HNZM2

**Experiment Overview:**

The purpose of this experiment is to determine if the type of feedback response after a gambling task affects future outcome predictions.

The reinforcement learning theory suggests that the feedback negativity should be larger when feedback is unexpected. The study that I replicated however, found that two recent studies showed that the feedback negativity was unaffected by outcome probability. To further examine this issue, participants in this replication study made reward predictions on each trial of a gambling task where objective reward probability was indicated by a cue. Participants made reward predictions following the cue, but prior to their gambling choice.

**Description of Procedure:**

Participants will first consent to participating in the experiment and then fill out a page collecting demographic information. After submitting that form they will be taken to an instructions page, where they will be told that they will be participating in a short gambling task, where 4 doors are presented and some prizes are behind a certain number of doors. The participants will then choose what door they believe holds a prize behind it. Before each trial, the number of doors with prizes behind it will be shown briefly as well as a fixation mark, and then before picking a door the participants will be asked whether or not they believe they will win the following trial. Following the first door choice the participant will be presented with either positive or negative feedback (green plus sign or circle). These steps will be repeated until all trials are completed.

**Experimenters Manual:**

In order to run this experiment, one only needs to run the file called: ‘GamblingTask1.py’.

To run open HNZM2 file in pycharm and then click on ‘GamblingTask1.py’ and then press the green arrow sign on the right of the program.

**Easy adaptations That Can Be Made To This Program:**

The number of trials can be changed by going down to the function called:

counter()

Within that function the experimenter will see an if statement that looks like:

if window.trialNumber <= 2

Place the desired number of trials in place of the **2**. Please keep in mind that numbers in code begin with 0. So, if you are looking to have 3 trials you would put in 2.

Also go to the part of the code that says:

window.preDeterminedResults = pseudorandomAssignment(4, 0.5)

Then within the parentheses the number 4 must also be changed to at least one above the number of trials that you have placed in the window.trialNumber. This will keep the window.TrialNumber index from going out of range when you are drawing out the pseudorandomized results.

To change the color of the blinking label:

Go into the ‘FunctionsHolder.py’ and go down to the lblColor() function. Underneath that function you will see:

lblColor.blink('pink','purple')

Replace this with the colors that you want to flash at the end of the program. Make sure to use the string of colors: example 🡪 ‘red’, ‘blue’

**Data Collection**

Demographic information can be found in a file called:

DemographicInfo.csv

Demographic information will include:

Name, Age, Gender, University Attended, Student Status, and Focus of Study.

Participant Information gathered from the experiment can be found in file called:

ParticipantInfo.csv

The columns include what trial the participant is on, how many doors were shown to have prizes behind them, the participant’s subsequent prediction of outcome, and whether or not they were shown positive or negative feedback. Please follow headers for further clarification.

To open either file in finder, right click on the file name, and then press **reveal in finder.**

Please note that for the sound effect file in this program to work, if the computer you are opening the program on does not have ‘Pygame’ it must be installed. This can easily be done by writing at the top of the program:

import pygame

A small red bulb will pop up on the left of the words. Click on the red bulb and install pygame in order to have appropriate sound effects.

**Program Highlights:**

This program is interactive in that it allows the participant to directly click on any of the 4 doors to make a choice. The keyboard is also connected to certain pages of the program. At the end there are two different animations, a typewriter animation that spells out how much the participant has earned after all the trials as well as a blinking label. To go along with the animation at the end there is also a typewriter sound effect.

What makes this program intricate is the use of automatically turning pages that gets the participant from one point of the experiment to the next. Through the use of timers each page is timed for a specific amount (please see hashtags in code for all the times of each specific page). Each time and function linked to a timer and had to be planned carefully so that each trial was carried out appropriately. Some of the functions were created in a separate module referred to as ‘FunctionsHolder’ and imported later into the main file. Most of the animations will be in this folder.