")
plt.show()
```



In [ ]:
```
```

These results are inconclusive, indicating that perhaps this is not the right response variable to base analysis around. Moving forward, we will experiment with relative reaction time as described earlier and cooperation/defection following the partner's first defection instead of cooperation percentage broadly.